МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Санкт-Петербургский государственный университет
телекоммуникаций им. проф. М.А.Бонч-Бруевича»

На правах рукописи

Мутханна Аммар Салех Али

РАЗРАБОТКА И ИССЛЕДОВАНИЕ КОМПЛЕКСА МОДЕЛЕЙ И МЕТОДОВ ИНТЕГРАЦИИ ГРАНИЧНЫХ ВЫЧИСЛЕНИЙ В СЕТЯХ СВЯЗИ ПЯТОГО И ШЕСТОГО ПОКОЛЕНИЙ

2.2.15. Системы, сети и устройства телекоммуникаций
Диссертация на соискание ученой степени
доктора технических наук

Научный консультант
доктор технических наук, профессор
Кучерявый Андрей Евгеньевич

Санкт-Петербург – 2023

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования. Развитие сетей связи за последние 20 лет можно охарактеризовать появлением целого ряда новых технологий, которые вначале обеспечили возможности предоставления достаточно широкого спектра услуг пользователям сети, включая масштабное развитие мобильной телефонии, а впоследствии и мобильной передачи данных и видео, а затем к настоящему времени привели к принципиальному изменению как в принципах построения сети, так и в предоставляемых пользователям услугах. При этом ключевую роль в формировании нового облика сетей связи и предоставляемых сетью услугах сыграли концепции Интернета Вещей и Тактильного Интернета. Первая способствовала появлению сетей высокой и сверхвысокой плотности, а вторая – сетей связи с ультра малыми задержками. Все это вместе взятое потребовало комплексного использования ресурсов всевозможных сетей и преобразовало сети связи из гомогенных в гетерогенные.

В начале третьего десятилетия 21 века стало ясно, что нескоординированное развитие мобильных и фиксированных сетей связи не способствует решению проблемы интегрирования всех ресурсов всех сетей для предоставления современных услуг всем пользователям сетей связи общего пользования и на этапе формирования подходов к реализации сетей связи шестого поколения 6G появилась новая концепция развития сетей связи, в основе которой лежит понимание необходимости интеграции не только разнообразных технологий в рамках тех или иных сетей, но и интеграции сетей связи в единую сеть. Эта концепция называется интегрированные сети Космос-Воздух-Земля-Море SAGSIN (Space-Air-Ground-Sea). Представляется, что эта

концепция определяет множество научных проблем и задач, по крайней мере, на ближайшее десятилетие. В связи с изложенным диссертационная работа, в которой решается научная проблема разработки и исследования комплекса моделей и методов интеграции граничных и туманных вычислений в сетях связи пятого и шестого поколений для глобального фрагмента Воздух-Земля концепции SAGSIN является актуальной.

Разработанность темы исследования. Существует множество работ в области сетей 5G и последующих поколений, высокоплотных и сверхплотных сетей, сетей связи с ультра малыми задержками как теоретического, так и экспериментального плана.

Определяющий вклад в теоретические и экспериментальные исследования этих научных проблем внесли российские и зарубежные ученые В. М. Вишневский, К.Е.Самуйлов, Ю. В. Гайдамака, Б. С. Гольдштейн, В. Г. Карташевский, А. И. Парамонов, А. Е. Кучерявый, Е.А.Кучерявый, М. С. Маколкина, Д.А.Молчанов, Р. В. Киричек, А. П. Пшеничников, В. К. Сарьян, С. Н. Степанов, М. А. Сиверс, Н. А. Соколов, В. О. Тихвинский, М. А. Шнепс-Шнеппе, M. Dohler, G. Fettweis, J. Hosek, A. A. Ateya, M. Maier, M. Z. Shafig и другие.

Настоящая диссертация в отличие от известных подходов к построению и исследованию сетей связи направлена на решение научной проблемы разработки и исследования комплекса моделей и методов интеграции граничных и/или туманных вычислений в сетях связи пятого и шестого поколений для глобального фрагмента Воздух-Земля концепции SAGSIN.

Обеспечение эффективности интеграции глобального фрагмента сети SAGSIN может быть достигнуто только при интеграции различных технологий телекоммуникаций, обеспечивающих собственно построение как наземных и воздушных сетей. Поэтому в диссертационной работе при разработке комплекса

моделей и методов интеграции граничных и/или туманных вычислений исследуются проблемы их интеграции для сетей беспилотных летательных аппаратов (БПЛА), программно-конфигурируемых сетей SDN (Software Defined Networks), сетей взаимодействия устройство-устройство D2D (Device-to-Device), сетей автомобильного транспорта VANET (Vehicular Ad Hoc Networks), беспилотных автомобилей, Интернета Вещей,

По каждой из этих технологий также существует достаточно большое число работ отечественных и зарубежных ученых В. М. Вишневского, К.Е.Самуйлова, Ю. В. Гайдамаки, В. Г. Карташевского, А. И. Парамонова, А. Е. Кучерявого, Е.А.Кучерявого, Д.А.Молчанова, С.Д.Андреева, Р. В. Киричка, В. К. Сарьяна, Р.Л. Смелянского, Е.М.Хорова, Ian F. Akyildiz, M. Dohler, G. Fettweis, J. Hosek, A. A. Ateya, M. Maier, Halim Yanikomeroglu, M. Z. Shafig, Tarik Taleb и других.

Объект и предмет исследования. Объектом исследования являются сети связи пятого и последующих поколений. Предметом исследования является комплекс моделей и методов интеграции граничных и туманных вычислений в сетях связи пятого и шестого поколений.

Цель и задачи исследования. Целью диссертационной работы является разработка и исследование комплекса моделей и методов интеграции граничных и/или туманных вычислений в сетях связи пятого и шестого поколений.

Цель работы достигается путем последовательного решения следующих задач:

- анализ существующего положения в области исследования сетей связи пятого и последующих поколений, роли и места граничных и/или туманных вычислений, а также сетей беспилотных летательных аппаратов (БПЛА), программно-конфигурируемых сетей SDN (Software Defined Networks), сетей взаимодействия устройство-устройство D2D (Device-to-Device), сетей

автомобильного транспорта VANET (Vehicular Ad Hoc Networks), беспилотных автомобилей, Интернета Вещей в развитии сетей и систем связи;

- разработка метода иерархической кластеризации мультиконтроллерной сети, включающую в себя кластеры с головными узлами и централизованный контроллер,
- разработка метода построения мультиконтроллерной сети, основанного на интегральном решении задач по размещению контроллеров в мультиконтроллерных сетях, использующего как метаэвристический алгоритм, так и алгоритм балансировки нагрузки,
- разработка модифицированного алгоритма хаотического роя сальп для использования в иерархических кластерных сетях clus-CSSA,
- разработка модели сети с мобильными серверами граничных вычислений на БПЛА и метода выгрузки трафика с наземной сети на мобильные серверы граничных вычислений на БПЛА с трехуровневой процедурой выгрузки,
- анализ эффективности предложенных методов построения сетей Воздух-Земля по сравнению с существующими по параметрам задержки, энергетической эффективности и доли заблокированных задач по выгрузке трафика,
- разработка метода построения сети с использованием технологий MEC, SDN и D2D для поддержки приложений беспилотных автомобилей,
- разработка метода прогнозирования трафика на основе CNN - LTP-CNN с реализацией алгоритма прогнозирования на туманных узлах,
- разработка метода размещения SDN-контроллеров в мультиконтроллерных сетях на мобильных узлах сетей VANET, например, автобусах, для обеспечения связи в плотных и сверхплотных сетях 6G и взаимодействия с туманной средой устройств сети.

Научная новизна. Научная новизна работы состоит в следующем:

1. Решена научная задачи, отличающаяся от известных тем, что предложен метод построения мультиконтроллерной сети, основанный на интегральном решении задач по размещению контроллеров в мультиконтроллерных сетях, базирующийся на метаэвристическом алгоритме вследствие сложности решаемых задач, и алгоритме балансировки нагрузки, позволяющем обеспечить наилучшее использование ресурсов контроллеров в таких сетях.
2. Для решения научной задачи интеграции размещения контроллеров в мультиконтроллерных сетях и балансировки нагрузки предложено использовать в отличии от известных решений иерархическую кластеризацию мультиконтроллерной сети, включающую в себя кластеры с головными узлами и централизованный контроллер, что обеспечивает балансировку нагрузки в разработанном методе построения сети.
3. Для решения научной задачи интеграции контроллеров в мультиконтроллерных сетях и балансировки нагрузки в отличии от известных решений разработан модифицированный алгоритм хаотического роя сальп для использования в иерархических кластерных сетях clus-CSSA, что позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1мс до 10мс по сравнению как с широко известными метаэвристическими алгоритмами роя частиц PSO (Particle Swarm Optimization) и серого волка GWO (Grey Wolf Optimization), так и с предыдущей версией хаотического алгоритма роя сальп CSSA (Chaotic Salp Swarm Algorithm). При этом для наиболее сложного случая задержки величиной в 1мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза.
4. Для решения научной задачи интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и

сверхвысокой плотности разработаны модель сети, отличающаяся от известных тем, что предложено для уменьшения задержки и энергопотребления в такой сети использовать мобильные серверы граничных вычислений на БПЛА и метод выгрузки трафика с наземной сети на мобильные серверы граничных вычислений на БПЛА, отличающийся от известных тем, что процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, который определяет сложность вычисляемой задач и по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений и на 30-40% по сравнению с сетью с использованием только наземных граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш около 10% по сравнению с использованием неоптимизированного алгоритма.

5. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение энергопотребления до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений. Кроме того,

использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш в 5-10% по сравнению с использованием неоптимизированного алгоритма.

6. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение доли заблокированных задач по выгрузке трафика в десятки раз по сравнению с сетью без использования технологий граничных вычислений, в разы по сравнению с сетью с использованием только наземных граничных вычислений. Использование оптимизации на основе метаэвристического хаотического роя сальп не дает практически значимого эффекта по сравнению с неоптимизированным алгоритмом. Кроме того, определены зависимости значений задержки, энергопотребления и доли заблокированных задач по выгрузке трафика от плотности сети.
7. В отличии от известных решений предложен новый метод построения сети с использованием технологий MEC, SDN и D2D для поддержки приложений беспилотных автомобилей. Предлагаемая архитектура направлена на преодоление двух основных проблем автомобильных сетей – высокой плотности трафика и наличия непокрытых зон в автомобильной сети связи. При этом, разработан также алгоритм кластеризации на основе взаимодействий D2D для транспортных средств в непокрытых зонах и для выгрузки трафика сети в регионах с интенсивным движением. Результаты моделирования показывают, что предложенная архитектура дает 74% прироста производительности системы в терминах вероятности блокировки задач.
8. Разработан метод прогнозирования трафика на основе CNN - LTP-CNN, который предсказывает трафик сети IoT на базе состояния сети за

предыдущий интервал времени, отличающийся от известных тем, что алгоритм прогнозирования реализован на туманных узлах, которые представляют собой основную часть сетей IoT/5G. Результаты показывают, что разработанный LTP-CNN может предсказывать трафик сети IoT с точностью около 90%.

9. Разработан метод размещения SDN-контроллеров в мультиконтроллерных сетях, отличающийся тем, что контроллеры могут располагаться на мобильных узлах сетей VANET, например, автобусах, для обеспечения связи в плотных и сверхплотных сетях 6G и взаимодействия с туманной средой устройств сети, что позволяет уменьшить задержку на 60% по сравнению с традиционными моделями граничных вычислений на базе SDN, а также снизить потребляемую энергию на 72% по сравнению с методом Fog-MEC на базе SDN.

Теоретическая значимость работы. Теоретическая значимость диссертационной работы состоит, прежде всего, в разработке и исследовании комплекса моделей и методов интеграции граничных и/или туманных вычислений в сетях связи пятого и шестого поколений для глобального фрагмента Воздух-Земля концепции SAGSIN. Это закладывает основы для перехода сетей связи общего пользования в среднесрочной перспективе к интегрированным сетям.

Важными результатами, имеющими существенную теоретическую ценность, представляются метод построения мультиконтроллерной сети, основанный на интегральном решении задач по размещению контроллеров в мультиконтроллерных сетях, базирующийся на метаэвристическом алгоритме вследствие сложности решаемых задач, и алгоритме балансировки нагрузки, позволяющем обеспечить наилучшее использование ресурсов контроллеров в таких сетях, решения по иерархической кластеризации мультиконтроллерной

сети, модифицированный алгоритм хаотического роя сальп для использования в иерархических кластерных сетях clus-CSSA, трехуровневая процедура выгрузки трафика, модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности. Самостоятельную научную ценность имеет метод размещения SDN-контроллеров в мультиконтроллерных сетях на мобильных узлах сетей VANET, например, автобусах, для обеспечения связи в плотных и сверхплотных сетях 6G и взаимодействия с туманной средой устройств сети.

Практическая ценность работы. Практическая ценность работы состоит в создании научно-обоснованных рекомендаций по интеграции граничных и/или туманных вычислений в современных сетях связи с учетом массового внедрения новых услуг связи, включая услуги телеприсутствия, что реализуется как в методиках планирования сетей связи ПАО «ГИПРОСВЯЗЬ», так и в международных стандартах (рекомендациях) Сектора стандартизации электросвязи Международного союза электросвязи.

Реализация результатов работы. Полученные в диссертационной работе результаты внедрены в ПАО «ГИПРОСВЯЗЬ» при разработке методик планировании сетей связи пятого поколения, в ФГУП НИИР при выполнении государственных контрактов по научно-техническому и методическому обеспечению выполнения Министерством цифрового развития, связи и массовых коммуникаций функций Администрации связи Российской Федерации в Секторе стандартизации электросвязи Международного союза электросвязи в работах по разработке стандартов (вкладов), в РУДН при создании научного центра моделирования беспроводных сетей 5G, в Уфимском университете науки и технологий при чтении лекций, проведении практических занятий и лабораторных работ, в Санкт-Петербургском государственном университете телекоммуникаций им. проф. М.А. Бонч-Бруевича при проведении работ по

Мегагранту «Исследование сетевых технологий с ультра малой задержкой и сверхвысокой плотностью на основе широкого применения искусственного интеллекта для сетей 6G» по соглашению № 075-15-2022-1137 с Министерством науки и высшего образования РФ, чтении лекций, проведении практических занятий и лабораторных работ. Акты реализации диссертационных исследований представлены в Приложении Б.

Методы исследования. При проведении исследований были использованы методы теории телетрафика и теории массового обслуживания, теории оптимизации, теории вероятностей, а также метаэвристические алгоритмы и методы имитационного моделирования.

Основные положения, выносимые на защиту.

1. Метод построения мультиконтроллерной сети, основанный на интегральном решении задач по размещению контроллеров в мультиконтроллерных сетях, базирующийся на метаэвристическом алгоритме вследствие сложности решаемых задач, и алгоритме балансировки нагрузки, позволяет обеспечить наилучшее использование ресурсов контроллеров в таких сетях.
2. Модифицированный алгоритм хаотического роя сальп для использования в иерархических кластерных сетях clus-CSSA позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1мс до 10мс по сравнению как с широко известными метаэвристическими алгоритмами роя частиц PSO (Particle Swarm Optimization) и серого волка GWO (Grey Wolf Optimization), так и с предыдущей версией хаотического алгоритма роя сальп CSSA (Chaotic Salp Swarm Algorithm). При этом для наиболее сложного случая задержки величиной в 1мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза.

3. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности, в котором процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, определяющий сложность вычисляемой задачи, а по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА. При этом результаты моделирования доказали, что обеспечиваются уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений и на 30-40% по сравнению с сетью с использованием только наземных граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш около 10% по сравнению с использованием неоптимизированного алгоритма.
4. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение энергопотребления до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш в 5-10% по сравнению с использованием неоптимизированного алгоритма.
5. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение доли заблокированных задач по

выгрузке трафика в десятки раз по сравнению с сетью без использования технологий граничных вычислений, в разы по сравнению с сетью с использованием только наземных граничных вычислений. Использование оптимизации на основе метаэвристического хаотического роя сальп не дает практически значимого эффекта по сравнению с неоптимизированным алгоритмом. Кроме того, определены зависимости значений задержки, энергопотребления и доли заблокированных задач по выгрузке трафика от плотности сети.

6. Метод построения сети с интеграцией технологий MEC, SDN и D2D для поддержки приложений беспилотных автомобилей и алгоритм кластеризации на основе взаимодействий D2D для транспортных средств в непокрытых зонах и для выгрузки трафика сети в регионах с интенсивным движением дает 74% прироста производительности системы в терминах вероятности блокировки задач.
7. Метод прогнозирования на основе CNN - LTP-CNN, который предсказывает трафик сети IoT по информации о состояния сети за предыдущий интервал времени, реализующийся на туманных узлах, которые представляют собой основную часть сетей IoT/5G позволяет предсказывать трафик сети IoT с точностью около 90%.
8. Метод размещения SDN-контроллеров в мультиконтроллерных сетях, который предусматривает размещение контроллеров на мобильных узлах сетей VANET, например, автобусах, для обеспечения связи в плотных и сверхплотных сетях 6G и взаимодействия с туманной средой устройств сети, позволяет уменьшить задержку на 60% по сравнению с традиционными моделями граничных вычислений, а также снизить потребляемую энергию на 72% по сравнению с методом Fog-MEC.

Достоверность результатов. Степень достоверности полученных результатов подтверждается корректным применением математического аппарата, результатами натурного и имитационного моделирования, а также широким спектром публикаций и выступлений как на российских, так и на международных научных конференциях.

Апробация работы. Основные положения диссертационной работы обсуждались и были одобрены на следующих конгрессах, конференциях и семинарах: Международной конференции по проводным и беспроводным сетям и системам следующего поколения NEW2AN (Санкт-Петербург, 2015 - 2020); Международной конференции «Распределенные компьютерные и телекоммуникационные сети: управление, вычисление, связь» DCCN (Москва, 2017- 2021); Международной научно-технической и научно-методической конференции «Актуальные проблемы инфотелекоммуникаций в науке и образовании» АПИНО (Санкт-Петербург, 2019–2022); Региональной научно-методической конференции магистрантов и их руководителей «Подготовка профессиональных кадров в магистратуре для цифровой экономики» ПКМ (Санкт-Петербург, 2022); Всероссийской научно-технической конференции, посвященной Дню радио (Санкт-Петербург, 2017, 2021); 4-й Международной конференции по сетям будущего и распределенным системам ICFNDS (Санкт-Петербург, 2017-2020); Международном конгрессе ультрасовременных телекоммуникаций и систем управления (ICUMT, 2018-2021), IEEE Конференции молодых ученых России в области электротехники и электронной техники (ElConRus 2017), IEEE Международной конференции по электротехнике и фотонике (EexPolytech 2019), 4-я Международной конференции МЕС по большим данным и умному городу (ICBDSC 2019), XXII

Международной конференции по программным вычислениям и измерениям (SCM 2019)), Международном симпозиуме по потребительским технологиям (ISCT 2018), 20-ой международной конференции по передовым коммуникационным технологиям (ICACT, 2018). IEEE конференции Системы синхронизации, генерации и обработки сигналов в телекоммуникациях (SINKHROINFO 2017), 18-ой конференции Ассоциации «Открытые инновации» и семинару по информационной безопасности и защите информационных технологий (FRUCT-ISPIT 2016), 38-ой Международной конференции по телекоммуникациям и обработке сигналов (TSP 2015), 15-ой Международной конференции “Проводные/беспроводные интернет-коммуникации” IFIP WG (WWIC 2017). 14-ой международной научно-технической конференции «Перспективные технологии в средствах передачи информации» ПТСПИ (Владимир, 2021), Молодежной научной школе по прикладной теории вероятностей и телекоммуникационным технологиям (АРТСТ-2017). Были сделаны доклады на пленарных заседаниях следующих конференций: Международной научно-технической и научно-методической конференции «Актуальные проблемы инфотелекоммуникаций в науке и образовании» АПИНО (Санкт-Петербург 2022), Международной конференции «Распределенные компьютерные и телекоммуникационные сети: управление, вычисление, связь» DCCN (Москва 2022) и 5-ой Международной школе прикладной теории вероятностей, Коммуникационные технологии и наука о данных (АРТСТ-2020).

Публикации. Основные результаты диссертации изложены в 137 опубликованных работах, в том числе в 24 работах, опубликованных в журналах из перечня ВАК Министерства образования и науки Российской Федерации; 87 работах в изданиях, включенных в международные базы цитирования; 2

результатах интеллектуальной деятельности; 6 отчетах о НИР; 18 работах в других научных изданиях и материалах конференций.

Соответствие паспорту специальности. Диссертационная работа выполнена по специальности 2.2.15. Системы, сети и устройства телекоммуникаций и соответствует следующим пунктам паспорта специальности: 2, 4, 7, 8, 9, 10, 11, 12, 18.

Структура и объем работы. Диссертация состоит из введения, 6 глав, заключения, списка литературы и двух приложений. Работа содержит 393 страницы машинописного текста, 75 рисунков, 25 таблиц и список литературы из 366 наименований.

Личный вклад автора. Все основные результаты диссертационной работы получены автором самостоятельно. Экспериментальные исследования проведены под научным руководством автора при его непосредственном участии.

ГЛАВА 1. ПЕРЕХОД К ИНТЕГРИРОВАННОЙ АРХИТЕКТУРЕ СЕТИ 6G

1.1 Интегрированная архитектура сети «воздух-земля»

Целями данной диссертации являются изучение вопроса интеграции уровней сетей (наземной и летающей), разработка подходящего метода интеграции с использованием таких технологий, как граничные вычисления, искусственный интеллект (ИИ) и программно-конфигурируемые сети (SDN). Интегрированная архитектура выбрана на основе рекомендаций и обзора литературы по сетям 6G и ключевым технологиям, спецификациям, технологиям связи внутри и между уровнями, сценариям использования и важнейшим показателям производительности.

В данной диссертации рассматривается архитектура наземной сети, способной обрабатывать трафик со всех уровней. Численное моделирование и симуляция работы позволят предсказать производительность сети. Кроме того, для всех ключевых элементов, включая высокопроизводительные граничные вычислительные узлы, контроллеры SDN, коммутаторы OpenFlow и интерфейсы, будут определены характеристики и установлены технические требования. Наземная сеть, которая выступает ядром системы, будет спроектирована с учетом требований приложений 6G. Ключевые особенности проектирования сетей радиодоступа (RAN) 6G заключаются в создании сверхплотных сетей, гетерогенных сетей, интеллектуальных и адаптивных сетей, а также в повышении энергоэффективности. Интеграция наземного и летающего уровней сетей является главной задачей, при этом используются искусственный интеллект и распределенные вычисления на границе для

достижения бесшовного соединения, непрерывного покрытия и эффективного распределения ресурсов.

Ядро сети рассматривается с использованием парадигмы программно-конфигурируемых сетей (SDN), а для обеспечения гибкости, доступности, надежности и масштабируемости применяются методы оптимизации роевого интеллекта. В ядре сети используются мультиконтроллерные структуры SDN, коммутаторы OpenFlow, подход программируемости сети, виртуализация и оркестрация, сервис-ориентированная архитектура, нарезка сетевых ресурсов, интеллектуальное управление ресурсами, функциональная совместимость и открытые интерфейсы. Также целью данной работы является интеграция всех рассматриваемых компонентов и общая оценка системы.

Система предусматривает взаимодействие наземной и летающей сети, отвечающей требованиям 6G.

Интегрированные летающие сети являются ключевыми компонентами сетей 6G. Ожидается, что эти сети будут играть важную роль в реализации требований 6G к беспроводной связи. Более того, интеграция таких сетей с наземными сетями является очень востребованной. Однако такая инфраструктура сталкивается со многими проблемами, включая взаимодействие гетерогенных каналов, обработку огромного объема трафика и требования безопасности. Кроме того, еще одной проблемой проектирования являются требования сетей 6G. Достижение сверхвысокой доступности и надежности с субмиллисекундной задержкой для комплексных сетей является большой проблемой.

Интеграцию уровней можно увидеть в структуре сети, представлены также методы интеграции для этих уровней. Интеграция осуществляется с использованием новых технологий, включая граничные вычисления,

искусственный интеллект (AI) и программно-конфигурируемые сети (SDN). Методология этой архитектуры может быть представлена следующим образом.

Общие положения и характеристика устройств 6G и основные ключевые технологии.

Для рассматриваемых уровней сетей сосредоточимся на выделении основных требуемых спецификаций, поддерживаемых внутри межуровневых технологий связи, основных сценариев использования и ключевых показателей эффективности (KPIs).

Архитектура наземной сети обрабатывает все типы трафика. Также численное моделирование и симуляция работы с использованием высокоуровневого программного обеспечения будет постоянно поддерживать прототипирование аппаратного обеспечения и поможет прогнозировать производительность и интерпретировать результаты. Кроме того, разработка модели включает в себя определение точных технических характеристик, например, мощности, пропускной способности и интерфейсов, необходимых устройств и оборудования.

Сюда входят модули радиодоступа, высокопроизводительные граничные вычислительные узлы, контроллеры SDN, коммутаторы OpenFlow и сетевые интерфейсы. Также определены характеристики всех ключевых устройств сети и ее элементов.

Сети радиодоступа

Наземная сеть – это сеть наземной связи, которая должна обеспечить приложениям 6G заявленные требования. Наземный уровень 6G действует как сердце системы. Весь трафик от всех уровней передается в наземную сеть, что

накладывает большие ограничения на проектирование этой части. Общая структура наземного уровня предлагаемой системы состоит из двух основных частей: сети радиодоступа (RAN) и ядра сети. RAN представляет собой средства и решения радиосвязи, развернутые для предоставления услуг наземной связи и поддержки наземных пользователей. Однако ядро сети предназначено для трех уровней. Ядро сети управляет трафиком остальных уровней таким образом, чтобы соответствовать требованиям систем 6G.

Проектирование сетей радиодоступа (RAN) 6G включает в себя несколько ключевых аспектов и технологий, отвечающих изменяющимся требованиям будущих систем беспроводной связи. Хотя стандарты 6G еще не полностью определены, в настоящее время имеется несколько направлений исследований и новых технологий для проектирования сетей RAN 6G. Ниже приведены некоторые ключевые аспекты:

- **Использование спектра**. Ожидается, что сети RAN 6G будут использовать широкий спектр частотных диапазонов, включая более высокочастотные диапазоны, такие как терагерцовые (ТГц) и субтерагерцовые (суб-ТГц) частоты. Эти частотные диапазоны предоставляют обширные ресурсы для передачи данных с высокой скоростью и для массового подключения устройств. Разработка эффективных методов использования спектра, таких как усовершенствованное формирование луча, агрегация несущих и динамическое разделение спектра, будет иметь решающее значение для максимизации пропускной способности и эффективности RAN 6G;
- **Massive MIMO и формирование луча**. Технология Massive Multiple-Input Multiple-Output (MIMO) в сочетании с передовыми методами формирования луча станет неотъемлемой частью RAN 6G. Massive MIMO предполагает использование большого количества антенн как на базовых

станциях, так и на пользовательских устройствах, что обеспечивает пространственное мультиплексирование и улучшенную спектральную эффективность. Формирование луча позволяет направить радиосигнал на нужного пользователя, увеличивая зону покрытия, пропускную способность и качество сигнала. Эти технологии будут усовершенствованы в 6G для поддержки еще большего числа антенн и более сложных алгоритмов формирования луча;

- **Сверхплотные сети**. В сетях RAN 6G будут использоваться сверхплотные сети, в которых базовые станции и точки доступа размещаются плотно и согласованно. Такое уплотнение сети увеличивает пропускную способность, зону покрытия и общую производительность системы. Для устранения помех и оптимизации использования доступных ресурсов будут использоваться такие передовые технологии, как уплотнение сети, развертывание малых сот и динамическое объединение базовых станций;
- **Гетерогенные сети**. RAN 6G будут включать в себя различные технологии беспроводного доступа, включая традиционные сотовые сети, спутниковую связь и новые беспроводные технологии. Интеграция этих гетерогенных сетей обеспечит бесперебойное подключение, непрерывное покрытие и эффективное распределение ресурсов. Проектирование RAN 6G будет сосредоточено на разработке протоколов и алгоритмов для поддержки бесшовных переключений, выбора сети и эффективной маршрутизации трафика между различными передающими устройствами. Это станет ключевым аспектом проектирования, поскольку конечной целью данной работы является полная интеграция наземных, и летающих коммуникационных уровней. Развертывание новых технологий, включая ИИ, распределенные граничные вычисления и SDN, станет средством достижения этих требований;
- **Интеллектуальные и адаптивные сети**. В сетях RAN 6G будут использоваться методы искусственного интеллекта и машинного обучения (ML),

чтобы сделать сети более интеллектуальными и адаптивными. Алгоритмы AI/ML могут быть использованы для проактивной оптимизации сети, прогнозируемого распределения ресурсов, динамического управления спектром, минимизации помех и самовосстановления. Эти интеллектуальные и адаптивные функции позволят сети подстраиваться под изменяющиеся условия, оптимизировать производительность и предоставлять индивидуальные услуги на основе требований пользователей и состояния сети;

- **Энергоэффективность**. Поскольку потребление энергии становится критической проблемой, в сетях RAN 6G особое внимание будет уделяться повышению энергоэффективности. Такие методы проектирования, как использование энергоэффективного оборудования, механизмов управления питанием и интеллектуальных алгоритмов планирования перехода в спящий режим, будут применяться для минимизации энергопотребления при сохранении производительности.

Ядро сети. Ядро сети будет создаваться с использованием парадигмы программно-конфигурируемых сетей (SDN). Для управления огромным количеством трафика в сети будет использоваться SDN с несколькими контроллерами. Чтобы обеспечить необходимую безопасность, гибкость, доступность, надежность и масштабируемость, ядро сети будет оптимизировано с помощью алгоритмов роевого интеллекта. Проблема размещения и распределения контроллеров будет решена с помощью предлагаемого алгоритма, который будет реализован на контроллерах SDN.

Ядро сети разработано с использованием мультиконтроллерной структуры SDN.

Ядро сети SDN будет иметь следующие характеристики:

- **Программируемость сети**. Ожидается, что ядро сети SDN для сетей

6G обеспечит высокий уровень программируемости сети. Это позволит операторам сети динамически конфигурировать и управлять сетевыми ресурсами, услугами и протоколами через интерфейсы прикладного программирования SDN (API). Такая программируемость позволит гибко и эффективно распределять сетевые ресурсы в соответствии с изменяющимися требованиями и характером трафика. Это позволит реализовать сетевой трафик гетерогенного характера от различных приложений и коммуникационных шин;

- **Виртуализация и оркестрация**. Сеть SDN для систем 6G будет использовать методы виртуализации сетевых функций (NFV) и оркестрации. Для реализации множества виртуализированных сетевых функций в ядро сети будет внедрена платформа NFV MANO. Виртуализация позволяет отделять сетевые функции от аппаратного обеспечения и запускать их в виртуальной среде. Оркестрация относится к автоматизированной координации и управлению виртуализированными сетевыми функциями для развертывания, масштабирования и управления сетевыми услугами. Виртуализация и оркестрация обеспечивают эффективное использование ресурсов, масштабируемость и гибкость ядра сети;
- **Сервис-ориентированная архитектура**. в системах 6G будет использоваться сервис-ориентированная архитектура (SBA) для ядра сети SDN. SBA позволяет разделить сетевые функции на небольшие компоненты (сетевые сервисы), которые можно переиспользовать. Эти сервисы могут динамически комбинироваться для создания пользовательских сетевых фрагментов, адаптированных к конкретным приложениям или требованиям к услугам. SBA способствует гибкости, масштабируемости и эффективному предоставлению услуг в ядре сети;
- **Сегментация сети**: Сегментация сети будет ключевой особенностью ядра сети SDN для сетей 6G. Она позволит создавать несколько

логических сетей (срезов) на общей физической инфраструктуре, каждая из которых будет оптимизирована для конкретных случаев использования, приложений или требований к услугам. Сетевые срезы обеспечивают индивидуальное распределение ресурсов, гарантированное качество обслуживания (QoS) и изоляцию между различными сетевыми сервисами, что позволяет ядру сети эффективно поддерживать различные сценарии использования 6G с различными требованиями к производительности сети. Это станет ключевым аспектом интеграции трех уровней сетей;

- **Интеллектуальное управление ресурсами**. Ядро сети SDN будет включать в себя механизмы интеллектуального управления ресурсами. Эти механизмы будут использовать алгоритмы AI и ML для оптимизации распределения ресурсов, маршрутизации трафика и балансировки нагрузки на основе состояния сети в реальном времени, требований пользователей и требований к производительности. Интеллектуальное управление ресурсами повысит эффективность сети, масштабируемость и QoS в сетях 6G;
- **Функциональная совместимость и открытые интерфейсы**. Ядро сети SDN будет использовать открытые интерфейсы и стандартизированные протоколы для обеспечения совместимости между сетевыми компонентами, доменами и поставщиками. Это будет способствовать беспрепятственному взаимодействию различных сетевых элементов, поддержке среды с множеством операторов и продвижению инноваций, позволяя сторонним приложениям и услугам использовать возможности ядра сети.

Диссертация посвящена объединению всех компонентов и общей оценке системы.

1.2 Услуги сетей пятого и шестого поколения 5G/6G

Задержка имеет решающее значение в современных приложениях, таких как здравоохранение, беспилотное вождение, умные дома и умные отрасли. Такие приложения требуют сверхвысокой надежности, высокой доступности и сверхнизкого времени отклика [1]. Требования таких приложений отличаются друг от друга; однако все они имеют общие ограничения с точки зрения задержки, надежности и доступности [2]. Требования к задержке и надежности накладывают ограничения на разработку коммуникационной сети, обеспечивающей такие приложения. Эти приложения заявлены в определенной категории услуг 5G/6G, которой являются услуги uRLLC [3].

uRLLC — это приложения 5G/6G, требующие экстремальных задержек и надежности. Приложения uRLLC имеют разные требования к задержке, которые можно разделить на три основные группы. В первую группу входят приложения uRLLC, которым требуется сквозная задержка 5 мс или выше. В эту группу входят такие приложения, как AR, VR и MR. Вторая группа содержит приложения uRLLC, которым требуется сквозная задержка в одну миллисекунду, например Тактильный Интернет. Третья группа содержит субмиллисекундные приложения, такие как голографическая связь.

Мы можем перечислить часть таких uRLLC приложений следующим образом.

1) Умные фабрики. Использование устройств и точность контролируются в режиме реального времени для быстрого производства и облегчения процесса переработки на фабриках. Наличие большого количества производственных линий представляет собой серьезную проблему с точки зрения задержки и надежности. Поэтому для большинства сервисов требуется очень низкая задержка до 5 миллисекунд. Такое применение вызывает большой интерес с анонсом парадигмы индустрии 4.0 [4, 5];

2) Интеллектуальные транспортные системы. Автономное вождение и облегчение дорожного движения требуют масштабируемой инфраструктуры, высоконадежной связи с очень малой задержкой и специальных станций для обеспечения безопасности дорожного движения. Максимально допустимая задержка для большинства автомобильных приложений составляет 5 миллисекунд [6, 7];

3) Роботы и дистанционное управление. В качестве примера роботов и дистанционного управления можно привести дистанционную хирургическую операцию в местах, пострадавших от природных и техногенных катастроф [8];

4) Виртуальная реальность (VR). Многие приложения, требующие очень высокой чувствительности и точности обработки данных, такие как удаленная хирургия, нуждаются в технологии VR. Поддержка этих сервисов требует очень малого времени отклика [9];

5) Дополненная реальность (AR). Технология AR используется в нескольких приложениях, включая дистанционное обучение, медицинские услуги, игры, умные города и обучение пожарных борьбе с пожарами без человеческих потерь [10]. Для таких приложений требуется сквозная задержка не более 5 миллисекунд [11]. Это необходимо для достижения требуемого качества восприятия (QoE);

6) Здравоохранение. Сюда входят удаленная операция, удаленная диагностика и выполнение опасных операций с использованием роботизированных систем с участием человека. Такие приложения основаны на связи в реальном времени, которая должна обеспечиваться с очень низкой задержкой [12]. Это должно передать человеческие ощущения тактильным роботам с максимальной чувствительностью. Такие приложения требуют непрерывной задержки в пределах 1-5 миллисекунд [13];

7) Интеллектуальные сети. К интеллектуальным сетям предъявляются важные требования с точки зрения надежности и задержки, поэтому требуется очень низкая задержка, чтобы соответствовать этим требованиям и приложениям, в которых используются интеллектуальные сети [14];

8) Тактильный Интернет. Он представляет собой четвертую волну традиционного Интернета, которая позволяет передавать человеческие ощущения и действия в режиме реального времени. Основное приложение, которое будет поддерживать Тактильный Интернет, — это тактильные коммуникации, для которых требуется непрерывная задержка не более одной миллисекунды [15]. Такая проблема с задержкой связана с физическими параметрами, определяемыми человеческими ощущениями, как представлено в [16];

Цифровой двойник

Цифровой двойник (Digital Twin, DT) – это виртуальное представление реального объекта, процесса или системы, созданное с использованием цифровых технологий [17]. DT может быть представлен в виде компьютерной модели, которая в точности имитирует физические характеристики и поведение оригинала. Данные, собранные с помощью цифрового двойника, могут быть использованы для оптимизации процессов и предсказания будущих состояний объектов или систем, а также для выявления их возможных улучшений. Технология DT становится все более популярной в различных отраслях промышленности, и ожидается, что в ближайшие годы она будет продолжать развиваться. Цифровые двойники также могут использоваться для оптимизации процесса миграции микросервисов [18]. Моделируя процесс миграции, можно выявить и решить любые потенциальные проблемы до того, как все будет приведено в исполнение.

Технология DT также позволит повысить и безопасность сети 6G [19]. Благодаря созданию виртуальной копии физического объекта или процесса в сетях 6G станет возможным обнаруживать любой несанкционированный доступ или изменения по отношению к оригиналу двойника. Это позволит сетям связи шестого поколения предпринять соответствующие действия для защиты реального объекта (процесса) от любых злонамеренных действий.

Архитектура цифрового двойника в 6G представляет собой сложную систему, объединяющую физический и цифровой миры [20]. Это сочетание аппаратных средств, программного обеспечения и данных, которое позволяет создать цифровое представление реального объекта. В основе архитектуры лежит сам цифровой двойник. Компьютерная 3D-модель создается на основе данных, полученных от датчиков, камер и других источников, и впоследствии может быть использована для мониторинга, анализа и управления имитируемого объекта. После создания модели, DT подключается к облачной платформе, где реализуется хранение и анализ данных цифрового двойника, и далее на основе анализа создаются прогностические модели в целях предвидения будущих явлений и принятия опережающих мер.

Однако существует несколько трудностей и проблем, связанных с технологией цифровых двойников, которые необходимо решить. Одним из самых больших препятствий к внедрению технологии цифровых двойников является связанная с ней высокая стоимость [21]. Для реализации и обслуживания этой технологии требуется значительное количество различных ресурсов (аппаратного и программного обеспечения, человеческих, временных). Еще одной проблемой является сложность данных, которые необходимо собирать и анализировать [22]. Технология DT все еще относительно новая, и до сих пор существуют некоторые проблемы с ее точностью и надежностью. Для того, чтобы цифровой двойник был полезен и применим, собранные данные

должны быть точными, а сам двойник необходимо поддерживать в актуальном состоянии, что может оказаться непростой задачей.

Голограммы

Голограммы – это трехмерные изображения, созданные путем проецирования света через ряд линз и зеркал [23]. В прошлом эта технология использовалась для создания реалистичных изображений предметов, но сейчас она используется для создания реалистичных изображений людей. Голограммы в рамках миграции микросервисов могут быть использованы для создания более глубокого погружения клиентов в процесс обслуживания. Например, клиент может взаимодействовать с представителем службы поддержки в виртуальной среде, задавая вопросы и получая ответы в режиме реального времени [24]. Это может быть особенно полезно для клиентов, которые находятся в отдаленных районах.

Кроме того, новой революционной технологией явилась Holo-Gauze (голографическая ткань), позволяющая проецировать крупномасштабные 3D-голограммы [25]. Это легкая прозрачная ткань, которая используется для создания трехмерного голографического изображения. Ткань состоит из крошечных отражающих частиц, расположенных по определенной схеме. Когда свет проходит через ткань, она создает трехмерное изображение, которое кажется парящим в воздухе. Holo-Gauze также невероятно универсальна. Ее можно использовать самыми разными способами, от проекционного отображения до создания интерактивных впечатлений. Она также может быть использована для создания виртуальной реальности, позволяя зрителям исследовать трехмерную среду [26]. Кроме того, Holo-Gauze намного дешевле традиционных систем 3D-проекции, а также проста в настройке и использовании, что делает эту технологию отличным вариантом для лиц и

компаний с ограниченным бюджетом и для тех, кто не обладает большими техническими знаниями.

Как и у любой новой технологии, у голограмм есть ряд проблем, которые необходимо преодолеть, прежде чем они получат повсеместное внедрение. Одной из самых больших проблем является стоимость, поскольку для голограмм требуется специализированное оборудование и материалы, которые могут быть дорогостоящими [27]. Кроме того, так как технология все еще относительно новая, отсутствует стандартизация и совместимость между различными системами. Еще одной проблемой является то, что голограммы требуют больших вычислительных мощностей для генерации и отображения изображений, а программное обеспечение, используемое для их создания, часто является сложным и трудным в использовании [28]. До сих пор существуют и некоторые технические ограничения голограмм. Например, разрешение изображения все еще относительно низкое по сравнению с другими видами медиа, а также проекции подвержены искажениям, если зритель слишком далеко отходит от дисплея. С голограммами сложно взаимодействовать, так как они требуют от пользователя определенного положения для правильного просмотра.

Робот-аватар

Роботы-аватары – это сгенерированные компьютером персонажи, которые могут быть использованы для представления человека в виртуальном мире [29]. Они могут использоваться для различных целей, от обеспечения виртуального присутствия в игре или виртуальном мире до более реалистичного представления человека в виртуальной среде. Эта технология также может использоваться для управления роботами и транспортными средствами, поскольку аватары запрограммированы на различные навыки и способности: в них может быть заложен функционал на выполнение таких задач, как навигация по виртуальному миру или даже оказание помощи другим аватарам [30]. Это

позволяет обеспечить более реалистичное взаимодействие между людьми в виртуальном мире.

Проблемы, связанные с использованием роботов-аватаров в 6G, многочисленны и разнообразны. Во-первых, существует проблема создания реалистичного аватара, который может взаимодействовать с людьми естественным образом [31], точно имитируя поведение и эмоции человека. Это требует проведения большого количества исследований и разработок. Во-вторых, существует проблема интеграции аватара в существующие сервисы [32]. Необходимо большое количество технических знаний, чтобы обеспечить беспрепятственное взаимодействие аватара с существующими сервисами. В-третьих, существует проблема обеспечения безопасности и надежности аватара [33]. Чтобы избежать утечку данных и подверженность атакам, необходимо принятие большого количества мер безопасности.

Сверхплотные сети (IoT). Интернет вещей (IoT) – это быстро растущая сеть подключенных к сети различных устройств, способных взаимодействовать в первую очередь друг с другом, а также с другими сетями. Поскольку количество подключенных устройств постоянно возрастает, растет и потребность в обеспечении возможности их подключения становятся необходимыми сверхплотные сети IoT. Сверхплотные сети IoT предназначены для обеспечения высокого уровня связности и пропускной способности ячеистой сетевой архитектуры, обеспечивающей эффективную передачу данных между несколькими устройствами [34]. Этот тип сети разработан специально для обеспечения безопасности и энергоэффективности сети IoT и идеально подходит для приложений, требующих большого количества подключенных устройств в ограниченном пространстве, таких как «умный город», автоматизация производства и здравоохранения.

Потенциальные преимущества сверхплотных сетей IoT в 6G огромны, включая широкий спектр новых приложений и услуг, а также более высокую скорость передачи данных, более безопасную передачу данных и более эффективное использование спектра [35]. Кроме того, приложения и услуги таких сетей смогут обслуживать миллионы устройств, подключенных на ограниченной территории, что позволит более эффективно собирать и анализировать данные [36].

Кроме того, устройства IoT становятся все более популярными и используются для предоставления различных услуг. Эти устройства подключены к Интернету и могут использоваться для мониторинга и управления аспектами различных услуг, предоставляемых пользователям [37]. Например, система «умный дом» может использоваться для мониторинга и управления температурой, освещением и безопасностью дома. К тому же, интегрируя устройства IoT с технологией блокчейн, поставщики смогут создать безопасную, распределенную и автоматизированную систему предоставления услуг [38]. Симбиоз IoT и блокчейна может обеспечить ряд преимуществ и в области миграции микросервисов [39].

Однако совместная работа IoT и технологии блокчейн сопряжена с рядом проблем. Наиболее значительной проблемой является масштабируемость [40]. IoT-устройства генерируют большое количество данных, и блокчейн-сети должны быть в состоянии справиться с этой нагрузкой. Другим вопросом является безопасность, поскольку устройства IoT уязвимы для атак, а сети блокчейн должны быть способны защитить данные от злоумышленников [41]. Для этого необходимо, чтобы сеть блокчейн была безопасной и устойчивой к атакам. Существует также проблема совместимости [42]: устройства IoT должны иметь возможность взаимодействовать друг с другом и с сетью блокчейн одновременно. Для этого требуется такая архитектура сети блокчейн, которая

способна поддерживать множество протоколов и стандартов, используемых в сетях IoT.

1.3 Требования, предъявляемые к современным сетям связи

Так как услуги URLLC представляют собой взаимодействие машин и людей в реальном времени и с высокой надежностью обмена информацией, а также с гарантией доставки пакетов, они предъявляют жесткие требования к среде передачи, которые в полной мере необходимо соблюдать. В рекомендации ITU-R M.2083-0 ключевыми принципами проектирования сетей связи пятого поколения обозначены гибкость и разнообразие для обслуживания множества различных сценариев использования [43]. При этом описываемые требования имеют разную значимость для разных сценариев использования, упомянутых в разделе 1.1 текущей главы. Общий список требований к сетям 5G приведен в таблице 1.

Таблица 1.3.1 – Требования к сетям связи пятого поколения

Возможность	**Требования**	**Сценарий**
Пиковая скорость передачи данных по линии вниз	20 Гбит/с	eMBB
Пиковая скорость передачи данных по линии вверх	10 Гбит/с	eMBB
Пользовательская скорость передачи данных по линии вниз	100 Мбит/с	eMBB
Пользовательская скорость передачи данных по линии вверх	50 Мбит/с	eMBB
Задержка	4 мс	eMBB
	1 мс	URLLC

Мобильность	500 км/ч	eMBB, URLLC
Плотность подключений	10^6/км^2	mMTC
Пропускная способность	10 Мбит/(с·м^2)	eMBB

В таблице 1.3.1. к требованиям, относящимся к тем, что выдвигают сети URLLC, относятся пункты по задержке и мобильности. Круговая задержка должна быть неощутимой для человека и составлять максимум 1 мс, так как приложения, которые входят в группу URLLC, очень чувствительны ко времени [43, 44]. Телемедицина предполагает, что любое движение врача на другом конце канала связи должно передаваться по сети и воспроизводиться на стороне получателя мгновенно, что позволяло бы проводить удаленные хирургические операции, например в ситуациях, если пациент и врач находятся в разных странах, и невозможно организовать личную встречу. При этом поддержка мобильности должна осуществляться даже при максимальной скорости передвижения объектов в 500 км/ч. Более того, сети связи с ультрамалыми задержками требуют надежности передачи данных выше, чем 99,9999% [44].

Реализация этих требований позволяет обеспечить высокую надежность и точность передачи данных в режиме реального времени, что необходимо для решения задач в области автономной техники, телемедицины, индустриальной автоматизации и т. д. Для построения подходящего уровня сетевой инфраструктуры необходимо использовать передовые технологии, такие как, многолучевую многопозиционную передачу (multi-user MIMO), более широкие спектры частот и высокочастотные диапазоны, а также интеллектуальную оптимизацию работы сетей и устройств. Кроме того, необходимо также внедрять новые технологии аналитики и обработки данных и Искусственный Интеллект

для обработки больших объемов информации и принятия решений на основе полученных данных [43, 45, 46]. Архитектура должна быть гибкой и масштабируемой, состоять из слоев, взаимодействующих друг с другом по отдельным сетевым интерфейсам [45]. Должен осуществляться контроль и эффективное распределение ресурсов сети. В целом, реализация услуг URLLC требует комплексного подхода к развитию сетевой инфраструктуры и применению передовых технологий.

Таблица 1.3.2 – Сравнение 5G и 6G

Параметры	**5G**	**6G**
Начало эксплуатации	2020	2030
Скорость передачи информации	20 Гб/с	1 Тб/с
Частотные диапазоны	3 - 300 ГГц	1 - 10 ТГц
Точность/погрешность	m - уровень	cm - уровень
Покрытие	2 D	3D
Плотность подключений (/км^2)	10^3 устройств	10^6 устройств
Задержка (мс)	4 - 5	0.1
Джиттер	1 мс	1 μс
Надежность (%)	99.999	99.9999999
Безопасность	Низкая	Высокая с использованием технологий Блокчейн

Срок службы аккумуляторных батарей	Ограничен размером батареи	Неограниченный срок службы батареи (используются устройства с нулевым энергопотреблением и микросхемы с функцией получения энергии извне)
Спектральная эффективность (б/с/Гц)	30	100
Поддержка мобильности (скорость транспортного средства)	Более 500 км/ч	Более 1000 км/ч
Спутниковая связь	Нет	Да

Мобильность: максимально допустимая скорость будет увеличена с 500 км/ч до 1000 км/ч.

Надежность: будет достигнута надежность 99,99% для поддержки беспилотных транспортных средств, включая БПЛА и коллаборативную робототехнику.

Сетевая задержка: показатели сквозной задержки уменьшатся в 10 раз.

Пропускная способность: максимальная пропускная способность 1 Тбит/с будет необходима для 6G, что в 1000 раз быстрее, чем 5G. Ожидается 100-кратный рост пропускной способности.

Энергоэффективность и эффективность использования спектра: 100-кратное повышение энергоэффективности.

Надежность – это вероятность того, что система будет безотказно работать в течение определенного периода времени. Поскольку сети 6G будут сильно распределенными, основной задачей в вопросах обеспечения надежности будет являться эффективная координация работы вычислительных узлов. Для достижения этой цели необходимы соответствующие протоколы передачи данных между узлами сети, а также надежная опорная сеть, способная обеспечить передачу больших объемов трафика, генерируемого в процессе хранения и получения данных. Под доступностью понимается вероятность того, что система будет работать исправно в любой момент времени. Распределенные решения ИИ для сетей 6G являются перспективным вариантом для сокращения времени обучения при одновременном снижении потребления ресурсов, что позволит повысить доступность систем и сервисов основанных на ИИ. Форм-фактор является важной переменной, поскольку он ограничивает ресурсы, включая энергию, доступные для связи с внешними распределенными системами.

Безопасность – это способность системы к самостоятельному обеспечению своей защищённости путём своевременного выявления угроз и принятия необходимых мер эффективной защиты сервисов, развернутых в системе, а также данных, которыми обмениваются устройства и пользователи. В случае сетевых сервисов 6G, распределенные алгоритмы AI/ML потребуются для локального обучения моделей выявлению и устранению угроз, что позволит сохранить конфиденциальность данных конечных пользователей. Кроме того, под безопасностью понимается и способность системы не причинять вреда человеческой жизни, окружающей среде или частной собственности. Поскольку в сетях 6G будут распространены сценарии использования, в которых под

угрозой может оказаться человеческая жизнь, например, при использовании автомобилей с автономным управлением, то анализ роли AI/ML приобретает особенную важность.

Доступность обеспечивается только после достижения надежности. Надежность – это вероятность того, что система в целом работоспособна, а доступность – вероятность того, что она будет работоспособна в определенный момент времени. Доступность гарантирует, что не будет отказано в авторизованном доступе к системе. Преимущество распределенных систем заключается в том, что наличие множества узлов и каналов связи способствует предотвращению возможных сбоев. Современные тенденции исследований в области граничных вычислений направлены на повышение доступности системы путем тщательного планирования процессов передачи задач и данных от конечных устройств к граничным серверам, с использованием механизмов, способных принимать решения об этой передаче на основе статистики сети и данных о вычислительных возможностях граничных серверов. Также доступность может быть повышена за счет переназначения задач с отказавших узлов на работоспособные, хотя и обычные отказы все еще являются проблемой, ведь задача, вызвавшая сбой на одном узле, может вызвать тот же сбой и на другом узле. Поскольку доступность и надежность взаимосвязаны, важно отметить, что обе характеристики должны быть взвешены относительно друг друга, так как для различных систем может потребоваться и их различная величина.

1.4 Современные технологии для поддержки услуг в сетях 5G/6G

1.4.1 Программно-конфигурируемые сети.

В настоящее время все типы современных приложений требуют определенного типа сети. так как их приложения требуют сети, которая характеризуется высокой скоростью, чтобы приложение работало эффективно,

как требуется, и достигало своей цели, для которой оно создано. Кроме того, эти типы быстрых сетей обеспечивают передачу данных в огромных объемах, что помогает обслуживать большое количество пользователей без проблем с трафиком, предлагая все виды услуг и приложений очень эффективно и достигая своей цели. Этот тип быстрых сетей связан с двумя основными концепциями - "взаимосвязанный центр обработки данных" и "виртуализация серверов". Благодаря этим двум технологиям запрос на данный тип сети очень быстро увеличивается в очень широком диапазоне. Помимо всего этого, сеть ассоциируется с несколькими сетевыми продуктами в качестве аппаратного компонента, большим количеством распространяемых протоколов, необходимых для реализации различных приложений в различных областях с несколькими функциями и программными компонентами, которые необходимы в большинстве приложений. Этот тип сети содержит огромное количество устройств в виде коммутаторов, которые отвечают за направление пакетов по отдельности по наилучшему пути, основанному на их назначении и протоколе, которому они следуют. Весь процесс принятия решений, который происходит для пути пакетов как в коммутаторах, так и в маршрутизаторах, всегда собирается на одном устройстве. В то время как все виды сетевого интеллекта и способность принимать решения применяются при распределении на нескольких компонентах сетевого оборудования. Для начала инициализации любого нового сетевого устройства или сервиса, который затем требует огромного количества реконфигураций всех узлов в сети в широком диапазоне, что приводит к тому, что это описывается рутинной работой и требует большого количества предложений для реализации. Здесь возникает проблема способности сети работать самостоятельно выполнять свою основную функцию для каждой прикладной цели автоматически.

В настоящее время IP-адреса являются основными компонентами, на которых базируются все сети, чтобы иметь возможность определять и выделять любые приложения и серверы. И это очень удобно в условиях статической сети, так как каждое устройство определяется уникальным IP-адресом, в то время как, с другой стороны, это будет очень трудно применить в сетях, которые работают в широком виртуальном диапазоне. Способ справиться с такой сложной средой с помощью классических сетей помогает сэкономить время и силы. Это необходимо для условий виртуальной машины и конфигурации сети. Это помогает упростить управление широкими сетями виртуальным способом. Итак, от менеджеров требуется решить проблему физической инфраструктуры, которая является причиной повышения сложности процесса управления. Для того чтобы администраторам было проще управлять виртуализированной сетью, необходимо решить проблему физической инфраструктуры, так как она является основной причиной трудностей, с которыми сталкивается администратор в процессе управления. Так как это основная причина сложности, решение которой поможет легко контролировать сеть.

В настоящее время поставщики используют новые методы управления, такие как программное обеспечение плоскости управления, чтобы помочь данным перемещаться и следовать по наилучшему достигнутому пути без перегрузок и осложнений. Все это помогает руководству гарантировать, что сеть будет работать наилучшим образом и сможет использоваться во многих областях и целях, другими словами, чтобы получить максимальное преимущество от сети. По этому сценарию плоскость управления на основе коммутатора предлагает администратору сети ограниченную возможность повысить эффективность потока данных во всей сети. При этом сеть структурируется как сплошная сеть, что, к сожалению, затрудняет изменение программируемости сети для достижения различных потребностей или требований клиента. Кроме того, это

иногда заставляет поставщиков создавать сеть с программируемой системой управления, которую легко сломать и гибко изменить. Именно поэтому большое количество сотрудников нанимается для того, чтобы помогать в качестве администратора сети и вносить изменения в компоненты сети в ручном режиме. Стремление к этому быстро растет, поскольку множество приложений, необходимых для ряда продуктов и услуг, предлагаемых сетями, должны быть в этом формате. Помимо всего этого, трафик видео, огромные объемы данных и рост использования мобильных устройств в настоящее время ставят перед операторами сетей множество серьезных проблем, поскольку все эти требования к приложениям стремительно растут. Операторы мобильной и телекоммуникационной связи сталкиваются с перегруженностью спектра, смещением интернет-протокола и быстрым ростом числа мобильных пользователей. Операторы центров обработки данных также сталкиваются с огромным ростом числа серверов и виртуальных машин, которые быстро увеличиваются в широком диапазоне, чтобы удовлетворить потребности клиентов в приложениях или различных услугах. Поэтому операторы предпочитают решать все эти проблемы с помощью сети, которая характеризуется высокой эффективностью, гибкостью, масштабируемостью и еще большей гибкостью.

Все эти проблемы решаются с помощью двух основных технологий - программно-определяемой сети и виртуализированной сети. Поскольку эти две технологии являются основным решением, которое способно преодолеть все эти проблемы и применить необходимые требования. Поскольку основная операционная точка SDN основана на плоскости управления, которая является групповой и централизованной таким образом, что помогает и предлагает решения для управления сетью и контроля, с которыми сталкиваются операторы. SDN работает на основе концепции отделения плоскости данных от плоскости

управления. Кроме того, она предлагает плоскость управления в программируемой форме, чтобы быть гибко контролируемой и обновляемой на основе любого приложения или услуги, требуемой от пользователя в данный момент времени. SDN имеет множество преимуществ, так как помогает упростить сеть, а также противостоять проблемам, ограничивающим функциональность и производительность сети, связанным с облачными вычислениями, организациями информационных технологий и сетевыми предприятиями. Таким образом, все это и многое другое - преимущества, которые предлагает SDN, чтобы быть в состоянии достичь требований, необходимых клиенту в настоящее время, и столкнуться с любыми проблемами в изменении сети очень гибким способом, не нанося вреда сетевой инфраструктуре и не сталкиваясь с проблемами управления или контроля.

Преимущества SDN:

1- Программируемость сети:

Программируемость сети в SDN помогает решить проблему традиционной сети, связанную с негибкостью и сложностью сети, которая стоит перед сетевым менеджером и заставляет оператора сети сталкиваться с множеством проблем. Оператор сети может управлять программируемостью сети и стандартизировать сеть таким образом, чтобы это не влияло на надежность работы сети или даже на пользовательский опыт работы с сетью. Есть два основных параметра, которые формируют и предлагают SDN огромные, огромные значения - это плоскость управления и структура плоскости данных. Избавляясь от всех сложностей, которые всегда возникают на уровне инфраструктуры, и предлагая большое видение как сервисов, так и приложений, SDN упрощает управление сетью и помогает гибко осуществлять виртуализацию сети.

При этом предлагается краткая информация об управлении потоком данных от начала пути в отдельных устройствах до конца пути на уровне сети.

Возможность контролировать поток данных в широком диапазоне сети помогает предложить администратору возможность реализовать сетевые потоки для достижения требуемой связности и иметь возможность определить специфические требования для каждого пользователя в отдельности. Благодаря всем этим достижениям SDN, сетевому менеджеру не нужно выполнять традиционный способ применения на каждом устройстве сетевых политик и протоколов по отдельности. В общем случае SDN отделяет функцию на плоскости управления от физических устройств, поскольку она работает с помощью контроллера, но эти контроллеры работают извне. При этом обеспечивается на плоскости управления возможность для программирования сети, и это для того, чтобы предложить возможность применять любые изменения, а затем направлять их на устройство или на сетевое оборудование и следить за тем, чтобы данные перемещались и достигали требуемого места по безопасному каналу. Это помогает достичь цели SDN, чтобы сгладить процесс объединения в сеть существующих устройств с новой архитектурой. Помимо всего этого, контроллер SDN также помогает оператору сети расширить возможности трафика за счет использования видеотрафика. Это позволяет операторам сети управлять перегрузкой узлов и ограничивает любые сложности, с которыми сталкивается пользователь из-за трафика в сети.

2- Виртуализация.

SDN предоставляет сети огромные возможности по управлению центром обработки данных в широком диапазоне. В настоящее время центр обработки данных играет важную роль в проблемах масштабируемости, частично обусловленных ростом виртуальных машин. Как и в традиционной сети, процесс приведения таблицы адресов Media Access Control и изменение места виртуальной машины может повлиять на пользователей в их практическом

использовании приложения. Это приведет к тому, что пользователь получит плохой опыт использования приложения. Виртуализация сети рассматривается как приложение SDN, что дает большие возможности для использования огромных центров обработки данных. SDN может позволить нескольким поставщикам услуг соединить свои физические и виртуальные серверы, локальные и удаленные, частные и публичные облака в единую логическую сеть. В результате каждый клиент сможет получить свой собственный изолированный вид поставщика услуг. Это очень помогает клиентам получить несколько точек доступа на сеть центра обработки данных в зависимости от их требований и потребностей. SDN — это перспективный путь, который дает сети большое преимущество, помогая строить гибкие модели, которые предоставляют модели услуг рядом с операторами виртуальных сетей и предлагают предприятиям возможность управления центрами обработки данных и трафиком.

3- Конфигурация устройства и устранение неполадок.

Благодаря SDN устройство способно конфигурировать и устранять неполадки с помощью единой точки сети, что делает более очевидным признание важной цели для сети, которая работает динамически, способна конфигурироваться и быть удобной с окружающими требованиями. Кроме того, SDN помогает обеспечить возможность применения новых идей, связанных с областью сети, предоставляя программируемую платформу для их тестирования на всех протоколах и политиках, связанных с проблемами трафика. Поток данных, отделенный от тестового потока, облегчает внедрение в сеть новых протоколов и идей. Помимо всего этого SDN обеспечивает форму сети, которая помогает изолировать пакеты управления маршрутизацией от аппаратной коммутации. Объединение SDN и Ethernet позволяет успешно реализовать реальный сетевой интеллект.

Проблемы в SDN.

SDN обязуется содействовать конфигурации и исполнению, а также средствам любого рода обновления, но SDN все еще является новой разработанной системой, которая только что была создана. Многие основные вопросы все еще не ясны и полностью не решены, между любой стандартизацией и совместимостью, которые являются более важными. Хотя Open Networking Foundation (ONF) рассматривается как определение для SDN. Но открытый сетевой поток рассматривается без каких-либо инструментов, в то время как SDN – единственный стандарт, который не имеет инструментов, но рассматривается как готовое решение. В сравнении с открытым потоком нет драйвера с открытым исходным кодом, в то время как контроллер в SDN разработан, только язык программирования на высоком уровне отсутствует. Разработчики приложений поставщиков сетевых устройств, потребители сетевых устройств все еще отсутствуют в любом разработанном SDN приложении. SDN предоставляет широкую область для новых технологий, связанных с сетью. Однако трансформация традиционной сети в SDN может быть очень сложной. Некоторые общие проблемы, связанные с SDN, как способность компьютерных систем к наследию конфиденциальности сетевых устройств и производительность, связанная с централизованным управлением, кроме того, это также имеется недостаток опыта в технической поддержке. В настоящее время проводится много исследований в области SDN, но использование SDN все еще ограничено расширением тестов и исследований в большом количестве случаев и прототипов, чтобы быть в состоянии обеспечить зависимость от возможности применения в реальном мире.

Использование SDN для различных сетей и приложений.

1- Потоковая передача данных:

Сеть VANET, серверы, такие как онлайн-игры, мультимедийные приложения, потоковая передача данных и видеоконференции, наиболее важной

идеей которых является потоковая передача данных. Сети и приложения, связанные с потоковой передачей данных, получили более широкое применение и разделяют большое количество трафика данных через сеть. Например, для VANET основной вопрос заключается в том, как разделить трафик потоковой передачи данных между узлами автомобильной сети от сервера потоковой передачи. Поскольку эта сеть характеризуется высокой мобильностью и большим количеством изменений топологии, также требуется очень низкая задержка. Есть два основных преимущества, которых не может достичь традиционная система VANET по сравнению с FSDN VANET. Контроллер SDN является первым, поскольку он содержит все знания о маршрутизаторах транспортных средств, RSUCs и ресурсах, управляемых с помощью RSUs и оркестровки тумана. Все это позволяет определить наилучшую конфигурацию, которая может быть развернута на сервисе. Поставщик услуг является вторым, который выделяет необходимый автомобиль и предполагает, что этот необходимый автомобиль недоступен для RSU. FSDN VANET способна динамически адаптироваться к любым изменениям, связанным с топологией, и может быть реконфигурирована для того, чтобы качество восприятия достигло рациональной реконфигурации для себя.

2- Услуга Lane-Change:

Служба смены полосы движения использует множество параметров, которые необходимо разработать в случае микроскопических данных о трафике, таких как состояние дороги и переменные транспортного потока. Это поддерживает множество приложений, таких как "система предупреждения Hella” и система обнаружения слепых зон от Mobileye. Эти приложения работают в основном на предупреждение водителя о слепых зонах и игнорируют переменные в дорожном движении. Это результат недостатка данных, собираемых конечными пользователями транспортных средств для принятия

правильных решений. Основная цель состоит в том, чтобы автомобиль мог принять решение о смене полосы движения с помощью высокого уровня управления наиболее согласованным и оптимизированным способом, а не принимать решение в зависимости только от отдельного автомобиля. Контроллер SDN является всеобщим для сети и несколько видов данных только дорожной системы существенно более функционально для проектирования динамического изменения системы полосы движения в отличие от традиционного подхода. Планируемая структура содержит BS, RSU, RSUC, которые рассматриваются как противотуманные устройства, предоставляющие экстренные услуги, которые необходимы. Для применения услуги система смены полосы движения должна быть применена как на граничных устройствах, так и на плоскостях приложений. Такое решение требует системной проверки данных в дискретное время, чтобы убедиться, что происходят какие-либо обновления, так как это очень помогает получить важные данные от приложений других пользователей. Эти данные сохраняются на базовой станции или RSUCs в течение короткого промежутка времени, в то время как в центре обработки данных они сохраняются в течение длительного времени. Помимо всего этого, в центре обработки данных, на базовой станции или RSUC можно получить такие необходимые переменные параметры, как скорость, время, пространственные интервалы, карта движения и многое другое.

Все эти приложения доступны и могут использовать преимущества SDN, чтобы иметь возможность управлять важными требованиями и решать множество проблем, с которыми сталкиваются некоторые приложения сейчас, что сделало их насущной необходимостью в нашей жизни в настоящее время.

1.4.2 **Виртуализация сетевых функций (NFV)**

Виртуализация сети абстрагирует аппаратные ресурсы и встроенное программное обеспечение сети в единую логическую сущность (виртуальную

сеть). Существует два типа виртуализации: внутренняя и внешняя виртуализация. Внутренняя виртуализация имеет один программный контейнер на сервере для представления сети в виде функциональности. Внешняя виртуализация представляет несколько физических ресурсов как единую виртуальную сущность. NFV — это одна из самых больших разработок в процессе эволюции сети, которая использует платформу виртуализации и аппаратное обеспечение коммерческого производства (COTs) для достижения функциональности сетей связи. Она не зависит от аппаратных устройств и использует программные модули, такие как брандмауэры и шлюзы (маршрутизаторы, коммутаторы). Основными преимуществами NFV являются снижение CAPEX и OPEX за счет уменьшения затрат на оборудование и снижение энергопотребления, большая гибкость для развития, увеличение или уменьшение масштаба услуг и развертывание новых инновационных услуг при минимальных рисках. NFV — это программное обеспечение на некоторых аппаратных средствах, что упрощает масштабирование количества виртуальных машин.

Преимущества технологии NFV

- NFV снизила стоимость сетевой инфраструктуры за счет использования виртуальных ресурсов (т.е. виртуальных машин, VMs) и замены аппаратных решений, сделала огромный шаг вперед в отрасли телекоммуникаций. NFV минимизировала CAPEX (капитальные затраты) и OPEX (операционные затраты) расходы, которые разделили сетевую инфраструктуру на провайдеров, и время, используемое для внедрения новых услуг и приложений, которые будут разработаны;
- NFV оптимизирует для конечных пользователей предоставление ресурсов с высоким качеством обслуживания (QoS) и обеспечивает производительность VNF, например, низкую задержку;

- NFV обладает большей гибкостью, чем традиционные сети, поскольку операторы сети используют новые услуги на той же аппаратной платформе и могут вводить услуги для клиентов в зависимости от их требований, динамически изменяя производительность NFV;
- NFV снижает потребление энергии за счет объединения сетевых устройств;
- NFV разделяет аппаратное и программное обеспечение. NFV использует коммерческое программируемое оборудование такое, как серверы, коммутаторы и системы хранения данных, оно отделяет программные функции от аппаратных без установки новых программ или оборудования, может быть использовано при необходимости как Firewall с открытым исходным кодом операторами сети в виртуальной машине (VM) на платформе X86. И можно выполнять сетевые функции на платформе, основанной на обработке данных, например, обработку сигналов в опорной сети сотовой связи;
- NFV не требует дополнительной установки нового аппаратного устройства, так как оператор мобильной связи может активировать любое программное обеспечение, если виртуальные ресурсы (BM, контейнеры) в какой-то момент времени делают NFV экономически эффективным;
- NFV повышает безопасность основных архитектурных компонентов, применяя методы безопасности в виде виртуальной инфраструктуры менеджера (VIM), главным компонентом которой является гипервизор и прикрепленные к нему VM/гостевые ОС для отражения атаки и разрушения данных;
- NFV не зависит от SDN, его можно использовать отдельно, но SDN и NFV дополняют друг друга. Эта технология виртуализирует и облегчает динамическую мобильность контроллеров SDN в необходимом месте. SDN полезен для NFV тем, что позволяет подключать сеть с помощью программ,

чтобы функционирование сети зависело от мониторинга и анализа трафика сети. Приведем практические примеры VNF: vRouters, virtual content delivery servers, virtual VPN servers, vFirewalls.

Инфраструктура NFV. Инфраструктура NFV включает в себя аппаратные и программные ресурсы, аппаратные блоки включают в себя обработку, хранение и возможность подключения к VNFs через уровень виртуализации, который дает возможность использовать программное обеспечение над аппаратными ресурсами с помощью виртуализированной инфраструктуры. Примерами уровня виртуализации являются гипервизор и решения виртуализации контейнеров, такие как Docker. NFV состоит из трех компонентов:

1. Сетевая функция виртуализации (VNF):

VNF отвечает за функциональность программного обеспечения для выполнения сетевых операций конкретной инфраструктурой NFVI на одной или нескольких виртуальных машинах (VM). VNF — это виртуальные приложения, такие как маршрутизация, процессы для глубокой инспекции пакетов и управления трафиком.

2. Инфраструктура виртуализации сетевых функций (NFVI):

NFVI имеет программные и аппаратные ресурсы для управления и поддержки сети, обработки, виртуального хранилища и различных VNF.

3. Управление и оркестровка виртуализации сетевых функций (NFV-MANO):

NFV-MANO обеспечивает архитектурную структуру среди сервисов интерфейсов VNFs и NFVI отдельных модулей элементов и API из плоскости управления и оркеструет их соответствующие подкомпоненты.

1.4.3 Граничные вычисления

Термин "мобильные граничные вычисления" (MEC) был создан Европейской группой отраслевых спецификаций (ISG) и Институтом

телекоммуникационных стандартов (ETSI). В группу ISG входят Nokia, Intel, Vodafone, IBM, Huawei, NTT DOCOMO и другие. MEC также признана European 5G Infrastructure Public Private Partnership в качестве одной из главных развивающихся технологий для сетей 5G.

В соответствии с [47] "MEC предлагает среду IT-услуг и возможности облачных вычислений на границе мобильной сети, в сетях радиодоступа (RAN) и в непосредственной близости от мобильного устройства". MEC предоставляет облачные услуги в RAN и соединяет оконечные устройства непосредственно с ближайшей точкой, предоставляющей облачные услуги, вместо непосредственного мобильного трафика между оконечными устройствами и опорной сетью. Развертывание MEC на базовых станциях улучшает вычислительные возможности и помогает избежать узких мест в сети и сбоев системы. В соответствии с техническим документом, опубликованным ETSI, MEC можно охарактеризовать следующим образом [48].

1) Обособленность.

Платформы MEC могут работать изолированно от остальных сетевых структур, при этом они имеют возможность доступа к локальным ресурсам. Это очень важно для сценариев связи по типу «машина-машина». Отделение MEC от других сетевых структур также делает эту технологию менее уязвимой.

2) Расположение.

Сеть MEC может быть развернута максимально приближено к источнику информации. Таким образом, MEC помогает анализировать и материализовывать большие данные. Данная технология также полезна для оконечных устройств для обработки данных приложений, требовательных к ресурсам, таких как видеоаналитика, дополненная реальность и т.д.

3) Снижение задержки. Поскольку услуги MEC могут быть предоставлены в ближайшем расположении к мобильным пользователям,

качество обслуживания (Quality Of Service – QoS) для мобильных устройств в отношении задержки и пропускной способности будет улучшено. Простыми словами, в сети будет сверхнизкая задержка и высокая пропускная способность;

4) Определение местоположения. Граничные распределенные устройства могут использовать сигналы достаточно низкого уровня мощности для определения местоположения каждого подключенного устройства. Это будет полезно для всех бизнес-ориентированных сценариев использования, включая аналитику, услуги на основе местоположения и т. д.;

5) Сетевая контекстная информация. Компании, использующие приложения, которые зависят от данных реального времени, могут извлечь выгоду из платформы MEC в своем бизнесе. Эти приложения могут оценить пропускную способность сети и перегруженность радиоэфира. В будущем это поможет им принимать разумные решения для лучшего предоставления услуг клиентам.

Технологии, связанные с MEC

Mobile Cloud Computing (MCC – Мобильные Облачные Вычисления), Cloudlets (Облачные Системы), Fog Computing (Туманные Вычисления) and Local Cloud (Локальное Облако) – технологии, связанные с MEC. Далее мы подробно объясним эти термины.

– MCC

В целом, MCC объединяет все преимущества MEC, мобильного Интернета и Облачных Вычислений [49]. Основной целью облачных вычислений является предоставление изолированных и виртуализированных вычислительных ресурсов, ресурсов хранения и связи для поддержки и повышения производительности для обслуживания оконечных пользователей [50]. Google, Ростелеком, Microsoft Azure, Amazon EC2 и Aneka – примеры компаний,

предоставляющих услуги облачных вычислений. MCC предоставляет различные ресурсы и услуги по требованию, к примеру, какие-либо приложения, сетевые услуги, серверные и вычислительные ресурсы, которые, в свою очередь, легко управляются. Однако, MCC менее полезен в средах, содержащих интенсивные (требовательные к QoS-параметрам) вычислительные задачи, поскольку облачные серверы расположены далеко от конечных пользователей. Например, некоторые приложения, подключенные к облачному серверу, могут столкнуться с проблемами в виде задержек или отключений во время использования приложений.

Локальные облачные вычисления (Local Cloud)

Локальное облако управляется внешними или внутренними источниками, такими как организации или группы [51]. Локальное облако может быть реализовано в локальной сети, которая координируется с удаленным облачным сервером для поддержки конфиденциальности информации. Оно может быть активировано путем установки программного обеспечения на локальном сервере, которое связано с облачным сервером. Однако, хотя локальное облако и оказывает положительный эффект для сетевой задержки, из-за нехватки ресурсов оно подвержено некоторым ограничениям в вычислениях [52].

– Cloudlet

Cloudlet – это небольшой центр данных, который обычно развертывается в беспроводной точке доступа, удаленной (как WiFi) от мобильных пользователей (например, в общественных местах, таких как торговый центр, больница и т.д.) (рис. 1.1) [53]. Cloudlet можно рассматривать как перспективное решение проблем высоких задержек и энергопотребления. Эта технология подразумевает расположение центра обработки данных перед ближайшими облачными серверами [54]. Основная идея Cloudlet заключается в том, чтобы приблизить

облачные сервисы к оконечным устройствам для поддержки приложений, чувствительных к задержкам [55]. Однако, облако состоит из ограниченных ресурсов. Следовательно, оно не может быть использовано в интенсивных вычислительных средах. Поэтому в качестве базы для реализации данной технологии могут выступать точки доступа Wi-Fi, что в свою очередь говорит о зависимости Cloudlet от надежного и постоянного подключения к Интернету. В связи с этим возникают некоторые проблемы в области конфиденциальности и безопасности [56].

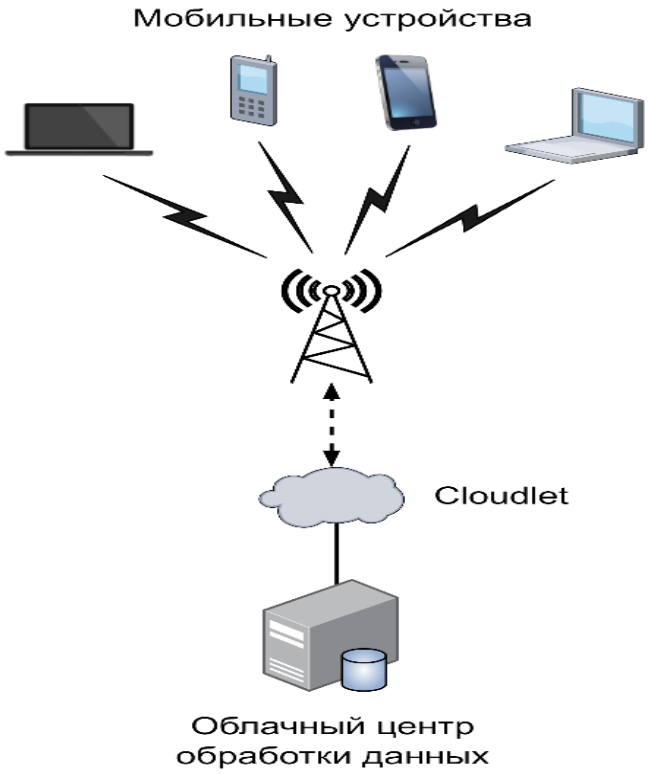


Рисунок 1.4.3.1 – Cloudlet.

Туманные вычисления

Технология Fog Computing (также известная как edge computing или fogging) была представлена компанией CISCO для предоставления облачных услуг оконечным устройствам на границе сети. Туманные вычисления были специально разработаны для сценариев IoT. В Fog Computing вычисления, в

основном, выполняются на уровне локальной сети, шлюза IoT или «туманного» узла. Туманные вычисления позволяют отдельным устройствам собирать данные с различных датчиков в сети, и в соответствии с полученной информацией, выполнять определенные действия. Например, умный пылесос, который может получить данные о загрязнении того или иного объекта в помещении и сразу отправиться на очищение указанной области.

Туманные вычисления обеспечивают более низкую задержку по сравнению с MCC. Однако туманные вычисления страдают от некоторых ограничений, поскольку они зависят от непрерывного беспроводного соединения, которое должно быть постоянным для выполнения сложных задач.

Технологии MEC и Fog Computing широко используются в сфере беспроводных вычислений, но в некоторых отношениях они отличаются. Говоря более точно, в Fog Computing вычислительные мощности находятся между оконечными устройствами и центром обработки данных (на уровне локальной сети) [57]. В то время как в среде MEC облачные услуги выполняются в пределах RAN (Radio Access Network – Сеть Радиодоступа), поэтому MEC более популярна в сетях четвертого (4G) и пятого (5G) поколения.

Туманные вычисления — это новый тип технологий, который способен определить вычисления и сеть для приложения IOX, применяемого на огромном пространстве. Он также рассматривается как дополнительная часть для дизайна облачных вычислений, памяти и сети. Кроме того, это помогает ускорить вычисления на границе сети и обеспечить многоуровневые задачи, которые помогают в распределении ресурсов среди системы и приложений. Это может быть выполнено на границе устройств или сетевых методов для использования вычислений везде. Туманные вычисления дают возможность добавить несколько различных составляющих таких, например, как система

проектирования, приложение, занятость для программного обеспечения, безопасность и управление для вычислительных методов и сетей [55].

Туманные вычисления строят свою сетевую архитектуру на основе граничных устройств вблизи оконечного пользователя, чтобы иметь возможность собирать огромное количество данных, хранящихся в памяти, решать вычислительные и коммуникационные задачи, необходимые для передачи всех видов конфигураций, обработки, контроля, измерения и управления. Эта технология характеризуется своей способностью расширять граничные вычисления, другими словами, она работает в основном на идее расширения облачных вычислений. Некоторыми из основных свойств туманных вычислений являются следующие:

1) Туманные вычисления зависят от структуры огромного количества различных устройств с возможностью децентрализации, чтобы обеспечить возможность участвовать всем устройствам вместе и реализовывать любые функции для сети и приложений.

2) Слово "туман" не привязано к определенной области технологий. Оно способно обозначать неоднородность как устройств, так и интерфейсов [55].

3) Туманные вычисления состоят из платформы, которая работает виртуально и способна поддерживать фундаментальную функцию сети или приложения, и сервисов, которые недавно добавлены и все еще применяются в тестовой среде [55].

4) Характеристики, которые ассоциируются с туманными вычислениями, включают, например, программируемость связи, осознание местоположения, управление мобильностью.

Узел тумана состоит из физической инфраструктуры, которая предоставляет средства ко всем ресурсам, расположенным на границе сети. Эти

ресурсы делятся на две основные отдельные группы. Первая предназначена для устройств с ограниченными ресурсами, например, маршрутизаторов, коммутаторов, точек доступа и оконечных устройств. Другая группа классифицируется для ресурсов большей емкости, например, как устройства IOx и Cloudlets [55].

Облако определяется как хорошо известный компьютерный ресурс или как масса компьютеров, которые характеризуются тем, что очень хорошо подключены к интернету и всегда готовы к использованию всеми оконечными устройствами, такими как мобильные устройства. Таким образом, получается, что туман характеризуется высокой мобильностью и малым радиусом действия для облачного центра обработки данных, который расположен на границе интернета. Туманные вычисления способны поддерживать все тяжелые ресурсы и реактивные мобильные приложения, которые требуют ультра малой задержки и достаточно высокой пропускной способности, основанной на количестве пользователей, которые работают интерактивно в передаче и получении данных [55].

IOX является одной из версий для туманных устройств Cisco, которые действуют на сбор приложений в гостевой операционной системе (GOS), которая работает и управляет одним или несколькими компьютерами виртуальных машин на маршрутизаторе Connected Grid Router (CGR). Cisco анонсировала IOX как "основу для создания приложений для IOX". Это единственная первая версия программного обеспечения для IOS, выполненная в виде Linux. IOX рассматривается как шкала, которая работает для выявления тумана в сети [55].

Основная функция, которую преследовала компания Cisco, создавая облако, заключалась в управлении всеми сетевыми приложениями и сервисами в промежуточном блоке путем предоставления им возможности хранения и

вычисления. Это помогает снизить огромную нагрузку и объем трафика, доступного в облаке, разделив нагрузку процесса обработки на все сетевые ресурсы" [55].

Туман как ключевая вспомогательная технология:

Все приложения, связанные с сетью транспортных средств, могут работать на основе туманной технологии, например, как построение архитектуры Интернета транспортных средств (IOV) или, другими словами, туманных вычислений транспортных средств с помощью интеграции между туманными вычислениями и обычными автомобильными ad-hoc сетями. Действительно, транспортные средства являются устройствами, которые характеризуются как интеллектуальные мобильные устройства и оснащены большим количеством датчиков, кроме того, они имеют возможность собирать основные данные о дорожном движении с помощью вычислительных и коммуникационных возможностей. Данные могут быть собраны с датчиков внутри транспортных средств. При такой структуре узлы тумана могут располагаться на границе автомобильных сетей, что значительно помогает в процессе сбора информации более точным способом, а также дает возможность в режиме реального времени организовывать, обрабатывать и хранить информацию о дорожном движении. Существует три основных слоя, которые можно определить как "умные" транспортные средства, собирающие данные, придорожные устройства/узлы тумана как слой тумана, и облачные серверы как слой облака.

Поскольку умные транспортные средства способны вычислять, общаться и иметь возможность хранения данных, которые могут быть применены в режиме реального времени, это играет важную роль в качестве источника информации в туманной вычислительной автомобильной системе. Совокупная информация в интеллектуальных транспортных средствах, собранная несколькими датчиками, составляет около 25 ГБ/ч в день. Умные автомобили

начинают обрабатывать часть этой информации, чтобы иметь возможность принимать решения в реальном времени, с другой стороны, иная информация будет распределяться и отправляться на узлы туманных вычислений, чтобы иметь возможность использовать ее в другой раз для любого типа анализа, например, для планирования управления движением. Придорожные устройства расположены в нескольких районах города, при этом узлы тумана становятся областью обработки собранных данных и передают их на облачные серверы. Взаимодействие может быть расширено как система промежуточного программного обеспечения, которая помогает установить связь между интеллектуальными транспортными средствами и облачными серверами в туманной вычислительной автомобильной системе. Эти узлы характеризуются большим количеством задач и предоставляют больше различных услуг, связанных с транспортными средствами таких, например, как умный светофор, навигация и потоковое видео. Они должны обеспечивать мониторинг на уровне города, постоянно хранить информацию и играть централизованную роль в системе управления. Узлы тумана будут работать над распределением данных среди окружающей информации, чтобы иметь возможность принять наилучшее решение.

Архитектура MEC

Прежде чем мы обсудим архитектуру MEC, мы представим краткую историю и общую архитектуру сотовой сети с точки зрения RAN.

История RAN в сотовых сетях

Первое поколение сотовых сетей (1G) было представлено в начале 1980-х годов. 1G была аналоговой системой связи и поддерживала мобильность. Позже была открыта система цифровой сигнализации, использующая метод множественного доступа с временным разделением каналов (TDMA), и на смену

1G пришло второе поколение сотовых сетей (2G), которое предложило лучшее качество голосовых услуг и высокую безопасность по сравнению с 1G. Затем было выпущено третье поколение сотовых сетей (3G) с более высокой скоростью передачи и согласованностью мультимедийных приложений с использованием RAN с ограниченной поддержкой данных. С быстрым ростом использования Интернета через RAN, 4G завоевал рынок сотовых сетей, обеспечивая наилучшее QoS для пользователей [58].

RAN является жизненно важной частью любой сотовой сети, сеть радиодоступа облегчает связь между пользователями и с основной сетью. В традиционных радиосетях устройства подключаются к операторам мобильных сетей (MNO) через RAN. RAN охватывает широкий географический диапазон, который делится на несколько ячеек, где каждая ячейка состоит из одной БС, а БС связаны друг с другом через радиолинии или стационарные линии связи с контроллером радиосети (RNC). Основной функцией RAN является выполнение некоторых функций управления мобильной связью и управление узлами BS. Информация шифруется перед отправкой в основную сеть. RNC связаны с одной или двумя транзитными сетями. С момента коммерциализации 4G сотовые сети стали более эффективными, чем раньше, поскольку они обеспечивают высокую скорость передачи данных, низкую задержку и высокую пропускную способность.

Эволюция системной архитектуры ядра долгосрочной эволюции (LTE) может быть представлена с HetNets и некоторыми другими системами, такими как General Packet Radio Service (GPRS) и Universal Mobile Telecommunications System (UMTS) [59]. На рис. 1.2 показан общий вид сотовых сетей, где ядро сети соединено проводными каналами связи, такими как Ethernet или Интернет-протокол (IP) с RAN, а RAN соединена с мобильными пользователями беспроводным способом. Соединение между БС и транзитной сетью

осуществляется с помощью кабельной сети Ethernet через RAN, которая поддерживает высокую скорость передачи данных. Значительный рост IP из Интернета в сети связи общего пользования заставляет поставщиков услуг LTE внедрять его. Объем IP-данных между опорной сетью и RAN инкапсулируется с помощью туннельного протокола GPRS с шифрованием IPsec.

В последнее время идея облачной RAN (C-RAN) была представлена некоторыми сетевыми операторами. C-RAN предлагает централизованные вычисления, совместную радиосвязь и экономную инфраструктуру энергопотребления [60]. В частности, она объединяет все вычислительные ресурсы БС в центральный пул. В C-RAN радиосигналы от распределенных антенн могут собираться удаленными устройствами и передаваться на облачный сервер по оптическим каналам связи. Использование C-RAN позволяет сократить количество сотовых площадок и предоставить устройствам более качественные услуги, при этом сохраняется аналогичное покрытие и снижаются эксплуатационные расходы.

Рисунок 1.4.3.2 – Инфраструктура сотовой сети

Трехуровневая архитектура

Как показано на рис. 1.3, MEC — это уровень, который существует между облачным уровнем и уровнем оконечных устройств. Поэтому данная инфраструктура известна как трехслойная (облако, MEC и оконечные

устройства) [61]. MEC в основном работает в паре с облачным сервером для поддержки и повышения производительности оконечных устройств.

На рисунке 1.4 показана структура сети MEC. Идея заключается в том, что для каждого целевого устройства предусмотрен свой граничный сервер с определенными приложениями и услугами. Такой сервер может быть развернут локально в определенной области или помещениях, к примеру, в торговых центрах, парках, автобусные терминалы и т.д.

Для обеспечения доступа оконечных устройств к сервисам и приложениям могут использоваться компоненты сотовой сети, к примеру, базовые станции. Платформа MEC может быть установлена в фиксированном месте, например, в торговом центре или на любом движущемся объекте, таком как автобус. MEC может быть развернута в многотехнологичном месте агрегации сот (3G/4G). Многотехнологичная площадка агрегации сот может существовать на открытом воздухе или в помещении. Технология MEC предлагает экосистемные способы продвижения облачных услуг от опорной сети к БС и эффективной оптимизации услуг RAN. Таким образом, ключевое значение MEC заключается в предоставлении услуг облачных вычислений путем предоставления облачных ресурсов, таких как вычислительные ресурсы, хранилище и сеть, на границе сети, чтобы удовлетворить требования приложений для выполнения их задач.

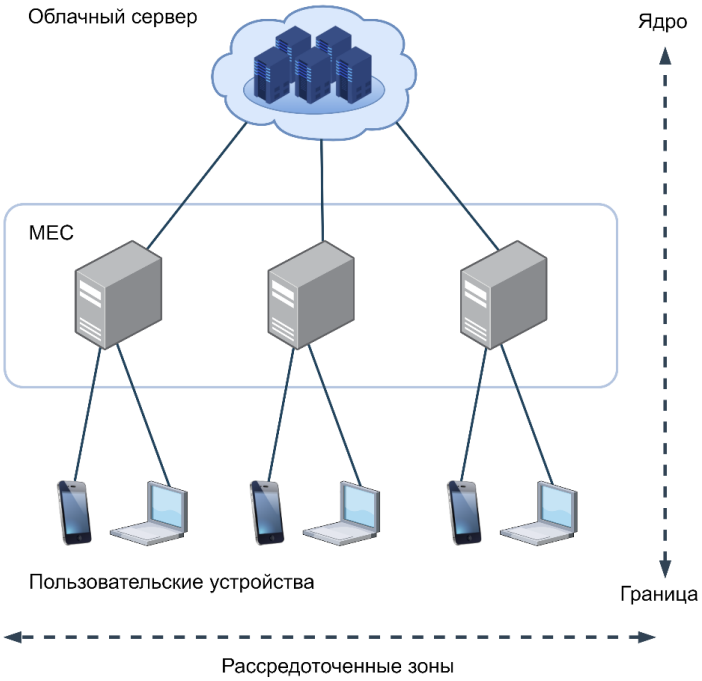


Рисунок 1.4.3.3 – Трехуровневая архитектура MEC.

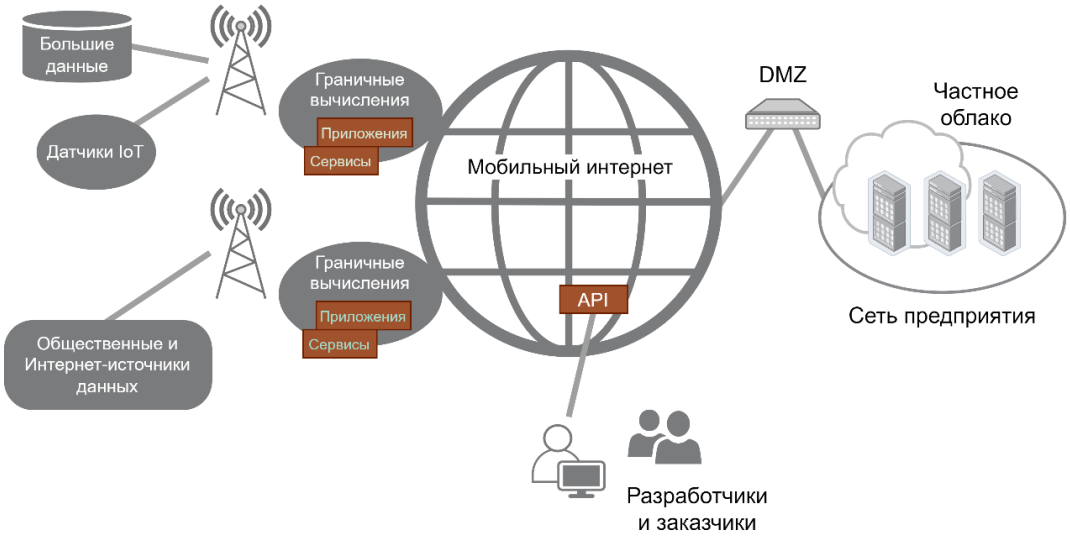


Рисунок 1.4.3.4 – Инфраструктура MEC.

Особенности MEC

Как уже упоминалось в предыдущих разделах, существует множество преимуществ, связанных с MEC, которые оказываются полезными как для поставщиков услуг приложений (ASP – Application Service Provider), так и для

операторов сети (MNO – Mobile Network Operator). Кроме того, они полезны для поставщиков сетевого оборудования, контент-провайдеров, поставщиков промежуточного программного обеспечения и для OTT-услуг [62]. Концепция MEC сосредоточена на основных показателях, таких как высокая пропускная способность и низкая задержка, которые достигаются за счет размещения данных на серверах MEC, а не на централизованных облачных серверах. Более того, потребление энергии также является одной из основных проблем. Вычислительные задачи переносятся во внешние системы с богатыми ресурсами для увеличения времени работы оконечных устройств от аккумулятора. Кроме того, распределенные виртуальные серверы обеспечивают масштабируемость и надежность. Что касается игроков рынка (MNOs, ASPs и оконечных пользователей), преимущества MEC заключаются в следующем.

MNO могут разрешить доступ к RAN сторонним поставщикам для реализации своих приложений и услуг более гибким и адаптируемым способом. Эти вспомогательные услуги могут приносить прибыль за счет взимания платы на основе используемых услуг, таких как пропускная способность, хранение данных и другие IT-услуги. Некоторые другие услуги, такие как услуги DVR и OTT, предоставляемые кабельными операторами, могут быть более быстрыми, поскольку их услуги могут находиться на серверах MEC. ASP могут получать доход благодаря использованию платформы MEC "инфраструктура как услуга" (IaaS) на границе сети, что делает услуги ASP масштабируемыми наряду с низкой задержкой и высокой пропускной способностью. ASP также могут получить доступ к активности пользователей в режиме реального времени, что может помочь в разработке более эффективных приложений. RAN перестраивается на доступе с учетом услуг и предлагает данные о местоположении устройства, нагрузке на ячейку и перегрузке сети. Конечные пользователи могут экспериментировать с быстрыми вычислительными

приложениями посредством вычисления методов загрузки, которые управляются серверами MEC в RAN. Кроме того, тесная ассимиляция RAN и физически близких серверов может улучшить QoE пользователя, например, в кэширование видео, высокой пропускной способности браузера, лучшего DNS и т.д.

MEC Приложения

Платформа MEC — это новый доход для мобильных операторов, которая еще не достигла достаточной зрелости. Тем не менее, можно обсудить некоторые приложения, которые могут адаптировать MEC следующим образом:

1) Дополненная реальность

В последнее время приложения дополненной реальности (AR) охватили мобильные технологии (например, Wikitude, Layar, Google и т.д.). AR позволяет пользователю ощутить окружающую среду в реальном времени, интегрируя виртуальные и реальные объекты, существующие одновременно [63], [64]. Новые AR-приложения могут адаптивно интегрировать звуковые и визуальные компоненты (например, телевизионные программы, игры, распознавание объектов и т.д.). Однако AR-приложения обычно требуют высокой пропускной способности и низкой задержки для улучшения QoE. Платформы MEC признаны эффективной технологией для приложений, чувствительных к задержкам в области AR. MEC может улучшить AR-системы, например, оптимизировать пропускную способность за счет использования ресурсов публичного облака на границе сети вместо передачи трафика данных непосредственно в опорную сеть. Таким образом, разгрузка интенсивных вычислительных задач на ближайший облачный сервер является более эффективной и оптимизированной для QoS пользователя. Взаимодействие мозга и компьютера - пример AR-приложений, которые работают на основе обнаружения ритмов мозговой активности

человека. Информация, получаемая датчиками в реальном времени, требует интенсивных вычислительных ресурсов, которые обрабатываются платформами MEC.

2) Доставка и кэширование данных.

Технология MEC играет важную роль в оптимизации производительности веб-сайта (например, реорганизация веб-разметки, кэширование HTML-контента, изменение размеров веб-компонентов и т.д.). Пользователь отправляет HTTP-запросы, которые проходят через сервер MEC. Этот сервер управляет запросами пользователей, обрабатывая ряд задач для загрузки веб-страниц на интерфейс пользовательского устройства. Поскольку сервер MEC развернут рядом с пользователем, эти запросы и ответы выполняются в реальном времени. Платформы MEC эффективны по времени по сравнению с традиционными интернет-платформами, где запросы пользователей обрабатываются серверами, расположенными далеко от их местонахождения. Кроме того, MEC может обнаруживать и анализировать производительность сети в часы пиковой нагрузки. Например, когда разные пользователи одновременно транслируют видео в условиях перегруженной сети, графическое разрешение снижается до более низкого, MEC поможет справиться с запросами пользователей, избегая отказа в обслуживании.

Технологии реализации MEC

Реализация платформ MEC стала возможной благодаря нескольким ключевым технологиям, которые также имеют решающее значение в развитии мобильных сетей пятого поколения. В частности, 5G включает в себя множество новых технологий, обеспечивающих надежную связь, повышенное энергосбережение, низкую задержку, решающих задачу с нехваткой радиочастотного спектра и растущего объема данных от оконечных устройств. Кроме того, ожидается, что сети 5G будут способствовать гибкому и

программируемому развертыванию услуг и основных сетевых функций с помощью программно-конфигурируемых сетей (SDN – Software Define Network) и виртуализации сетевых функций (NFV – Network Function Virtualization). Ожидается, что эти технологии будут играть ключевую роль и в развитии технологии MEC в том числе. Подробное исследование технологий предлагается в [65], в данной же работе будут представлены наиболее важные из них для MEC.

Виртуализация

Виртуализация позволяет операторам платформ MEC выполнять одновременно несколько независимых задач одного и тоже программного обеспечения на одном физическом сервере. Эти задачи могут получать доступ к базовым физическим ресурсам, будучи изолированными друг от друга. Такая изолированность позволяет им работать, не мешая (или даже не зная) о существовании других, работающих на том же сервере. Виртуализации происходит на базе виртуальных машин (VM – Virtual Machine)

В настоящее время VM являются основным средством развертывания виртуализированных версий программного обеспечения в облачных средах. Программный уровень абстракции (гипервизор), расположенный между физическим оборудованием и виртуальными машинами, разрешает виртуальным машинам использовать базовые ресурсы процессора, хранения данных и сети. Поверх операционной системы (OS – Operation System) главного сервера каждая VM запускает свою собственную OS пользователя. Стоит отметить также, что несмотря на плюсы виртуализации на основе гипервизора в изолирование ресурсов и распределении нагрузок, данная технология обладает незначительными накладными расходами.

В качестве альтернативы существует виртуализация на базе контейнеров. В этом случае виртуализированные версии программного обеспечения не

требуют запуска разделенной OS и могут совместно использовать ресурсы базового хоста. Изолирование контейнеров в данном случае происходит за счёт определенных доработок основной OS. Такой вид виртуализации помогает сократить время запуска дубликатов программ, что в целом приводит к повышению производительности [66].

Помимо разделения физических ресурсов, виртуализация также дает возможность миграции виртуальных машин или контейнеров. В частности, речь идёт про перемещение вычислительных ресурсов с одного физического сервера на другой. Это оказывает положительное влияние на производительность при сценариях мобильности пользователей или для снижения энергопотребления центра обработки данных.

При перемещении экземпляра с одного хоста на другой, в сети происходят задержки, для этого существует механизм миграции в реальном времени. В то время как механизм миграции в реальном времени виртуальных машин существует уже давно, и широко используется облачными провайдерами, в случае с миграцией контейнеров он начал использоваться сравнительно недавно. Стоит отметить, что технологии виртуализации и миграции в реальном времени играют ключевую роль для MEC. Кроме того, эти технологии являются основой для SDN и NFV.

NFV и SDN

Задача реализация сетевых функций в виде программных модулей, которые могут работать на аппаратном обеспечении общего назначения, является основой технологии NFV. Она отделяет базовое оборудование от программного обеспечения, используя ранее рассмотренные технологии виртуализации. При таком сценарии услуги и некоторые сетевые функции больше не обязаны и не приурочены к специализированному оборудованию.

Вместо этого они могут быть реализованы на узлах общего назначения. По этой причине сетевые функции могут быть развернуты в более надежных и подходящих для них местах, что положительно сказывается на эффективности предоставления услуг.

Кроме того, в таких сценариях происходит отделение компонентов, отвечающих за контроль и управление, от транспортной плоскости, что говорит об использовании технологии SDN. Благодаря логически централизованному контроллеру, который управляет политиками в сети и решениями о передаче (транспортировке) данных, SDN помогает упростить и повысить эффективность управления. Стоит также отметить, что контроллер помогает повысить скорость развертывания новых услуг.

Использование технологий SDN и NFV совместно позволяет развертывать программные модули в гибкой и программируемой форме, тем самым облегчая управление и конфигурирование сети. Более того, эти технологии очень важны для операторов сетей в быстром развертывании новых программных функций с ограниченными финансовыми затратами. Например, автоматическое развертывание виртуальных ресурсов может быть использовано технологией NFV для поддержания работоспособности сети с внезапным увеличением трафика, генерируемого приложениями IoT на определенном объекте.

В случае с платформами MEC технология NFV может развернуть необходимые виртуальные ресурсы вблизи оконечных устройств, а управление виртуальными машинами или контейнерами будет осуществляться с помощью SDN, наряду с контролем политик и маршрутизацией трафика.

Выгрузка вычислений

Как правило, вычислительные задачи с оконечных устройств с ограниченным объёмом ресурсов передаются для обработки в облако. В данном

сценарии очевидна необходимость использования именно платформы MEC по причине её положительного влияния на QoS-параметры сети. В целом, передача вычислений с оконечных устройств несёт в себе ряд преимуществ. Например, увеличение автономной работы или запуск ресурсно-требовательных программ на устройствах с малыми объемами вычислительных ресурсов. Выгрузку вычислений с оконечных устройств можно разделить на три части:

1) Отсутствие выгрузки. Отсутствие выгрузки означает, что задача вычисляется локально на самих устройствах без использования сервера MEC. В действительности большинство мобильных устройств вычисляют задачи именно так. Факторами, влияющими на производительность локальных вычислений, являются скорость считывания информации и мощность процессора;

2) Полная выгрузка. Полная выгрузка или иногда ее называют бинарной выгрузкой означает, что все вычислительные задачи передаются на сервер MEC для обработки. При полной выгрузке время выполнения задачи делится на время передачи данных и время вычисления данных. Факторами, влияющими на эффективность полной выгрузки, являются пропускная способность канала, состояние канала и вычислительная мощность сервера MEC;

3) Частичная выгрузка. Частичная выгрузка разделяет вычислительную задачу на две части в соответствии с планированием алгоритма выгрузки вычислений: часть задачи обрабатывается локально, другая выполняется посредством сервера MEC. Частичная выгрузка требует учета таких факторов, как потребление энергии, вычисление данных задачи, скорость передачи данных и распределение вычислительных ресурсов. Эти факторы не только влияют на общую эффективность системы, но и воздействуют друг на друга. Изменения в некоторых факторах приведут к изменению состояния всей системы. Частичная

выгрузка является наиболее сложным алгоритмом из трёх, поскольку она требует объединения и учёта ряда взаимозависимых факторов.

Прежде, чем инициировать выгрузку вычислений, сервер MEC выполняет несколько шагов, как показано на рис. 1.5. Во-первых, он проверяет легальность (правомерность использования данного вида услуг) пользователей, которые обращаются за этой услугой. Если пользователь легален, вычислительные ресурсы выделяются для него в соответствии с потребностями его задачи. В обратном случае, он не обслуживается.

Существуют параметры, определяющие эффективность алгоритма выгрузки вычислений, среди них: временная задержка, использование полосы пропускания и применимость алгоритма.

1) Временная задержка. Временная задержка – это сумма времени отклика системы и времени обработки данных, за которое пользователь или приложение запрашивает вычислительные услуги. Основными факторами, влияющими на длительность временной задержки, являются пропускная способность канала, эффективность алгоритма разгрузки вычислений и объем данных задачи. Временная задержка уникальна для каждой задачи;

2) Использование полосы пропускания. В процессе выгрузки вычислений будет передаваться огромное количество данных, что может вызвать перегрузку канала, что приведёт к большим временным задержкам. Если транспортной средой является радиоканал, то, очевидно, частотный спектр ограничен. Одним из решений является увеличение используемой полосы пропускания;

3) Применимость алгоритма. Различные сценарии применения предъявляют различные требования к энергии и вычислительным ресурсам. Например, онлайн-игры и видео-сервисы требуют гораздо меньшей временной задержки, чем кэширование данных, а узлы, которым требуется обрабатывать

огромные объемы данных, потребляют много энергии. Поэтому модель разгрузки вычислений, применимая ко всем сценариям, невозможна.

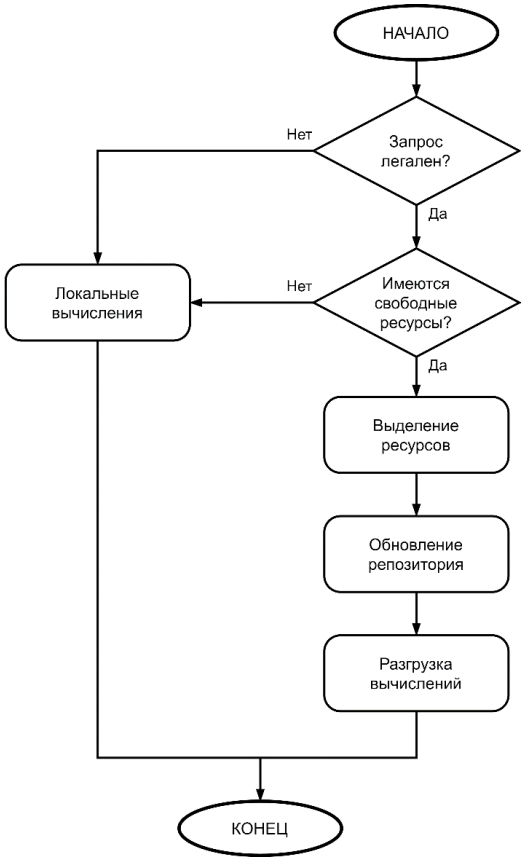


Рисунок 1.4.3.5 – Диаграмма выгрузки вычислений

Оптимальный сценарий может быть определен путем измерения характеристик пользовательских требований со всех доступных областей.

1.4.4 **Искусственный Интеллект в сетях связи**

Помимо новых решений по передаче данных в сети и модернизации самой ее архитектуры, для сетей связи пятого, шестого и последующих поколений не менее важной становится задача обработки трафика. Объемы трафика, генерируемого и пересылаемого в сетях, значительно увеличиваются, с увеличением числа устройств, подключенных к сети, и потребностей

пользователей в предоставляемых сетями услугах. Количество трафика настолько велико, что его обработка с сохранением эффективности и гибкости всей сети становится под силу только активно развивающейся технологии Искусственного Интеллекта (ИИ) . Рекомендация ITU-R M.2083-0 обязывает разработчиков внедрять ИИ в сети связи как составной элемент обработки трафика и его прогнозирования, управления сетью и т. п., что объясняется наличием ряда задач, необходимых к решению для достижения целей построения сетей URLLC:

- Идентификация трафика в сети, не увеличивающая задержку при передаче данных;
- Мониторинг состояния сети в реальном времени;
- Прогнозирование нагрузок на элементы сети и на сегменты сети в целом;
- Прогнозирование потребностей абонентов в тех или иных услугах;
- Прогнозирование передвижения пользователей по географической области;
- Идентификация и прогнозирование вредоносного трафика и аномалий в сети с применением опережающих атаки решений и др.

Эти задачи попадают под область задач, решаемых Искусственным Интеллектом. Использование ИИ в сетях связи становится возможным одновременно с построением сети на основе ранее описанных технологий – Программно-конфигурируемых сетей SDN и Виртуализации сетевых функций NFV, которые обеспечивают программируемость сети и «прозрачность» ее управления. Программное обеспечение, посредством которого происходит управление сетью, может так же осуществлять использование алгоритмов машинного обучения и обработки больших данных.

Сервисы, применяющие технологии ИИ, могут располагаться как на верхних уровнях сети в качестве приложений, получающих трафик от нижних уровней и оперативно его обрабатывающих, так и быть заложены в программное обеспечение на контроллерах сети, и решать вышеописанные задачи в локальных зонах.

1.4.5 **Микросервисная архитектура**

Традиционный подход к разработке приложений имеет ряд недостатков в вопросе применимости для облачных сервисов сетей пятого, шестого и последующих поколений. Разработка по традиционному подходу проходит ряд тяжеловесных этапов, приложение долго тестируется и только после этого вводится в эксплуатацию, архитектура является монолитной и труднопереносимой, не масштабируемой и вовсе не гибкой, что противоречит современным требованиям к сетям и к их компонентам.

В последние несколько лет многие компании поддержали новый подход к разработке приложений и активно начали использовать концепцию микросервисной архитектуры вместо традиционной.

Микросервисная архитектура в сущности позволяет разделить мощности одного монолитного приложения на компоненты, представляющие из себя части функционала большого сервиса. Действия, выполняемые конкретным микросервисом, определяются разработчиками приложения.

Получившаяся группа микросервисов, на которые дробится приложение, работает едино, каждый элемент группы взаимодействует со всеми остальными элементами или с их частью, а на выходе такой системы получается уникальный продукт, по функционалу не отличающийся от первоисточника, однако более надежный и отказоустойчивый в работе, с огромным приростом в производительности и масштабируемости всей системы. Технологии, на основе

которых строится микросервисная архитектура, так же определяются ее разработчиками.

Каждый микросервис, в частности, может быть реализован посредством виртуализации на уровне ядра, чаще всего упоминающейся, как контейнеризация [31]. Контейнеризация – особый вид виртуализации, который подразумевает отсутствие гипервизора в организации архитектуры множественных окружений в одной вычислительной единице [39, 40]. В привычном, традиционном подходе к виртуализации, в инфраструктуре гипервизора создаются изолированные виртуальные окружения, главным образом работающие благодаря основной функции гипервизора – обеспечению взаимодействия гостевых операционных систем с хостовой.

При контейнеризации же окружение создается посредством возможностей самого ядра. Можно рассматривать этот процесс подобно делению мощностей ядра на составляющие, доступ к каждой из которых предоставляется конкретному контейнеру без возможности использования остальных ресурсов системы. Новый этап в прогрессе виртуализации дал возможность исключить влияние гипервизора на систему, выражающееся как понижение производительности создаваемых сервисов и неудобство упаковки и доставки приложений.

Таким образом, микросервисная архитектура является предпочтительным подходом для разработки приложений в сетях связи пятого и последующих поколений в виду повышения масштабируемости системы и обеспечения ее надежности и переносимости, а также ускорения скорости развертывания сервисов.

1.5 Анализ зарубежных публикаций в области исследований граничных вычислений

В этом разделе рассмотрим дополнительную информацию об основных технологиях и методах в MEC и HetNets.

Выгрузка вычислений в MEC

В последние годы интенсивно изучается вопрос выгрузки вычислительных задач на серверы MEC [67-80].

Методы минимизации задержки

В [67], [68] были разработаны методы выгрузки вычислений для оптимизации задержки выполнения задач. В исследовании [67] для решения стохастической задачи оптимизации двух временных масштабов был использован подход марковского процесса принятия решений. В частности, на большем временном масштабе необходимо определить, выполнять ли задачу в режиме локальных вычислений на устройстве или выгрузить задачу на сервер MEC для удаленных вычислений, в то время как на меньшем временном масштабе политика скорости передачи для размера входных данных должна быть адаптирована к информации со стороны канала. Вычислительные задачи планируются на основе состояния очереди в буфере задач, состояния выполнения локального вычислительного блока, а также состояния блока передачи. Исследуя среднюю задержку для каждой задачи и среднее потребление энергии на пользовательском устройстве, авторы сформулировали проблему задержки с ограничением мощности как задачу минимизации. Для получения оптимальной общей политики планирования задач авторы предложили алгоритм одномерного поиска. Воспользовавшись идеей SDN, авторы работы [68] исследовали проблему разгрузки вычислений в сверхплотной сети для оптимизации задержки при сохранении времени

автономной работы MD. Сформулированная проблема оптимизации представляла собой смешанную целочисленную нелинейную программу, которая является NP-трудной. Для того чтобы решить проблему, они разложили ее на две подпроблемы: подпроблему размещения задач и подпроблему распределения ресурсов.

В [69] рассматривается компромисс между задержкой и надежностью при выгрузке вычислений на граничный сервер. В предложенной схеме предполагается, что пользователь может разделить свою задачу на подзадачи и выгрузить их на несколько близлежащих граничных серверов. Задача системы формулируется как задача минимизации и оптимизации для совместной задержки и вероятности сбоя вычислительной выгрузки. Поскольку исходная задача трудноразрешима, были предложены три алгоритма, основанные на эвристическом поиске, соответственно, для решения исходной задачи путем оптимизации выбора граничных узлов-кандидатов, упорядочения выгрузки вычислений и распределения задач.

В [70] была изучена проблема разделения многопользовательских вычислений с учетом разделения задач нескольких пользователей вместе с планированием выгруженных задач на облачных ресурсах. Вместо того, чтобы обеспечить минимальное время выполнения для каждого отдельного устройства, сценарий был направлен на достижение минимального среднего времени выполнения для всех устройств в зависимости от количества предоставленных ресурсов в облаке. Для решения этой задачи авторы разработали автономный эвристический алгоритм SearchAdjust. На основе SearchAdjust авторы разработали онлайн-алгоритм для решения проблемы, который может быть развернут в практических системах.

Политика выгрузки кода в мобильных устройствах и политика ценообразования/предоставления серверов в провайдерах услуг выгрузки кода

(CSP) изучались независимо друг от друга. В частности, системные модели как для стороны пользователя, так и для стороны CSP не отражали адекватно их практические аспекты. Исследование в [71] рассматривает обе стороны и учитывает их в интегрированной системе мобильной выгрузки кода (MCO) одновременно для разработки практической модели. Используя метод функций Ляпунова плюс штраф, авторы предложили политику выгрузки кода, тактовой частоты локального процессора и выбора сетевого интерфейса для мобильных устройств. Кроме того, они предложили политику ценообразования услуг MCO и предоставления серверов для CSP в сценариях конкуренции и сотрудничества. Алгоритмы Com-UC и Com-PC для мобильных устройств и CSP были предложены в случае сценария конкуренции для оптимизации каждой стоимости для каждого ограничения стабильности очереди. Алгоритм Coo-JC был предложен для оптимизации их суммарной стоимости для мобильных пользователей и CSP в случае сценария сотрудничества.

В [72] авторы исследовали сценарий, в котором два мобильных пользователя получают энергию от беспроводной передачи энергии от точки доступа (AP) и могут выгрузить часть или все свои вычислительно-интенсивные задачи, критичные к задержке, на AP, связанную с граничным сервером. Однако при этом более дальний мобильный пользователь страдает от эффекта удвоенной близости-дальности. Для решения этой проблемы предлагается кооперативная связь в виде ретрансляции ближайшим мобильным пользователем для выгрузки вычислений, целью которой является оптимизация общей энергии передачи AP с учетом ограничений оптимизации задач. Поскольку исходная задача оптимизации является NP-трудной, она может быть решена двухфазным методом. На первом этапе принимаются оптимальные решения по выгрузке вычислений путем решения задачи максимизации суммы энергосбережения для

заданной мощности передачи энергии. На втором этапе с помощью метода бисекционного поиска находится минимальная мощность передачи энергии.

Методы минимизации энергопотребления

В работе [73] предложен метод выгрузки вычислений для MEC с использованием идеи графа вызовов, который моделирует общую компьютерную программу как набор процедур, связанных друг с другом посредством взвешенного направленного графа. Целью авторов было найти оптимальное разбиение графа вызовов, определяющее, какие процедуры должны выполняться в локальном или удаленном режиме. С выбором параметров радиосвязи, таких как мощность передачи, оптимальное разбиение получено совместно для оптимизации потребления энергии мобильным терминалом при ограничении задержки с учетом времени передачи, времени выполнения, одноканальной и многоканальной стратегий передачи, и доказано, что может быть получено глобальное оптимальное решение. Наконец, была предложена субоптимальная стратегия для решения более простой версии исходной задачи, чтобы найти компромисс между сложностью и производительностью предложенного метода.

В исследовании [74] рассматриваются возможности MEC для решения прикладных проблем, связанных с управлением энергией на IoT-устройствах с ограниченными источниками питания, при этом обеспечивается обработка визуальной информации с низкой задержкой и высоким качеством. Используя приложение для распознавания лиц, которое является жизненно важным в сценариях реального времени, таких как катастрофы, авторы предложили новый алгоритм принятия решений по выгрузке, который анализирует компромиссы в вычислительной политике для выгрузки визуальной информации для ее обработки. При низких и высоких рабочих нагрузках в сети этот алгоритм также

анализирует влияние на энергопотребление при принятии решений при различных требованиях к потреблению визуальной информации. Для решения проблемы компромисса между энергоэффективностью и пропускной способностью был предложен алгоритм, основанный на использовании машинного обучения в мобильных ad hoc-сетях, названный устойчивым алгоритмом граничной маршрутизации на основе политики, управляемой искусственным интеллектом. Предложенный алгоритм ориентирован на энергосбережение и устойчивость, что улучшает географические характеристики выбора пути маршрутизации для повышения пропускной способности.

В [75] была рассмотрена архитектура сети малых сот для выгрузки вычислений, целью которой является минимизация общего потребления энергии всеми элементами системы с учетом ограничений на вычислительные возможности и задержку в обслуживании. При этом авторы использовали алгоритм косяка рыб для получения оптимального решения.

В работе [76] был предложен метод выгрузки вычислений от одного мобильного устройства MD (Mobile Device) к нескольким граничным устройствам с целью минимизации энергопотребления и задержки выполнения задач MD путем совместной оптимизации частоты центрального процессора (CPU) MD и решения о распределении задач.

В работе [77] авторы исследовали проблемы максимизации скорости вычислений в системе MEC с беспилотным летательным аппаратом (БПЛА) в режиме частичной выгрузки и в режиме бинарной выгрузки, с учетом некоторых оптимизационных ограничений в отношении энергосбережения и скорости БПЛА. Для решения сформулированных задач были предложены двухэтапный алгоритм и трехэтапный альтернативный алгоритм, соответственно.

Методы минимизации накладных расходов

Авторы в [78] стремились минимизировать накладные расходы системы путем изучения многопользовательской задачи выгрузки для MEC в многоканальной беспроводной среде. Они показали, что сформулированная задача оптимизации является NP-трудной. Поэтому был использован теоретико-игровой метод для получения эффективной задачи выгрузки. Проблема оптимизации распределенной задачи принятия решения о выгрузке среди мобильных устройств MD формулируется как многопользовательская задача игры о выгрузке. Были проанализированы структурные характеристики игры и доказано, что для игры существует точка равновесия Нэша. Для получения точки равновесия Нэша был разработан алгоритм для распределенной задачи схемы выгрузки.

Стремясь минимизировать накладные расходы для MD в MEC, авторы в [79] разработали динамическую структуру выгрузки, учитывающую локальные накладные расходы в MD, а также вычислительные и коммуникационные ограничения в сети. Они сформулировали проблему как задачу классификации с несколькими метками и предложили глубокое контролируемое обучение для поиска оптимального решения.

Многопользовательская сеть MEC, поддерживаемая беспроводной передачей энергии (WPT), была рассмотрена в [80], где каждое энергосберегающее беспроводное устройство (WD) принимает бинарную стратегию выгрузки вычислений. Авторы провели максимизацию взвешенной суммарной скорости выгрузки вычислений всех устройств путем совместной оптимизации выбора режима вычислений по отдельности и распределения времени передачи в системе (на WPT и вычислении выгрузки).

Методы выбора ячеек в сетях HetNet

Проблема выбора ячеек или объединения пользователей была достаточно подробно изучена для сетей HetNet. Совместная активация и выбор ячеек (CAS, cell activation and selection) предложена в [81] для оптимизации энергоэффективности сети при долгосрочных ограничениях скорости пользователя. Сформулированная проблема оптимизации трудноразрешима. Поэтому авторы предложили трехслойный итерационный алгоритм для решения этой задачи. Первый слой проверяет параметр энергоэффективности, используя метод бисекции; второй слой поочередно оптимизирует показатели CAS; третий слой решает задачи (CS , cell selection) и активации клеток (CA, cell activation), используя двойное разложение и итерацию с фиксированной точкой, соответственно.

Исследование в [82] было посвящено проблемам задержки и надежности мобильных транзитных сетей и проверке их влияния на сети LTE-A HetNets. В частности, был предложен алгоритм выбора ячеек с учетом транзита для сетей LTE-A HetNets с улучшенной беспровдной оптической связью (FiWi). В статье рассматриваются факторы ограничения производительности современных волоконно-оптических транзитных инфраструктур и предлагаются различные решения. На основе заданных условий транзита в отношении задержки и надежности представлен алгоритм распределенной балансировки нагрузки для объединения пользователей в FiWi-LTE HetNets. Представленный алгоритм оценивается и анализируется численно путем сравнения его производительности с современными альтернативными подходами в отношении вероятности блокировки, средней задержки, процента прерывания обслуживания и средней достижимой пропускной способности.

В [83] была предложена модель нисходящего трафика для отдельной мобильной станции. Исследование было направлено на оптимизацию задержки пакетов в масштабах всей сети путем изучения оптимальной схемы выбора ячеек

и алгоритма распределения идентичных ресурсов. Были предложены стратегии объединения пользователей с учетом QoS с различными характеристиками, которые легко реализовать в распределенном режиме.

В [84] рассматривается совместная оптимизационная задача выбора ячеек, распределения подканалов и распределения мощности для передачи по нисходящей линии связи в многосотовых ортогональных сетях HetNet с ортогональным частотным разделением множественного доступа. Для решения этой сформулированной задачи оптимизации авторы преобразуют задачу в две подзадачи, т.е. выбор ячеек и распределение подканалов для фиксированного распределения мощности и распределение мощности для фиксированного выбора ячеек и распределения подканалов. Получено локально оптимальное решение для совместной исходной задачи путем поочередного решения двух подпроблем. Для первой подпроблемы получено глобально оптимальное решение, основанное на теории графов. Также получено оптимальное решение Каруша-Куна-Таккера с помощью алгоритма низкой сложности на основе метода аппроксимации вариации двух выпуклых функций оптимизации. Кроме того, рассматриваются сценарии пропорциональной справедливости и многоантенного приемника.

1.6 Анализ методов выгрузки трафика в системе многоуровневых граничных вычислений

В последнее время мобильные устройства Интернета вещей (IoT) и беспроводные датчики становятся все более популярными и играют все более важную роль во всех аспектах повседневной жизни. Кроме того, ожидается, что к 2025 году количество объектов с датчиками, подключенных к сети, превысит 75 миллиардов, как показано на рис. 1.6. При этом мировой рынок IoT уже вырос до 14,4 трлн долларов в 2022 году [85,86]. Более того, благодаря наличию стабильного и высокоскоростного интернета появилось много новых технологий

и приложений в виртуальной и дополненной реальности, видеоиграх, распознавании лиц, электронном здравоохранении, социальных сетях, транспорте и обработке естественного языка [87]. Многие из этих приложений требуют экспоненциального роста скорости передачи данных и вычислений за пределами возможностей встроенных мобильных устройств IoT. Несмотря на улучшение вычислительной мощности, набора функций и сенсорных возможностей, эти устройства по-прежнему обладают ограниченной вычислительной мощностью и энергией, в то время как большинство приложений и услуг считаются требующими больших вычислительных ресурсов [88,89].

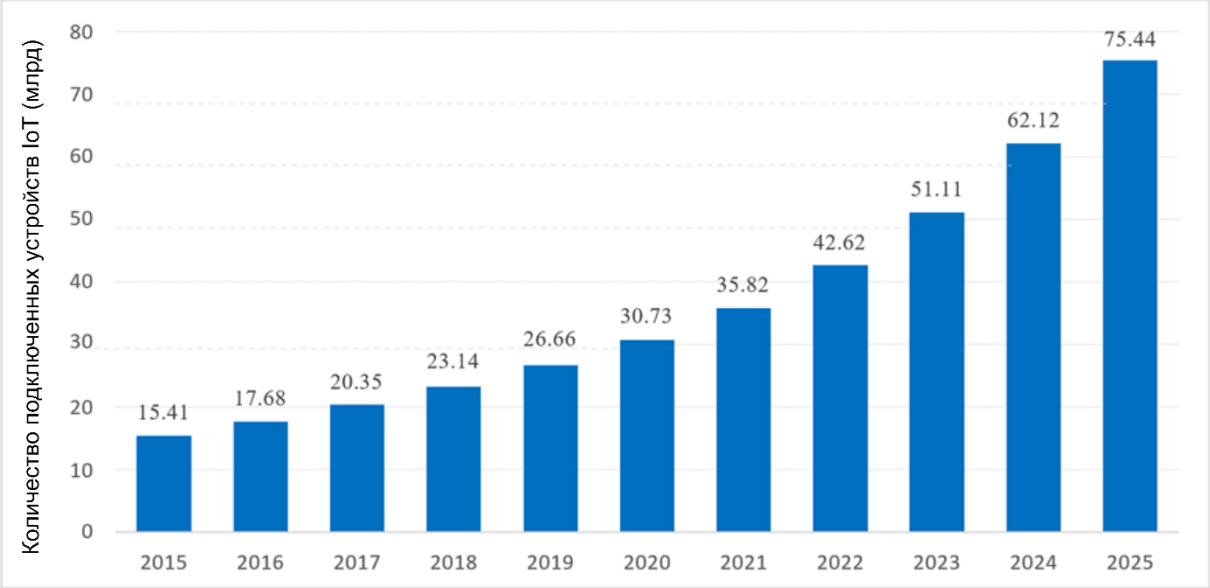


Рисунок 1.6.1 – Статистика пользователей мобильных IoT-устройств [90]

В целом, для экономии энергии и продления срока службы аккумуляторов этих устройств необходимы серьезные изменения аппаратного и программного обеспечения, которые можно разделить на четыре основных подхода [91]:

• Внедрение нового поколения полупроводниковых технологий: несмотря на то, что транзисторы меньшего размера и потребляют меньше энергии, для

оптимальной работы батареи требуется больше транзисторов. В результате спрос на электроэнергию возрастает;

• Уменьшение затрат энергии: в этом методе компоненты устройства переходят в спящий режим для экономии энергии, например затемнение монитора, когда он не используется;

• Медленное выполнение программ: когда тактовая частота процессора удваивается, энергопотребление почти удваивается. Если половина уменьшает тактовую частоту, время выполнения удваивается, но потребляется только четверть энергии;

• Выгрузка вычислений: мобильная система не выполняет интенсивных вычислений; вместо этого трафик будет выгружен и задачи удаленно выполнены на многофункциональном ресурсе, таком как облако или граничные ресурсы; тем самым продлевая срок службы батареи мобильной системы, как показано на рис. 1.7.

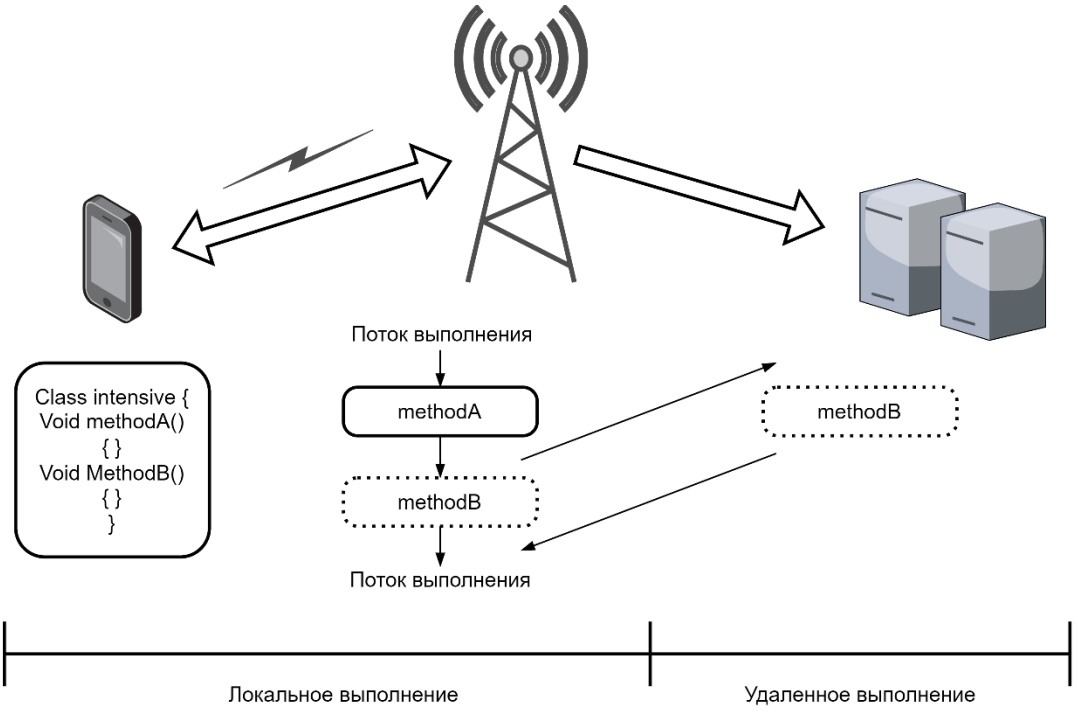


Рисунок 1.6.2 – Вычислительная выгрузка

Мобильные облачные вычисления считаются одним из доступных достаточно мощных ресурсов, которые могут уменьшить ограничения устройств IoT с помощью концепции выгрузки вычислений, при которой интенсивные вычислительные задачи выгружаются и выполняются в централизованном облаке по беспроводной сети [92,93]. Несмотря на это, основной проблемой, которая делает мобильные облачные вычисления неприемлемым решением для чувствительных к задержкам приложений и приложений реального времени, является высокая задержка [94]. Кроме того, мобильные облачные устройства подвержены различным угрозам безопасности на разных этапах передачи данных [95].

Чтобы решить эти проблемы, в исследовательских работах было установлено, что использование ресурсов и услуг, наиболее близких к мобильным сетям IoT, является экономически эффективным решением с малой задержкой. В результате возникла новая парадигма вычислений, известная как граничные вычисления [96,97]. Во-первых, в 2009 году была разработана концепция облачных сервисов, которые могут предоставлять облачные услуги IoT-устройствам в пределах радиодиапазона своей точки доступа Wi-Fi [98]. Однако такой сервис плохо масштабируется для крупномасштабных сетей IoT. Кроме того, радиодиапазон точки доступа Wi-Fi ограничен, что не может поддерживать мобильность пользователей. Во-вторых, специальное облако считается еще одним вариантом, в котором вычислительная мощность используется для локальной обработки приложений с интенсивными вычислениями [99,100]. Однако поиск подходящего мобильного компьютера в непосредственной близости при обеспечении отправки обработанных данных обратно в источник устройства является одним из ограничений концепции ad-hoc. Кроме того, отсутствует контроль каналов связи для обеспечения

надежности вычислений. При этом решение вопросов, связанных с безопасностью и конфиденциальностью, не гарантируются.

Большая задержка, накладные расходы на трафик, проблемы безопасности и конфиденциальности считаются наиболее существенными недостатками с точки зрения мобильных сетей IoT. Таким образом, перемещение возможностей облачных вычислений на граничные узлы (базовые станции), которые находятся ближе к мобильным устройствам и связаны с облачными серверами через базовую сеть, является наиболее подходящим решением для мобильных сетей IoT. Эту технологию назвали мобильными граничными облачными вычислениями [101,102], что проиллюстрировано на рис. 1-3. В представленной диссертации решается научная проблема разработки и исследования комплекса моделей и методов интеграции граничных и туманных вычислений в сетях связи пятого и шестого поколений для глобального фрагмента Воздух-Земля концепции SAGSIN, что позволяет в том числе решить задачи, связанные с выгрузкой вычислений в архитектуре граничных вычислений для мобильных сетей IoT, высокоплотных и сверхплотных, и гарантировать производительность мобильных устройств и их приложений с точки зрения задержки и энергопотребления.

Мотивация и область применения

Мобильные граничные вычисления стали новой вычислительной парадигмой и ключевой технологией, позволяющей реализовать концепции IoT, сетей связи пятого и последующих поколений. Кроме того, текущее исследование находится на стыке мобильных вычислений и беспроводных сетей, где существование многих исследовательских возможностей привело к возникновению очень большой и продуктивной научной области. Более того, большинство исследователей как из академических кругов, так и из промышленности изучали множество вопросов, связанных с граничными

вычислениями, включая выгрузку вычислений, системное и сетевое моделирование, однопользовательское и многопользовательское распределение ресурсов, стратегию кэширования контента и управление мобильностью [104].

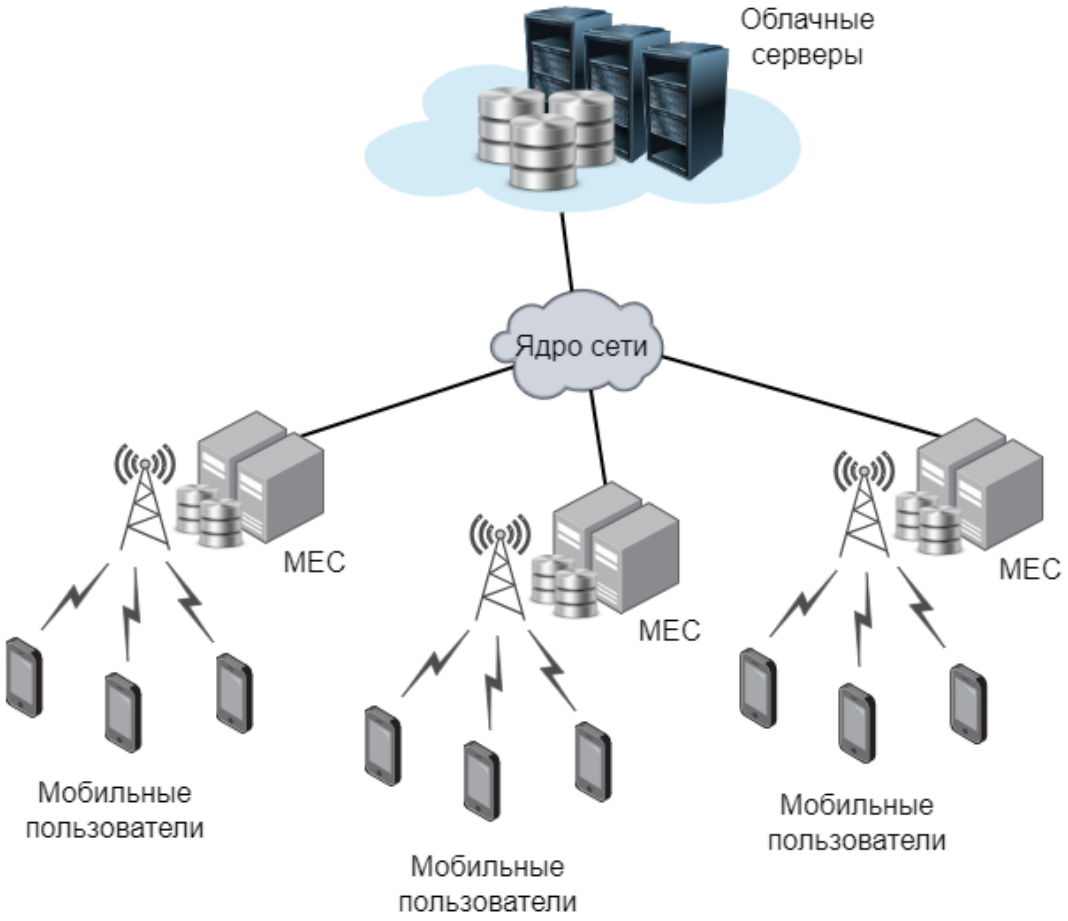


Рисунок 1.6.3 – Архитектура мобильных граничных вычислений [103].

Недавние разработки в области методов выгрузки вычислений для систем граничных вычислений усилили интерес к исследованиям, направленным на минимизацию энергопотребления, снижение системных затрат, эффективное распределение ресурсов вычислений и радиосвязи, максимизацию полезности системы и/или выполнение требований к задержке для сетей IoT [101,105]. Тем не менее, с учетом существующей работы, развертывание эффективной платформы для выгрузки вычислений в граничных вычислениях для нескольких

пользователей в изменяющихся во времени беспроводных средах по-прежнему остается сложной задачей.

В данной диссертации рассматривается развертывание эффективной платформы для выгрузки вычислений для технологии граничных вычислений для мобильных сетей IoT. Задачи при этом решаются с использованием теоретического анализа, формулирования математических моделей, разработки алгоритмов и обширного моделирования с помощью симуляторов на основе MATLAB и Python.

Анализ международной деятельности в области исследований выгрузки трафика

В последнее время были разработаны различные модели и подходы оптимизации, чтобы устранить проблемы, с которыми сталкиваются устройства IoT. Этот раздел представляет собой углубленное изучение существующих исследований моделей выгрузки вычислений и объединяет все существующие работы в иерархическую структуру, называемую систематическим обзором моделей выгрузки вычислений в граничных вычислениях, как показано на рис. 1.8. Кроме того, в следующих подразделах будет рассмотрен обзор общих моделей, основанных на этой таксономии, и связанных с ними проблем.

Цели модели

Что касается первой категории таксономии, цели модели представляют собой такие цели, которые должны быть достигнуты для принятия решения о выгрузке, а именно: минимизация задержки выполнения, минимизация энергопотребления, минимизация общих затрат, максимизация пропускной способности, минимизация использования радио, предпочтения пользователя и/или оптимизация вычислительных ресурсов. Мы можем классифицировать существующие модели на три основные категории: монообъективные, биобъективные и многоцелевые. Одноцелевая модель указывает, что

пользовательское оборудование будет переносить или выгружать вычисления для достижения только одной цели из упомянутых целей. В то время как в бицелевой модели пользовательское оборудование достигает только двух целей для применения выгрузки. И, наконец, в многокритериальной модели для принятия решения о выгрузке может быть выполнено несколько целей.

Монообъективная цель

В этом подразделе минимизация энергопотребления или уменьшение задержки выполнения на устройствах IoT является ключевой целью анализируемых публикаций.

Джанг и др. [106] предложили структуру выгрузки энергоэффективных вычислений (EECO) для многопользовательских мобильных пограничных облачных вычислений в гетерогенных сетях связи пятого поколения. При этом вычислительная задача выгрузки и распределения радиоресурсов совместно формулируются как задача оптимизации, основной целью которой является снижение потребления энергии. Кроме того, был разработан трехэтапный алгоритм EECO, чтобы справиться с системной сложностью этой проблемы и найти решение о выгрузке вычислений эффективным образом. O (max (I2 + N, IK + N)) — временная сложность этого алгоритма, где I, N и K обозначают количество итераций, количество мобильных устройств и количество доступных каналов. Экспериментальные результаты подтвердили, что предлагаемая структура снижает потребление энергии системой до 15% по сравнению с локальным исполнением. Более того, предлагаемая структура может оптимально решить выгрузку вычислительных задач для крупномасштабных мобильных устройств. Однако мобильное приложение рассматривается как единое целое, которое будет выгружено или нет; тем самым потребляется больше ресурсов из-за огромного количества данных, которые будут передаваться по сети. Кроме того, данные приложения не защищены в процессе передачи.

Та же цель [106] достигается в [107], в которой набор беспроводных сенсорных узлов объединяется для обеспечения совместных вычислений. Кроме того, предполагается, что мобильные приложения разделены на независимые задачи, которые будут назначены для выполнения одному из равноправных узлов взаимодействия, где выбор узла основан на компромиссе между справедливостью и потреблением энергии. Численные результаты показывают, что разработанная политика может снизить энергопотребление системы до 10% по сравнению с локальным исполнением. Тем не менее, основным недостатком предлагаемого метода является то, что трудно гарантировать уровень качества обслуживания и качества восприятия IoT-устройствами, поскольку достаточно сложно идентифицировать набор узлов беспроводных датчиков в непосредственной близости и гарантировать, что они доставят обработанные данные обратно в исходный узел. Кроме того, в этом исследовании рассматривается только один пользовательский сценарий для мобильного устройства.

Уменьшение задержки мобильных приложений для мобильных устройств с ограниченными ресурсами является основной целью [108] и [109]. В частности, Као и др. [108] сформулировали задачу целочисленной оптимизации, которая оптимизирует назначение вычислительной задачи, в которой совместно учитываются профиль конкретного приложения, связность каналов и доступность вычислительных ресурсов. Кроме того, для решения этой проблемы и поиска решения поставленной задачи предлагается новый подход с полностью полиномиальной аппроксимацией. При этом разработан алгоритм онлайн-обучения, гарантирующий разницу в производительности по сравнению с оптимальной стратегией. Результаты моделирования показали, что предложенный метод снижает задержку на 16% по сравнению с эвристическим подходом.

Рен и др. [109] рассмотрели совместное распределение вычислительных и коммуникационных ресурсов для многопользовательской граничной мобильной вычислительной системы на основе TDMA. При этом исследуются три различные модели выгрузки сжатия видео, а именно локальное сжатие, сжатие на граничном облаке и выгрузка с частичным сжатием, в которых формулируется задача оптимизации, целью которой является уменьшение задержки взвешенной суммы для пользователей мобильных устройств. Численные результаты доказали, что выгрузка с частичным сжатием может эффективно снизить системную задержку до 61% и 45% по сравнению с локальным и граничным сжатиями соответственно. Однако авторы в [108] не учли оптимизацию трафика данных, возникающую при выполнении задачи, а также не защитили данные приложения от атак при передаче на граничный сервер. Кроме того, в [109] не рассматриваются передовые методы беспроводной связи, которые могут повысить производительность граничных вычислительных систем.

Биобъективная цель

В этом разделе освещаются распространенные модели, которые решают только две задачи разгрузки вычислений в системе мобильных граничных вычислений.

Минимизация энергопотребления и уменьшение задержки выполнения приложений для мобильных граничных вычислительных систем считается главной целью в работах [110,111]. В [110] исследуется межуровневая оптимизация выгрузки вычислений и мощности передачи на физическом и прикладном уровнях соответственно. Кроме того, для уменьшения сложности выгрузки вычислений предлагается эффективная структура передачи сообщений, основанная на топологической структуре приложения, которая может применяться как к традиционной, так и к параллельной реализации

обработки и взаимодействия. Энергопотребление при ограничениях буфера сформулировано как задача оптимизации в [111]. Там разрабатывается онлайн-алгоритм, основанный на оптимизации Ляпунова, который оптимально решает, как распределить мощность передачи и полосу пропускания для выгрузки задач пользователей мобильных устройств на уровень мобильных граничных вычислений. Результаты моделирования показали, что потребление энергии и задержку выполнения можно минимизировать, применяя выгрузку вычислений на уровне мобильных периферийных вычислений. Тем не менее, в [110] данные приложений не защищают от кибератак при передаче. Кроме того, помехи при передаче между мобильными пользователями не рассматриваются в [111], где они сильно влияют на производительность системы. При этом требование о времени выполнения вычислительных задач не предъявляется.

Многоцелевые модели

В этом разделе рассмотрим статьи, посвященные выгрузке многоцелевых вычислений для мобильных граничных облачных вычислений.

Лю и др. [112] оптимизировали совместно мощность беспроводной передачи и вычисление вероятности выгрузки путем формулирования задачи оптимизации модели организации очередей, целью которой является минимизация потребления энергии, задержки выполнения и ценовых затрат для многопользовательских мобильных граничных облачных вычислений. Для получения оптимального решения применяются метод скаляризации (масштабирования) и метод внутренних точек. Основное ограничение данного исследования заключается в том, что требования разнообразия вычислительных задач в стратегии управления ресурсами не учитываются, а оно является более значимым и эффективным в мобильной граничной вычислительной системе.

Другая идея, касающаяся выгрузки многокритериальных вычислений, представлена в [113,114].

По сравнению с предыдущим исследованием Сюй и др. [113] предложили многоцелевую модель вычислений для Интернета транспортного средства в граничных вычислениях, минимизируя потребление энергии, уменьшая задержку выполнения задач и уменьшая скорость балансировки нагрузки для граничных вычислительных устройств, что является основной целью этой модели. При этом разработан алгоритм связи между транспортными средствами, который предназначен для поиска маршрута выгрузки для вычислительных задач. Кроме того, для решения задачи оптимизации и получения решения используется генетический алгоритм. Для оценки сгенерированного решения используются простые аддитивные методы взвешивания и принятия решений по множеству критериев.

В [114] Чжан и др. исследовали многоцелевое распределение ресурсов для многопользовательских мобильных граничных вычислительных систем с множественным доступом с ортогональным частотным разделением, в которых время и энергия объединены в системную полезность, которая будет максимизирована посредством формулировки задачи оптимизации. На основе модифицированного метода Ньютона и концепции приоритета выгрузки вычислений разработан несложный алгоритм ранжирования для получения решения, близкого к оптимальному. Результаты моделирования доказали, что предложенный алгоритм может обеспечить почти оптимальную производительность. Однако авторы в [114] не рассматривали вычислительные ресурсы и распределение радиоресурсов вместе, что может привести к значительной трате другого ресурса [115].

Природа выгрузки

После рассмотрения целей, которые влияют на решение о выгрузке, характер выгрузки может указать, какая часть приложений должна быть выгружена для удаленного выполнения на пограничном сервере. В соответствии

с предыдущими исследованиями распространенные модели можно разделить на модели с полной или частичной выгрузкой приложений. Полная выгрузка приложения означает, что устройство IoT выгрузит все приложение на удаленные серверные узлы. В то время как разделение приложений указывает на то, что только интенсивные компоненты отделяются и выгружаются на удаленный узел пограничного сервера для работы во время выполнения.

Полная выгрузка

В этом подразделе освещаются распространенные модели, в которых задачи приложения рассматриваются как отдельные компоненты, которые будут обрабатываться локально или будут выгружаться и обрабатываться удаленно.

Процесс принятия решений по Маркову был использован для планирования задач для систем MEC в [116], где состояние очереди буфера задач, состояние блока передачи и состояние выполнения локального блока обработки исследуются для определения вычислительной мощности и планирования выгрузки задач. Кроме того, задержка и энергопотребление вычислительной задачи формулируются как задача, целью которой является уменьшение задержки. После этого разрабатывается эффективный алгоритм одномерного поиска для решения этой проблемы и получения оптимальной политики планирования задач. Наконец, результаты моделирования показали, что предложенный алгоритм может обеспечить более низкую задержку выполнения по сравнению с другими эталонными тестами. Однако основным недостатком этой работы считается отсутствие защиты передаваемых данных мобильных приложений от кибератак.

В других работах [106] и [117] предложена энергоэффективная модель выгрузки вычислений для многопользовательской системы MEC. В частности, Чжан и др. [106] предложили новый энергоэффективный механизм для MEC в гетерогенных сетях связи пятого и последующих поколений. Кроме того, они

формулируют стоимость энергозатрат на вычислительную задачу и передачу файлов как задачу снижения энергопотребления системы при ограничении задержки. Затем разрабатывается трехэтапная схема выгрузки для решения этой задачи за приемлемое время и получения оптимального решения. Численные результаты показали, что предложенный метод выгрузки может снизить потребление энергии до 15% по сравнению с другими базовыми вариантами.

Чен и др. [117]. изучали проблему выгрузки MEC в условиях многоканальных беспроводных помех. Затем они приняли решение о выгрузке задач на основе теоретико-игрового метода, в котором равновесие Нэша может быть достигнуто с помощью алгоритма распределенной выгрузки. Численные результаты показали, что предложенный алгоритм не только улучшает производительность выгрузки, но и хорошо масштабируется для большого числа мобильных пользователей.

Недавно Алам и др. [118] предложили автономный метод выгрузки для систем MEC, в котором проблема распределения ресурсов моделируется как процесс Маркова при принятии решений, а основной целью является минимизация задержки сервисных вычислений. Затем для поиска оптимального решения разрабатывается основанный на глубоком Q-обучении алгоритм. Экспериментальные результаты показывают, что предложенный метод и алгоритм могут повысить производительность системы в отношении времени выполнения и энергопотребления.

Частичная выгрузка

В этом подразделе освещаются общие модели, которые частично делят вычислительные задачи в приложении на две основные части, некоторые из них будут выполняться локально на устройстве IoT, а другая часть будет передаваться и обрабатываться удаленно.

В [119] выгрузка вычислений и планирование компонентов оптимизированы совместно с помощью предлагаемого подхода для многокомпонентных мобильных приложений. Затем формулируется задача линейной оптимизации с функцией полезности, которая может найти компромисс между экономией энергии и задержкой соединений, порядком приоритета компонентов и приложением времени выполнения. После этого авторы используют измерения реальных данных для решения этой проблемы. Результаты показали, что предлагаемый подход может снизить общие накладные расходы.

В [120] скорость вычислений, коэффициент загрузки и мощность передачи совместно оптимизируются с помощью невыпуклой формулировки задачи с целью минимизации энергопотребления и задержки выполнения приложений для устройств IoT. Затем разрабатывается локально оптимальный алгоритм для эффективного получения решения этой задачи. Результаты моделирования подтвердили, что предложенный алгоритм может значительно снизить потребление энергии и задержку по сравнению с локальным подходом. Однако общим недостатком [119] и [120] является то, что данные IoT-приложений подвержены различного рода атакам в процессе передачи на сервер.

В следующих работах также рассматривали минимизацию задержки выполнения задачи [121,122]. В [121] рассматривается сценарий выгрузки в Device-to-Device (D2D) для MEC, в котором управление помехами и частичная выгрузка интегрированы с использованием метода множественного доступа с ортогональным частотным разделением. При этом потребление энергии, распределение ресурсов и частичная выгрузка совместно рассматриваются посредством формулировки задачи нелинейного программирования. После этого разрабатывается эффективный и новый алгоритм для решения этой проблемы. Результаты оценки подтвердили, что предлагаемое решение может

уменьшить задержку выполнения по сравнению с локальным выполнением, решениями со случайной и полной выгрузкой.

В [122] Ван и др. сформулировали приложение частичной выгрузки для системы с несколькими транспортными средствами как задачу оптимизации. Затем для решения этой проблемы и получения оптимального действия предлагается новый алгоритм. Результаты моделирования доказали, что предложенный алгоритм снижает задержку выполнения по сравнению с локальным выполнением и решениями с Deep-Q-Network.

Число пользователей

В предложенной таксономии в зависимости от среды MEC, модели выгрузки вычислений можно разделить на три основные категории: число пользователей или устройств IoT, количество узлов пограничных серверов и количество уровней. В этом подразделе рассматриваются общие модели, касающиеся числа пользователей или устройств IoT, которые подразделяются на два подкласса, называемые однопользовательскими и многопользовательскими. При этом две другие категории (т. е. количество узлов пограничных серверов и количество уровней) будут подробно рассмотрены в следующих подразделах.

Один пользователь

Для выгрузки однопользовательских вычислений Mao et al. предложили подход к выгрузке для системы MEC в [123] и [124]. В частности, в [123] исследовалась зеленая система MEC с устройствами сбора энергии, в которых задержка выполнения и невыполнение задачи совместно оптимизируются посредством постановки задачи. Затем разрабатывается алгоритм на основе оптимизации Ляпунова для решения задачи и получения решения о выгрузке с использованием значений мощности передачи и частотах процессорного цикла без запроса информации о запросе задачи, процессе сбора энергии и беспроводном канале. Результаты моделирования подтвердили, что

предложенный алгоритм может сократить время выполнения за счет разумного развертывания стратегии выгрузки вычислений. В [124] планирование задач и распределение мощности передачи совместно оптимизируются посредством выпуклой формулировки задачи с целью уменьшения взвешенной суммы задержки выполнения и потребления энергии. Результаты моделирования показали, что планирование задач позволяет добиться меньшей задержки по сравнению с традиционными подходами.

В [125] представлена стратегия выгрузки на основе подробной детализации, в которой минимизация энергопотребления с соблюдением ограничения по задержке является основной целью. Кроме того, авторы предложили алгоритм оптимизации бинарного роя частиц для решения задачи и получения решения о выгрузке задачи. Результаты моделирования показывают, что предложенный алгоритм может снизить энергопотребление устройств IoT до 25% по сравнению с подходом локального исполнения.

Недавно в [126] была предложена эффективная схема выгрузки, в которой облачные и граничные вычислительные ресурсы совместно взаимодействовали для IoT. Конкурентоспособные по вычислительным ресурсам IoT-устройства также рассматриваются с целью уменьшения задержки выполнения. Затем разрабатывается итеративный и эвристический алгоритм для эффективного решения этой проблемы. Наконец, результаты моделирования показывают, что предложенный алгоритм может уменьшить задержку выполнения по сравнению с другими базовыми алгоритмами. Однако ограничение на потребление энергии в данном исследовании не рассматривается.

Несколько пользователей

Что касается многопользовательской выгрузки вычислений, в [127] представлен подход к выгрузке с учетом энергопотребления для устройств IoT в гетерогенных сетях, в котором вычислительные ресурсы, энергопотребление,

состояния канала и требования к задержке совместно оптимизируются посредством постановки задачи. После этого исходная проблема разбивается на две подзадачи, а затем предлагается итеративная структура для получения решения по распределению мощности и эффективной выгрузке. Результаты моделирования подтвердили, что предложенный подход может найти близкое к оптимальному решение.

В [128] проблема выгрузки многопользовательских вычислений формулируется как задача разбиения графа, и для решения этой проблемы предлагается решение, основанное на спектральной кластеризации. Экспериментальные результаты показывают, что разработанный подход может минимизировать потребление энергии на передачу в процессе выгрузки.

Минимизация энергопотребления и ограничения по задержке были рассмотрены в [129]. В [129] задачи выгрузки и распределения пропускной способности совместно сформулированы как задача оптимизации для многопользовательской системы MEC, в которой минимизация общей стоимости задержки и энергии является основной целью. Для решения этой проблемы и получения решения, близкого к оптимальному, разработан алгоритм на основе Deep-Q-Network. Численные результаты показали, что разработанный алгоритм способен снизить общую стоимость по отношению к сценарию локального исполнения.

Количество граничных узлов

Что касается количества узлов граничного сервера, модели выгрузки вычислений делятся на две основные категории: одиночный узел и несколько узлов. В одиночном узле вычислительные задачи приложения IoT могут быть выгружены только на один узел граничного сервера. Принимая во внимание, что в среде с несколькими узлами существует набор узлов граничных серверов, в

которых задачи приложения могут быть выгружены и выполнены. В этом разделе модели выделены в соответствии с этими двумя категориями.

Одиночный узел

Для среды с одним узлом Чжао и др. [130] максимизировали количество приложений IoT, которые обслуживаются граничными вычислительными узлами, при этом удовлетворяются требования к задержке для этих приложений. В частности, авторы предложили новую модель совместного планирования, в которой данные и код приложения отправляются планировщику (т. е. основному компоненту этой модели). Затем планировщик решает, будет ли выгруженное приложение выполняться на виртуальной машине MEC или оно будет делегировано в удаленное интернет-облако. Это решение зависит от двух ограничений: доступности вычислительных ресурсов на граничном вычислительном узле и приоритетов приложений (приложение с малой задержкой имеет более высокий приоритет). Что касается предыдущего ограничения, если имеется свободный узел виртуальной машины с достаточными ресурсами, приложение будет выгружено и выполнено на этом узле, а затем результат будет доставлен обратно на устройство IoT. Для второго ограничения, когда мощность граничных вычислений не является достаточной, планировщик отправляет программу в удаленное интернет-облако. Далее в работе представлена политика сотрудничества на основе приоритетов, в которой для каждого уровня приоритета определены различные пороги буфера. В случае полностью загруженного буфера приложения отправляются в интернет-облако. Используется также рекурсивный алгоритм низкой сложности для определения локального оптимального порога. Авторы сравнили предложенное решение с тремя другими политиками, которые касаются только MEC стратегии сотрудничества на основе принципа «первым пришел, первым обслужен» и небуферной стратегии. Предлагаемая стратегия может повысить вероятность

завершения приложения в пределах допустимой задержки. Тем не менее, обращение с приложением как с единым блоком считается одним из основных недостатков этой модели. Считается, что лучше разделить приложение на задачи и выгружать только интенсивные задания.

Более того, максимальное использование виртуальных машин на серверах MEC, для которых перегрузка базовой станции eNB, задержка в сети и стоимость миграции виртуальных машин выступают в качестве ограничений, является основной целью [131], в которой авторы сформулировали проблему распределения виртуальных машин как марковский процесс принятия решений, а затем предложили алгоритм для вычисления оптимальной политики выделения подходящей виртуальной машины каждому устройству IoT. Результаты доказали, что при увеличении стоимости миграции ВМ лучше выделять ее локально на обслуживающей базовой станции eNB.

В [132] основными целями считаются сокращение задержки выполнения для выгруженных приложений и минимизация общего энергопотребления MEC. Авторы предположили, что каждый сервер MEC подключен только к одной из ближайших eNB. Что касается результатов моделирования, предлагаемая индексная политика является более затратной, чем оптимальная политика, на 6,5% в худшем случае с точки зрения стоимости системы. Тем не менее, как в [131], так и в [132] авторы не рассматривали большее количество вычислительных узлов для каждого приложения для дальнейшего уменьшения задержки выполнения. Кроме того, в [131] потребление энергии не учитывается в модели, а в [132] не рассматривается интерференционное ограничение.

В работах [133] и [134] представлено решение по выгрузке для систем MEC, в которых минимизация общего энергопотребления с учетом требований к задержке выполнения задач является основной целью. В частности, в [133] предложена модель выгрузки вычислений для многопользовательской системы

MEC, а затем разработан алгоритм, который может распределять вычислительные ресурсы и полосу пропускания системы для устройств IoT. Далее этот алгоритм был эффективно аппроксимирован дискретизацией по энергии для уменьшения его сложности и получил решение, близкое к оптимальному. Результаты моделирования доказали, что предложенная модель может снизить потребление энергии по сравнению с подходом локального исполнения. В [134] коммуникационные и вычислительные ресурсы совместно оптимизируются посредством невыпуклой формулировки задачи. Затем эта проблема разбивается на две подзадачи для эффективного получения оптимального решения. И в работе [133], и в работе [134] были свои недостатки, связанные с проблемами защиты передаваемых данных от разных типов атак.

Несколько узлов

Что касается среды с несколькими узлами, Уис и др. в работе [135] рассмотрели эффективность взаимодействия eNB в отношении характеристик кластера с точки зрения задержки выполнения и ограничений по энергопотреблению и предложили три различные стратегии оптимизации процесса кластеризации. Авторы использовали одноэтажное здание с квартирами 10 x 10 м в сетке 5 x 5 для оценки этих стратегий, где предполагается, что каждая квартира оборудована базовой станцией eNB. Затем эти eNB классифицируются на три группы: обслуживающие eNB, расположенные в центре сетки, eNB , которые отделены от обслуживающего eNB не более чем двумя стенами, и, наконец, удаленные eNB. В первой стратегии формируется кластер eNB, чтобы уменьшить общую задержку. Поэтому все активные узлы eNB должны участвовать в вычислительном кластере. Тем не менее, эта стратегия обеспечивает снижение задержки при использовании кластеризации до 22%, но, к сожалению, не учитывает энергопотребление в качестве ограничения. Вторая стратегия решала эту проблему, принимая потребляемую

мощность кластера в качестве ограничений. Вторая стратегия обеспечивает снижение энергопотребления до 61%, но, к сожалению, это может быть привести к увеличению потребления энергии для некоторых узлов eNB, чем для других. В результате последняя стратегия преодолевает эту проблему путем равномерного распределения потребляемой мощности по узлам eNB, и если есть узел eNB, у которого потребляемая мощность выше, чем у других, распределение нагрузки может быть изменено, если это возможно.

В [136] Уис и др. расширили свою работу, рассмотрев многопользовательский случай, в котором минимизация энергопотребления кластера и гарантия требуемой задержки для каждого устройства IoT является основной целью. Авторы использовали ту же модель сети и предложили метод оптимизации, при котором кластер eNB является более масштабируемым в зависимости от запроса приложения устройства IoT и его требований. Основная цель этого исследования — совместная оптимизация кластеров для всех активных запросов устройств IoT одновременно, чтобы эффективно распределять вычислительные и коммуникационные ресурсы между устройствами IoT и улучшать качество взаимодействия. Результаты этой модели сравниваются с другими тремя различными сценариями, которые включают отсутствие кластеризации, статическую кластеризацию и последовательную оптимизацию кластеров. Что касается коэффициента удовлетворенности пользователей, предлагаемая модель может обеспечить удовлетворение до 94% устройств IoT, что лучше, чем в других сценариях. Но среднее энергопотребление на одно устройство IoT значительно выше, чем в сценариях без кластеризации и последовательной оптимизации кластеров. Кроме того, среднее значение задержки лучше в последовательных сценариях оптимизации кластеров, чем в предлагаемом.

Танзил и др. в [137] предложили новую модель, которая максимально использует локальные вычислительные ресурсы MEC при минимальном использовании удаленного облака. Эта модель формирует кластер ресурсов соседей eNB на основе канонического подхода к совместной игре, в то время как эти eNB могут делиться своими ресурсами друг с другом. Каждый узел eNB может получать вознаграждение, если он обрабатывает вычисления для устройств IoT, которые подключены к другому eNB в этом кластере. Эта коалиция сохраняется в течение определенного периода времени, после чего могут формироваться новые коалиции. Авторы сформулировали задачу оптимизации с функцией полезности для решения проблемы совместного использования ресурсов eNB. Эта модель сравнивается с изолированными eNB (где нет взаимодействия между eNB и каждый узел работает индивидуально) и с большим фемто-облаком (где все eNB в системе участвуют в вычислениях). Результаты показывают, что предложенный подход лучше, чем другие сценарии, в отношении сокращения задержки выполнения задач. Недостатком работы [137] можно считать то, что в предложенном методе необходимо образование новых коалиций.

В [138] предложен метод размещения онлайн-приложений для среды MEC, в которой приложение устройства IoT и его физическая вычислительная система моделируются в виде графов, а вычислительные ресурсы (т. е. доступность и потребности) аннотируются. Затем разрабатывается точный алгоритм для линейного назначения графа приложения дереву физического графа, в котором совместно рассматривается назначение узла и связи, а также включение нескольких типов вычислительных ресурсов в узлах. Далее этот алгоритм эффективно аппроксимируется с полиномиально-логарифмическим коэффициентом конкуренции для размещения дерева графа приложений. Теоретическая оценка показала, что предложенный алгоритм в среднем может

достигать практически хороших результатов. Однако передаваемые данные приложения недостаточно защищены от различных типов атак во время передачи.

В [139] была представлена модель компромисса между вычислениями и коммуникациями для нескольких граничных вычислительных сетей, где вычислительные задачи реплицируются на несколько граничных узлов, а затем копируются на несколько копий, что может ускорить фазу выгрузки с использованием большего числа ресурсов. Численные результаты показали, что время выгрузки уменьшается пропорционально и линейно в бинарном и частичном случаях выгрузки соответственно. Однако в этом исследовании не рассматривалась проблема балансировки нагрузки между несколькими граничными узлами, которая оказывает большое влияние на снижение задержки и энергопотребления устройств IoT.

Количество уровней

В зависимости от количества уровней в MEC модели загрузки вычислений можно разделить на две основные категории: одноуровневые и многоуровневые. В следующих подразделах общие модели, касающиеся количества уровней, будут подробно рассмотрены.

Одноуровневый

Рассматривая модели выгрузки одноуровневых вычислений, Би и др. [140] предложили модель выгрузки для многопользовательской системы MEC, в которой все вычислительные задачи рассматриваются как единое целое, которое будет выполняться локально или удаленно. Эта модель совместно оптимизировала выбор режима отдельных вычислений и распределение времени передачи с целью максимизации скорости вычислений для всех устройств IoT. Результаты моделирования доказали, что предложенная модель может значительно превзойти сценарии локального выполнения и полной выгрузки.

Однако рассмотрение вычислительных задач как единого объекта, подлежащего выгрузке или нет, является основным недостатком данной работы, при которой по сети будут передаваться огромные объемы данных. Более того, данные приложения не защищены в процессе передачи.

Алгоритмы глубокого обучения с подкреплением используются в [141] и [142] для выбора модели выгрузки вычислений для многопользовательской системы MEC. В [141] распределение пропускной способности и решение о выгрузке совместно оптимизируются посредством модели, целью которой является уменьшение общей задержки выполнения задач и соответствующей энергии. В работе разработан распределенный алгоритм, основанный на глубоком обучении, для преодоления проблемы масштабирования для этой задачи и для эффективного получения решения о выгрузке. Численные результаты подтвердили, что предложенный алгоритм может сэкономить до 39,4 % и 21 % от общей стоимости по сравнению со сценариями локальной и периферийной обработки. В [142] Хван и др. сформулировали интегрированную модель для распределения беспроводных ресурсов и задачи выгрузки, целью которой является максимизация взвешенной суммы скорости вычислений. Был разработан онлайн-алгоритм, основанный на глубоком изучении, для эффективного получения решения, близкого к оптимальному. Общим недостатком обоих рассмотренных выше подходов (то есть [141] и [142]) является то, что данные приложений устройств IoT уязвимы для различных типов атак в процессе передачи.

В [143] предложены оффлайн и онлайн решения по оптимизации для одного пользователя с одной многоантенной системой MEC, в которых задачи выгрузки и распределения энергии объединены с целью минимизации потребления энергии при передаче. Во-первых, для получения оптимального решения используются априорные знания о задаче и информация о состоянии

канала. Затем это оптимальное решение используется для дальнейшего использования эвристического и онлайн-подхода, который может получить близкое к оптимальному решение для статических и изменяющихся во времени сценариев каналов. Численные результаты показали, что предложенное решение может значительно снизить потребление энергии по сравнению с другими эталонными решениями.

Многоуровневый

Рассматривая многоуровневые модели с выгрузкой вычислений, Мэн-Хис и др. [144] оптимизировали совместно вычислительные и коммуникационные ресурсы и решение о выгрузке для многопользовательской системы MEC с помощью постановки задачи, целью которой является снижение общей нагрузки на систему с точки зрения энергии, вычислений и задержки. При этом использовались полуопределенное ослабление, попеременная оптимизация и последовательная настройка для решения этой задачи и получения локального оптимального решения с помощью трехэтапного алгоритма. Численные результаты предлагаемой модели показали, что можно сэкономить общие затраты по сравнению с подходом локального исполнения.

В других работах [145] и [127] представлены решения по выгрузке задач для мобильных систем облачных вычислений, чтобы эффективно минимизировать общее энергопотребление устройств IoT. В частности, в [145] предложена модель граничных вычислений с помощью облачных вычислений и сформулирована задача целочисленной оптимизации, в которой минимизация энергопотребления с учетом крайнего срока завершения задачи является основной целью. В работе предложен полуопределенный алгоритм ослабления и стохастического отображения для решения задачи. Результаты моделирования доказали, что предложенный алгоритм может снизить энергопотребление по отношению к сценариям локальной и облачной обработки.

Ли и др. в [127] сформулировали модель выгрузки с учетом энергопотребления для многопользовательской среды, в которой неоднородность вычислительных ресурсов, требования к задержке, энергопотребление устройств IoT и состояния канала совместно учитываются. Проблема решается итеративно на основе распределения мощности передачи и политик для выгрузки задач. Результаты моделирования подтвердили, что предложенная структура конкурентоспособна по отношению к оптимальному решению. Тем не менее, в этих исследованиях не рассматривается балансировка нагрузки между узлами граничных серверов, что играет решающую роль для уменьшения задержки и энергопотребления устройств IoT.

В [146] представлены многосерверные, многопользовательские, многозадачные модели выгрузки вычислений для систем граничных вычислений. В [146] Хван и др. сформулировали различные политики задач в реальном времени, используя при этом для решения задач методы оптимизации и алгоритмы ослабления и глубокого обучения, которые позволяют получить близкое к оптимальному решение о выгрузке. Численные результаты показали, что предложенный алгоритм может получить близкое к оптимальному решение менее чем за 1 миллисекунду.

1.4.6 Проблема мобильности

Категория проблем мобильности в предложенной таксономии связана с управлением вычислительными задачами, выгружаемыми посредством перемещения устройств IoT между различными граничными узлами. В общих моделях выгрузки используются три разных подхода к решению проблемы мобильности: управление питанием, миграция виртуальных машин и выбор пути. В подходе управления питанием мощность передачи узла пограничного сервера адаптируется во время выполнения выгруженных задач и этот подход применим для ограниченной мобильности устройств IoT (например, устройства

IoT медленно перемещаются внутри здания). Подход миграции виртуальной машины переносит виртуальную машину, отвечающую за выполнение выгруженных задач, на другой, более подходящий вычислительный узел, и этот подход применим, когда мобильность устройств IoT не ограничена (например, устройство IoT выполняет передачу обслуживания на новый обслуживающий узел). В подходе выбора пути модель выгрузки отвечает за выбор лучшего маршрута связи между устройством IoT и вычислительным узлом, и этот метод полезен, когда необходимо переместить большой объем данных между обоими вычислительными узлами. В следующих подразделах будут подробно рассмотрены распространенные модели выгрузки в отношении мобильности.

Управление мощностью

Рассматривая подход к управлению мощностью, Max и др. [147] предложили эффективный облачный алгоритм для адаптации мощности передачи малых сот с целью максимизации количества обрабатываемых приложений в MEC при ограничении задержки. В частности, малая сота может адаптировать (т. е. увеличивать или уменьшать) мощность передачи для движения устройства IoT, используя грубые и динамические точные настройки, чтобы избежать переключения на новую соту до получения результата выгруженной задачи, если это возможно. Результаты моделирования показали, что предложенный алгоритм может успешно увеличить количество приложений, доставляемых на устройства IoT, в отличие от приложений, не поддерживающих облачные технологии. Расширение [147] дано в [148], в котором время для регулировки мощности передачи применяется индивидуально для каждого устройства IoT с учетом качества канала, что увеличивает долю доставляемых приложений. Однако основным недостатком [147] является поздняя адаптация управления мощностью, когда достаточное качество канала не гарантируется за требуемое время.

Кроме того, управление мощностью передачи и миграция виртуальных машин используются совместно для управления мобильностью устройств IoT и максимальной масштабируемости для многопользовательских многооблачных систем [149]. В частности, задержка обработки, транзитная задержка и задержка передачи формулируются в виде математической модели для MEC, затем разрабатывается эффективный эвристический алгоритм, основанный на методе оптимизации роя частиц, чтобы сбалансировать рабочую нагрузку между облаками, и повысить таким образом эффективность MEC. Экспериментальные результаты показывают, что предложенный метод не только превосходит традиционные методы по количеству обслуживаемых пользователей, но также может предоставлять масштабируемый сервис с различными типами сценариев требований.

Миграция виртуальной машины VM

Что касается подхода к миграции VM, был изучен процесс принятия решения о миграции VM для многопользовательской мобильной системы микрооблачных вычислений с несколькими экземплярами служб в [150]. Авторы предложили автономный алгоритм для поиска наилучшей конфигурации для размещения на основе априорных сведений о входящих и исходящих экземпляров окна просмотра. Затем предлагается онлайн-алгоритм для решения этой задачи в режиме реального времени. Наконец, предлагается эффективный метод оптимального определения размера окна просмотра с целью минимизации верхней границы стоимости размещения. Результаты моделирования подтвердили, что предложенные офлайн- и онлайн-алгоритмы могут минимизировать затраты в отношении политик без миграции и с миграцией соответственно. Кроме того, они применимы для более широкого класса задач динамического распределения ресурсов.

Максимизация пропускной способности системы распределенных центров обработки данных рассматривается в [151], в которой для улучшения доступа к облаку предлагается протокол на основе разделения локаторов/идентификаторов. Процесс миграции VM на новый граничный узел будет определяться исходя из требуемых и доступных вычислительных ресурсов на граничных серверных узлах, а также с заданным порогом задержки или джиттера. Моделирование и реальные оценки подтвердили, что предложенное решение может значительно увеличить пропускную способность системы. Стоит отметить, что в этих исследованиях [150,151] не рассматривается оптимизация потребляемой энергии устройств IoT во время выполнения задачи, которая считается одним из основных показателей в процессе выгрузки решений.

В работе [152] представлено эффективное решение для улучшения процесса миграции путем подхода к прогнозированию на основе мобильности. Этот подход позволил найти компромисс между качеством восприятия и общими накладными расходами, при котором запрос на услугу IoT разбивается на несколько частей, которые будут оптимально обрабатываться в наиболее подходящих микроцентрах обработки данных на основе оценки пропускной способности передачи данных между сервером и IoT-устройством, а также оценка временных окон для выполнения передачи данных IoT-устройством. Результаты моделирования подтвердили, что предложенный подход может уменьшить задержку передачи примерно на 35% по сравнению с известными. Сун и др. в [153] предложили облачную сетевую архитектуру для мобильных облачных вычислений, чтобы уменьшить задержку E2E и удовлетворить требования к качеству обслуживания. Затраты на миграцию и выгоды были совместно оптимизированы с помощью формулировки проблемы со смешанным временем, цель которой состоит в том, чтобы максимизировать общую прибыль

от процесса динамической миграции. Было использовано эвристическое решение с использованием инструмента, основанного на квадратичном программировании со смешанным временем. Результаты моделирования доказали, что предложенная архитектура может сократить задержку выполнения и затраты на миграцию по сравнению с подходом без миграции. Однако эти исследования (например, [152,153]) не касаются проблемы балансировки нагрузки между граничными вычислительными узлами.

Али и др. в [154] предложили новый энергоэффективный подход к распределению ресурсов для системы MEC. Этот подход оптимально назначает задачу устройства IoT наиболее подходящему серверу и выделяет надлежащие ресурсы для устройства IoT. Кроме того, недорогой энергоэффективный алгоритм, основанный на глубоком обучении, предназначен для решения сложной многомерной задачи распределения ресурсов, в которой совместно учитываются разная нагрузка запросов устройств IoT и их разнородный характер. Результаты моделирования показывают, что разработанный алгоритм позволяет значительно снизить энергопотребление и повысить скорость обслуживания по сравнению с другими алгоритмами.

Выбор пути

Что касается подхода к выбору пути, Зденек и др. [155] предложили алгоритм выбора пути для передачи данных выгруженной задачи между малой сотой и устройством IoT с целью минимизации потребления энергии и задержки передачи. Для достижения этой цели может быть принудительно выполнен процесс передачи обслуживания новой обслуживающей соте. Результаты моделирования подтвердили, что предложенный алгоритм может минимизировать задержку передачи до 9% по сравнению с традиционными методами доставки. Кроме того, удовлетворенность устройств IoT по задержкам увеличивается примерно на 6,5%. Расширение [155] предлоджено в [156], в

котором сложность разработанного ранее алгоритма уменьшена. При этом задержка передачи сводится к минимуму, а удовлетворенность устройства IoT по задержке увеличивается примерно на 28%. Общим недостатком обоих рассмотренных выше подходов (т. е. [155] и [156]) является то, что предложенный алгоритм не может быть использован для случая, когда расстояние между вычислительным местоположением и устройством IoT слишком велико.

Плачи и др. в [157] предложили марковский алгоритм принятия решений для управления мобильностью устройства IoT в системах MEC. В частности, выгрузка вычислительных и коммуникационных ресурсов граничного узла, а также прогноз движения IoT-устройства динамически используются для определения размещения ВМ и определения наиболее подходящего пути между IoT-устройством и граничным узлом для возврата результатов на IoT-устройства. Результаты моделирования доказывают, что предложенный алгоритм может сократить время выгрузки примерно на 10 % по сравнению с известными подходами.

Ю и др. в[158] предложили алгоритм частичной выгрузки с учетом динамической мобильности для системы MEC. В этом алгоритме выгруженные данные назначаются наиболее подходящему граничному узлу для обработки на основе прогнозирования движения IoT-устройства, в котором основной целью является минимизация потребления энергии с удовлетворением требований к задержке. Результаты моделирования показывают, что разработанный алгоритм позволяет значительно снизить энергопотребление по сравнению с традиционными подходами.

Выводы по главе 1

1. Анализ развития современных сетей связи показал, что в начале третьего десятилетия 21 века дальнейшее нескоординированное развитие мобильных и фиксированных сетей связи не способствует решению проблемы интегрирования всех ресурсов всех сетей для предоставления современных услуг всем пользователям сетей связи общего пользования и на этапе формирования подходов к реализации сетей связи шестого поколения 6G появилась новая концепция развития сетей связи, в основе которой лежит понимание необходимости интеграции не только разнообразных технологий в рамках тех или иных сетей, но и интеграции сетей связи в единую сеть. Эта концепция называется интегрированные сети Космос-Воздух-Земля-Море SAGSIN (Space-Air-Ground-Sea).
2. В связи с изложенным требуется решение научной проблемы разработки и исследования комплекса моделей и методов интеграции граничных и/или туманных вычислений в сетях связи пятого и шестого поколений для глобального фрагмента Воздух-Земля концепции SAGSIN.
3. Обеспечение эффективности интеграции глобального фрагмента сети SAGSIN может быть достигнуто только при интеграции различных технологий телекоммуникаций, обеспечивающих собственно построение как наземных и воздушных сетей. Поэтому в диссертационной работе при разработке комплекса моделей и методов интеграции граничных и/или туманных вычислений должны быть исследованы проблемы их интеграции для сетей беспилотных летательных аппаратов (БПЛА), программно-конфигурируемых сетей SDN (Software Defined Networks), сетей взаимодействия устройство-устройство D2D (Device-to-Device), сетей автомобильного транспорта VANET (Vehicular Ad Hoc Networks), беспилотных автомобилей, Интернета Вещей. Отдельное внимание при

этом, как показывает проведенный анализ, должно быть уделено задачам выгрузки трафика, что является универсальным методом распределения ресурсов для всех вышеприведенных технологий.

4. На основании проведенного анализа современных направлений развития сетей связи были сформулированы цель и задачи диссертационной работы.

ГЛАВА 2. ИНТЕГРАЛЬНОЕ РЕШЕНИЕ ПРОБЛЕМЫ РАЗМЕЩЕНИЯ КОНТРОЛЛЕРОВ В БАЛАНСИРОВКИ НАГРУЗКИ

Наиболее эффективным методом построения ядра сетей связи пятого и последующих поколений в настоящее время представляется использование мультиконтроллерных программно-конфигурируемых сетей SDN. Поскольку создание ядра сети влечет за собой достаточно большие затраты при планировании сетей связи пятого и последующих поколений, одно из приоритетных мест в исследованиях такой проблемы занимают вопросы оптимизации построения ядра сети. К настоящему времени существует целый ряд алгоритмов для размещения контроллеров в мультиконтроллерных сетях, основанных на метаэвристических методах вследствие сложности решаемых задач, и алгоритмов балансировки нагрузки, позволяющих обеспечить наилучшее использование их ресурсов. Однако интегрального решения проблемы размещения контроллеров и балансировки нагрузки пока найдено не было. Именно решению такой проблемы и посвящена настоящая глава. С целью достижения поставленной цели в работе предложено совместно использовать кластеризацию сети и метаэвристический хаотический алгоритм «роя сальп», хорошо зарекомендовавший себя в предыдущих исследованиях по проблемам построения мультиконтроллерных сетей. С учетом интегрального решения проблемы размещения контроллеров на базе кластеризации мультиконтроллерной сети и балансировки нагрузки алгоритм «роя сальп» в главе модифицирован. Анализ эффективности предложенного решения проведен путем сравнения результатов моделирования как с широко известными метаэвристическими алгоритмами «роя частиц» (PSO) и «серого волка» (GWO), так и с предыдущей версией хаотического алгоритма «роя сальп» (CSSA).

2.1 Введение

Развитие сетей и систем связи в направлении создания сетей связи пятого и последующих поколений ставит все новые и новые задачи перед исследовательскими центрами во всем мире. Появление концепций Интернета Вещей и Тактильного Интернета [159] привело к созданию сверхплотных сетей и сетей связи с ультрамалыми задержками, принципиально изменивших представления о трафике, поступающем на сети, и о требованиях к качеству обслуживания в таких сетях. Это привело к необходимости пересмотра представлений о построении ядра сети. При этом наиболее подходящей новой технологией для построения сетей связи пятого и последующих поколений были признаны программно-конфигурируемые сети SDN (*аббр. от англ.* Software-Defined Network) [160, 161].

Научной проблемой, исследуемой в главе, является разработка алгоритма, обеспечивающего интегральное решение для оптимального размещения контроллеров в мультиконтроллерных сетях, основанных на метаэвристических методах вследствие сложности решаемых задач, и балансировки нагрузки, позволяющей обеспечить наилучшее использование ресурсов контроллеров в таких сетях.

Основной вклад в данной главе заключается в:

– разработке алгоритма иерархической кластеризации мультиконтроллерной сети для решения проблемы интеграции размещения контроллеров в мультиконтроллерных сетях и балансировки нагрузки;

– разработке модифицированного алгоритма хаотического «роя сальп» (CSSA, аббр. от англ. Chaotic Salp Swarm Algorithm – хаотический алгоритм «роя сальп») для использования в иерархических кластерных сетях clus-CSSA.

Глава организована следующим образом: в разделе (2) приводится аналитическая информация о проблеме размещения контроллеров и балансировки нагрузки в мультиконтроллерных сетях SDN, в разделе (3) представлена разработанная кластерная архитектура для балансировки нагрузки в мультиконтроллерных сетях, в разделе (4) – сетевая модель исследуемой мультиконтроллерной сети SDN; в разделе (5) представлена решаемая оптимизационная задача, в разделе (6) – оценка характеристик разработанных решений с использованием кластеризации и метаэвристического модифицированного хаотического «роя сальп» в сравнении с другими известными алгоритмами.

2.2 История вопроса и соответствующие работы

Размещение контроллеров в SDN является одной из критических проблем и привлекает большое внимание в литературе [162]. Проблема размещения контроллеров в мультиконтроллерных сетях впервые была исследована в [163], а затем и [164].

В [165] авторы разработали метод балансировки нагрузки для динамического добавления контроллеров в заданную сеть, при этом коммутаторы могут мигрировать между контроллерами в зависимости от нагрузки на последние. Была представлена идея динамического назначения между коммутатором и контроллером и предложен эффективный алгоритм для балансировки нагрузки и миграции коммутаторов. Однако в исследовании не было учтено существенное влияние первоначального размещения контроллеров на их загрузку.

В [166] авторы модифицировали задачу *k*-центра для повышения эффективности распределения ресурсов между клиентами. Клиентам выделяются ограниченные ресурсы с учетом ограничений на пропускную

способность соответствующих объектов. Формулируется оптимизационная задача и вычисляется решение с помощью линейного и смешанного целочисленного программирования, когда в заданном радиусе доступно необходимое количество объектов с достаточной пропускной способностью.

В [167] авторы предложили метаэвристические решения для размещения контроллеров с использованием алгоритмов оптимизации «роя частиц» (PSO, *аббр. от англ.* Particle Swarm Optimization) и светлячков и сравнили результаты со случайным размещением контроллеров. Результаты моделирования показывали, что оба алгоритма работают лучше с точки зрения задержки и более быстрой сходимости. В приведенном анализе результатов для индийской сети TATA число контроллеров составляет 20. Однако оптимальность наблюдаемого числа контроллеров не была определена.

В работе [168] авторы представили решения на основе алгоритма игры с ненулевой суммой для оптимального размещения нескольких контроллеров. В игре с ненулевой суммой каждый контроллер имеет механизм оптимизации, который вычисляет функцию вознаграждения и сравнивает свое собственное значение вознаграждения для экономии затрат и улучшения качества обслуживания (QoS*, аббр. от англ.* Quality of Service) путем оптимизации расположения контроллеров. Аналогично, в [169] авторы представили игровую модель для изучения размещения нескольких контроллеров. Эта модель учитывает несколько метрик, которые включают задержку связи между контроллерами и коммутаторами, накладные расходы на связь между контроллерами и нагрузку на них. На основе этих метрик в главе сформулирована оптимизационная задача с двумя противоречивыми целями: минимизация задержки и накладных расходов на связь.

В [170] авторы предложили использовать теорию очередей для размещения нескольких контроллеров. Был использован CSSA для решения задачи оптимизации. Авторы провели сравнение разработанного алгоритма с другими метаэвристическими алгоритмами. Результаты моделирования доказали, что предложенный алгоритм превосходит другие метаэвристические алгоритмы и алгоритм на основе теории игр по своим характеристикам.

В [171] авторы сформулировали задачу оптимизации для размещения нескольких контроллеров, учитывая сценарий отказа одного контроллера в SDN. В работе был предложен метаэвристический алгоритм имитационного отжига для достижения глобального оптимального решения. Однако предварительное определение числа контроллеров привело к неоптимальному их размещению в динамически изменяющейся сети реального времени.

В [172] авторы предложили алгоритм разбиения сети, основанный на лувенской эвристической методике определения сообществ. Для идентификации сообществ они вычислили гаверсово расстояние между ребрами сети и присвоили ребрам вес, обратно пропорциональный расстоянию. Для определения расположения контроллеров были реализованы отдельные алгоритмы, использование которых позволяет найти компромисс между средней задержкой и нагрузкой на контроллер. Однако сложность алгоритмов значительно возрастает с размером сети.

Алгоритмы, основанные на *k*-means, эвристике, Парето и многоцелевой оптимизации для размещения нескольких контроллеров, не решаются за требуемое время. Для решения проблемы делаются

определенные допущения и приближения. В целях разработки простых в вычислительном отношении решений авторы работы [173] предложили

разделение сети и оптимизированную кластеризацию *k*-means для разделения сети на *k* подсетей, когда речь идет о задержке. Техника разбиения сети применяется для упрощения проблемы размещения контроллеров. По сравнению с приведенными выше методами подход на основе кластеризации *k*-средних снижает вычислительную сложность. Однако для алгоритма требуются предопределенные входные параметры, что делает его неприменимым в сценариях сетей, работающих в реальном времени.

В [174] авторы применили иерархическую кластеризацию *k*-means для разделения сети и расширили оптимизированную кластеризацию *k*-means, предложенную в [173], учитывая как задержку, так и балансировку нагрузки. Было использовано расстояние кратчайшего пути между узлами вместо евклидова расстояния, используемого в оптимизированном *k*-means. Алгоритм итеративно объединяет *k'* начальных центров и в итоге получает *k* центров. Предварительно заданные значения *k* и *k'* в иерархической кластеризации делают алгоритм неприменимым для глобальных вычислительных сетей (WAN, *аббр. от англ.* Wide Area Network) на базе SDN.

В [175] авторы разработали систему BalanceFlow, которая представляет собой типовое решение для кластеризации контроллеров, основанное на иерархическом развертывании. Основным преимуществом этого метода является гибкая настройка запросов потока, обрабатываемых каждым контроллером, без введения дополнительных задержек распространения. Он соответствует функции мультиконтроллеров в OpenFlow 1.2. Все контроллеры в BalanceFlow поддерживают свою собственную информацию о нагрузке и периодически публикуют ее друг другу через систему межконтроллерного взаимодействия. При изменении состояния трафика один из контроллеров BalanceFlow выбирается в качестве суперконтроллера, который разделяет

трафик и перераспределяет различные настройки потока между соответствующими контроллерами.

Новизна предлагаемых модели и метода заключается в рассмотрении балансировки нагрузки совместно с размещением контроллеров для мультиконтроллерных сетей SDN. Развертывание новой схемы кластеризации с новым разработанным метаэвристическим алгоритмом, т. е. CSSA, является основной новизной диссертационной работы, которая решает обе проблемы мультиконтроллерных сетей SDN. Насколько известно, это первая работа, в которой рассматриваются решения обеих проблем: размещение контроллеров и балансировка нагрузки.

2.3. Иерархическая кластеризация для мультиконтроллерных сетей SDN

Предложенный алгоритм иерархической кластеризации выполняет ее в динамическом режиме, основываясь на сетевом трафике и запросах потоков. Плоскость управления делится на кластеры SDN-контроллеров с головным SDN-контроллером для каждого. На рисунке 2.3.1 показана структура кластерной сети SDN. Сеть SDN состоит из централизованного альфа-контроллера (C_α) и кластеров обычных SDN-контроллеров с равным их числом в каждом кластере. В каждом кластере есть головной узел, который представляет бета-контроллер (C_β) и отвечает за настройку кластера и балансировку нагрузки между контроллерами, входящими в него. Основная цель разработанного алгоритма кластеризации – сбалансировать нагрузку между контроллерами плоскости управления и избежать отказов контроллеров. Это снижает стоимость сети и повышает ее общую доступность и надежность.

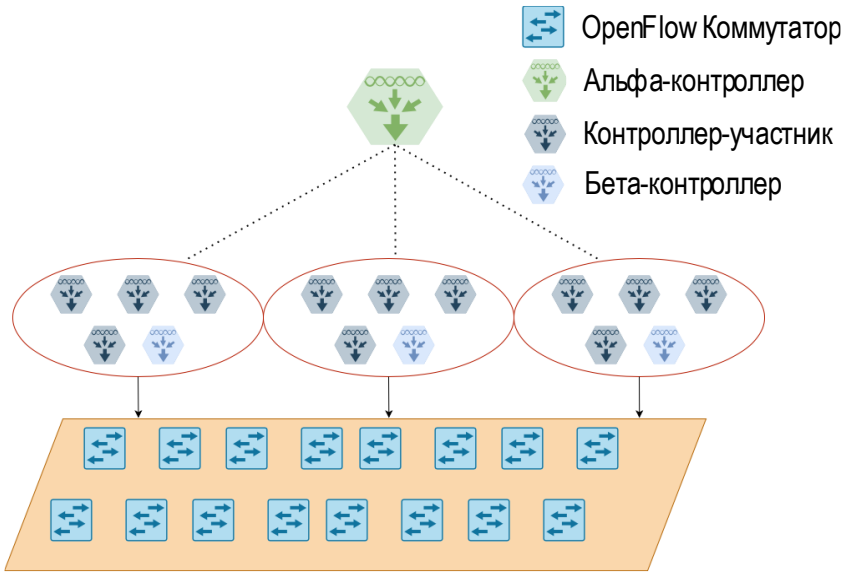


Рисунок. 2.3.1 – Общая структура кластерной мультиконтроллерной сети SDN

Процесс кластеризации делится на фазы 1) настройки контроллера, 2) соединения коммутаторов и 3) установившееся состояние. На этапе создания кластера альфа-контроллер формирует SDN-кластеры, выбирая бета-контроллеры и контроллеры-участники каждого кластера. Контроллер с минимальной ожидаемой нагрузкой выбирается в качестве бета-контроллера и берет на себя роль руководителя членов кластера. Все кластеры формируются однородно, т. е. количество SDN-контроллеров в каждом кластере одинаково. После формирования кластеров фаза настройки контроллеров заканчивается, и начинается фаза настройки соединений коммутаторов. На этапе установки соединений коммутаторов используется разработанный модифицированный алгоритм CSSA, представленный в разделе 2.5.

Каждому кластеру назначается группа OpenFlow-коммутаторов, и, кроме того, CSSA динамически распределяет коммутаторы между каждым контроллером-участником таким образом, чтобы достичь оптимальной эффективности задержки и стоимости, как CAPEX, так и OPEX. На этом этапе каждый SDN-контроллер принимает определенную роль, т. е. либо

члена кластера, либо бета и назначает оптимальные соединения с OpenFlow-коммутаторами, определяемые CSSA. Затем начинается фаза установившегося состояния. Каждый контроллер-член кластера управляет подключенными OpenFlow-коммутаторами и берет на себя ответственность за таблицы потоков таких коммутаторов. Каждый бета-контроллер управляет назначенными OpenFlow-коммутаторами как контроллерами-членами кластера и, кроме того, берет на себя роль головного узла своего кластера.

Бета-контроллер отслеживает трафик между узлами-участниками своего кластера и отправляет отчеты альфа-контроллеру SDN. Когда бета-контроллер обнаруживает дисбаланс нагрузки среди членов кластера, например, некоторые контроллеры перегружены более, чем на 90 % от максимальной нагрузки, или другие контроллеры недогружены – менее, чем на 30 % от максимальной нагрузки, он выполняет новую кластеризацию, называемую межкластеризацией. Процесс межкластеризации направлен на балансировку нагрузки среди контроллеров-участников путем перемещения роли головного узла на контроллер-участник с минимальной нагрузкой, который становится новым бета-контроллером, и использования CSSA для переподключения SDN-контроллеров кластера к OpenFlow-коммутаторам. Процесс межкластерного объединения состоит из трех фаз, как и общая кластеризация, однако бета-контроллер выполняет межкластерное объединение, в отличие от общей кластеризации, которая выполняется альфа-контроллером.

Работа сети продолжается до тех пор, пока нагрузка между кластерами не станет несбалансированной. Альфа-контроллер получает отчеты от бета-контроллеров и отслеживает трафик между различными кластерами; когда он обнаруживает дисбаланс нагрузки между ними, он выполняет новый раунд общей кластеризации, формируя новые кластеры с помощью ранее введенных

фаз. Этот метод поддерживает балансировку нагрузки между распределенными SDN-контроллерами, что позволяет достичь оптимального использования сети.

2.4. Математическое моделирование кластерной мультиконтроллерной сети SDN

Набор развернутых SDN-контроллеров в плоскости управления – это вектор *C*, и он определяется следующим образом:

$$C = \{C_1, C_2, C_3, \dots\dots\dots\dots, C_N\},\ C_\alpha \in C,\ C_\beta \subset C, \quad (1)$$

где N – общее количество развернутых SDN-контроллеров.

Общее количество созданных кластеров равно M, и оно отличается от раунда к раунду. Набор бета-контроллеров определяется следующим образом:

$$C_\beta = \left\{C_{\beta_1}, C_{\beta_2}, C_{\beta_3}, \dots\dots\dots\dots, C_{\beta_M}\right\}, \quad (2)$$
$$C_\beta \subset C \quad \forall\, C_{\beta_i} \in C.$$

Каждый сформированный кластер имеет набор развернутых контроллеров-членов, длина которого равна L. Набор контроллеров-членов для каждого кластера определяется из выражения:

$$C_{mi} = \left\{C_{mi,1}, C_{mi,2}, C_{mi,3}, \dots\dots\dots\dots, C_{mi,L}\right\}, \quad (3)$$
$$C_{mi} \subset C \qquad \forall\, C_{mi,j} \in C.$$

В плоскости данных сети SDN развернуто k OpenFlow-коммутаторов, распределенных между кластерами контроллеров. Каждый коммутатор имеет соединение с SDN-контроллером, которые распределяются с помощью

разработанного алгоритма размещения контроллеров, представленного в следующем разделе. Набор развернутых OpenFlow-коммутаторов определяется выражением:

$$S = \{S_1, S_2, S_3, \dots\dots\dots\dots, S_K\}. \qquad (4)$$

Каждый кластер SDN-контроллеров имеет набор подключенных коммутаторов, которые можно определить как:

$$S_{mi} = \{S_{mi,1}, S_{mi,2}, S_{mi,3}, \dots\dots\dots\dots, S_{mi,R}\}, \quad (5)$$
$$S_{mi} \subset S \qquad \forall\, S_{mi,j} \in S.$$

Соединения между коммутаторами и SDN-контроллерами указываются в матрице коммутации, где строки указывают на SDN-контроллер, а столбцы вводятся для OpenFlow-коммутаторов, при этом матрица T представляет общее количество подключенных коммутаторов на каждый SDN-контроллер.

Пример матрицы коммутации представлен в виде:

$$[0 \cdots 1 \vdots \ddots \vdots 1 \cdots 0\,], T = [2 \vdots 3\,]. \quad (6)$$

Одним из способов проверки производительности контроллера является оценка временного отклика контроллера, на который, в основном, влияет задержка в очереди. Контроллеры могут быть смоделированы с помощью многосерверной модели очередей $M/M/s$, где предполагается, что каждый контроллер имеет s ядер [176]. Передаваемые пакеты поступают на контроллер с определенной интенсивностью, соответствующей процессу Пуассона, образуя единую очередь на контроллере.

Среднее время ответа T_i контроллера C_i представляет собой сумму времени ожидания в очереди и времени обработки и может быть рассчитано по формуле Erlang C как функция интенсивности поступления λ_i и интенсивности обслуживания μ:

$$T_i(\lambda) = \frac{C\left(s, \frac{\lambda_i}{\mu}\right)}{s\mu_i - \lambda_i} + \frac{1}{\mu}, \qquad (7)$$

где $C(s, \lambda/\mu)$ – вероятность того, что все серверы системы используются, и любой прибывающий пакет будет поставлен в очередь, и может быть рассчитана следующим образом:

$$C\left(s, \frac{\lambda}{\mu}\right) = \frac{\left(\frac{(s\rho)^c}{s!}\right)\left(\frac{1}{1-\rho}\right)}{\sum_{k=0}^{s-1}\frac{(s\rho)^k}{k!} + \left(\frac{(s\rho)^c}{s!}\right)\left(\frac{1}{1-\rho}\right)} = \frac{1}{1 + \left(\frac{1}{1-\rho}\right)\left(\frac{s!}{(s\rho)^c}\right)\sum_{k=0}^{s-1}\frac{(s\rho)^k}{k!}}, \qquad (8)$$

$$\rho = \frac{\lambda_i}{s\mu}, \qquad (9)$$

где ρ – коэффициент использования сервера, который является показателем стабильности системы.

Система имеет стабильное распределение только в том случае, если ρ меньше единицы. Когда поступивших заявок в очереди больше, чем серверов контроллера, переход все равно будет только с $s\mu$ и не более, а контроллер будет находиться в состоянии максимальной пропускной способности.

Интенсивность поступления запросов на контроллер может быть рассчитана как сумма средних интенсивностей поступления запросов от коммутаторов, подключенных к контроллеру:

$$\lambda_i = \sum_{k_i} \lambda_s \qquad (10)$$

Средняя нагрузка на контроллер C_i может быть рассчитана как среднее количество запросов, поставленных в очередь и обработанных. Используя (7), средняя нагрузка на контроллер Li может быть рассчитана по формуле (11):

$$L_i(\lambda) = s\rho + \frac{\rho}{1-\rho} C\left(s, \frac{\lambda_i}{\mu}\right). \qquad (11)$$

2.5. Проблема размещения контроллеров с точки зрения задержки и эффективности затрат

2.5.1. Формулировка проблемы

В связи с динамическим изменением нагрузки на сеть проблема размещения контроллеров должна решаться динамически – чтобы достигались требуемые показатели сети. В этом разделе эта проблема сформулирована с точки зрения задержки, использования и стоимости сети. Задача размещения контроллеров при этом направлена на получение динамически оптимального числа SDN-контроллеров, которое обеспечивает требуемую задержку между распределенными OpenFlow-коммутаторами и SDN-контроллерами, а также минимальную стоимость сети. Это оптимальное количество динамически изменяется в зависимости от нагрузки на сеть. Более того, задача направлена на определение оптимальных соединений между коммутаторами и SDN-контроллерами таким образом, чтобы достичь требуемых задержки и стоимости. Модифицируем ранее разработанную нами задачу размещения контроллеров с учетом задержки и стоимости, представленную в [170], для кластерной сети SDN.

Задача размещения контроллеров определяется таким образом, чтобы распределить новые контроллеры или сократить существующие в соответствии с динамическим изменением сетевого трафика. Проблема оптимизации формулируется для получения оптимального количества SDN-контроллеров и оптимального количества контроллеров-участников на кластер таким образом, чтобы достичь эффективности задержки, стоимости, использования и балансировки нагрузки. Задача оптимизации представляет собой функцию минимизации, целью которой является уменьшение общего числа SDN-контроллеров в сети N_T, общего числа SDN-контроллеров на кластер N_C, общей стоимости SDN-контроллеров, включающей как CAPEX, так и OPEX, и средней задержки между контроллером и подключенными коммутаторами, включающей распространение, постановку в очередь и обработку.

Задача формулируется следующим образом:

$$\text{Min } f(N_T, N_C, C, D). \qquad (12)$$

Ограничения:

$$T_i^j \leq T_{\text{thr}}, \ \ \forall\, i\epsilon\, \gamma \ \wedge \ j\epsilon\, \beta, \qquad (13)$$

$$U_{lb} \leq U_i^j \leq U_{ub}, \ \ \forall\, i\epsilon\, \gamma \ \wedge \ j\epsilon\, \beta, \qquad (14)$$

$$U_{C-lb} \leq U_C^j \leq U_{C-ub}, \ \ \forall\, j\epsilon\, \beta, \qquad (15)$$

где f – нелинейная функция общего количества развернутых SDN-контроллеров N_T, общего количества SDN-контроллеров на кластер N_C; C – общая стоимость SDN-контроллеров, включающая CAPEX и OPEX; D – средняя задержка между контроллером и подключенными коммутаторами, включающая распространение, постановку в очередь и обработку.

Задача представляет собой многоцелевую оптимизацию с множеством ограничений, которая может быть решена соответствующим метаэвристическим

алгоритмом с тремя ограничениями. Первое ограничение указывает, что среднее время ответа T_i^j контроллера C_i, принадлежащего кластеру j, должно быть меньше порогового значения T_{thr}, которое является предопределенным значением; это имеет место для всех SDN-контроллеров в наборе доступных контроллеров [γ] для набора развернутых кластеров [β]. T_{thr} предопределено таким образом, чтобы удовлетворить требуемое QoS. Второе ограничение вводится для поддержания уровня использования каждого SDN-контроллера. Показатель использования U_i^j контроллера C_i в кластере j должен находиться в пределах, с одной стороны, нижней границы использования SDN-контроллера U_{lb} (т. е. минимального значения использования SDN-контроллера, ниже которого он должен быть отключен для достижения экономической эффективности), а с другой – верхней (т. е. максимального значения использования SDN-контроллера, выше которого контроллер может быть перегружен). Верхний и нижний пределы U_{ub} и U_{lb} предопределены таким образом, чтобы достичь требуемого QoS. Этот показатель использования SDN-контроллера применяется для сопоставления с показателями использования мощности, хранения и обработки данных. Третье ограничение учитывает поддержание общего индекса использования кластера j между максимальным (U_{C-ub}) и минимальным пределами (U_{C-lb}). Это ограничение введено для поддержания баланса нагрузки между кластерами и для предотвращения выхода из строя фрагментов сети.

2.5.2. Использование системы

В этом подразделе представлена фитнес-функция для сформулированной задачи. Это функция использования, которая применяется для сравнения различных решений и указания на лучшее решение путем отображения переменных или событий на реальные числа:

$$U: V \to R. \qquad (16)$$

Использование времени – это первая функция полезности, которая применяется для отображения временной реакции SDN-контроллеров. Функции потерь хорошо подходят для данного случая, поэтому для моделирования использования времени можно применить любую их форму. Здесь рассматривается квадратичная функция, так как она математически проста благодаря своей симметрии.

Временная полезность SDN-контроллера C_i, принадлежащего кластеру j, равна $U^j_{T\text{-}Ci}$ и определяется следующим образом:

$$U^j_{T-Ci} = \{\alpha + \delta\left(T_{\text{thr}} - T_i^j(\lambda)\right)^2, \quad T_i^j(\lambda) \leq T_{\text{thr}} \;\; 0, T_i^j(\lambda) > T_{\text{thr}} \quad \forall\, i \in \gamma \;\wedge\; j \in \beta, \qquad (17)$$

где α и δ – константы, значения которых не влияют на решение. Первой константе α может быть присвоено определенное значение, которое представляет собой минимальное ненулевое значение времени использования $U^j_{T\text{-}Ci}$, возникающее, когда время реакции равно пороговому значению.

Эти константы могут быть определены следующим образом:

$$\alpha = U_{T-\text{thr}} \quad \forall\, U_T \in [0,1], \qquad (18)$$

$$\beta = \frac{(1 - U_{T-\text{thr}})}{T_{\text{thr}}^2} \quad \forall\, U_T \in [0,1]. \qquad (19)$$

При пороговом времени использования каждого контроллера 70 %, время использования может быть пересчитано следующим образом:

$$U_{T-Ci}^{j} = \{0{,}7 + \frac{0{,}3}{T_{\text{thr}}^{2}}\left(T_{\text{thr}} - T_{i}^{j}(\lambda)\right)^{2}, \quad (20)$$

$$T_{i}^{j}(\lambda) \leq T_{\text{thr}}\ \ 0, T_{i}^{j}(\lambda) > T_{\text{thr}} \quad \forall\, i \in \gamma\ \wedge\ j \in \beta.$$

Функция использования времени кластера j – U^{j}_{T} может быть рассчитана как нормализованное среднее значение использования времени каждого члена кластера (21).

Второй функцией полезности, которую следует рассмотреть, является функция полезности затрат, которая отображает стоимость используемых контроллеров. Под стоимостью в основном понимаются оба термина: CAPEX и OPEX, связанные с развернутыми контроллерами. Квадратичная функция потерь также представляет собой подходящую функцию для выражения утилизации затрат. Стоимость использования каждого контроллера в наборе доступных контроллеров может быть определена по выражению (22), где $U^{j}_{C\text{-}Ci}$ – функция использования затрат контроллера C_i в кластере j, а ρ – константа, которая не влияет на решение, независимо от того, какое значение ей присвоено. Правильное значение ρ может быть определено следующим образом, для $U^{j}_{C\text{-}Ci}$ между 0 и 1 (23). Функция использования затрат в кластере j – U^{j}_{C} может быть рассчитана как нормализованное среднее значение использования затрат каждого члена кластера (24).

$$U_T^j = \frac{\sum_{N_C} U_{T-Ci}^j}{N_C} \quad (21$$

$$= \frac{\left(\sum_{N_C}\{0{,}7 + \frac{0{,}3}{T_{\text{thr}}^2}\left(T_{\text{thr}} - T_i^j(\lambda)\right)^2,\ T_i^j(\lambda) \le T_{\text{thr}} 0\ ,\ T_i^j(\lambda) > T_{\text{thr}} \quad \forall\, i\epsilon\,\gamma\ \wedge\ j\right)}{N_C}$$

$$U_{C-Ci}^j = \{\rho\left(U_{ub} - U_i^j\right)^2 \quad \forall\ U_i^j\ \in\ [U_{lb}\ , U_{ub}] \quad 0 \quad \forall\ U_i^j \quad (22$$

$$\notin\ [U_{lb}\ , U_{ub}] \quad \forall\, i\epsilon\,\gamma\ \wedge\ j\epsilon\,\beta, \quad)$$

$$\rho = \frac{1}{(U_{ub} - U_{lb})^2} \quad \forall\, U_C\ \in\ [0{,}1] \quad (23$$

$$)$$

$$U_C^j = \frac{\sum_{N_C} U_{C-Ci}^j}{N_C} \quad (24$$

$$= \frac{\left(\sum_{N_C}\{\rho(U_{ub} - U_i^j)^2 \quad \forall\ U_i^j\ \in\ [U_{lb}\ , U_{ub}] \quad 0 \quad \forall\ U_i^j\ \notin\ [U_{lb}\ , U_{ub}] \quad \forall\, i\epsilon\right)}{N_C}$$

Общая полезность каждого развернутого SDN-контроллера может быть рассчитана как взвешенная сумма полезности времени и затрат, и таким образом можно оценить общую полезность каждого кластера.

Общая полезность контроллера C_i в j-м кластере равна U^j_{Ci} и рассчитывается следующим образом:

$$U_{Ci}^j = \delta_C U_{C-Ci}^j + \delta_T U_{T-Ci}^j, \qquad (25)$$

где δ_C и δ_T – весовые коэффициенты затрат и времени, соответственно.

Функция общего использования системы U_{cont} представляет собой среднее значение использования каждого контроллера и рассчитывается следующим образом:

$$U_{cont} = \frac{\sum_{\beta} \sum_{N_C} U_{Ci}^{j}}{|\gamma|}. \qquad (26)$$

Более того, средняя полезность затрат $U_{C\text{-}cont}$ и средняя полезность времени $U_{T\text{-}cont}$ всех доступных контроллеров могут быть рассчитаны следующим образом:

$$U_{C-cont} = \frac{\sum_{\beta} \sum_{N_C} U_{C-Ci}^{j}}{|\gamma|}, \qquad (27)$$

$$U_{T-cont} = \frac{\sum_{\beta} \sum_{N_C} U_{T-Ci}^{j}}{|\gamma|}. \qquad (28)$$

Общая полезность каждого сформированного кластера может быть рассчитана таким же образом, как и для отдельных контроллеров. Общая полезность кластера j равна U^J и может быть рассчитана как взвешенная сумма полезностей времени и затрат кластера j:

$$U^{j} = \delta_C U_C^{j} + \delta_T U_T^{j}. \qquad (29)$$

Функция среднего использования каждого сформированного кластера U_{clus} и рассчитывается следующим образом:

$$U_{Clus} = \frac{\sum_{\beta} U^{j}}{|\beta|}. \qquad (30)$$

Более того, средняя полезность затрат $U_{C\text{-}clus}$ и средняя полезность времени $U_{T\text{-}clus}$ всех сформированных кластеров вычисляются по выражениям:

$$U_{C-clus} = \frac{\sum_{\beta} U_C^{j}}{|\beta|}, \qquad (31)$$

$$U_{T-clus} = \frac{\sum_{\beta} U_T^{j}}{|\beta|}. \qquad (32)$$

2.5.3. Хаотический алгоритм «роя сальп» для кластерной сети SDN

В этом подразделе представляется процедура оптимизации на основе CSSA для решения задачи размещения контроллеров в кластерной сети SDN в постановке, сформулированной в подразделе 5.1. Разработанный алгоритм является модифицированной версией CSSA, представленного в [170].

Алгоритм CSSA – это метаэвристический популяционный алгоритм, имитирующий поведение сальпы в океанах. Это недавний тип оптимизаторов PSO, который моделирует поведение живых роев в реальной жизни. Рой сальп состоит из сальпы-лидера и сальп-последователей, которые движутся по цепочке вслед за лидером к позиции пищи, которая является наилучшей для них. Позиция сальпы моделируется как d-мерное пространство поиска, где d представляет собой количество переменных в определенной задаче, точно так же, как и другие алгоритмы на основе роя.

Вектор позиции n сальп в пространстве поиска имеет вид $X^j = [x_1^j, x_2^j, x_3^j, \ldots\ldots, x_d^j]$, $j = 1, 2, \ldots, n$, и лидер обновляет свою позицию, используя следующее уравнение:

$$X_i^1 = \begin{cases} F_i + C_1\big((ub_i - ul_i)C_2 + lb_i\big), \\ C_3 \geq 0 \; F_i - C_1\big((ub_i - ul_i)C_2 + lb_i\big), \quad C_3 < 0, \end{cases} \tag{33}$$

где X_i^1 обозначает положение сальпы-лидера в измерении i^{th}; F_i – положение пищи в измерении i^{th}; ub_i и lb_i – верхняя и нижняя границы в измерении i^{th}; C_1, C_2, и C_3 – коэффициенты модели (представляют собой случайные числа, которые используются для решения определенных задач).

Первый коэффициент C_1 вводится для обеспечения баланса между разведкой и эксплуатацией, представляет собой наиболее важный параметр в алгоритме и определяется следующим образом:

$$C_1 = 2\mathrm{e}^{-\left(\frac{4t}{\mathrm{T_{max}}}\right)^2}, \qquad (34)$$

где t – текущая итерация: $\mathrm{T_{max}}$ – максимальное число итераций.

C_2 и C_3 – случайные числа, которые равномерно генерируются со значениями от 0 до 1. Сальпы-последователи обновляют свои позиции на основе закона движения Ньютона, используя следующее уравнение:

$$X_i^k = \frac{1}{2}\left(X_i^k + X_i^{k-1}\right) \quad 2 \le k < n, \quad (35)$$

где X_i^k – положение k^{th} последователей сальпа в i^{th} измерении; n – общее число частиц сальп.

Чтобы избежать спада в локальных оптимумах и низкой скорости сходимости, введем хаотические карты в рассматриваемый оптимизатор «роя сальп». Хаотические карты вводятся для обновления оптимизатора вместо случайных чисел.

Используем логистическую карту для настройки значения второго коэффициента C_2 следующим образом:

$$C_2^t = \omega(t), \qquad (36)$$

$$\omega(t+1) = a\omega(t)[1-\omega(t)], \quad a = 4, \qquad (37)$$

где $\omega(t)$ – значение логистической карты на итерации t^{th}, с начальным условием 0,7, т. е. $\omega(0) = 0{,}7$.

Для *NP*-трудной задачи, смоделированной в подразделе 2.5.1, мы стремимся определить оптимальное количество SDN-контроллеров и кластеров, а также оптимальное соединение для каждого коммутатора в наборе коммутаторов *S*. Путем итераций несколько роев сальп параллельно ищут оптимальные решения, которые представляют собой оптимальное количество контроллеров, и кластеров, а также оптимальное распределение контроллеров между коммутаторами. При этом для получения оптимальных решений вводятся три вложенных алгоритма, основанных на ранее представленном хаотическом сальпе.

Алгоритм 1 представляет псевдокод CSSA, разработанного для задачи размещения контроллеров, поставленной в подразделе 2.5.1, где каждая сальпа представляет контроллер в сети. Выходом этого алгоритма является оптимальное количество SDN-контроллеров. Алгоритм 2 представляет псевдокод CSSA, разработанного для оптимального соединения SDN OpenFlow-коммутаторов на основе оптимального количества контроллеров, рассчитанного алгоритмом 1. Каждая сальпа в алгоритме 2 представляет все доступные соединения для всех коммутаторов с их выделенными контроллерами, и это *M*-мерный вектор, каждое измерение которого представляет коммутатор. Выходом второго алгоритма является наилучшее распределение контроллеров между коммутаторами. Алгоритм 3 представляет собой псевдокод CSSA, разработанного для оптимальной кластеризации. Выходом алгоритма 3 является оптимальное число кластеров.

Алгоритм 1. *CSSA для поиска оптимального числа контроллеров*

1: ***Initialize*** u_b, l_b, T_{max}, d, n
2: ***Initialize*** *positions of salps* x_i $(i = 1, 2, 3, \ldots, n)$
3: ***While*** $(t \leq t_{max})$
4: *Calculate the fitness function of each salp position using Eq. (15)*
5: *F = The best salp position*

6: Update the value of C_1 using (23)
7: Get the value of chaotic map (Logistic) w(t)
8: For (i = 1: i ≤ n) do
9: if (i == 1)
10: Update the position of the leading salp using (22, 23, 25, 26)
11: else
12: Update the position of the follower salps using (24)
13: end if
14: end for
15: Adjust salps based on the upper and lower bounds
16: Calculate the best connections of switches for the best salp (call Algorithm 2)
17: Update the best salp based on the results of Algorithm 2
18: Calculate the best number of clusters (call Algorithm 3)
19: t← t+1
*20: **Return** F*

Алгоритм 2. *CSSA для оптимального соединения переключателей с контроллерами*

*1: **Initialize** ub, lb, T_{max}, d, n*
*2: **Initialize** positions of salps x_i (i = 1, 2, 3, …, n)*
*3: **While** ($t \leq t_{max}$)*
4: Calculate the fitness function of each salp position using (15)
5: F = The best salp position
6: Update the value of C1 using (23)
7: Get the value of chaotic map (Logistic) w(t)
8: For (i = 1: i ≤ n) do
9: if (i == 1)
10: Update the position of leading salp using (22, 23, 25, 26)
11: else
12: Update the position of follower salp using (24)
13: end if
14: end for
15: Adjust salps based on the upper and lower bounds
16: t← t+1
*17: **Return** F*

Алгоритм 3. *CSSA для оптимальной кластеризации*

*1: **Initialize** ub, lb, T_{max}, d, n*
*2: **Initialize** positions of salps x_i (i = 1, 2, 3, …, n)*
*3: **While** ($t \leq t_{max}$)*
4: Calculate the fitness function of each salp position using (19)
5: F = The best salp position
6: Update the value of C1 using (23)
7: Get the value of chaotic map (Logistic) w(t)
8: For (i = 1: i ≤ n) do
9: if (i == 1)
10: Update the position of leading salp using (22, 23, 25, 26)

11: *else*
12: *Update the position of follower salp using (24)*
13: *end if*
14: *end for*
15: *Adjust salps based on the upper and lower bounds*
16: *t← t+1*
17: ***Return*** *F*

2.6. Оценка эффективности

2.6.1. Настройка моделирования

Разработанный оптимизированный алгоритм кластеризации смоделирован в среде Matlab с использованием процессора Intel Core i7 и оперативной памяти 8 ГБ. Для оценки производительности рассмотрим реальные топологии крупномасштабных сетей WAN, чтобы лучше проиллюстрировать эффективность предложенной схемы. Для моделирования выбрано 15 типовых реальных топологий из набора данных Internet Topology Zoo. Эти топологии рассматриваются как крупномасштабные сети и представлены в таблице 1 с их характеристиками. Каждая точка присутствия в топологии сети рассматривается как сетевой коммутатор. Предполагается, что поток является случайной величиной, подчиняющейся распределению Пуассона. В таблице 2.6.1.1 представлены рассматриваемые параметры моделирования. В качестве SDN-контроллера используется NOX-контроллер.

Таблица 2.6.1.1 – Топологии сети, рассмотренные для моделирования

Ссылка. Номер	Топология	Количество
1	IBM	18
2	Oxford	20

3	FCCN	23
4	AGIS	25
5	Viatel	83
6	GEANT	27
7	TATA	169
8	GTS CE	187
9	PalmettoNet	49
10	OTEGlobe	78
11	DFN	47
12	GARR	36
13	ULAKNET	76
14	RNP	28
15	Carnet	43

Таблица 2.6.1.2 – Параметры моделирования

Параметр	Описание	Количество
λ_S	Средняя интенсивность запросов коммутатора	[1500, 3000]
μ	Интенсивность обслуживания контроллера (запрос/с)	30000
T_{thr}	Порог задержки (мс)	2

Параметр	Описание	Количество
U_{ub}	Верхняя граница индекса использования контроллера	0,9
U_{lb}	Нижняя граница индекса использования контроллера	0,7
$U_{C\text{-}ub}$	Верхняя граница индекса использования кластера	0,9
$U_{C\text{-}lb}$	Нижняя граница индекса использования кластера	0,7
$T_{\max}$ (1)	Максимальное количество итераций для алгоритма 1	50
$T_{\max}$ (2)	Максимальное количество итераций для алгоритма 2	30
$T_{\max}$ (3)	Максимальное количество итераций для алгоритма 3	30
δ_C	Весовой коэффициент затрат	18
δ_T	Весовой коэффициент времени	25

2.6.2. Результаты моделирования

В интересах оценки производительности рассматриваются три сценария моделирования для исследования влияния на оптимальные топологические решения (число SDN-контроллеров) для следующих параметров: 1) порога среднего времени отклика SDN-контроллера T_{thr} (Сценарий I), 2) верхнего индекса использования U_{ub} (Сценарий II), верхней границы индекса использования

кластера $U_{C\text{-}ub}$ (Сценарий III). Для Сценариев I и III – U_{ub} установлена на 0.9; для Сценариев II и III – T_{thr} установлен на 2 мс; для Сценариев I и II – $U_{C\text{-}ub}$ установлена на 0.9. Разработанный оптимизированный метод кластеризации реализуется для каждой из пятнадцати топологий глобальной сети для четырех случаев (наборов параметров), представленных в таблице 2.6.2.1.

Таблица 2.6.2.1 – Четыре набора параметров для трех сценариев моделирования

Ссылка. Номер	Сценарий I	Сценарий II	Сценарий III
Случай(1)	$T_{\text{thr-1}} =$ 1 мс	U_{ub1} = 0.8	$U_{C\text{-}ub1}$ = 0.8
Случай (2)	$T_{\text{thr-2}}$ = 2 мс	U_{ub2} = 0.90	$U_{C\text{-}ub2}$ = 0.0
Случай (3)	$T_{\text{thr-3}}$ = 3 мс	U_{ub3} = 0.2	$U_{C\text{-}ub3}$ = 0.2
Случай (4)	$T_{\text{thr-4}}$ = 4 мс	U_{ub4} = 0.4	$U_{C\text{-}ub4}$ = 0.4

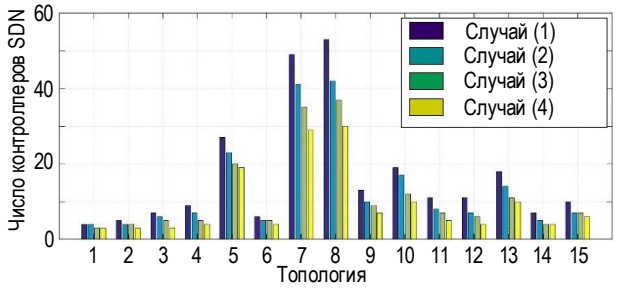
Результаты показывают, что оптимальное количество активных SDN-контроллеров уменьшается по мере увеличения порогового времени отклика и индекса максимального использования каждого SDN-контроллера, а значит, уменьшается и оптимальное количество кластеров. С увеличением порогового времени отклика и индекса максимальной загрузки SDN-контроллер обрабатывает большее количество коммутаторов. Это происходит за счет задержки, которая должна поддерживать требуемое QoS системы и нагрузки на контроллер, которая увеличивает вероятность сбоя и, кроме того, потребляет больше энергетических ресурсов.

Разработанный оптимизированный метод кластеризации реализован для случайной сети с плоскостью данных из 100 коммутаторов, чтобы оценить производительность CSSA для сетей большого масштаба. Основной целью

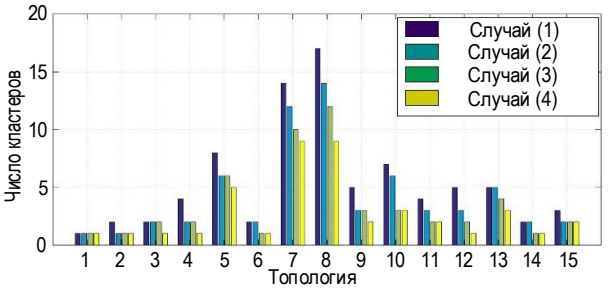
данного сценария моделирования является оценка влияния вариации параметров сети на производительность оптимизированного метода кластеризации. Производительность системы измеряется в широком диапазоне порогового времени обработки SDN-контроллера T_{thr} верхнего индекса использования SDN-контроллера U_{ub} и верхней границы индекса использования кластера $U_{C\text{-}ub}$.

Изменение этих параметров является критическим и оказывает значительное влияние на оптимальные топологические решения; особенно для крупномасштабных сетей с плотным развертыванием; более того – большое влияние на QoS сети.

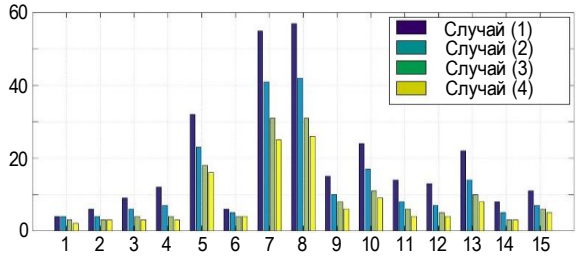
Далее для моделирования рассматриваются три сценария с десятью случаями. В первом сценарии оценивается влияние изменения порогового времени обработки SDN-контроллера T_{thr}, во втором сценарии – верхнего индекса использования SDN-контроллера U_{ub}, а в третьем сценарий – верхней границы индекса использования кластера $U_{C\text{-}ub}$.



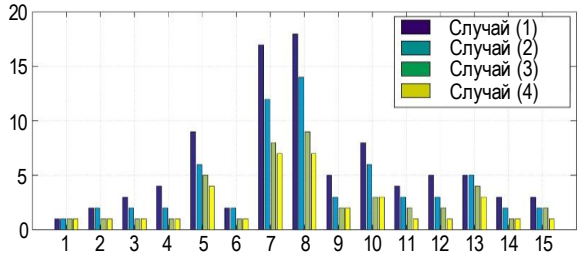
a)



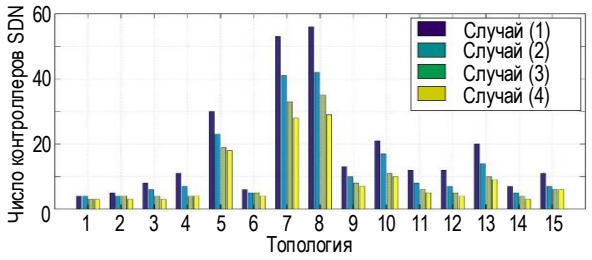
a)



b)

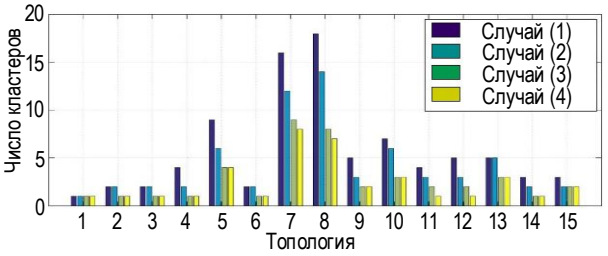


b)



c)

Рисунок 2.6.2.1 – Оптимальное число SDN-контроллеров для каждой топологии Сценария I (a), Сценария II (b), Сценария III (c)

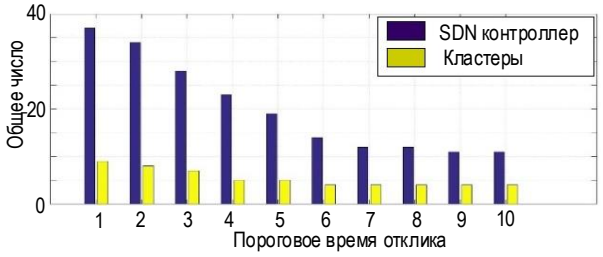


c)

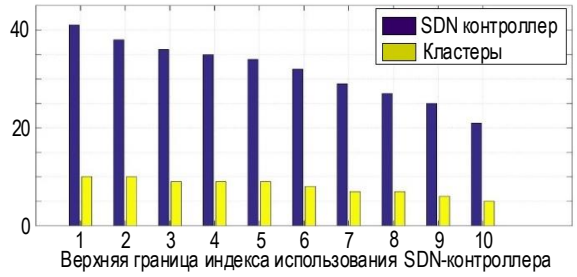
Рисунок 2.6.2.2 – Оптимальное число кластеров для каждой топологии Сценария I (a), Сценария II (b), Сценария III (c)

Для каждого сценария рассматривается десять значений каждого параметра, а каждое значение представляет собой вариант моделирования. В таблице 4 приведены значения рассматриваемых параметров для каждого варианта по каждому сценарию.

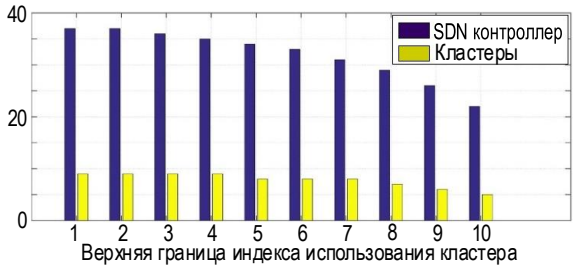
На рисунке 4 представлены результаты для трех упомянутых в таблице 4 сценариев моделирования.



a)



b)



c)

Рисунок 2.6.2.3 – Оптимальное число SDN-контроллеров и кластеров для Сценария I (a), Сценария II (b), Сценария III (c)

Таблица 2.6.2.2 – Десять наборов параметров для трех сценариев моделирования

Ссылка. Номер	Сценарий I	Сценарий II	Сценарий III
Вариант (1)	$T_{\text{thr-1}} =$ 1 мс	U_{ub1} = 0,86	$U_{C\text{-}ub1}$ = 0,86
Вариант (2)	$T_{\text{thr-2}}$ = 2 мс	U_{ub2} = 0,87	$U_{C\text{-}ub2}$ = 0,87
Вариант (3)	$T_{\text{thr-3}}$ = 3 мс	U_{ub3} = 0,88	$U_{C\text{-}ub3}$ = 0,88
Вариант (4)	$T_{\text{thr-4}}$ = 4 мс	U_{ub4} = 0,89	$U_{C\text{-}ub4}$ = 0,89
Вариант (5)	$T_{\text{thr-5}}$ = 5 мс	U_{ub5} = 0,90	$U_{C\text{-}ub5}$ = 0,90
Вариант (6)	$T_{\text{thr-6}}$ = 6 мс	U_{ub6} = 0,91	$U_{C\text{-}ub6}$ = 0,91
Вариант (7)	$T_{\text{thr-7}}$ = 7 мс	U_{ub7} = 0,92	$U_{C\text{-}ub7}$ = 0,92
Вариант (8)	$T_{\text{thr-8}}$ = 8 мс	U_{ub8} = 0,93	$U_{C\text{-}ub8}$ = 0,93
Вариант (9)	$T_{\text{thr-9}}$ = 9 мс	U_{ub9} = 0,94	$U_{C\text{-}ub9}$ = 0,94
Вариант (10)	$T_{\text{thr-10}}$ = 10 мс	U_{ub10} = 0,95	$U_{C\text{-}ub10}$ = 0,95

Из рисунка 4a (Сценарий I) видно, что по мере увеличения порогового времени отклика SDN-контроллера общее количество оптимальных SDN-контроллеров и, соответственно, кластеров, необходимых для сети, уменьшается. Это аналогично Сценарию II (см. рисунок 4b) и Сценарию III (см. рисунок 4c). Однако влияние изменения порогового времени отклика на оптимальное количество SDN-контроллеров и кластеров значительнее, чем влияние индексов использования.

Разработанный CSSA сравнивается с генетическим алгоритмом (GA, *аббр. от англ.* Genetic Algorithm), алгоритмом PSO и алгоритмом оптимизации «серого волка» (GWO, *аббр. от англ.* Grey Wolf Optimization). Это самые последние алгоритмы, используемые для решения задачи размещения контроллеров в мультиконтроллерных сетях SDN. Они реализованы для ранее представленной случайной сети со 100 OpenFlow-коммутаторами, для указанных трех сценариев в таблице 4. Для такого сравнения рассматриваются две основные метрики: доля отказов в обслуживании со стороны контроллера и использование системы контроллеров в целом. Естественно, что при этом анализируются случаи для различных значений длительности задержки. На рисунках 5 и 6 представлены процент отказов и общее использование системы для каждого алгоритма.

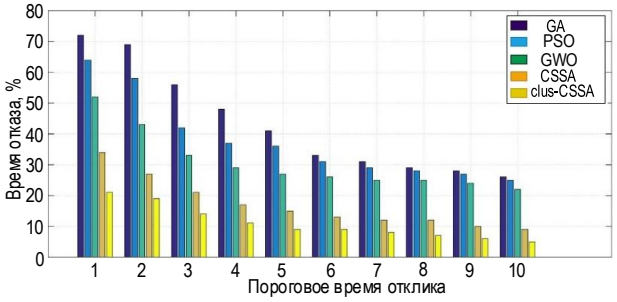


Рисунок 2.6.2.4 – Доля отказов в обслуживании со стороны контроллера при использовании сравниваемых алгоритмов

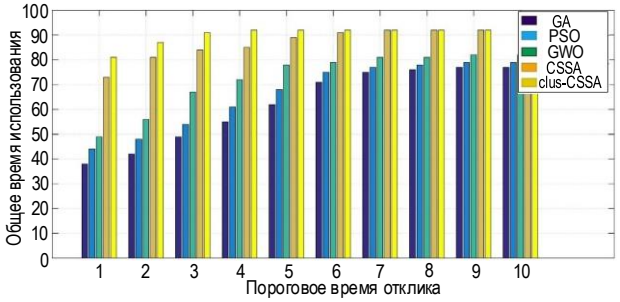


Рисунок 2.6.2.5 – Общее использование системы для сравниваемых алгоритмов

Результаты показывают, что разработанный оптимизированный CSSA достигает более высокой эффективности, чем другие алгоритмы.

2.7. Выводы по главе 2

В данной главе представлено решение научной проблемы размещения контроллеров в мультиконтроллерных сетях и балансировки нагрузки, новизна которого (решения) состоит в следующем.

Во-первых, предложен метод построения мультиконтроллерной сети, основанный на интегральном решении задач по размещению контроллеров в таких сетях, базирующийся на метаэвристическом (вследствие сложности решаемых задач) алгоритме и алгоритме балансировки нагрузки, позволяющем обеспечить наилучшее использование ресурсов контроллеров. Во-вторых, предложено использовать иерархическую кластеризацию такой сети, включающую в себя кластеры с головными узлами и централизованный контроллер, что обеспечивает балансировку нагрузки в разработанном методе построения сети. В-третьих, разработан модифицированный алгоритм CSSA для использования в иерархических кластерных сетях clus-CSSA.

Разработанный метод построения мультиконтроллерной сети позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1 до 10 мс по сравнению как с широко известными метаэвристическими алгоритмами

PSO и GWO, так и с предыдущей версией CSSA. При этом для наиболее сложного случая задержки величиной в 1 мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза.

Далее еще будет реализован модифицированный алгоритм CSSA для сети Интернета Вещей высокой плотности.

ГЛАВА 3 МОДЕЛЬ ИНТЕГРАЦИИ ГРАНИЧНЫХ ВЫЧИСЛЕНИЙ В СТРУКТУРУ СЕТИ «ВОЗДУХ-ЗЕМЛЯ» И МЕТОДЫ ВЫГРУЗКИ ТРАФИКА ДЛЯ СЕТЕЙ ИНТЕРНЕТА ВЕЩЕЙ ВЫСОКОЙ И СВЕРХВЫСОКОЙ ПЛОТНОСТИ

Глава посвящена научной проблеме интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности. Подобные проблемы являются наиболее актуальными сегодня в связи с появлением концепции интегрированных сетей космос-воздух-земля-море. Разработана модель сети, в которой предложено для уменьшения задержки и энергопотребления использовать мобильные серверы граничных вычислений, расположенные на беспилотных летательных аппаратах (БПЛА), а также метод выгрузки трафика с наземной сети на мобильные серверы граничных вычислений на БПЛА. При этом процедура выгрузки трафика является трехуровневой, а на оконечных устройствах используется программный профилировщик, который определяет сложность вычисляемой задачи и по результатам его работы механизм принятия решения делает вывод о необходимости выгрузки трафика. Для оптимизации структуры сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности с целью минимизации задержки и энергопотребления при выгрузке трафика с наземной сети на серверы граничных вычислений БПЛА разработан метаэвристический алгоритм на основе хаотического роя сальп. Результаты моделирования показали, что предложенные модель и метод обеспечивают существенное уменьшение задержки и энергопотребления, а также доли заблокированных задач при выгрузке трафика по сравнению с известными решениями.

3.1. Использование алгоритма хаотического роя сальп для расчета оптимальных позиций БПЛА

Развитие сетей связи в настоящее время осуществляется в направлении создания интегрированных сетей связи космос-воздух-земля-море SAGSIN (Space-Air-Ground-Sea Integrated Networks) [176]. Это ни коей мере не отрицает достижений в области сетей связи пятого и шестого поколений, сетей связи высокой и сверхвысокой плотности, сетей связи с ультрамалыми задержками [177,178]. Напротив, интеграция ресурсов в сети SAGSIN должна позволить эффективно использовать возможности таких сетей.

Сценарии высокоплотного и сверхплотного построения сетей связи [179,180] призваны в полной мере использовать возможности концепции Интернета Вещей, а сети с ультрамалой задержкой – концепции Тактильного Интернета. Для решения задач по построению таких сетей требуется использовать новые технологии в области сетей и систем связи [181]. К таким технологиям относятся, например, распределенные граничные вычисления (MEC) [182,183,184,185] и беспилотные летательные аппараты (UAVs) [186,187,188,189,190].

Беспилотные летательные аппараты могут быть использованы для сетевой поддержки в современных сетях для решения разнообразных задач, таких как увеличение покрытия сети, при чрезвычайных ситуациях, граничные вычисления на базе БПЛА и т.д. [191, 192, 193, 194, 195, 196].

В этой главе исследуется возможность использования БПЛА в качестве мобильных серверов граничных вычислений для поддержки наземных высокоплотных и сверхплотных сетей Интернета вещей с целью уменьшения задержки и энергопотребления, а также уменьшения доли заблокированных задач при выгрузке трафика в воздушный сегмент сети (БПЛА). Эта цель

достигается путем разработки модели сети, метода выгрузки трафика и оптимизации полученных решений с использованием метаэвристического алгоритма на базе хаотического рода сальп.

3.1.1 Модель сети

Разработанная модель сети представлена на рисунке 3.1.1. Она включает в себя наземный и воздушный сегменты.

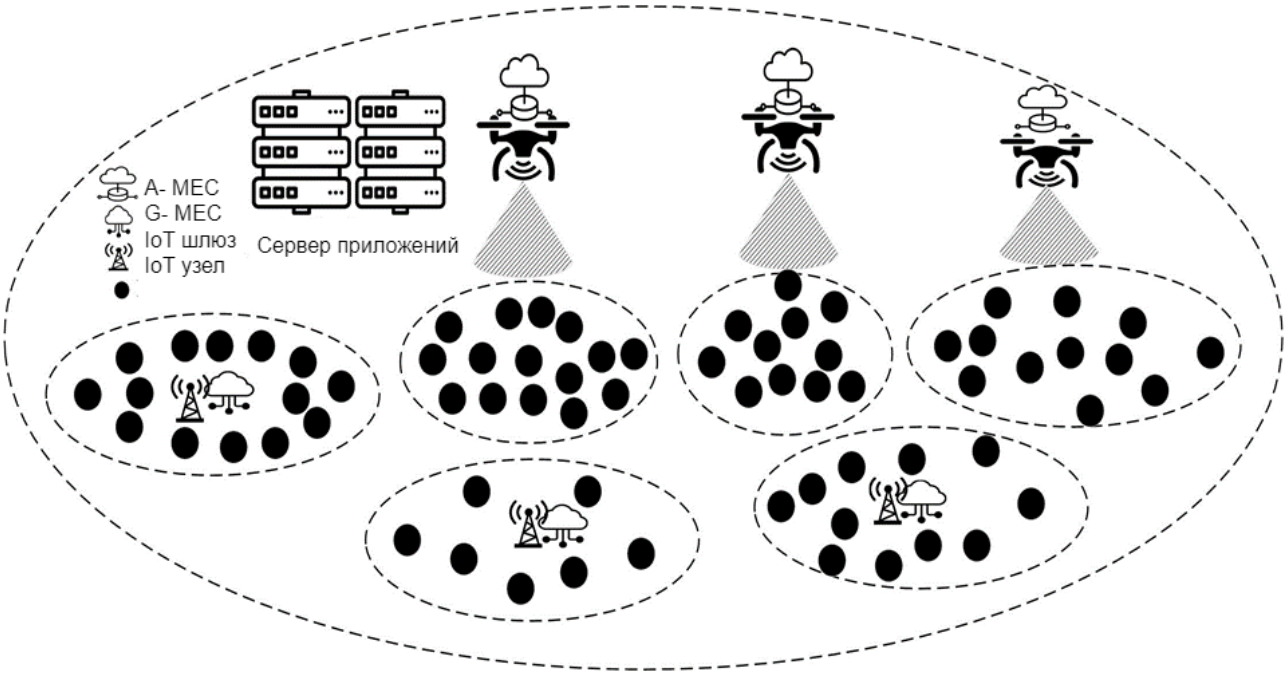


Рисунок 3.1.1 - Разработанная модель сети.

А) Наземный сегмент.

Наземный сегмент представляет собой сеть Интернета вещей с высокой плотностью или сверхвысокой плотностью с распределенными оконечными устройствами. Это четырехуровневая сеть, для обеспечения функционирования которой используется технология распределенных граничных вычислений.

Первый уровень включает в себя распределенные устройства IoT, например датчики и исполнительные механизмы. Эти устройства используются

для измерения параметров окружающей среды. Сценарий высокоплотного и сверхплотного развертывания предполагает, что таких устройств будет очень большое количество, они могут быть распределены по широкому диапазону областей, поддерживают различные интерфейсы, в том числе интерфейс «воздух-земля» (air-to-ground, A2G) с беспилотными летательными аппаратами БПЛА.

Наземные распределённые серверы граничных вычислений G-MEC (Ground MEC) представляют собой второй уровень такой сети.

Третий уровень образуется шлюзами сети Интернета Вещей. Этот уровень представляет собой интерфейс между оконечными устройствами и облачным сервером сети Интернета Вещей [201]. При этом предполагается, что все шлюзы IoT в рассматриваемой сети подключены и к воздушной плоскости.

Четвертый уровень – это прикладной уровень, а именно: облачный сервер приложений. Все собранные данные передаются через IoT-шлюзы или воздушный сегмент на сервер приложений, где данные анализируются и хранятся. Принятые данные предварительно обрабатываются на оконечных и граничных устройствах, что обеспечивается на основе разработанного метода выгрузки трафика.

Б) Воздушный сегмент.

Воздушный сегмент включает в себя множество беспилотных летательных аппаратов, например, микродронов и квадрокоптеров, развернутых для поддержки сетей Интернета Вещей высокой и сверхвысокой плотности. Каждый БПЛА имеет два интерфейса связи: «воздух-земля» A2G (Air-to-Ground) и «воздух-воздух» A2A (Air-to-Air). A2G — это интерфейс, используемый для связи с наземным сегментом, а интерфейс A2A используется для связи между

беспилотными аппаратами.

Каждый БПЛА имеет граничный сервер A-MEC(Air MEC) для предоставления вычислительных ресурсов наземной сети. A-MEC представляет собой микро облачный сервер обеспечивающий процедуру выгрузки трафика.

Каждый БПЛА обслуживает кластер оконечных устройств Интернета Вещей. При этом устройства IoT решают присоединиться ли к кластеру узла БПЛА или связаться с IoT-шлюзом на основе индикатора уровня принимаемого сигнала (RSSI). Оконечное устройство IoT решает также к какому БПЛА присоединиться или присоединиться к шлюзу IoT, сравнивая их RSSI.

3.1.2 Математическая модель исследуемой сети и ее составляющих

Прежде всего, определим необходимые для исследования параметры и переменные. В таблице 3.1 приведены обозначения всех рассматриваемых параметров и переменных.

Таблица 3.1.2.1 - Параметры и переменные

Обозначение	***Описание***
D	Набор БПЛА, развернутый в воздушной плоскости
R_{MEC-A}	Вычислительные ресурсы сервера МЭК-А
N	Общее число развернутых БПЛА в воздухе
h	Высота БПЛА
x, y	Характеристики местоположения БПЛА (двумерные наземные координаты)
x_{Di}, y_{Di}	Характеристики местоположения i -го БПЛА, D_i (двумерные наземные координаты)
h_{Di}	Высота i- го БПЛА D_i
P^{Di}	Текущее местоположение i- го БПЛА, $Д_i$
T^{Di}	Траектория БПЛА D_i
G_x^{Di}	Множество x местоположений БПЛА D_i, формирующих траекторию T^{Di}
G_y^{Di}	Множество y местоположений БПЛА D_i, формирующих траекторию T^{Di}
I	Набор оконечных устройств IoT, развернутых в наземном сегменте

M	Общее количество устройств IoT, развернутых в наземном сегменте
P^{Ij}	Текущее местоположение IoT-устройства I_j
x_{Ij}, y_{Ij}	Характеристики местоположения устройства IoT I_j (местоположение xy)
C	Набор сформированных кластеров
CM_j^{Ci}	j - й член i- го кластера (C_i)
K_i	Общее количество членов i -го кластера (C_i)
G	Набор шлюзов IoT, развернутых в наземном сегменте
W	Общее количество IoT-шлюзов, развернутых в наземном сегменте
$L_{Ii,Dj}$	Расстояние между оконечным устройством IoT I_i и БПЛА D_j
$L_{Ii,Gj}$	Расстояние между оконечным устройством IoT I_i и шлюзом IoT G_j
dDi	Продолжительность полета БПЛА D_i
h_{Gj}	Высота IoT-шлюза G_i
x_{Gi}, y_{Gi}	x-y местоположение шлюза IoT G_i
t_s	Продолжительность временного интервала
γ_{nts}^{Di}	Коэффициент усиления канала БПЛА D_i для n- го кадра
γ_0	Принимаемая мощность на эталонном расстоянии один метр
$\Omega_{C\text{-}Ij,Di}$	Энергия, потребляемая в восходящем соединении между IoT-устройством I_j и БПЛА D_i
$\Omega_{C\text{-}Di,Ij}$	Энергия, потребляемая в нисходящем соединении между БПЛА D_i и IoT-устройством I_j
σ^2	Мощность шума
B	Ширина полосы шума
S	Количество данных в задаче, т. е. размер в байтах
Z	Объем данных, полученных в результате обработки выгруженной задачи
$\Omega_{M\text{-}Di}$	Энергозатраты БПЛА D_i на механические операции
V_{Di}^{nts}	Вектор скорости БПЛА D_i на n-ом кадре
m_{Di}	Масса БПЛА D_i
UAV_{ID}	Идентификация БПЛА
$D_{clus,Ii}$	Решение IoT-устройства I_i присоединиться к кластеру БПЛА
NC_{bit}	Требуемые (CPU) циклы для обработки одного бита данных,
$D_{h\text{-}IoT}$	Время, необходимое для обработки вычислительной задачи локально на устройстве IoT.
$\Omega_{exec\text{-}IoT}$	Энергия, необходимая для обработки вычислительной задачи локально на устройстве IoT.
$\Omega_{available\text{-}aft\text{-}IoT}$	Оставшаяся энергия IoT-устройства после обработки задачи с использованием IoT-ресурсов
$\Omega_{available\text{-}bef\text{-}IoT}$	Доступный уровень энергии устройства IoT перед обработкой выгруженной задачи.
$\Omega_{Th\text{-}IoT}$	Пороговый уровень энергии оконечного устройства IoT
$\delta_{alloc\text{-}IoT}$	Ресурсы обработки оконечного устройства IoT, выделенные для вычислительной задачи

ω_{IoT}	Общие ресурсы обработки оконечного устройства IoT
$\beta_{F\text{-}IoT}$	Бинарное решение о выгрузке определяется устройством IoT
$\beta_{\Omega\text{-}IoT}$	Бинарное энергетическое решение о выгрузке, принятое устройством IoT
$B_{D\text{-}IoT}$	Бинарное решение о выгрузке по времени определяется устройством IoT
τ	Пороговое время вычислительной задачи, которое поддерживает требования QoS.
$D_{h\text{-}G\text{-}MEC}$	Общее время, необходимое для обработки вычислительной задачи на сервере G-MEC, запрошенное оконечным устройством IoT.
$D_{exec\text{-}G\text{-}MEC}$	Общее время, необходимое для выполнения вычислительной задачи на сервере G-MEC.
$\delta_{alloc\text{-}G\text{-}MEC}$	Ресурсы обработки блока G-MEC, выделенные для вычислительной задачи
$\omega_{G\text{-}MEC}$	Общие ресурсы обработки блока G-MEC
$T_{up\text{-}Ij\text{-}Gi}$	Общее время восходящей линии связи между оконечным устройством IoT I_j и шлюзом IoT G_i
$T_{up\text{-}Gi\text{-}Ij}$	Общее время нисходящего канала между шлюзом IoT G_i и оконечным устройством IoT I_j
$T_{u\text{-}tx\text{-}Ij\text{-}Gi}$	Общая задержка передачи по восходящей линии связи между оконечным устройством IoT I_j и шлюзом IoT G_i
$T_{d\text{-}tx\text{-}Gi\text{--}Ij}$	Общая задержка передачи по нисходящей линии связи между шлюзом IoT G_i и оконечным устройством IoT I_j
T_{pro}	Задержка распространения
$\beta_{AF\text{-}G\text{-}MEC}$	Бинарное решение о принятии запроса на выгрузку, принятое G-MEC
$D_{h\text{-}A\text{-}MEC}$	Общее время, необходимое для обработки вычислительной задачи на сервере A-MEC, выгруженной оконечным устройством IoT.
$t_{up\text{-}Ij\text{-}Di}$	Общее время восходящей линии связи, необходимое для выгрузки данных задачи с оконечного устройства IoT I_j и БПЛА D_i
$t_{dw\text{-}Di\text{-}Ij}$	Общее время нисходящей линии связи, необходимое для отправки результата с БПЛА D_i на устройство IoT I_j
$D_{exec\text{-}A\text{-}MEC}$	Общее время, необходимое для выполнения вычислительной задачи на сервере A-MEC.
$\delta_{alloc\text{-}A\text{-}MEC}$	Ресурсы обработки блока A-MEC, выделенные для вычислительной задачи
$\omega_{A\text{-}MEC}$	Общие ресурсы обработки блока A-MEC
$T_{u\text{-}tx\text{-}Ij\text{-}Di}$	Суммарная задержка передачи по восходящей линии связи между оконечным устройством IoT I_j и БПЛА D_i
$T_{d\text{-}tx\text{-}Di\text{--}Ij}$	Суммарная задержка передачи по нисходящей линии связи между БПЛА D_i и оконечным устройством IoT I_j
$\beta_{D\text{-}A\text{-}MEC}$	Бинарное решение о времени принятия запроса на выгрузку, принятое A-MEC
$\beta_{\Omega\text{-}A\text{-}MEC}$	Бинарное энергетическое решение о принятии запроса на выгрузку, принятое A-MEC.

$I()$	Индикатор режима
$\Omega_{Th\text{-}IoT}$	Пороговый уровень энергии оконечного устройства IoT
$\Omega_{available\text{-}aft\text{-}A\text{-}MEC}$	Остаток энергии БПЛА после выполнения задачи с использованием ресурсов A-MEC
$\Omega_{Th\text{-}A\text{-}MEC}$	Пороговый уровень энергии БПЛА
$\Omega_{exec\text{-}A\text{-}MEC}$	Потребление энергии из-за вычислений при выполнении выгруженной задачи в A-MEC
$\beta_{AF\text{-}A\text{-}MEC}$	Бинарное решение о принятии запроса на выгрузку, принятое A-MEC
Φ	Набор позиций БПЛА D_i в каждом временном интервале
α_D	Весовой коэффициент времени
α_Ω	Весовой коэффициент энергии
V_{max}	Максимальная скорость БПЛА
$x_{min\text{-}Di}$, $x_{max\text{-}Di}$, $y_{min\text{-}Di}$, and $y_{max\text{-}Di}$	Минимальное и максимальное положения в координатах xy БПЛА D_i.
$\delta_{alloc\text{-}A\text{-}MEC\text{-}max}$	Максимальные вычислительные ресурсы A-MEC, которые могут быть выделены для вычислений
$\delta_{alloc\text{-}A\text{-}MEC\text{-}Tj}$	Ресурсы обработки A-MEC, выделенные для вычислений, связанных с задачей T_j
P^i	Вектор положения сальп
n_s	Общее количество сальп
F	Положение источника энергии
U_b	Верхняя граница, связанная с каждым измерением в пространстве поиска
L_b	Нижняя граница, связанная с каждым измерением в пространстве поиска
C_1, C_2, and C_3	Коэффициенты алгоритма роя сальп (SSA)
Δ	Текущая версия SSA
Δ_{max}	Максимальное количество итераций SSA
m	Количество измерений в пространстве поиска
$R_{b\text{-}Di\text{-}nts}$	Достижимая скорость выгрузки в n-м временном интервале

В воздушном сегменте создается набор беспилотных летательных аппаратов D, определенный в (1). Каждый БПЛА имеет граничный сервер MEC-A с возможностью ресурсов обработки $R_{MEC\text{-}A}$.

$$D = \{D_1, D_2, D_3, \cdots, D_N\} \qquad \forall N \in \mathbb{R} \qquad (1)$$

где N – общее количество развернутых БПЛА в воздушной плоскости. Трехмерная декартова система координат с координатами (x, y, h)

используется для определения местоположения БПЛА и устройств IoT, как показано на рисунке 3.1.2. Местоположение БПЛА характеризуется двумерной наземной координатой x и y и высотой h. Таким образом, текущее местоположение БПЛА D_i равно P^{Di} и характеризуется x_{Di}, y_{Di} u h_{Di}. Оконечные устройства IoT расположены в плоскости xy в разных местах. Каждый БПЛА движется по траектории T^{Di} с фиксированной высотой $\underline{h_{Di}}$, имея два набора местоположений xy, $G_x{}^{Di}$ и $G_y{}^{Di}$.

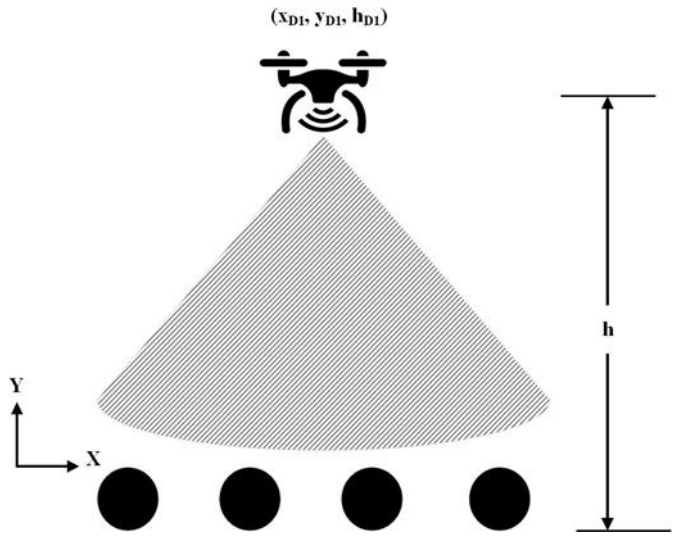


Рисунок 3.1.2 - БПЛА в трехмерных координатах

Сети IoT используют набор оконечных устройств I, определенный в (2). Расположение устройств IoT I_j равно P^{Ij} и характеризуется x_{Ij} и y_{Ij}. Некоторые устройства сгруппированы в кластеры БПЛА, а остальные узлы взаимодействуют непосредственно со шлюзами. Набор развернутых кластеров равен C, определенному в (3).

$$I = \{I_1, I_2, I_3, \cdots, I_M\} \qquad \forall M \in \mathbb{R} \quad (2)$$

$$C = \{C_1, C_2, C_3, \cdots, C_N\} \qquad \forall N \in \mathbb{R} \qquad (3)$$

$$C_i = \left\{D_i, {CM_1}^{Ci}, {CM_2}^{Ci}, \cdots, {CM_{ki}}^{Ci}\right\} \quad \forall K_i \in \mathbb{R}, C_i \subset C,$$

$$D_i \in D\ ,\ {CM_j}^{Ci} \in I \qquad (4)$$

где M – общее количество развернутых устройств IoT, CM_j^{Ci} — j-й член i-го кластера, а K_i – общее количество членов i-го кластера.

Набор шлюзов IoT равен G и определяется следующим образом:

$$G = \{G_1, G_2, G_3, \cdots, G_W\} \qquad \forall W \in \mathbb{R} \qquad (5)$$

где W — общее количество шлюзов IoT.

Расстояние между оконечным устройством Интернета вещей I_i и БПЛА D_j равно $L_{Ii,Dj}$ в соответствии с (6), а расстояние до шлюза G_j, равно $L_{Ii,Gj}$ как это определено в (7)

$$L_{I_{i,D_j}} = \sqrt{\left(x_{Ii} - x_{Dj}\right)^2 + \left(y_{Ii} - y_{Dj}\right)^2 + {h_{D_j}}^2} \quad (6)$$

$$L_{I_{i,G_j}} = \sqrt{\left(x_{Ii} - x_{Gj}\right)^2 + \left(y_{Ii} - y_{Gj}\right)^2 + {h_{G_j}}^2} \qquad (7)$$

Каждый БПЛА имеет продолжительность полета d^{Di}, которая зависит от энергии, потребляемой в процессах связи, вычислений и полета. Процессы, происходящие на одном БПЛА, отличаются от процессов происходящих на

другом; таким образом, продолжительность полета разных БПЛА неодинакова. Общее время полета разбивается на временные интервалы с длительностью интервала t_s, при этом процессы связи и вычислений выполняются в этих интервалах. В течение каждого временного интервала t_s местоположение БПЛА аппроксимируется как фиксированное. Набор местоположений *xy* соответствует траектории БПЛА D_i и может быть определен следующим образом:

$$G_x{}^{Di} = \{x_{Di}{}^{ts}, x_{Di}{}^{2ts}, x_{Di}{}^{3ts}, \cdots, x_{Di}{}^{qts}\} \quad \forall D_i \in D(8)$$

$$G_y{}^{Di} = \{y_{Di}{}^{ts}, y_{Di}{}^{2ts}, y_{Di}{}^{3ts}, \cdots, y_{Di}{}^{qts}\} \quad \forall D_i \in D(9).$$

При применении неортогонального множественного доступа (NOMA) все оконечные устройства одновременно используют общую длительность кадра t_s для передачи данных по восходящему и нисходящему каналам. Предполагается, что связь между БПЛА и устройствами IoT осуществляется в пределах прямой видимости. Таким образом, усиление канала для *n*-го кадра можно рассчитать следующим образом.

$$\gamma_{nts}{}^{Di}(x_{Di}{}^{nts}, y_{Di}{}^{nts}) = \frac{\gamma_0}{\left(x_{Di}{}^{nts} - x_{Ij}{}^{nts}\right)^2 + \left(y_{Di}{}^{nts} - y_{Ij}{}^{nts}\right)^2} \quad \forall n \in \mathbb{R}, I_j \in I,\ D_i \in D(10)$$

где γ_0 — коэффициент усиления канала на эталонном расстоянии в один метр, когда передаваемая мощность составляет один ватт.

3.1.3 Модель энергопотребления

Потребление энергии БПЛА происходит тремя способами: потребление энергии из-за вычислений, потребление энергии из-за связи и потребление энергии из-за механических операций.

Энергия, потребляемая из-за вычислений, рассчитывается на основе

количества выгружаемых данных. Энергия, потребляемая из-за связи, основана на объеме данных, передаваемых по восходящей и нисходящей линиям связи [202]. Энергия, потребляемая в восходящем соединении между IoT-устройством I_j и беспилотным летательным аппаратом D_i, равна $\Omega_{\text{C-Ij,Di}}$ и рассчитывается на основе объема выгружаемых данных S, как отмечено в (11). При этом энергия, потребляемая в нисходящем соединении между беспилотным летательным аппаратом D_i и устройством IoT I_j, составляет $\Omega_{\text{C-Di,Ij}}$ и рассчитывается на основе объема данных, не связанных с обработкой запроса на выгрузку, Z, как в (12). Канал моделируется аддитивным белым гауссовым шумом мощности σ^2 и шириной полосы B. Энергия, потребляемая при связи по восходящей и нисходящей линиям связи, рассчитывается следующим образом.

$$\Omega_{C-Ij,Di}(s, {x_{Di}}^{nts}, {y_{Di}}^{nts}) = \frac{\sigma^2 t_s\left(2^{S/Bt_s}-1\right)}{{\gamma_{nts}}^{Di}({x_{Di}}^{nts}, {y_{Di}}^{nts})} \cdot \frac{1}{\left(1-\left(2^{S/Bt_s}-1\right)\right)} \quad \forall n \in \mathbb{R}, I_j \in I,\ D_i \in D \qquad (11)$$

$$\Omega_{C-Di,Ij}\left(Z, {x_{Di}}^{nts}, {y_{Di}}^{nts}\right) = \frac{\sigma^2 t_s\left(2^{Z/Bt_s}-1\right)}{{\gamma_{nts}}^{Di}({x_{Di}}^{nts}, {y_{Di}}^{nts})} \cdot \frac{1}{\left(1-\left(2^{Z/Bt_s}-1\right)\right)} \quad \forall n \in \mathbb{R}, I_j \in I,\ D_i \in D \qquad (12)$$

Потребление энергии из-за механических операций, $\Omega_{\text{M-Di}}$, смоделировано в [203] и может быть рассчитано для длительности временного интервала на основе вектора скорости БПЛА V и массы БПЛА m_{Di} следующим образом:

$$\Omega_{M-Di}({x_{Di}}^{nts}, {y_{Di}}^{nts}) = 0.5 m_{Di} t_s \|{v_{Di}}^{nts}\|^2 \ \forall n \in \mathbb{R}, D_i \in D \qquad (13)$$

$$\|{v_{Di}}^{nts}\| = \sqrt{({x_{Di}}^{nts} - {x_{Di}}^{nts+1})^2 + ({y_{Di}}^{nts} - {y_{Di}}^{nts+1})^2} \Big/ t_s \quad \forall\, n \in \mathbb{R},\ D_i \in D \qquad (14)$$

3.1.4 Формирование кластера

Кластеры формируются по раундам, по две фазы на каждый раунд; фаза формирования кластера и фаза обеспечения связи. В начале каждого раунда каждый БПЛА транслирует сообщение о соединении в формате, представленном на рисунке 3.1.4.

Сообщение о присоединении				
Идентификация				**Технические характеристики канала**
Идентификатор, ID	Расположение			
$БПЛА_{ID}$	x_{Di}	y_{Di}	h_{Di}	

Рисунок 3.1.4 - Формат сообщения о присоединении

Сообщение о присоединении содержит идентификацию БПЛА: идентификатор БПЛА (UAV_{ID}), координаты местоположения БПЛА (x,y и h) и технические характеристики канала связи. Оконечные устройства IoT получают сообщения о присоединении от БПЛА и решают, следует ли присоединиться к БПЛА или взаимодействовать со шлюзом IoT.

Оконечное устройство IoT вычисляет расстояние до БПЛА $L_{Ii,Dj}$ на основе (6) и расстояние до IoT-шлюза $L_{Ii,Gj}$ на основе (7) и выбирает кратчайший путь. Устройства IoT, которые получают несколько сообщений о присоединении, выбирают то, у которого кратчайший путь связи, и сравнивают этот путь с путем

к ближайшему шлюзу IoT. Более того, IoT-устройства вне зоны действия шлюзов IoT выбирают ближайший БПЛА для присоединения. Таким образом, решение о присоединении к кластеру БПЛА D_{clus} принимается в соответствии с (15).

$$D_{clus-I_{i,}} = I\left(L_{I_{i,G_j}}, \min\left(L_{I_{i,D_j}}\right)\right) =$$

$$\begin{cases} 0 & IF\ \ (L_{I_{i,G_j}} \leq \min\left(L_{I_{i,D_j}}\right)) \\ 1 & IF\ \ (L_{I_{i,G_j}} > \min\left(L_{I_{i,D_j}}\right)) \vee (L_{I_{i,G_j}} \to \infty) \end{cases} \quad \forall\ D_j \in D,\ I_i \in I \tag{15}$$

Оконеченные устройства IoT при наличии кратчайшего пути к выбранному БПЛА отправляют ответное сообщение с положительным решением о кластеризации. Таким образом, формируются кластеры, в которых беспилотный летательный аппарат выступает в роли головного узла кластера, а этап формирования кластера заканчивается. Оконечные устройства IoT взаимодействуют и обмениваются данными с головным узлом кластера (БПЛА), который управляет кластером на фазе обеспечения связи.

3.1.5 Метод выгрузки трафика

Рассмотрим разработанный метод выгрузки трафика с наземной сети на мобильные серверы граничных вычислений на БПЛА. Оконечные устройства IoT обрабатывают данные, используя имеющиеся у них ресурсы, т. е. осуществляется локальное выполнение требуемых задач. Такое решение допустимо только для задач, требующих ограниченных ресурсов, в том числе и энергоресурсов. Задачи, требующие больших ресурсов, следует переносить на граничные серверы наземного сегмента или на граничные мобильные серверы

воздушного сегмента. Разработанный метод выгрузки является бинарным, поскольку предполагается, что БПЛА используются только для поддержки высокоплотных и сверхплотных сетей Интернета Вещей и не имеют иных локальных задач. На рисунке 3.1.5 представлен алгоритм реализации разработанного метода выгрузки с указанием основных шагов. Оконечные устройства и блоки MEC используют ранее разработанную структуру, представленную в [204].

БПЛА можно рассматривать как мобильные серверы граничных вычислений, которые получают выгруженные задачи от оконечных устройств IoT и либо обрабатывают их сами, либо переносят на другие граничные вычислительные узлы. Выгрузка трафика между наземным и воздушным сегментами осуществляется по восходящим и нисходящим интерфейсам с использованием дуплекса с частотным разделением (FDD). NOMA используется для улучшения связности и спектральной эффективности обслуживаемых кластеров.

Разработанный метод выгрузки состоит из трех уровней, каждый из которых представляет собой сервер граничных вычислений. На оконечных устройствах есть программный профилировщик, который вычисляет спецификацию задачи, включая объем данных, т. е. размер в байтах S и требуемые циклы центрального процессора (CPU) для обработки одного бита данных NC_{bit}.

Рисунок 3.1.5 - Алгоритм выгрузки трафика для разработанного метода

Планировщик ресурсов оконечного устройства предоставляет механизму принятия решений доступные в настоящее время ресурсы для решаемой задачи. Механизм принятия решений оконечного устройства рассчитывает необходимое время для обработки задачи $D_{h\text{-}IoT}$, необходимую энергию для обработки задачи $\Omega_{exec\text{-}oT}$ и остаточную энергию после обработки задачи с использованием

ресурсов IoT, $\Omega_{available\text{-}aft\text{-}IoT}$ следующим образом:

$$D_{h-IoT} = \frac{S.NC_{bit}}{\delta_{alloc-IoT}} \quad , \delta_{alloc-IoT} \in \omega_{IoT} \qquad (16)$$

$$\Omega_{exec-IoT} = S.NC_{bit}\mathrm{E}_{Cyc-IoT} \qquad (17)$$

$$\Omega_{available-aft-IoT} = \Omega_{available-bef-IoT} - \Omega_{exec-IoT} \qquad (18)$$

На основе рассчитанных параметров качества обслуживания (QoS), задержки и энергопотребления принимается решение об обработке на месте или передачи задачи на граничный сервер MEC. Бинарное решение выгрузки $\mathrm{B}_{\text{F-IoT}}$ рассчитывается следующим образом:

$$\beta_{\Omega-IoT} = I\left(\Omega_{available-aft-IoT}, \Omega_{Th-IoT}\right) = \begin{cases} 0 & IF \ (\Omega_{available-aft-IoT} \leq \Omega_{Th-IoT}) \\ 1 & IF \ (\Omega_{available-aft-IoT} > \Omega_{Th-IoT}) \end{cases} \qquad (19)$$

$$\beta_{\mathrm{D}-IoT} = I(D_{h-IoT}, \tau) = \begin{cases} 0 & IF \ (D_{h-IoT} \leq \tau) \\ 1 & IF \ (D_{h-IoT} > \tau) \end{cases} \qquad (20)$$

$$\beta_{\mathrm{F}-IoT} = \beta_{\Omega-IoT} \vee \beta_{\mathrm{D}-IoT} \qquad (21)$$

Пусть доступных ресурсов, включая энергопотребление оконечного устройства, достаточно для выполнения задачи. В этом случае бинарное решение о выгрузке минимально, т. е. $\beta_{F\text{-}IoT} = 0$, и устройство не выполняет

выгрузку. Однако, если ресурсов недостаточно, включая ограничения по энергопотреблению устройства, оконечное устройство решает передать задачу, т.е. $\beta_{F\text{-}IoT} = 1$, на соответствующий граничный сервер MEC.

Устройство, подключенное к шлюзу IoT, т. е. не являющееся членом кластера БПЛА, запрашивает выгрузку в G-MEC. Сообщение с запросом содержит необходимую информацию для обработки задачи и идентификации устройства. G-MEC обрабатывает полученные запросы на выгрузку и отправляет информацию о задаче в механизм принятия решений, чтобы решить, принять или отклонить запрос. Это зависит от доступных вычислительных ресурсов G-MEC и ограничений по QoS запрашиваемой задачи. Планировщик ресурсов G-MEC предоставляет механизму принятия решений доступные ресурсы для принятия решения о приеме или отклонении запрошенной задачи.

Механизм принятия решений G-MEC вычисляет время, необходимое для обработки задачи по запросу $D_{hG\text{-}MEC}$, следующим образом.

$$D_{exec-G-MEC} = \frac{S.NC_{bit}}{\delta_{alloc-G-MEC}} \quad , \delta_{alloc-G-MEC} \in \omega_{G-MEC} \quad (22)$$

$$D_{h-G-MEC} = D_{exec-G-MEC} + T_{up-Ij-Gi} + T_{dw-Gi-Ij} \quad (23)$$

$$T_{up-Ij-Gi} = T_{u-tx-Ij-Gi} + T_{pro} \quad (24)$$

$$T_{dw-Gi-Ij} = T_{d-tx-Gi-Ij} + T_{pro} \quad (25)$$

Затем механизм принятия решений определяет бинарное решение о принятии или отклонении запроса на выгрузку $\beta_{\text{AF-G-MEC}}$ путем сравнения общего времени, необходимого для обработки запрошенной задачи $D_{hG\text{-}MEC}$, со временем необходимым для обеспечения параметров QoS. При отрицательном решении задача завершается.

$$\beta_{\text{AF}-\text{G}-MEC} = I(D_{h-G-MEC}, \tau) = \begin{cases} 1 & IF \ \ (D_{h-G-MEC} \leq \tau) \\ 0 & IF \ \ (D_{h-G-MEC} > \tau) \end{cases} \quad (26)$$

3.1.6 Выгрузка в воздушный сегмент.

В случае, если механизм принятия решений устройств – членов кластера, подключенных к БПЛА, отклоняет локальное выполнение задачи, то устройства запрашивают выгрузку у соответствующего БПЛА через интерфейс восходящей линии связи. Сервер граничных вычислений A-MEC беспилотного летательного аппарата обрабатывает полученные запросы на выгрузку от обслуживаемых оконечных устройств, путем извлечения информации об устройстве и выполняемой задаче и предоставляет их механизму принятия решений A-MEC.

Механизм принятия решений также получает информацию о доступных в данный момент времени ресурсах от планировщика ресурсов и вычисляет время, необходимое для обработки запрошенной задачи, $D_{hA\text{-}MEC}$. Это время включает время восходящей линии связи, необходимое для выгрузки данных задачи в БПЛА $t_{up\text{-}Ij\text{-}Di}$, время нисходящей линии связи для отправки результата $t_{dw\text{-}Di\text{-}Ij}$ и время вычислений $t_{exec\text{-}A\text{-}MEC}$. Анализ расчетов времени передачи по восходящей и нисходящей линиям связи представлен в Приложении I к главе.

$$D_{h-A-MEC} = T_{up-Ij-Di} + T_{dw-Di-Ij} + D_{exec-A-MEC} \quad (27)$$

$$D_{exec-A-MEC} = \frac{S.NC_{bit}}{\delta_{alloc-A-MEC}} \quad , \delta_{alloc-A-MEC} \in \omega_{A-MEC} \quad (28)$$

$$T_{up-Ij-Di} = T_{u-tx-Ij-Di} + T_{pro} \qquad (29)$$

$$T_{dw-Di-Ij} = T_{d-tx-Di-Ij} + T_{pro} \qquad (30)$$

Далее механизмом принятия решений вычисляется время в двоичном формате и принимается решение о принятии или отклонении запроса на выгрузку $\beta_{DA\text{-}MEC}$ путем сравнения общего времени, необходимого для обработки запрошенной задачи, $D_{hA\text{-}MEC}$, со временем для обеспечения параметров QoS.

$$\beta_{\mathrm{D}-A-MEC} = I(D_{h-A-MEC}, \tau) = \begin{cases} 1 & IF \ (D_{h-A-MEC} \leq \tau) \\ 0 & IF \ (D_{h-A-MEC} > \tau) \end{cases} \quad (31)$$

При отрицательном решении, т. е. $\beta_{DA\text{-}MEC} = 0$, БПЛА D_i запрашивает ресурсы у соседнего БПЛА. БПЛА *Di* передает запрос на выгрузку БПЛА в рое. Связь между БПЛА осуществляется по ранее разработанной схеме маршрутизации, представленной в [205].

Кроме того, бинарное решение о принятии запроса по выгрузке $\beta_{\Omega\text{-}A\text{-}MEC}$ рассчитывается путем сравнения оставшейся энергии БПЛА D_i после выполнения запрошенной задачи $\Omega_{available\text{-}aft\text{-}A\text{-}MEC}$ с пороговым уровнем энергии БПЛА $Ом_{\ Th\ \text{-}A\text{-}MEC}$. БПЛА сначала рассчитывает расход энергии на вычисление запрашиваемой задачи $\Omega_{exec\text{-}A\text{-}MEC}$, как в (32), и расход энергии на передачу результатов расчета $\Omega_{\text{C-Di,Ij}}$, как в (12).

$$\Omega_{exec-A-MEC} = S.NC_{bit}\mathrm{E}_{Cyc-A-MEC} \quad (32)$$

$$\Omega_{available-aft-A-MEC} = \Omega_{available-bef-A-MEC} - \Omega_{exec-A-MEC} - \Omega_{C-Di,Ij}\left(R, {x_{Di}}^{nts}, {y_{Di}}^{nts}\right) \quad (33)$$

$$\beta_{\Omega-A-MEC} = I\left(\Omega_{available-aft-A-MEC}, \Omega_{Th-A-MEC}\right) = \begin{cases} 1 & IF \ (\Omega_{available-aft-A-MEC} \leq \Omega_{Th-A-MEC}) \\ 0 & IF \ (\Omega_{available-aft-A-MEC} > \Omega_{Th-A-MEC}) \end{cases} (34)$$

$$\beta_{\mathrm{AF}-A-MEC} = \beta_{\Omega-A-MEC} \wedge \beta_{\mathrm{D}-A-MEC} \, (35)$$

При отказе о приеме запроса на выгрузку БПЛА запрашивает выгрузку у соседнего БПЛА в рое с доступными ресурсами. Беспилотный летательный аппарат D_i передает запрос на выгрузку соседним беспилотным летательным аппаратам. Этот запрос принимается и обрабатывается с помощью описанного ранее алгоритма. Соседние узлы отправляют ответы для принятия или отклонения запросов на выгрузку, а БПЛА завершает задачу или выгружает ее на основе полученных ответов.

3.1.7. Оптимизация среднего энергопотребления и задержки для обработки задач, выгруженных на БПЛА, с помощью хаотического роя сальп

Рассмотренный ранее метод выгрузки в части потребляемой энергии и задержки, необходимых для выполнения вычислительных задач, в основном зависит от местоположения БПЛА. Основная цель этого подраздела оптимизировать энергопотребление и минимизировать среднюю задержку для выполнения задач выгрузки. Задача оптимизации определяется следующим образом:

$$\min_{\varphi}\left(\sum_{j=1}^{k}\alpha_D {D_{h-A-MEC}}^{Di,Ij}+\sum_{j=1}^{k}\alpha_\Omega\left(\Omega_{C-Di,Ij}\left(Z,{x_{Di}}^{nts},{y_{Di}}^{nts}\right)+\Omega_{C-Ij,Di}\left(s,{x_{Di}}^{nts},{y_{Di}}^{nts}\right)+\Omega_{exec-A-MEC}\right)\right)(36)$$

При ограничениях:

C1:

$$\sqrt{({x_{Di}}^{nts}-{x_{Di}}^{nts+1})^2+({y_{Di}}^{nts}-{y_{Di}}^{nts+1})^2}\Big/_{t_s}\leq V_{max}\quad\forall\, n\in\mathbb{R},\ D_i\in D\qquad(37)$$

C2:

$$X_{min-Di}<X_{Di}<X_{max-Di}\qquad\forall\ D_i\in D(38)$$

C3:

$$Y_{min-Di}<Y_{Di}\leq Y_{max-Di}\qquad\forall\ D_i\in D\qquad(39)$$

C4:

$$\sum_{j=1}^{N_{f-Di}}\delta_{alloc-A-MEC-Tj}\leq\delta_{alloc-A-MEC-max}\quad\forall\ \delta_{alloc-A-MEC-Tj},\delta_{alloc-A-MEC-max}\in\omega_{A-MEC}\qquad(40)$$

C5:

$$\beta_{\mathrm{AF}-A-MEC}\in\{0,1\}\qquad(41)$$

C6:

$$\sum_{j=1}^{N_{f-Di}} \beta_{\mathrm{AF}-A-MEC-Tj} \leq k_i \qquad (42)$$

где Ф — множество позиций БПЛА D_i в каждом временном интервале, α_D и α_Ω – весовые коэффициенты времени и энергии соответственно, $\delta_{alloc\text{-}A\text{-}MEC\text{-}max}$ –максимальные вычислительные ресурсы A -MEC, которые могут быть выделены для вычислений, V_{max} – максимальная скорость БПЛА, а $x_{min\text{-}Di}$, $x_{max\text{-}Di}$, $y_{min\text{-}Di}$, $y_{max\text{-}Di}$ – минимальные и максимальные координаты xy БПЛА *Di.*

Первое ограничение C1 вводится для того, чтобы скорость БПЛА в любой момент времени не превышала максимальной. Ограничения C2 и C3 вводятся для обеспечения того, чтобы каждый БПЛА покрывал только запланированную область полета и избегал столкновений между беспилотными летательными аппаратами роя. C4 гарантирует, что вычислительные ресурсы, направленные на обработку различных параллельно выполняемых задач, меньше, чем максимальные ресурсы БПЛА, выделенные для вычислений. Ограничение C5 гарантирует, что решение о выгрузке является бинарным. Решением является либо единица, либо ноль. Ограничение C6 гарантирует, что сумма запросов на выгрузку в кластере не превысит количество участников кластера.

ХАОТИЧЕСКИЙ РОЙ САЛЬП ДЛЯ ПОЛУЧЕНИЯ ОПТИМАЛЬНЫХ ПОЗИЦИЙ БПЛА

Алгоритм роя сальп (SSA) –это основанный на популяции сальп метаэвристический метод, который имитирует поведение сальп в океанах [206]. Это относительно новый класс оптимизации роя частиц (PSO), который

имитирует семейство сальп [207]. Популяция сальп состоит из двух типов сальп: сальпы - лидера и сальпы - последователя. Сальпа - лидер является первой сальпой в цепочке и отвечает за руководство роем. Сальпы, которые следуют за сальпой - лидером к источнику пищи, являются сальпами - последователями. Каждая сальпа занимает m-мерное пространство поиска, где m обозначает количество переменных в данной задаче, аналогично другим алгоритмам на основе роя [206]. Сальпы имеют вектор положения P^i из n_s элементов, который представляет общее количество развернутых роев и определяется следующим образом.

$$P^i = \left\{ {P_1}^i, {P_2}^i, {P_3}^i, \cdots, {P_m}^i \right\} \qquad , i = 1,2,3,4, \cdots, n_s \quad (43)$$

Сальпы ищут место пищи в пространстве и в результате поиска обновляют свои позиции до тех пор, пока не достигнут источника пропитания, т. е. оптимального решения. Согласно следующим уравнениям сальпа - лидер сначала обновляет свое положение; затем все следующие сальпы выслеживают его.

$$P_j^1 = \begin{cases} F_j + C_1 \left(\left(U_{b\,j} - U_{l\,j} \right) C_2 + L_{b\,j} \right), \ C_3 \geq 0 \\ F_j - C_1 \left(\left(U_{b\,j} - U_{l\,j} \right) C_2 + L_{b\,j} \right), \ C_3 < 0 \end{cases} , \ j = 1,2,3, \cdots, m \qquad (44)$$

$$P_j^z = \frac{1}{2} \left(P_j^z + P_j^{z-1} \right) \quad 2 \leq z < n_s \, , \ \ z \in \mathbb{Z}^+ (45)$$

где F указывает положение пищи, а U_b и L_b – верхняя и нижняя границы, связанные с каждым измерением в пространстве поиска. Работа алгоритма основана на коэффициентах C_1 , C_2 и C_3 , которые используются для обновления положения сальпы при каждой итерации. Коэффициент C_1 вводится для поддержания уровня стабильности между разведкой и эксплуатацией и

определяется следующим образом.

$$C_1 = 2e^{-\left(\frac{4\Delta}{\Delta_{max}}\right)^2} \qquad (46)$$

где Δ — текущая итерация, а Δ_{max} – максимальное количество итераций. Другие коэффициенты могут быть обновлены с использованием случайных функций со значениями от 0 до 1. Для предлагаемой модели мы вводим хаотические карты для обновления коэффициентов рассматриваемого SSA [208]. Хаотическая карта является логистической и определяется следующим образом:

$$y(\Delta + 1) = ay(\Delta)[1 - y(\Delta)] \quad , \quad a = 4.0, y(0) = 0.7 \qquad (47).$$

Таким образом, C_2 может быть обновлен следующим образом:

$$C_2{}^{\Delta} = y(\Delta) \qquad (48)$$

В SSA рой сальп параллельно выполняет поиск в рабочем пространстве, чтобы получить решение, представляющее собой набор оптимальных положений роя БПЛА в течение каждого временного интервала. Алгоритм 3.1 выполняет поставленную задачу и получает оптимальное решение. Алгоритм используется во вложенном цикле с ранее представленным алгоритмом выгрузки. Алгоритм роя сальп начинается с инициализации начальных параметров SSA на основе хаотических карт, включая нижнюю границу, верхнюю границу и максимальное количество итераций. Затем он случайным образом инициализирует n_s сальпы, представляющие количество местоположений БПЛА. Приспособленность различных сальп, в том числе лидера и последователей, определяется с помощью (36). Место пропитания считается местом с наибольшей приспособленностью сальпы. Параметры SSA обновляются и в результате обновляются местоположения сальпы. Процесс обновления местоположения сальпы повторяется до тех пор, пока не будет

найдено оптимальное решение или не будет достигнуто максимальное количество итераций.

Алгоритм 3.1 - Хаотическая SSA для оптимального положения БПЛА

Initialize U_b, L_b, Δ_{max}, m, n_s
Initialize positions of salps Pj (j= 1,2, 3,…,n_s)
While ($\Delta \leq \Delta_{max}$)
    Calculate the fitness function of each salp position
    F = The best salp position
    Update the value of C_1 (using (46))
    Calculate the logistic map's current value y(Δ)
    For (m = 1: $m \leq n_s$) do
        if (m == 1)
            Update the position of the leader (P_j^1) (using (44))
        else
            Update the position of follower salp (P_j^m) (using (45))
        end if
    end for
    Adjust salps based on U_b and L_b
    Call offloading algorithm
    Update the best salp based on the results of offloading algorithm
    Δ← Δ+1
Return F

3.1.8 Результаты моделирования

Разработанные решения были промоделированы в среде NS-3 на основе симулятора LIMoSim [34]. Кроме того, в процессе оценки характеристик

разработанных решений была использована среда Cloudsim. В качестве наземной сети рассматривалась сеть IoT с 1000 оконечными устройствами и двумя шлюзами, распределенными на площади 25 квадратных километров. Воздушная сеть состояла из четырех БПЛА, находящихся на равной высоте 30 метров. В таблице 3.1.8.1 представлены рассматриваемые параметры сети и моделирования, а в таблице 3.1.8.2 представлены минимальное и максимальное положения *xy* каждого БПЛА. Расположение шлюзов IoT представлено в таблице 3.1.8.3.

Чтобы оценить характеристики разработанных решений для архитектуры распределенных граничных вычислений воздух-земля и оптимизированной модели, были рассмотрены четыре системы, которые определяются следующим образом:

- **Система (1):** представляет традиционные сети IoT без поддержки наземных или воздушных граничных вычислений.
- **Система (2):** представляет сеть IoT, поддерживаемую только наземной системой распределенных граничных вычислений. В этой системе развернуты серверы G-MEC.
- **Система (3):** представляет собой сеть IoT с поддержкой распределенных наземных и воздушных граничных вычислений. В этом случае развернуты как серверы G-MEC, так и A-MEC.
- **Система (4):** представляет собой оптимизированную систему воздух-земля. При этом и серверы G-MEC, и серверы A-MEC поддерживают сеть IoT.

Энергопотребление, задержка и доступность для решения вычислительных задач рассматриваются как показатели функционирования систем.

Таблица 3.1.8.1 – Параметры моделирования

Параметр	**Значение**
Сетевая зона	5 x 5 км ^{2}
N	4
W	2
M	5 X 5 Km^2
B	4
σ^2	2
γ_0	1000
h_{Di}	20 MHz
t_s	−60 dBm
V_{max}	−30 dB
V_{Di}	30 m
$E_{Cyc\text{-}IoT}$	45 ms
$\Omega_{Th\text{-}IoT}$	50 m/s
$E_{Cyc\text{-}A\text{-}MEC}$	ϵ (1, 50) m/s
$\Omega_{Th\text{-}A\text{-}MEC}$	1J/GHz
$\omega_{A\text{-}MEC}$	ϵ [1.0,3.0] ГГц
$\omega_{G\text{-}MEC}$	ϵ [1.5,5.0] ГГц
ω_{IoT}	ϵ [0.5,1.0] ГГц
$\tau_{app\text{-}I}$	7 мс
$\tau_{app\text{-}II}$	10 мс
$\tau_{app\text{-}III}$	16 мс
$\tau_{app\text{-}IV}$	24 мс

Таблица 3.1.8.2 – Минимальное и максимальное положение с координатами XY для каждого БПЛА

D_i	$X_{min\text{-}Di}$	$X_{max\text{-}Di}$	$Y_{min\text{-}Di}$	$Y_{max\text{-}Di}$
1	4	5	4	5
2	4	5	0	1
3	0	1	4	5
4	0	1	0	1

Таблица 3.1.8.3 – Расположение шлюзов IoT

G_i	X_{Gi}	Y_{Gi}
1	1	2,5
2	4	2,5

Потребление электроэнергии оценивалось для трех систем: традиционной сети IoT, т. е. системы (1), развитой сети IoT «воздух-земля», т. е. системы (3), и оптимизированной сети IoT «воздух-земля», т. е. системы (4).

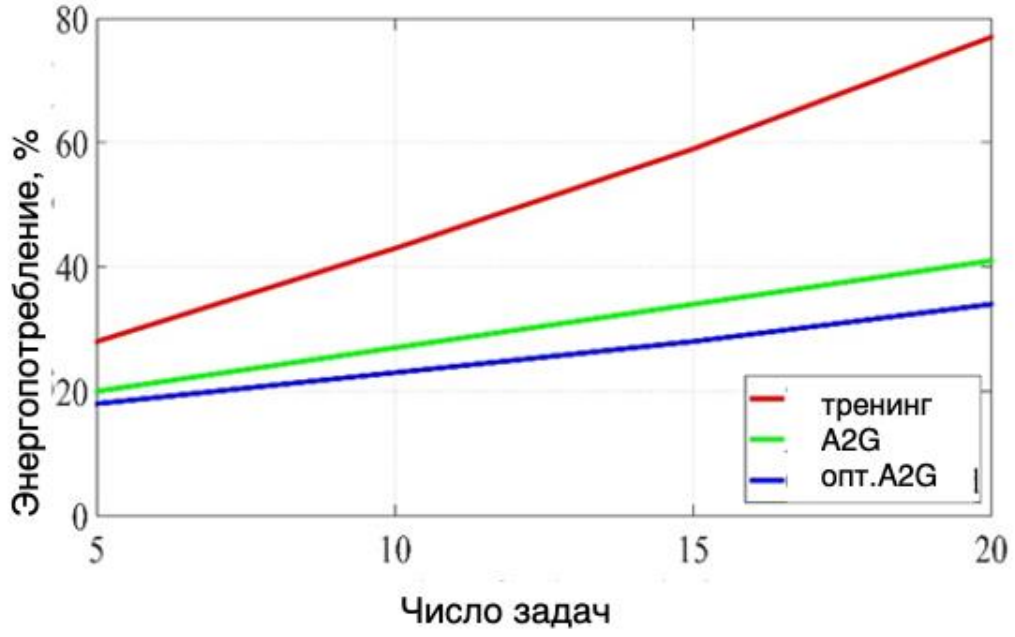


Рисунок 3.1.8.1 – Среднее энергопотребление при разном числе решаемых задач (в процентах от начального значения доступного энергопотребления)

На рисунке 3.1.8.1 представлено сравнение среднего энергопотребления в

процентах для трех систем при разных значениях числа решаемых задач для каждого оконечного устройства. Эти значения были зафиксированы при числе оконечных устройств 1000, а задачи относились к категории приложений II (обработка неподвижных изображений). Потребление энергии увеличивается по мере увеличения числа вычислительных задач на оконечных устройствах, однако разработанная модель воздух-земля сохраняет энергию для большего числа вычислительных задач по сравнению с традиционными сетями IoT. Предлагаемая модель снижает энергопотребление в среднем на 27% по сравнению с традиционной моделью IoT. Более того, оптимизированная сеть IoT «воздух-земля» позволяет экономить еще больше энергии. При этом достигается более высокая энергоэффективность в среднем на 6% по сравнению с моделью «воздух-земля».

На рисунке 3.1.8.2 показан процент энергопотребления для трех систем в четырех категориях приложений. В данном случае решались вычислительные задачи для разных типов приложений. В процессе оценки исследовались четыре разнородных типа приложений. Первый тип рассматриваемых приложений – это облегченные приложения, которые включают простые задачи, например такие, которые необходимы для обработки простых веб-страниц. Вторая категория приложений – это приложения на основе неподвижных изображений. Третья категория – это простые видео приложения, которые требуют достаточно простую обработку видео. Четвертая категория включает работу с 360-градусными панорамами и обработкой видео, например, дополненной реальности. При переходе из первой категории в четвертую задачи требуют все более высоких ресурсных возможностей, что снижает вероятность локального выполнения. Очевидно, что энергопотребление для решения задач более высоких категорий выше, чем у более низких категорий.

При этом число развернутых устройств составляло 1000 устройств, а число задач на каждом устройстве равнялось 10. Как видим, разработанная система «воздух-земля» снижает энергопотребление в среднем на 29% по сравнению с традиционными сетями, в то время как оптимизированная модель позволяет снизить потребление энергии еще на 9%. Можно заметить на рисунке 3.1.6, что разработанная система и оптимизированная система позволяют снизить энергопотребление на более высокий процент для приложений с более высокой рабочей нагрузкой, например, категории III и IV.

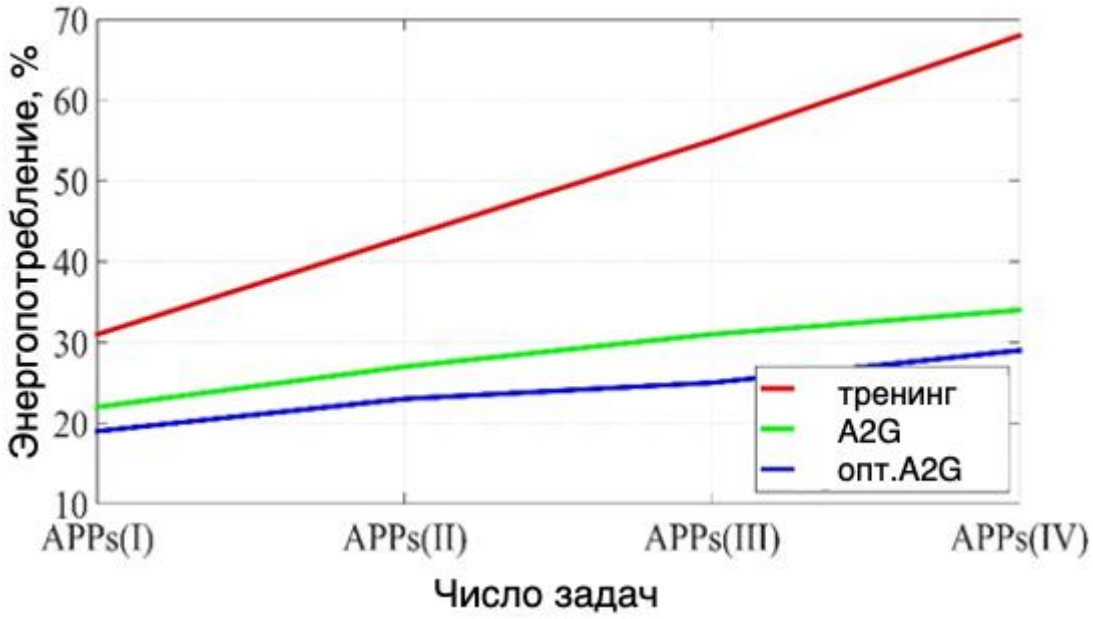


Рисунок 3.1.8.2 – Среднее энергопотребление для различных типов приложений

Влияние плотности расположения оконечных устройств на энергетические характеристики было также проанализировано для трех упомянутых систем. На рисунке 3.1.8.3 показан процент энергопотребления трех систем с разным числом оконечных устройств. Количество оконченных устройств в сети начиналась с 600 и было увеличено до 1200 устройств.

Рассматриваемые задачи относились ко второй категории, а число задач на каждом оконечном устройстве равнялось десяти. Разработанная модель «воздух-земля» снизила потребление энергии в среднем на 19% по сравнению с традиционной сетью, а оптимизированная модель обеспечила снижение еще на 4%. Для сети с большим числом развернутых оконечных устройств разработанная модель позволяет снизить потребление энергии на 27% по сравнению с традиционной моделью. Это связано с высокой нагрузкой на наземное исполнение при плотном развертывании традиционной модели.

Отметим, что разработанная схема выгрузки, естественно, снижает общее потребление энергии.

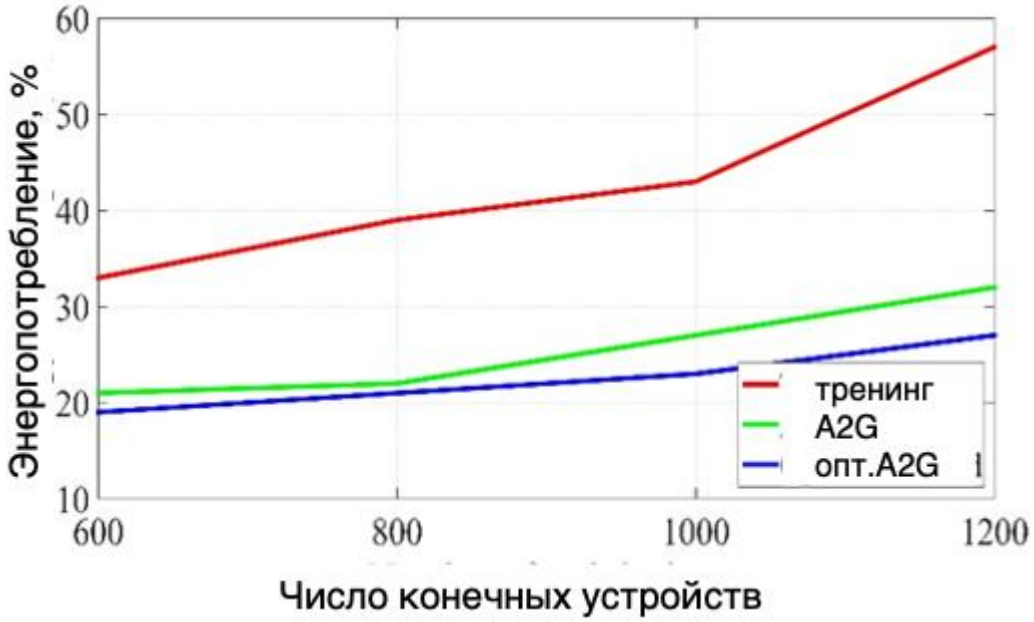


Рисунок 3.1.8.3 – Среднее энергопотребление при различном количестве развернутых узлов

Задержка – важнейший показатель сети, анализируемый в процессе оценки ее функционирования. Средняя задержка обработки рассмотренных вычислительных задач была измерена для ранее упомянутых четырех систем.

При этом изменялось число функционирующих устройств, требуемых к решению вычислительных задач и категорий приложений.

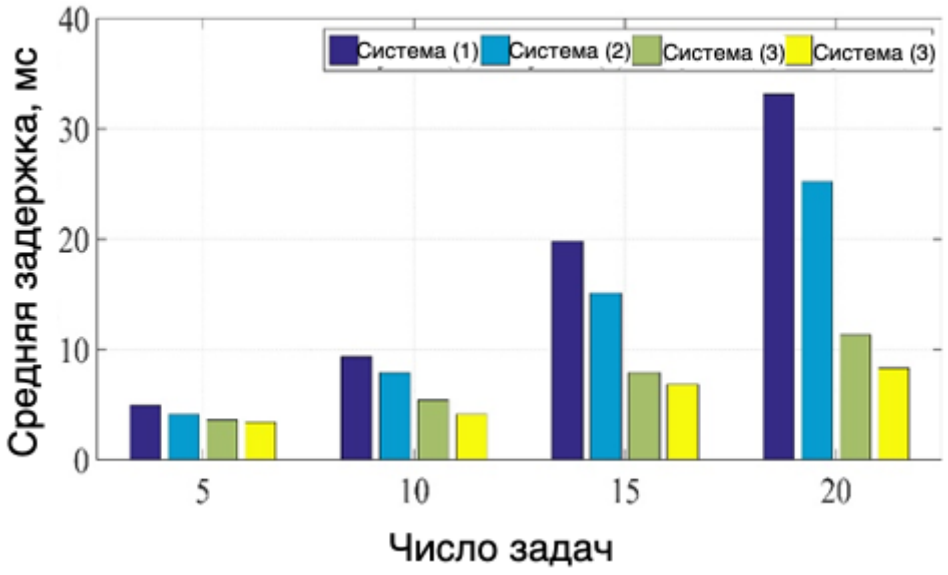


Рисунок 3.1.8.4 – Средняя задержка для четырех систем с разным числом оконечных устройств

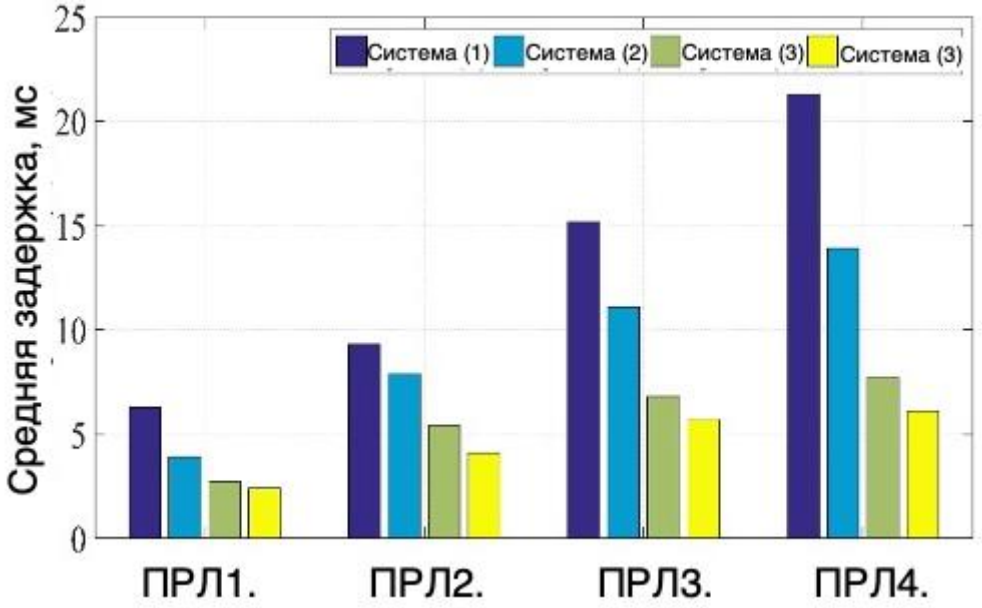


Рисунок 3.1.8.5 – Средняя задержка при различном числе вычислительных задач задач

На рисунке 3.1.8.4 представлена средняя задержка, необходимая для обработки вычислительных задач в каждой из четырех рассматриваемых систем с изменением числа задач, требуемых для выполнения каждым оконечным устройством. Эти показатели были определены для 1000 оконечных устройств и задач приложений категории II. Разработанная система «воздух-земля», т. е. система 3, обеспечивает более высокую эффективность системы по задержкам, чем традиционные системы без граничных вычислений и системы только с наземными граничными вычислениями. Это повышение эффективности возрастает с увеличением числа вычислительных задач на оконечных устройствах. Внедрение серверов A-MEC обеспечивает выгрузку вычислительных задач и, таким образом, сокращает общее время, необходимое для обработки этих задач. Более того, разработанная оптимизированная модель обеспечивает дополнительное снижение задержки в среднем еще на 2,5%.

На рисунке 3.1.8.5 показана средняя задержка обработки задач в каждой системе из четырех рассматриваемых систем для четырех категорий приложений. Эти значения были получены для сети с 1000 оконечными устройствами и десятью задачами на каждом оконечном устройстве. Разработанные система «воздух-земля» и оптимизированная система снижают среднюю задержку, необходимую для обработки вычислительных задач, особенно для задач категории IV. Кроме того, была измерена средняя задержка для четырех систем с разным числом оконечных устройств, результаты

представлены на рисунке 3.1.8.6. Эти показатели были оценены для случая, когда на каждом устройстве было десять задач второй категории. Как показывают результаты, с увеличением числа оконечных устройств традиционные модели не могут поддерживать требуемую задержку, в то время как разработанная система и оптимизированная система достигают существенно более высокой эффективности для обеспечения значений задержки, главным образом в сценариях плотного развертывания.

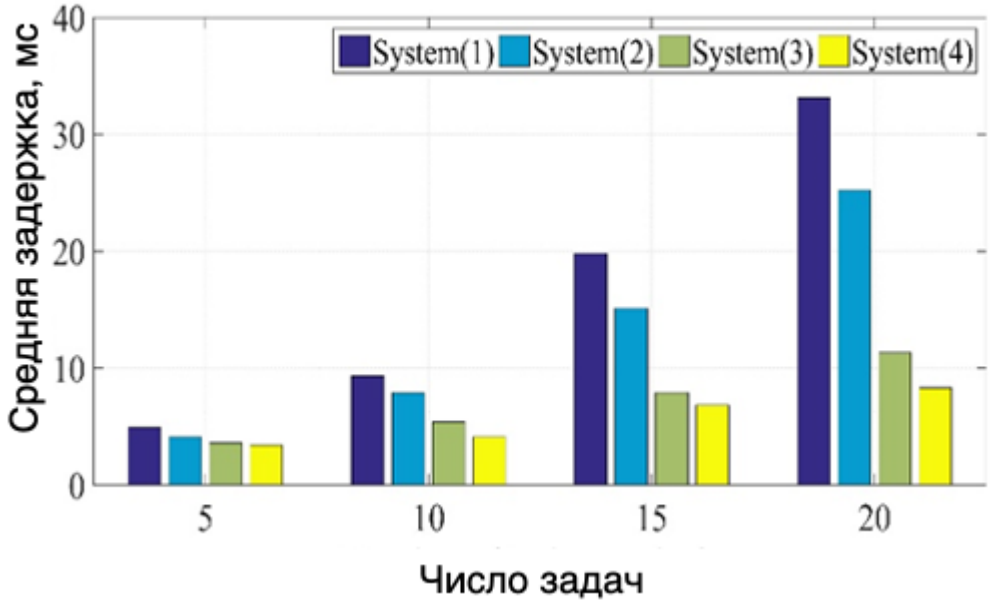


Рисунок 3.1.8.6 – Средняя задержка для четырех систем с разным числом вычислительных задач для оконечных устройств

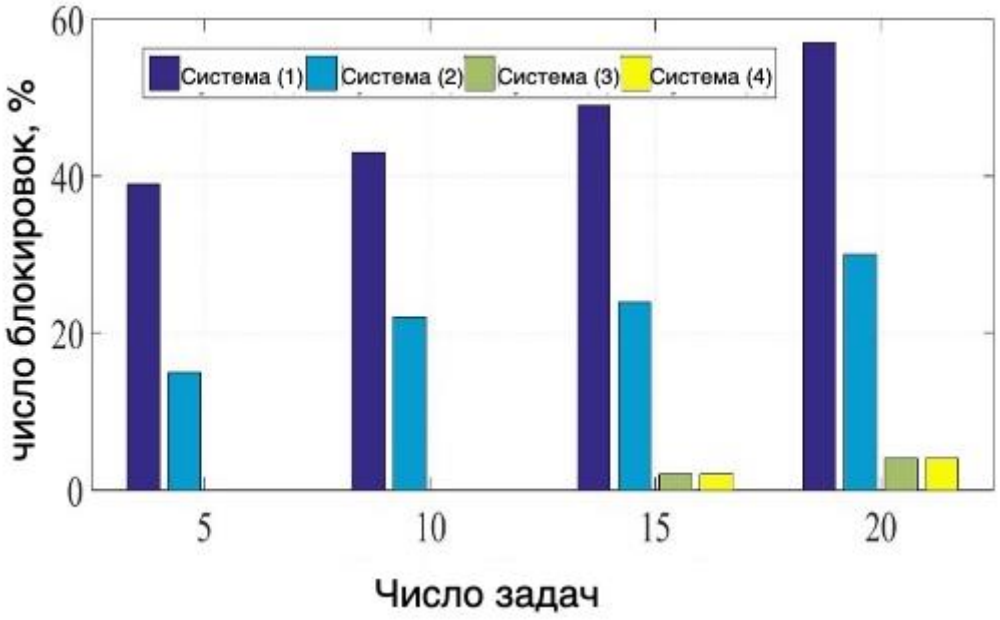


Рисунок 3.1.8.7 – Процентное соотношение заблокированных задач при различных значениях вычислительных задач для оконечных устройств

Третьим исследуемым показателем является процент заблокированных задач, который можно использовать в качестве меры доступности системы. Число заблокированных задач фиксировалось для каждой системы в разных случаях. На рисунке 3.1.8.7 представлен процент заблокированных задач для четырех упомянутых систем при различных значениях вычислительных задач для оконечных устройств. С увеличением числа задач процент заблокированных задач, естественно, увеличился во всех четырех системах, что связано с нехваткой вычислительных ресурсов. Однако процент заблокированных задач для разработанной системы «воздух-земля» меньше, чем у традиционных и наземных систем. Это связано с дополнительными ресурсами, предоставляемыми со стороны воздушного сегмента сети.

На рисунке 3.1.8.8 представлены результаты оценки процента заблокированных задач для четырех систем по четырем категориям приложений. Как показывают результаты, процент заблокированных задач увеличивается для

третьей и четвертой категорий приложений; однако это увеличение минимально для разработанной системы «воздух-земля» и оптимизированной системы. Это связано с тем, что для этих категорий приложений требуются более высокие вычислительные ресурсы, что уменьшает вероятность локального выполнения. Предложенные решения обеспечивает дополнительную возможность с использованием ресурсов воздушного сегмента сети для выгрузки данных, снижая вероятность блокировки задачи.

На рисунке 3.1.8.9 представлены результаты по заблокированным задачам в процентном соотношении для четырех систем для различных значений и разном числе оконечных устройств. По мере увеличения числа таких устройств увеличивается вероятность блокировки из-за высокой нагрузки на доступные ресурсы. Однако предлагаемые решения, как уже отмечалось выше, обеспечивают дополнительные ресурсы через воздушный сегмент сети.

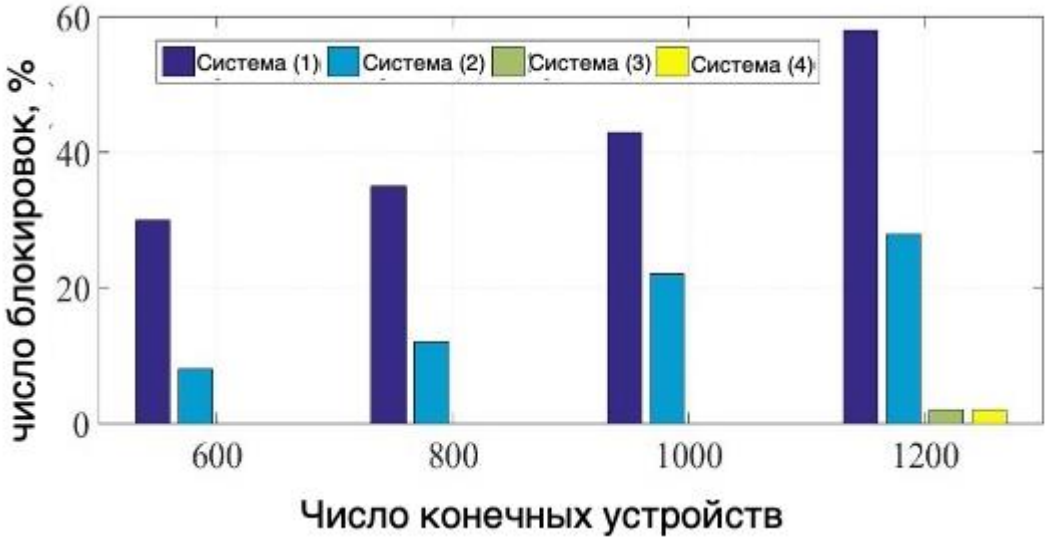


Рисунок 3.1.8.8 – Процентное соотношение заблокированных задач в

зависимости от категории приложений

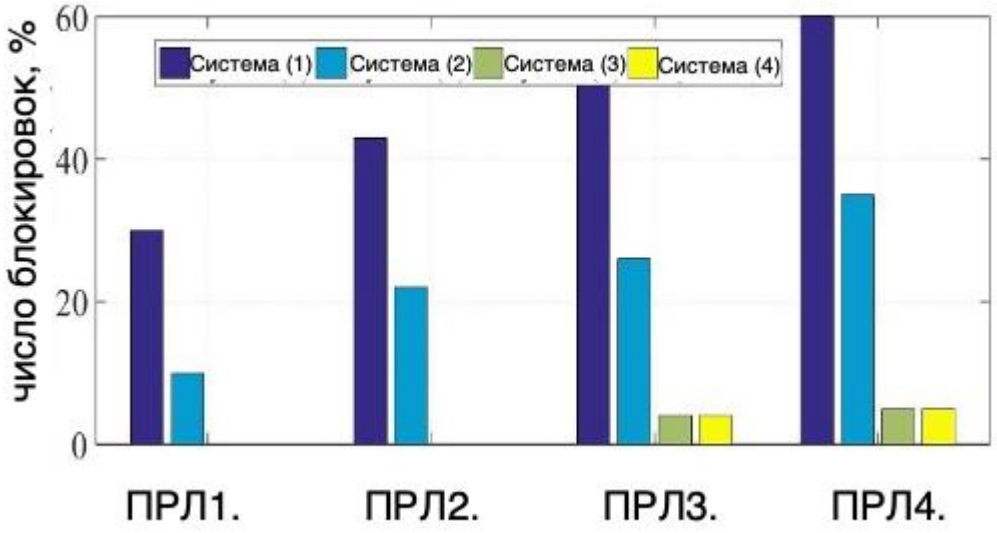


Рисунок 3.1.8.9 – Процентное соотношение заблокированных задач при разном числе оконечных устройств

На рисунке 3.1.8.10 показаны результаты моделирования для четырех систем для различной плотности устройств. Предложенные модель и метод показали более высокую эффективность для значений задержки во всем диапазоне плотности, особенно при высокой плотности наземной сети. Для сетей со сверхвысокой плотностью традиционные модели IoT не могут достичь сверхнизкой или даже низкой задержки. Однако предложенные модель и метод могут поддерживать приложения с ультра малой задержкой даже при сверхвысокой плотности наземной сети.

На рисунке 3.1.8.11 для четырех рассматриваемых систем представлены результаты моделирования для процента потребляемой энергии при различной

плотности наземной сети. Предложеные модель и метод снижают потребление энергии в среднем на 58% по сравнению с традиционными сетями IoT.

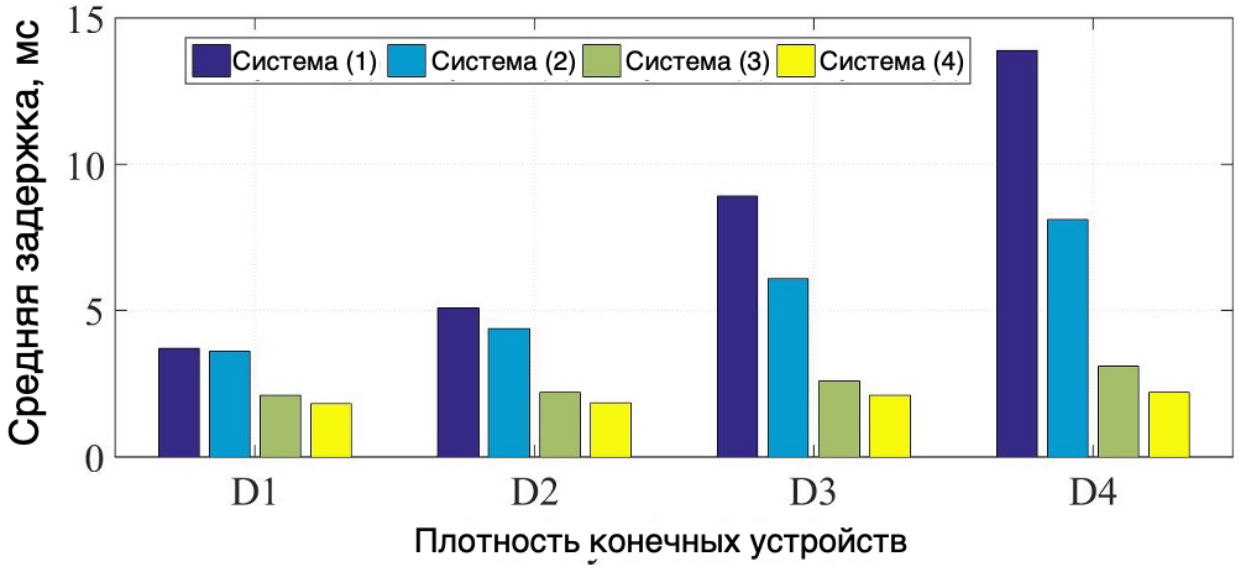


Рисунок 3.1.8.10 - Средняя задержка для различной плотности наземной сети

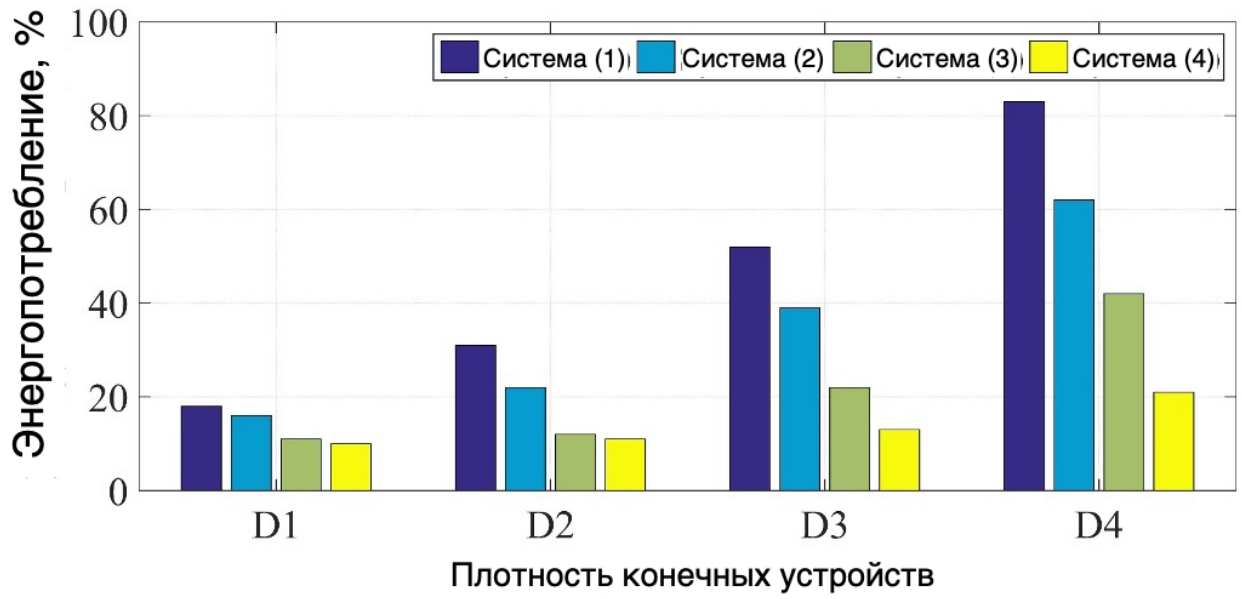


Рисунок 3.1.8.11 - Процент потребление энергии для различной плотности наземной сети

3.2 Алгоритм выгрузки с учетом энергопотребления и задержки для БПЛА

В последние годы все больший интерес вызывают беспилотные летательные аппараты (БПЛА), например дроны [209], [210]. Ожидается, что к моменту масштабного внедрения в эксплуатацию сотовой связи пятого поколения (5G) БПЛА будут иметь широкий круг применения. Эти приложения варьируются от простого мониторинга окружающей среды до сложных военных приложений с высоким уровнем безопасности [211]-[213]. Существует множество проблем связанных с разработкой сетей и приложений для БПЛА. Основные из них [214]-[216]: планирование траектории или пути, предотвращение столкновений и управление мобильностью, стоимость и безопасность, выгрузка данных и энергопотребление, задержка и совместимость с существующими системами и сотовыми сетями.

Часть этих проблем связана с ограниченными возможностями БПЛА из-за их небольшого размера, например, микро дронов, необходимых для многих приложений [211]. Приложения с интенсивными вычислительными задачами, например, на основе изображений или видео, требуют больших ресурсов обработки и энергии, которые влияют на работу в реальном времени и срок службы системы БПЛА или даже вызывают блокировку задач. Для того, чтобы продлить срок службы БПЛА и преодолеть ограничения использования батареи, энергетические ресурсы следует использовать рационально. Одним из способов повышения эффективности приложений и снижения энергопотребления является перенос вычислений на другие устройства в сети БПЛА с доступными ресурсами [217], [218]. Эти устройства могут находиться в воздушном сегменте или на земле [219]. Для БПЛА существует два возможных метода выгрузки данных: выгрузка с воздуха и выгрузка с земли. БПЛА может передать свои вычислительные задачи ближайшим БПЛА с доступными вычислительными и энергетическими ресурсами [220] или передать вычислительные задачи на наземные станции, подключенные к облачным серверам [221].

Последние технологии, которые будут использоваться для 5G, могут помочь в развертывании сетей БПЛА [222]. Эти технологии основываются на мобильных граничных вычислениях (MEC), программно-определяемой сетевой конфигурации (SDN) и виртуализации сетевых функций (NFV). MEC обеспечивает возможности облачных вычислений на границе сети радиодоступа (RAN) на расстоянии одного перехода от оконечного пользователя и, таким образом, уменьшает задержку вычислений [223], [224]. Развертывание MEC позволяет БПЛА эффективно переносить свои вычислительные задачи на граничные облачные серверы. Серверы MEC обрабатывают выгруженные вычислительные задачи и передают результаты на определенный БПЛА,

используя соответствующую ссылку [225], [226]. Кроме того, поскольку серверы MEC расположены близко к БПЛА, приблизительно на расстоянии одного телекоммуникационного перехода, обеспечивается более высокая эффективность с точки зрения задержки для выгруженных задач [227], [228]. В это же время сеть SDN может быть развернута для управления и контроля распределения ресурсов для БПЛА. Кроме того, контроллер SDN можно применять для оптимизации использования серверов MEC и обработки приложений БПЛА [229].

В этом подразделе для беспилотных летательных аппаратов разработан алгоритм выгрузки с учетом энергопотребления и задержки, позволяющий выгрузить вычислительные задачи посредством выгрузки с воздуха либо с земли. В БПЛА используется алгоритм принятия решений, в соответствии с которым задача выполняется локально на устройстве или же ее выполнение передается воздушным или наземным устройствам. Выгрузка наземного сегмента основана на разработанном алгоритме, представленном в [230]. Этот алгоритм был разработан для многоуровневой системы MEC, представленной в [231]. Предлагаемый алгоритм состоит из трех основных подпрограмм для продления срока службы сетей БПЛА за счет экономии ресурсов и за счет выгрузки задач. Кроме того, предложенный алгоритм снижает вероятность блокировки задачи из-за ограниченности ресурсов, а также уменьшает сквозную задержку обработки вычислительной задачи. Этот алгоритм разработан, в основном, для БПЛА небольших размеров с приложениями на основе изображений и видео.

В разделе 3.2.1 проанализированы работы, связанные с предлагаемым алгоритмом выгрузки. В разделе 3.2.2 представлены детали алгоритма выгрузки. Раздел 3.2.3 представляет систему моделирования и анализ результатов.

3.2.1 Существующие работы

Рассмотрим известные исследования, связанные с предлагаемыми

алгоритмами выгрузки для сетей БПЛА. В подразделе на основе проведенного анализа представлены преимущества и ограничения для каждой работы. Кроме того, новизна предложенного в диссертационной работе алгоритма сравнивается с этими исследованиями.

Луо *и др.* [232] предложили структуру для сети БПЛА, которая развернута для приложений, связанных со стихийными бедствиями. Платформа выгружает полученные данные, например, видеоданные, в удаленный облачный модуль. В системе используется механизм клиент-сервер, при котором клиент размещается на БПЛА, а сервер размещается в удаленной облачной единице. Клиент отвечает за получение видео, планирование данных с помощью контекстно-зависимого планировщика видео и выгрузку данных. Сервер получает выгруженные данные и предоставляет вычислительные ресурсы, необходимые для обработки полученных данных. Основным ограничением для этой платформы является задержка, поскольку данные выгружаются в удаленное облако, а не в граничное облако, расположенное ближе к сети БПЛА.

Lynskey [233] разработал алгоритм на основе MEC, который максимизирует вероятность выгрузки за счет оптимизации вычислительных ресурсов, используемых для обработки конкретной выгруженной задачи. Такая схема, в основном, используется для обработки выгруженных изображений с БПЛА. Представленный алгоритм был реализован на серверах MEC для поиска оптимальной комбинации выгруженных задач, которые могут быть обработаны в зависимости от мощности сервера MEC. В этой работе рассматривается процесс выполнения алгоритма и выгрузка только с земного сегмента.

Чжоу *и др.* [234] представили четыре варианта использования БПЛА, которые поддерживаются облачными и граничными вычислениями. В этой работе, в основном, определяются способы взаимодействия БПЛА с наземными

устройствами, в том числе разнородными вычислительными устройствами. Авторы провели исследование для БПЛА со структурой MEC. Этот сценарий представляет собой использование БПЛА для уменьшения трафика. В работе, в основном, рассматривается использование БПЛА для поддержки массивных сенсорных сетей и плотных сетей IoT. Имеется определенное сходство развертывания технологии MEC с предлагаемым в диссертационной работе решением. Однако система MEC, развернутая для предлагаемой в диссертационной работе, отличается тем, что она развернута в многоуровневой структуре. Кроме этого, в [234] рассматривается структура только для определенного приложения.

Валентино *и др.* [220] разработали алгоритм, который выгружает вычислительные задачи на ближайшие БПЛА. Предлагаемая система предполагает развертывание БПЛА кластерным способом. При этом каждый кластер состоит из группы близлежащих БПЛА, которые, в основном, выполняют общие задачи. Каждый кластер БПЛА имеет головной узел кластера, который управляет и контролирует члены кластера БПЛА. Предложенный алгоритм позволяет БПЛА разгружать свои задачи на другие кластеры с доступными ресурсами, если текущий кластер не имеет доступных вычислительных ресурсов. Только головной узел кластера решает, выполнять задачу локально (т.е. в текущем кластере) или передать ее соседнему кластеру с доступными ресурсами. Кроме того, выгрузка вычислительных задач между кластерами распределяет вычислительные задачи между БПЛА, что обеспечивает более высокую энергоэффективность и продлевает срок службы БПЛА. Каждый головной узел кластера отвечает за поиск доступных вычислительных и энергетических ресурсов в соседних кластерах. Этот алгоритм может быть развернут только для плотных сетей БПЛА с учетом кластеризации. В этой работе не учитывалась

выгрузка через наземный сегмент, а учитывалась только выгрузка по воздуху и только для достаточно большого числа БПЛА.

Юнг *и др.* [235] разработали адаптивный алгоритм выгрузки для БПЛА, который использует множественные соединения с использованием протокола TCP для процесса выгрузки. В этой работе рассматривается только выгрузка через наземный сегмент, когда алгоритм адаптивной выгрузки выбирает лучший наземный граничный сервер, на котором могут размещаться вычислительные задачи. Система разработана, в основном, для критически важных приложений и для продления срока службы БПЛА. Данные выгружаются с БПЛА на наземный сервер через несколько TCP соединений. Основным преимуществом этой работы является учет мобильности БПЛА при выгрузке и выполнении задач. Кроме того, учтена возможность роуминга между базовыми станциями при выполнении задачи, если БПЛА перемещается в зоне покрытия нескольких операторов мобильной связи.

Алгоритм с использованием SDN был предложен в [236] для поддержки вычислительных ресурсов БПЛА, находящегося над определенной географической областью и подключенного к одной базовой станции. Контроллер SDN использует алгоритм, который принимает выгруженные приложения и выбирает оптимальный облачный сервер с доступными ресурсами, обеспечивающий требуемую задержку QoS для выгруженной задачи. Облачные серверы, развернутые в системе, неоднородны: граничные облачные серверы, распределенные облачные серверы и удаленные облачные серверы в Интернете.

Новизна предложенного в диссертационной работе алгоритма по сравнению с описанными выше работами связана с использованием интегрированной выгрузки. БПЛА может использовать два варианта локального выполнения и два варианта выгрузки. Предложенный алгоритм использует

механизм принятия решений для выбора метода выгрузки, который наилучшим образом обеспечивает эффективность использования энергопотребления и уменьшения задержки. Более того, как воздушные, так и наземные процессы выгрузки отличаются от ранее представленных методов в предыдущих рецензируемых работах. Выгрузка наземного сегмента осуществляется по многоуровневой структуре системы MEC, связанной с наземными станциями. Эта многоуровневая структура была представлена в [231] и [237], а алгоритм выгрузки для такой структуры - в [230].

3.2.2 Алгоритм ингрированной выгрузки для БПЛА

Приложения и варианты использования БПЛА требуют развертывание легких и малых БПЛА, особенно для приложений видеонаблюдения [238]. Для большинства приложений БПЛА требуются вычислительные и энергетические мощности для сбора и анализа данных. Для видеоприложений БПЛА должен обрабатывать изображения с высоким разрешением, что требует больших вычислительных и энергетических мощностей [239]. Чтобы преодолеть вычислительные и энергетические ограничения для БПЛА, вычислительно ресурсоемкие задачи должны быть перенесены на близлежащие БПЛА с доступными вычислительными ресурсами. В этом подразделе представлен алгоритм выгрузки БПЛА с использованием MEC, основанный на уменьшении задержки и энергопотребления, для продления срока службы БПЛА и выполнения современных приложений сетей связи.

A. СТРУКТУРА СИСТЕМЫ

Предлагаемая система состоит из воздушного и наземного сегментов, как показано на рисунке 3.2.2.1 Воздушный сегмент состоит из БПЛА, которые работают вместе или по отдельности для предоставления требуемых услуг.

Наземный сегмент включает базовые станции и подключенные серверы MEC для обеспечения вычислительных возможностей на границе RAN рядом с оконечным пользователем.

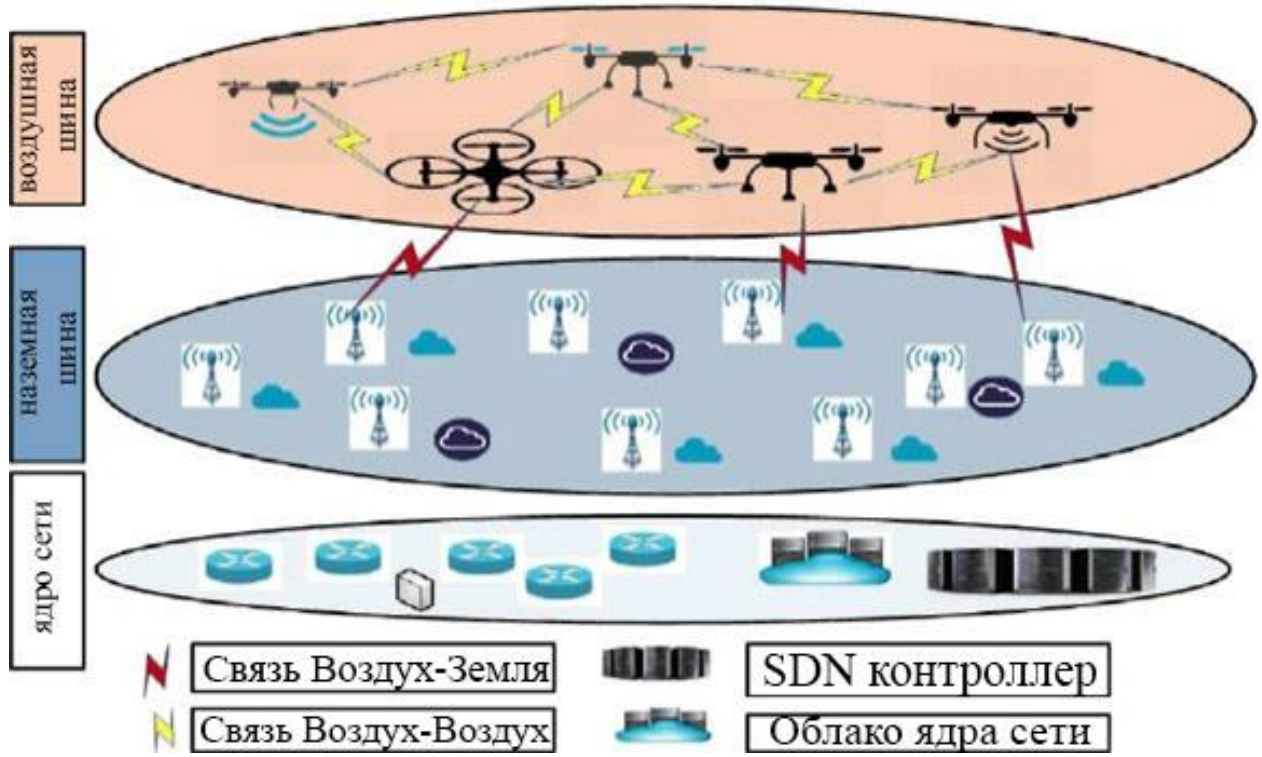


Рисунок 3.2.2.1. Структура многоуровневой системы БПЛА на базе MEC

Предлагаемый алгоритм выгрузки состоит из нескольких уровней выгрузки, которые показаны на рисунке 3.2.2.2 Каждый уровень представляет собой исполнительное устройство с вычислительными и энергетическими ресурсами. Если доступные ресурсы БПЛА могут справиться с вычислительной задачей в пределах требуемой задержки QoS, задача обрабатывается локально на БПЛА без выгрузки. Это нулевой уровень выгрузки, который подходит для простых задач, требующих небольших вычислительных и энергетических ресурсов.

Рисунок 3.2.2.2 - Основные уровни выгрузки

Первый уровень выгрузки – это когда выгрузка направлена на доступные поблизости БПЛА – это выгрузка по воздуху. Когда бортовых вычислительных ресурсов и энергии БПЛА недостаточно для выполнения определенной задачи, БПЛА может переложить рабочую нагрузку на один или несколько ближайших БПЛА, у которых есть доступные вычислительные и энергетические ресурсы. БПЛА может определить близлежащие БПЛА и запрашивать доступные ресурсы для определенной задачи для процесса выгрузки по воздуху.

Более высокий – второй уровень выгрузки включает в себя разнородные граничные облачные блоки, подключенные к наземным базовым станциям. Этот уровень грузки называется выгрузкой с земли. В [230] была представлена многоуровневая система MEC для поддержки приложений, чувствительных к задержкам, например, AR/VR и Tactile Internet. В диссертационной работе она используется в качестве наземной системы MEC для облегчения процесса выгрузки.

Третий уровень выгрузки представлен граничными микро-облаками, которые предоставляют вычислительные ресурсы на границе RAN на расстоянии одного коммуникационного перехода от БПЛА.

Четвертый уровень выгрузки включает граничные мини-облака, которые контролируют и управляют микро-облаками. Мини-облака имеют более высокие вычислительные и энергетические ресурсы.

Каждый БПЛА использует механизм принятия решений, в котором реализуется предложенный в диссертационной работе алгоритм выбора метода выгрузки, может ли текущая задача выполняться локально или ее необходимо выгрузить до соответствующего уровня. Более того, граничные вычислительные блоки, т. е. микро- и мини-облака, используют локальный механизм принятия решений для определения места выполнения определенной задачи на основе доступных вычислительных и энергетических ресурсов.

На рисунке 3.2.2.3 показана структура БПЛА и граничных вычислительных сегментов.

Рисунок 3.2.2.3. Структура системы выгрузки.

Этап (2) Наземные устройства/пограничные серверы. Предполагается, что в системе будут развернуты разнородные типы БПЛА различные по вычислительным и энергетическим возможностям. Эти БПЛА развернуты над определенными географическими регионами и управляются через ближайшие базовые станции сотовой связи. БПЛА развернуты для гетерогенных приложений и задач (например, наблюдение и обеспечение покрытия сотовой связи (беспилотник – базовая станция). Чтобы отличить БПЛА с задачей, которую необходимо выполнить, будем его называть исходным БПЛА (SUAV).

B. АЛГОРИТМ ВЫГРУЗКИ С УМЕНШЕНИЕМ ЗАДЕРЖКОИ И

ЭНЕРГОСБЕРЕЖЕНИЕМ

При разработке алгоритма были учтены перечисленные ниже условия:

1. БПЛА имеют неоднородные энергетические возможности;
2. БПЛА обладают разнообразными вычислительными возможностями;
3. БПЛА поддерживают различные приложения с отличающимися вычислительными потребностями и потребностями в QoS;
4. Учитывается только полная выгрузка, частичная выгрузка не учитывается;
5. Задержка в очереди не учитывается.

Далее определим все параметры, которые будут использоваться в следующих подразделах, а также представим алгоритм выгрузки и соответствующие математические расчеты. В таблице 1 представлены обозначения переменных и параметров. Различные шаги предложенных алгоритмов указаны в алгоритмах 1, 2 и 3. Алгоритм 1 представляет собой алгоритм для локальной обработки и выгрузки решений. Алгоритм 2 представляет основные этапы процесса выгрузки воздуха, а алгоритм 3 указывает этапы процесса выгрузки для наземного сегмента.

C. МОДЕЛЬ ВЫГРУЗКИ

Процесс выгрузки выполняется исходным БПЛА (SUAV), который решает, обрабатывать ли задачу локально или выгрузить ее. SUAV сначала вычисляет общий размер входных данных задачи в битах, включая программные данные *(Z)* и общее количество циклов ЦП (NCYC), необходимых для обработки текущей задачи длиной *Z* (бит).

SUAV направляет эти данные в механизм принятия решений SUAV. Механизм принятия решений также получает информацию о максимально допустимой задержке C̄ для обработки текущей задачи.

Механизм принятия решений исходного БПЛА вычисляет общее время, необходимое для выполнения текущей задачи локально на БПЛА (T_{SUAV}) (1), исходя из текущих доступных ресурсов. Механизм принятия решений проверяет двоичную временную переменную выгрузки $D_{T\text{-}off}$ (3), сравнивая (T_{SUAV}) с временем ожидания QoS C.

$$T_{SUAV} = \frac{N_{CYC}}{R_{SUAV}} \quad , R_{SUAV} \in f_{SUAV} \tag{1}$$

$$N_{CYC} = Z.K \tag{2}$$

$$D_{T-off} = I(T_{SUAV}, \tau) = \begin{cases} 0 & IF\ (T_{SUAV} \leq \tau) \\ 1 & IF\ (T_{SUAV} > \tau) \end{cases} \tag{3}$$

Таблица 3.2.2.1 - Ключевые параметры

Обозначение	**Параметры**
Z	Общий размер входных данных задачи в битах, включая программные данные
K	Количество циклов ЦП, необходимое для обработки одного бита входных данных задачи.
C	Максимально допустимая задержка задачи *i,* установленная таким образом, чтобы поддерживать QoS.
N_{CYC}	Общее количество циклов ЦП, необходимых для обработки задачи длиной Z (бит)
R_{SUAV}	Ресурсы обработки исходного БПЛА, выделенные для задачи *i*
R_{UAV}	Ресурсы обработки БПЛА, выделенные для выполнения задачи *i*
R_{Micro}	Ресурсы обработки граничного сервера микро-облака, выделенные для выгруженной задачи
R_{Mini}	Ресурсы обработки пограничного мини-облачного сервера, выделенные для выгруженной задачи
δ_{SUAV}	Потребление энергии на цикл ЦП для выполнения рабочей нагрузки для исходного БПЛА
δ_{UAV}	Потребление энергии на цикл ЦП для выполнения рабочей нагрузки на БПЛА
δ_{Micro}	Потребление энергии на один цикл ЦП граничного микро-облачного сервера
δ_{Mini}	Энергозатратность на один цикл ЦП граничного мини-облачного сервера
T_{SUAV}	Общее время выполнения задачи *i,* когда задача выполняется локально на исходном БПЛА
$T_{ex\text{-}UAV}$	Общее время выполнения определенной задачи, выполняемой на БПЛА

T_{UAV}	Суммарная временная задержка, необходимая для обработки задачи на БПЛА
$T_{ex\text{-}Micro}$	Общее время выполнения задачи, когда задача выполняется в микро-облаке
T_{Micro}	Общая задержка по времени, необходимая для обработки задачи в микро-облаке
$T_{ex\text{-}Mini}$	Общее время выполнения задачи, когда задача выполняется на мини-облаке
T_{Mini}	Суммарная задержка по времени, необходимая для обработки задачи в мини-облаке
E_{SUAV}	Энергозатраты на выполнение задачи *i* на исходном БПЛА
E_{UAV}	Энергозатраты на выполнение выгруженной задачи
E_{Micro}	Потребление энергии для обработки выгруженной задачи на граничном сервере микро-облака
E_{Mini}	Потребление энергии для обработки выгруженной задачи на граничном мини-облачном сервере
E_{th}	Пороговый уровень энергии БПЛА
$E_{th\text{-}Micro}$	Пороговый уровень энергопотребления граничного сервера микро-облака
$E_{th\text{-}Mini}$	Пороговый уровень энергопотребления граничного мини-облачного сервера
$E_{R\text{-}SUAV}$	Остаток энергии исходного БПЛА после выполнения задачи
$E_{R\text{-}UAV}$	Остаток энергии БПЛА после выполнения выгруженной задачи
$E_{R\text{-}Micro}$	Оставшаяся энергия граничного сервера микро-облака после выполнения выгруженной задачи
$E_{R\text{-}Mini}$	Оставшаяся энергия граничного мини-облачного сервера после выполнения выгруженной задачи
$E_{C\text{-}SUAV}$	Текущий уровень энергии исходного БПЛА
$E_{C\text{-}UAV}$	Текущий уровень энергии БПЛА
$E_{C\text{-}Micro}$	Текущий уровень энергопотребления граничного сервера микро-облака
$E_{C\text{-}Mini}$	Текущий уровень энергопотребления граничного мини-облачного сервера
$D_{air\text{-}off}$	Двоичная решающая переменная процесса выгрузки с воздуха, определяемая исходным БПЛА
$D_{ground\text{-}off}$	Двоичная решающая переменная процесса выгрузки с земли, определяемая исходным БПЛА
$D_{E\text{-}off}$	Переменная двоичного решения по энергии процесса выгрузки, определяемая исходным БПЛА
$D_{T\text{-}off}$	Двоичная временная переменная процесса выгрузки, определяемая исходным БПЛА
$D_{T\text{-}UAV\text{-}acc\text{-}off}$	Двоичная временная переменная принятия решения о выгрузке задачи, определяемая БПЛА
$D_{E\text{-}UAV\text{-}acc\text{-}off}$	Бинарная переменная энергии принятия решения о выгрузке задачи, определяемая БПЛА
$D_{T\text{-}Micro\text{-}acc\text{-}off}$	Двоичная временная переменная принятия решения о выгрузке задачи, определяемая граничным сервером микро-облачного хранилища
$D_{E\text{-}Micro\text{-}acc\text{-}off}$	Двоичная переменная принятия решения о выгрузке задачи, определяемая граничным сервером микро-облака
$D_{T\text{-}Mini\text{-}acc\text{-}off}$	Двоичная переменная времени принятия решения о выгрузке задачи, определяемая граничным сервером мини-облака
$D_{E\text{-}Mini\text{-}acc\text{-}off}$	Двоичная переменная принятия решения о выгрузке задачи, определяемая

	граничным мини-облачным сервером
D_{off}	Двоичная переменная решения процесса выгрузки, определяемая исходным БПЛА
I	Переменная индикатора режима
T_{comm}	Общая задержка связи между исходным БПЛА и ближайшим БПЛА
t_{pro}	Задержка распространения
t_{trans}	Задержка передачи
T_{tx}	Задержка для передачи входных данных задачи от исходного БПЛА к ближайшему целевому БПЛА
T_{rx}	Задержка времени обратной связи результатов расчета
C	Скорость света ($3*10^8$ м/с)
$d_{x,y}$	Расстояние между источником x и целью
R_b	Достижимая скорость передачи данных по восходящему каналу
ω	Пропускная способность системы
σ	Мощность шума в приемнике
h	Усиление канала
P	Передающая мощность исходного БПЛА
P_S	Мощность передачи базовой станции, подключенной к граничному серверу микро-облака
N	Общее количество полученных ответных сообщений / Общее количество ближайших БПЛА
η_c	Эффективность канала
f_{SUAV}	Суммарные вычислительные ресурсы исходного БПЛА/тактовая частота процессора
f_{UAV}	Суммарные вычислительные ресурсы БПЛА/тактовая частота процессора
f_{Micro}	Общие вычислительные ресурсы граничного сервера микро-облака / частота цикла ЦП
f_{Mini}	Общие ресурсы обработки граничного мини-облачного сервера / частота цикла ЦП

Алгоритм 1		**Алгоритм выгрузки с учетом задержки и энергопотребления для SUAV**
	1:	Initialize C, E_{th}
	2:	Calculate Z, K
	3:	Calculate T_{SUAV}
	4:	If ($T_{SUAV} \leq C$)
	5:	$D_{T\text{-}off} = 0$
	6:	Calculate E_{SUAV}, $E_{R\text{-}SUAV}$
	7:	If ($E_{R\text{-}SUAV} > E_{th}$)
	8:	$D_{E\text{-}off} = 0$
	9:	Выполнять задачу локально
	10:	else
	11:	$D_{E\text{-}off} = 1$

	12:	Проверить выгрузку по воздушному сегменту [алгоритм вызова 2]
	13:	end if
	14:	else
	15:	$D_{T\text{-}off} = 1$
	16:	Проверить выгрузку по воздушному сегменту [алгоритм вызова 2]
	17:	end if

Алгоритм 2		**Алгоритм выгрузки по воздушному сегменту**
	1:	SUAV передает сообщение-запрос
	2:	Ближайшие БПЛА получают сообщение запроса
	3:	Ближайшие БПЛА обрабатывают запрос:
	4:	Рассчитать $T_{ex\text{-}UAV}$, E_{UAV}, $E_{R\text{-}UAV}$
	5:	If ($E_{R\text{-}UAV} > E_{th}$)
	6:	$D_{E\text{-}acc\text{-}off} = 1$
	7:	else
	8:	$D_{E\text{-}acc\text{-}off} = 0$
	9:	end if
	10:	БПЛА передает ответное сообщение
	11:	SUAV получает ответные сообщения (N)
	12:	Количество = 0
	13:	For ($i = 1 : i \leq N$) do
	14:	If ($D_{E\text{-}UAV}$-acc-off == 1)
	15:	Calculate $T_{UAV}(i)$
	16:	If ($T_{UAV}(i) \leq$ C)
	17:	$D_{air\text{-}off} = 1$
	18:	Count ++
	19:	end if
	20:	else
	21:	$D_{air\text{-}off} = 0$
	22:	end if
	23:	End for
	24:	If ($D_{air\text{-}off}$ ==1)

	25:		If (Count > 1)
	26:		Select min(T_{UAV}(i))
	27:		end if
	28:		Перенесите задачу выгрузки на соответствующий ближайший БПЛА
	29:	else	
	30:		Проверить выгрузку наземного сегмента [алгоритм вызова 3]
	31:	end if	

Алгоритм 3	**Алгоритм выгрузки наземного сегмента**
1:	Send request message of type I
2:	Receive request message of type I
3:	Calculate T_{Micro}, E_{Micro}, $E_{R\text{-}Micro}$
4:	If ($T_{Micro} \leq$ Ͼ)
5:	$D_{T\text{-}acc\text{-}g\text{-}off} = 1$
6:	If ($E_{R\text{-}Micro} > E_{th\text{-}Micro}$)
7:	$D_{E\text{-}Micro\text{-}acc\text{-}g\text{-}off} = 1$
8:	else
9:	$D_{E\text{-}Micro\text{-}acc\text{-}g\text{-}off} = 0$
10:	end if
11:	else
12:	$D_{T\text{-}Micro\text{-}acc\text{-}g\text{-}off} = 0$
13:	end if
14:	If ($D_{T\text{-}Micro\text{-}acc\text{-}g\text{-}off}$ & $D_{E\text{-}Micro\text{-}acc\text{-}g\text{-}off}$ == 1)
15:	$D_{ground\text{-}off} = 1$
16:	Выгрузить задачу в микро-облако
17:	else
18:	Send request message of type II
19:	Receive request message of type II
20:	Calculate T_{Mini}, E_{Mini}, $E_{R\text{-}Mini}$
21:	if ($T_{Mini} \leq$ Ͼ)
22:	$D_{T\text{-}Mini\text{-}acc\text{-}g\text{-}off} = 1$
23:	If ($E_{R\text{-}Mini} > E_{th\text{-}Mini}$)
24:	$D_{E\text{-}Mini\text{-}acc\text{-}g\text{-}off} = 1$
25:	else
26:	$D_{E\text{-}Mini\text{-}acc\text{-}g\text{-}off} = 0$
27:	end if
28:	else
29:	$D_{T\text{-}Mini\text{-}acc\text{-}g\text{-}off} = 0$
30:	end if
31:	If ($D_{T\text{-}Mini\text{-}acc\text{-}g\text{-}off}$ & $D_{E\text{-}Mini\text{-}acc\text{-}g\text{-}off} = 1$)
32:	$D_{ground\text{-}off} = 1$
33:	Выгрузить задачу в мини-облако
34:	else
35:	$D_{ground\text{-}off} = 0$
36:	Заблокировать задачу
37:	end if

Если решение по времени принимается отрицательное, SUAV проверяет ограничения по энергии, вычисляя бинарную переменную решения по энергии выгрузки $D_{E\text{-}off}$. Решение по бинарной энергии (4) (5) (6) вычисляется путем сравнения остаточной энергии на исходном БПЛА $E_{R\text{-}SUAV}$ после потребления

энергии на выполнение текущей задачи с пороговым уровнем E_{th}. Пороговый уровень энергии представляет собой минимальный уровень энергии исходного БПЛА, после которого уровень энергии БПЛА считается критическим и не может поддерживать выполнение вычислительных задач.

$$E_{SUAV} = N_{CYC}\delta_{SUAV} \quad (4)$$

$$E_{R-SUAV} = E_{C-SAUV} - E_{SUAV} \quad (5)$$

$$D_{E-off} = I(E_{R-SUAV}, E_{th}) = \begin{cases} 1 \;\; IF(E_{R-SUAV} \leq E_{th}) \\ 0 \;\; IF(E_{R-SUAV} > E_{th}) \end{cases} \quad (6)$$

Если остаточный уровень энергии исходного БПЛА после потребления энергии на выполнение задачи больше порогового уровня энергии БПЛА, задача выполняется локально и нет необходимости выгружать задачу на какие-либо другие устройства. В этом случае бинарная переменная решения по энергии устанавливается равной нулю, чтобы указать, что нет необходимости в выгрузке. При этом, если оставшаяся энергия после выполнения текущей задачи будет меньше порогового уровня энергии БПЛА переменная решения об энергии устанавливается равной единице. В этом случае следует рассмотреть решение о выгрузке на воздушное либо на наземное устройство.

Для положительного решения о выгрузке, будь то решение по времени или по энергоемкости, БПЛА должен выгрузить вычислительную задачу. БПЛА сначала проверяет возможность выгрузки по воздуху путем определения доступных вычислительных ресурсов ближайших БПЛА. Для этого необходимо, чтобы поблизости находился БПЛА с достаточными энергетическими и вычислительными возможностями для выполнения текущей задачи в пределах времени задержки QoS. Если условия для выгрузки через воздушный сегмент

соблюдены, БПЛА принимает решение о выгрузке по воздушному сегменту на ближайший БПЛА. Однако, если выгрузка по воздуху не удалась из-за отсутствия достаточных энергетических и вычислительных ресурсов у окружающих БПЛА, осуществляется проверка возможности взаимодействия с наземным сегментом.

1) Процесс выгрузки через воздушный сегмент

Процесс выгрузки через воздушный сегмент состоит из двух основных подпроцессов: процесса обнаружения и собственно процесса выгрузки. Как только БПЛА принимает решение о выгрузке, он сначала проверяет возможность наличия выгрузки. БПЛА ищет ближайшие БПЛА, чтобы проверить, есть ли у одного из них доступные ресурсы для выполнения задачи. С этой целью БПЛА запускает процесс обнаружения окружающего пространства на наличие БПЛА с доступными ресурсами для выполнения задачи. Рисунок 3.2.2.4 иллюстрирует процесс обнаружения для процесса выгрузки через воздушный сегмент.

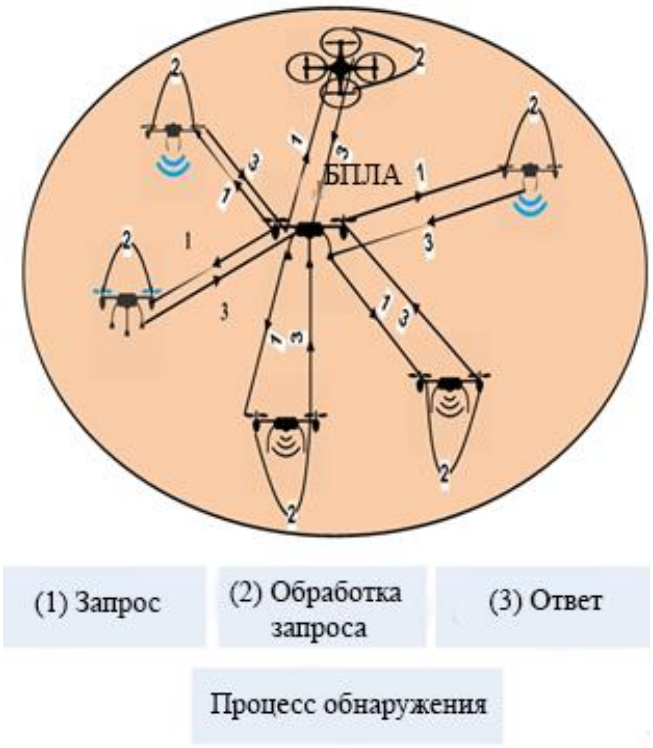


Рисунок 3.2.2.4 - Процесс обнаружения

В процессе обнаружения БПЛА рассылает сообщение о необходимости выгрузки всем окружающим БПЛА через соответствующий канал связи, например, WiFi [240]. Передаваемое сообщение содержит два основных поля: поле идентификации БПЛА и поле информации о задаче. На рисунке 3.2.5 показаны основные поля и подполя сообщения обнаружения. Поле идентификации содержит идентификационный номер БПЛА (ID), а также долготу, широту и высоту текущего местоположения БПЛА. Поле информации о задаче содержит информацию о рабочей нагрузке, которую необходимо выполнить. Эта информация включает в себя общий объем вычислительных данных задачи, подлежащей разгрузке, количество циклов ЦП, необходимых для обработки этих данных и ограничения QoS по задержке задачи, подлежащей выгрузке. Все находящиеся поблизости БПЛА получают сообщение об обнаружении, передаваемое БПЛА, и считывают полученную информацию.

Каждый БПЛА решает, может ли он выполнить задачу и принять запрос на выгрузку или отклонить запрос на выгрузку, если его ресурсов окажется недостаточно для выполнения задачи. Механизм принятия решений каждого близлежащего БПЛА определяет следующее:

1. Общее время, необходимое для выполнения задачи, определенной в сообщении запроса $T_{ex\text{-}UAV}$ на основе текущих доступных ресурсов;

2. Общее потребление энергии для обработки задачи, определенной в сообщении запроса E_{UAV}, исходя из текущих доступных ресурсов;

3. Остаток энергии БПЛА после обработки целевой задачи $E_{R\text{-}UAV.}$

$$T_{ex-UAV} = \frac{N_{CYC}}{R_{UAV}} \quad , R_{UAV} \in f_{UAV} \tag{7}$$

$$E_{UAV} = N_{CYC}\delta_{UAV} \tag{8}$$

$$E_{R-UAV} = E_{C-UAV} - E_{UAV} \tag{9}$$

Механизм принятия решений каждого БПЛА вычисляет переменную для энергии для принятия выгрузки $D_{E\text{-}acc\text{-}off}$ путем сравнения оставшейся энергии после выполнения задачи $E_{R\text{-}UAV}$ с пороговым уровнем энергии БПЛА. Если оставшаяся энергия меньше порогового уровня, механизм принятия решений отклоняет запрос на выгрузку и переменная решения об энергии принятия разгрузки $D_{E\text{-}acc\text{-}off}$ устанавливается равной нулю. В противном случае БПЛА принимает выгрузку и отправляет ответное сообщение с решением на БПЛА. Исходный БПЛА также должен проверить T_{UAV} с задержкой для QoS для положительных ответов.

$$D_{E-acc-off} = I(E_{R-UAV}, E_{th}) = \begin{cases} 0 & IF\ (E_{R-UAV} \leq E_{th}) \\ 1 & IF\ (E_{R-UAV} > E_{th}) \end{cases}$$

(10)

Ответное сообщение содержит три основных поля: поле идентификации, поле решения по энергии и поле спецификаций выполнения. На рисунке 3.2.2.5 показаны основные поля и подполя ответного сообщения. Поле решения об энергии представляет собой однобитовое поле, которое относится к соглашению о выгрузке: единица для согласия и ноль для отклонения. Поле спецификации выполнения заполняется, только если поле двоичного решения установлено в единицу. В этом поле указываются основные особенности процесса выполнения, если задачу решили выгрузить; эти функции включают в себя общее время выполнения задачи и общий расход энергии на обработку задачи.

Сообщение запроса на обнаружение			
Идентификация		**Спецификация**	
ID	Положение	Тип	Размер

Сообщение ответа				
Идентификация		**Состояние энергоресурса**	**Спецификация выполнения задачи**	
ID	Положение	Один/Ноль	Время выполнения	Энергопотребление

Рисунок 3.2.2.5 - Сообщения запросов на обнаружение и ответов

БПЛА получает ответные сообщения и принимает решение о выгрузке по воздушному сегменту на основании решений, содержащихся в полученных

сообщениях, и спецификаций выполнения задачи. Для всех положительных ответов исходный БПЛА вычисляет общую задержку обработки задачи на ближайшем БПЛА T_{UAV}. Эта задержка рассчитывается путем сложения времени задержки выполнения на близлежащем БПЛА с положительным ответом $T_{ex\text{-}UAV}$ и задержки связи (т.е., время доставки пакета для восходящей передачи входных данных задачи T_{tx} и временная задержка для получения вычисленных результатов T_{rx}).

$$T_{UAV} = T_{ex-UAV} + T_{tx} + T_{rx} \quad (11)$$

Задержка связи для доставки пакетов данных задачи T_{tx} представляет собой сумму времени на передачу входных данных задачи t_{trans} и задержки распространения t_{pro} от БПЛА к целевому БПЛА.

$$T_{tx} = t_{trans} + T_{pro} \quad (12)$$

$$t_{pro} = \frac{d_{SUAV,UAV}}{C} \quad (13)$$

$$t_{trans} = \frac{Z}{R_b} \quad (14)$$

Достижимая скорость передачи в восходящем канале может быть рассчитана с использованием формулы Шеннона-Хартли следующим образом [35]:

$$R_b = \omega \log_2\left(1 + \frac{hp}{\sigma}\right) \quad (15)$$

Затем механизм принятия решений БПЛА вычисляет переменную решения для выгрузки по воздуху $D_{air\text{-}off}$, сравнивая общую временную задержку для обработки выгруженной задачи на соседнем БПЛА T_{UAV} с задержкой QoS $\mathbb{C}$. Решение о выгрузке принимается, если общая задержка T_{UAV} меньше, чем задержка QoS $\mathbb{C}$.

$$D_{air-off} = I(T_{UAV}, \tau) = \begin{cases} 1 & IF\ (T_{UAV} \leq \tau) \\ 0 & IF\ (T_{UAV} > \tau) \end{cases} \quad (16)$$

Если поблизости находится более одного БПЛА с доступными ресурсами, подходящими для выполнения задачи, и положительным ответом на решение о воздушной выгрузке, БПЛА рассчитывает T_{UAV} для каждого близлежащего БПЛА с положительным ответом. Затем модуль принятия решений БПЛА вычисляет решение о выгрузке с воздуха для ближайшего БПЛА с наименьшим значением T_{UAV}. В случае отрицательного решения о выгрузке с воздуха исходный БПЛА принимает решение о выгрузке с земли через многоуровневую систему MEC.

2) ПРОЦЕСС РАЗГРУЗКИ ЧЕРЕЗ НАЗЕМНЫЙ СЕГМЕНТ

Когда не удается выполнить выгрузку с воздуха, БПЛА проверяет возможность выгрузки с земли. SUAV отправляет запрос на соответствующий граничный сервер микро-облака, подключенный к наземной базовой станции. Запрошенное сообщение содержит информацию об идентификационном номере, положении и спецификации задачи, представленные в таблице 3.2.2.2. Граничный сервер микро-облака получает запрос на выгрузку и начинает его обработку. Механизм принятия решений микро-облачного сервера рассчитывает общее время, необходимое для выполнения целевой задачи, исходя из текущих доступных ресурсов $T_{ex\text{-}Micro}$

$$T_{ex-Micro} = \frac{N_{CYC}}{R_{Micro}} \quad , R_{Micro} \in f_{Micro} \quad (17).$$

Затем механизм принятия решений микро-облачного сервиса вычисляет общее время, необходимое для обработки задачи T_{Micro}

$$T_{Micro} = T_{ex-Micro} + T_{tx} + T_{rx} \quad (18)$$

$$T_{tx} = t_{trans} + T_{pro} \quad (19)$$

$$t_{pro} = \frac{d_{SUAV,Micro-cloud}}{C} \quad (20).$$

Сервер микро-облака вычисляет переменную решения о времени для принятия выгрузки земли путем сравнения общего времени обработки задачи на сервере микро-облака T_{Micro} с временем задержки QoS Ƈ.

Задержка взаимодействия для доставки пакетов данных задачи *Ttx* представляет собой сумму времени передачи входных данных задачи Wans и задержки распространения t_{pro} от БПЛА к целевому БПЛА

$$D_{T-Micro-acc-off} = I(T_{Micro}, \tau) = \begin{cases} 1 & IF\ (T_{Micro} \leq \tau) \\ 0 & IF\ (T_{Micro} > \tau) \end{cases} (21).$$

Таблица 3.2.2.2 - Обмен сообщениями о выгрузке на земной плоскости

Сообщение	**Тип**	**Tx**	**Rx**	**Содержанеи**
Сообщение запроса	I	SUAV	Микро-облако	*Z, K,* Ƈ, SUAV Расположение
Сообщение ответа	I	Микро-облако	SUAV	Решение, $D_{ground-offloading}$ уровень разгрузки
Сообщение запроса	II	Микро-облако	Мини-облако	Z, K, Ƈ
Сообщение ответа	II	Мини-облако	Микро-облако	Решение, $D_{ground-offloading}$

Решение о положительном времени принятия выгрузки задачи принимается, если время обработки T_{Micro} меньше или, в худшем случае, равно времени ожидания QoS Ƈ. В этом случае механизм принятия решений сервера микро-облака проверяет решение об энергопотреблении. В противном случае сервер микро-облака перенаправляет задачу выгрузки на граничный сервер мини-облака.

Чтобы проверить доступность энергетических ресурсов сервера микро-облака, механизм принятия решений сначала рассчитывает общее потребление энергии для выполнения задачи E_{Micro}. Затем подсчитывается оставшаяся энергия микро-облачного сервера $E_{R\text{-}Micro}$, если задача выгружена и обработана

$$E_{Micro} = N_{CYC}\delta_{Micro} + T_{trans}P\eta_C \quad (22)$$

$$E_{R-Micro} = E_{C-Micro} - E_{Micro} \quad (23).$$

Механизм принятия решений вычисляет переменную решения об энергии принятия наземной выгрузки задачи $D_{E\text{-}Micro\text{-}acc\text{-}off}$, сравнивая оставшуюся энергию сервера микро-облачных вычислений $E_{R\text{-}Micro}$ с пороговым уровнем энергии сервера микро-облачных вычислений, при котором энергия сервера микро-облака находится на критическом уровне

$$D_{E-\text{Micro}-\text{acc}-\text{off}} = I(E_{R-\text{Micro}}, E_{\text{th}-\text{Micro}}) = \begin{cases} 0 & IF\ (E_{R-\text{Micro}} \leq E_{\text{th}-\text{Micro}}) \\ 1 & IF\ (E_{R-\text{Micro}} > E_{\text{th}-\text{Micro}}) \end{cases} \quad (24).$$

При достаточной энергоемкости принимается положительное энергетическое решение и сервер микро-облака отправляет ответное сообщение на БПЛА с согласием на выгрузку с земли. Если энергоемкости недостаточно и решение о принятии выгрузки отрицательное, сервер микро-облака отправляет сообщение-запрос типа II на соответствующий граничный сервер мини-облака. Граничный сервер мини-облака получает сообщение запроса и обрабатывает его, вычисляя переменные принятия решения о времени и энергии для принятия выгрузки ($D_{T\text{-}Mini\text{-}acc\text{-}off}$ и $D_{E\text{-}Mini\text{-}acc\text{-}off}$)

$$T_{ex-Mini} = \frac{N_{CYC}}{R_{Mini}}, \ R_{Mini} \in f_{Mini} \quad (25)$$

$$T_{Mini} = T_{ex-Mini} + T_{tx} + T_{rx} \quad (26)$$

$$D_{T-Mini-acc-off} = I(T_{Mini}, \tau) = \begin{cases} 1 & IF\ (T_{Mini} \leq \tau) \\ 0 & IF\ (T_{Mini} > \tau) \end{cases} \quad (27)$$

$$E_{Mini} = N_{CYC}\delta_{Mini} + T_{trans}P_S\eta_C \quad (28)$$

$$E_{R-Mini} = E_{C-Mini} - E_{Mini} \quad (29)$$

$$D_{E-\text{Mini}-\text{acc}-\text{off}} = I(E_{R-\text{Mini}}, E_{\text{th}-\text{Mini}}) = \begin{cases} 0 & IF\ (E_{R-\text{Mini}} \leq E_{\text{th}-\text{Mini}}) \\ 1 & IF\ (E_{R-\text{Mini}} > E_{\text{th}-\text{Mini}}) \end{cases} \quad (30)\ .$$

Только для положительных решений по времени и энергии механизм принятия решений граничного мини-облачного сервера принимает выгрузку. В противном случае граничный сервер мини-облака решает отклонить запрос на выгрузку. Модуль принятия решений модуля мини-облака отправляет ответное сообщение с решением на модуль микро-облака. Если и микро-, и мини-облачные устройства не могут принять запрос на выгрузку, SUAV блокирует задачу.

3.2.3 Оценка эффективности

Проведем оценку производительность предложенного алгоритма выгрузки с учетом задержки и энергии для сети БПЛА с гетерогенными возможностями. Наземные станции подключены к многоуровневой системе MEC, которая обеспечивает возможности для выгрузки с земли.

A. ПАРАМЕТРЫ МОДЕЛИРОВАНИЯ

Предложенный алгоритм был промоделирован для системы, состоящей из воздушного и наземного сегмента с помощью Matlab. Воздушный сегмент имеет пять дронов с разнородными вычислительными и энергетическими возможностями. Предполагается, что пять дронов случайным образом распределены по воздушному пространству радиусом R_{air} на высоте h в

диапазоне, указанном в таблице 3.2.3.1.

Таблица 3.2.3.1 - Параметры моделирования

Параметр	**Значение**
h	ϵ [50,100] m
R_{air}	2Km
R_{cell}	1Km
δ_{UAV}	1J/GHz
δ_{Micro}	1J/GHz
F_{UAV} (ID=1)	ϵ [0.5,1.5] GHz
F_{UAV} (ID=2)	ϵ [1.0,2.0] GHz
F_{UAV} (ID=3)	ϵ [0.5,1.0] GHz
F_{UAV} (ID=4)	ϵ [0.7,1.2] GHz
F_{UAV} (ID=5)	ϵ [0.5,1.5] GHz
ω	20 MHz
σ	-10 dBm
P	20 dBm
$f_{Micro\text{-}cloud}$	ϵ [2.0,4.0] GHz
$f_{Mini\text{-}cloud}$	ϵ [3.0,6.0] GHz

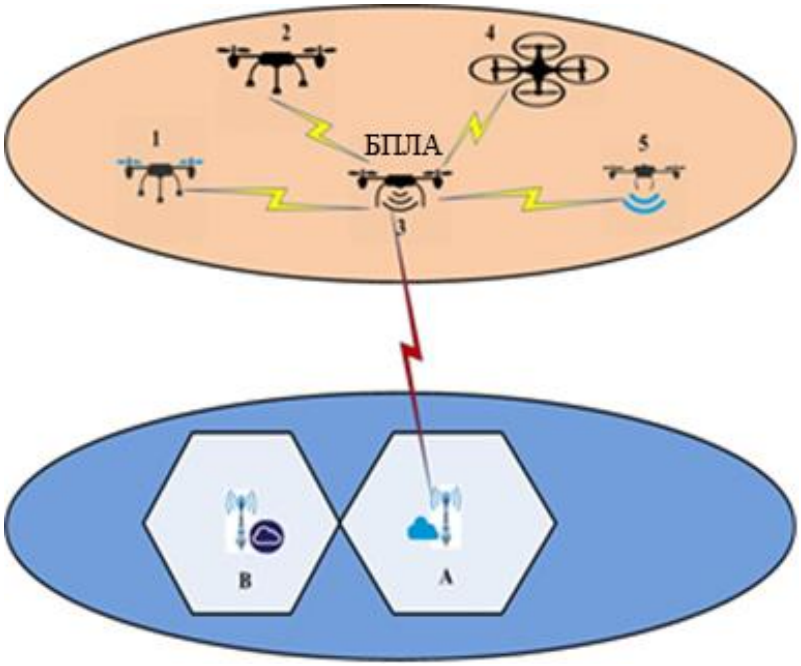


Рисунок 3.2.3.1 - Топология сети, рассматриваемая для моделирования

Траектория полета, введенная в [242], используется для БПЛА в воздушном сегменте соединения. Наземный сегмент состоит из двух сотовых ячеек радиусом *Rcell*, каждая ячейка имеет базовую станцию, расположенную в центре ячейки. Базовая станция ячейки *A* подключена к граничному серверу микро-облачной среды с вычислительными возможностями представленными в табл. 3. Базовая станция соседней соты *B* подключена к граничному мини-серверу с вычислительными возможностями, представленными в таблице 3.2.3, и оба граничных сервера взаимодействуют между собой. Топология, рассмотренная для оценки производительности, показана на рисунке 3.2.6. Параметры моделирования приведены в таблице 3.2.3.

Для целей моделирования рассматриваются три основных сценария с различными случаями для каждого сценария.

1. Сценарий (A)

На исходном БПЛА рассматриваются десять разнородных задач с разной нагрузкой, поступающие последовательно. Эти задачи выполняет исходный БПЛА на основе предложенного алгоритма выгрузки локально, либо путем выгрузки их на воздушное или наземное устройство в зависимости от доступных ресурсов. В таблице 3.2.3.1 указан объем рассматриваемых задач, который достаточно часто соответствует загруженности изображениями с разным разрешением. Предполагается, что БПЛА имеет идентификатор 3. Для этого сценария рассматриваются три случая, в каждом случае для каждой задачи предполагается определенное значение максимально допустимой задержки, соответствующее QoS. В таблице 3.2.3.2 указаны значения QoS с точки зрения задержки для каждой задачи в каждом случае.

Таблица 3.2.3.1 - Спецификация задания

Задача	**Задача(1)**	**Задача(2)**	**Задача (3)**	**Задача(4)**	**Задача(5)**
Z(kB)	200	270	320	250	370
Задача	**Задача**(6)	**Задача**(7)	**Задача**(8)	**Задача**(9)	**Задача**(10)
Z(kB)	400	450	390	500	520

Таблица 3.2.3.2 - QoS Значения задержки для различных задач

Задача	**Задача (1)**	**Задача (2)**	**Задача (3)**	**Задача (4)**	**Задача (5)**
C_1(ms)	10	13.5	16	12.5	18.5
C_2(ms)	15	20.25	24	18.75	27.75
C_3(ms)	20	27	32	25	37
Задача	**Задача (6)**	**Задача (7)**	**Задача (8)**	**Задача (9)**	**Задача (10)**
C_1(ms)	20	22.5	19.5	25	26

C_2(ms)	30	33.75	29.25	37.5	39
C_3(ms)	40	45	39	50	52

Сценарий (Б)

В этом сценарии предыдущие рассмотренные задачи случайным образом распределяются между 5 дронами. В таблице 3.2.3.3 указаны выделенные задачи для каждого БПЛА. Задачи выполняются одновременно, однако заявки на БПЛА с более чем одной задачей ставятся в очередь до тех пор, пока не будет выполнена текущая задача. В этом сценарии также рассматриваются три изученных случая в предыдущем сценарии.

Таблица 3.2.3.3 - Выделенные задачи для каждого дрона

БПЛА ID	**БПЛА 1**	**БПЛА 2**	**БПЛА 3**	**БПЛА 4**	**БПЛА 5**
Выделенная задача	3,7	1,6,9	2,4	10	5,8

Сценарий (С)

Этот сценарий предложен для оценки производительности и важности каждого уровня выгрузки. Система моделируется пять раз. каждый раз представлен результат моделирования. В первом случае система моделируется без выгрузки. Во втором случае система моделируется с учетом только выгрузки по воздуху и предполагается, что возможность выгрузки с земли отсутствует. Третий случай предполагает наличие только наземной выгрузки и отсутствие возможности воздушной выгрузки. В этом случае предполагается, что наземная выгрузка представляет собой однородные серверы MEC (т. е. микро-облака), а мини-облака отсутствуют. В четвертом случае предполагается только наземная выгрузка без воздушной выгрузки, при этом развернуты блоки микро-облака и

мини-облака. В последнем случае система моделируется с развертыванием обоих; выгрузка осуществляется по воздуху и по земле с многоуровневым MEC. Рассматриваемые задачи такие же, как и в первом сценарии с БПЛА с идентификатором 3. Таким образом, сценарий (A) можно считать пятым случаем сценария (B). Сценарий моделирования (C) рассматривается для указания важности и влияния каждого из рассмотренных уровней выгрузки на производительность.

Рисунок 3.2.3.4 иллюстрирует среднюю задержку для обработки каждой задачи в каждом случае сценария моделирования (A). Некоторые рассматриваемые задачи обрабатываются локально на БПЛА без необходимости выгрузки, в то время как другие задачи выгружаются из-за того, что имеющихся ресурсов недостаточно для обработки задач в рамках QoS с задержкой. На рисунке 3.2.3.5 показан уровень выгрузки для каждой задачи в каждом случае. Часть выгруженных задач выполняется на ближайших БПЛА, а остальные выгружаются на наземные многоуровневые серверы МЭК, где эти задачи и выполняются. Без выгрузки с земли часть задач БПЛА не решалась и поэтому часть задач блокировалась.

Результаты для сценария моделирования (B) представлены на рисунках 3.2.9 и 3.2.10. На рисунке 3.2.3.6 показана средняя задержка обработки каждой задачи в каждом рассматриваемом случае сценария (B), а на рисунке 3.2.10 представлен уровень выгрузки для каждой задачи в каждом случае, который указывает, где выполняется каждая задача. В этом сценарии всем БПЛА назначаются задачи, и это снижает вероятность того, что ближайший БПЛА имеет доступные ресурсы для поддержки выгрузки с воздуха. Задачи 1,2,3,5 и 10 распределяются между 5 дронами одновременно, таким образом, у каждого БПЛА есть задача для выполнения. Таким образом, эти задачи либо

выполняются локально (т. е. нулевой уровень выгрузки), либо выгружаются на земле (т. е. второй или третий уровни выгрузки).

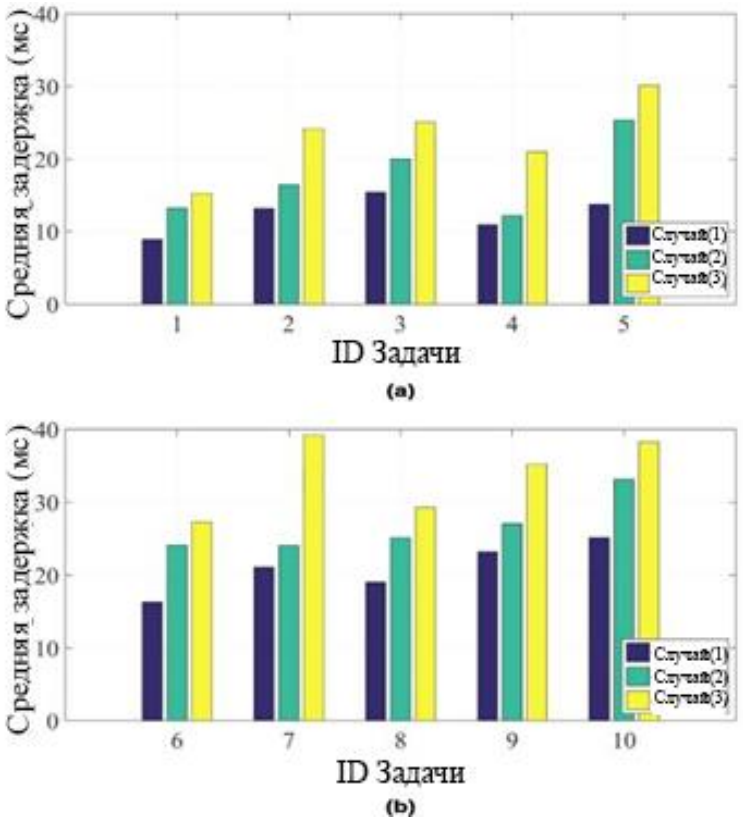


Рисунок 3.2.3.4 - Средняя задержка обработки задач для разных случаев сценария (А)

2. Сценарий (Б)

В этом сценарии рассмотренные ранее задачи случайным образом распределяются между пятью дронами. В табл. 6 указаны выделенные задачи для каждого БПЛА. Задачи выполняются одновременно, однако заявки на БПЛА с более чем одной задачей ставятся в очередь до тех пор, пока выполняется текущая задача. Три рассмотренных случая в предыдущем сценарии также рассматриваются и в этом сценарии.

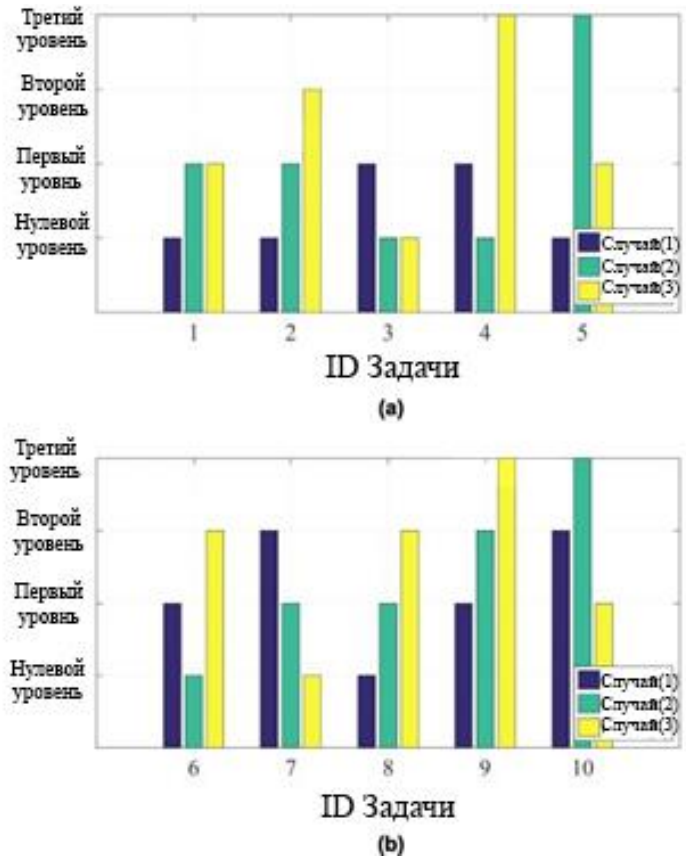


Рисунок 3.2.3.5 - Уровни выгрузки каждой задачи для разных случаев сценария (A)

3. Сценарий (C)

Этот сценарий предлагается для оценки производительности и важности каждого уровня выгрузки. В этом сценарии моделирование системы осуществляется пять раз; каждый раз представлен результат моделирования. В первом случае система моделируется без выгрузки. Во втором случае система моделируется с учетом только выгрузки по воздуху и предполагается, что возможность выгрузки с земли отсутствует. Третий случай предполагает наличие только наземной выгрузки и отсутствие возможности воздушной выгрузки. В этом случае предполагается, что наземная выгрузка представляет собой однородные серверы MEC (т. е. микро-облачные вычисления), а мини-облачные

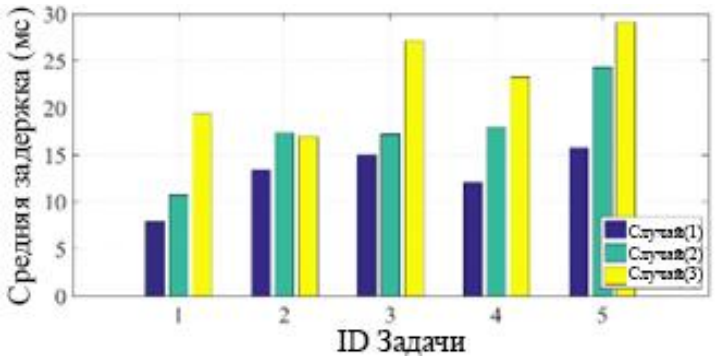
вычисления отсутствуют. В четвертом случае предполагается только наземная выгрузка без воздушной выгрузки, при этом используются как микро-облачные, так и мини-облачные серверы. В последнем случае система моделируется с использованием обоих вариантов выгрузок, как по воздушному сегменту, так и наземному с многоуровневым МЕС. Рассматриваемые задачи те же, что и в первом сценарии с исходным БПЛА с идентификатором 3. Таким образом, сценарий (A) можно считать пятым случаем сценария (B). Сценарий моделирования (C) рассматривается для указания важности и влияния каждого из представленных уровней выгрузки на производительность.

3.2.4 Результаты моделирования и анализ

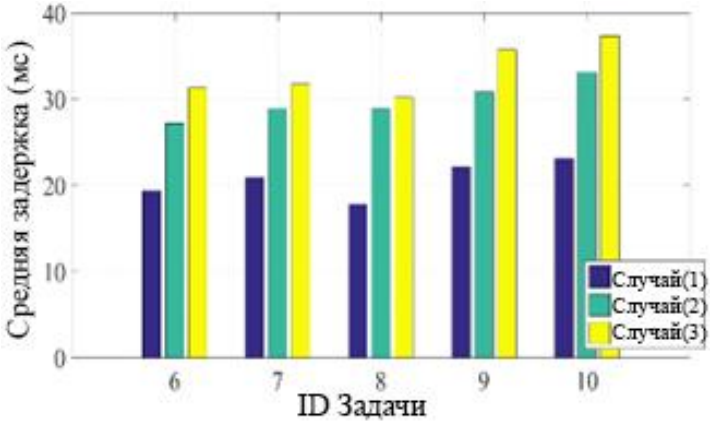
Рисунок 3.2.3.4 иллюстрирует среднюю задержку для обработки каждой задачи в каждом случае сценария моделирования (A). Некоторые рассматриваемые задачи решаются локально на БПЛА без необходимости для выгрузки, в то время как другие задачи выгружаются из-за того, что доступных ресурсов недостаточно для обработки задач в рамках требований к задержке. На рисунке 3.2.3.5 показан уровень выгрузки для каждой задачи в каждом случае. Часть выгруженных задач выполняется на ближайших БПЛА, а остальные выгружаются на наземные многоуровневые серверы МЕС, где эти задачи и выполняются. Без выгрузки с земли ряд задач БПЛА не могли быть решены, и поэтому некоторые задачи были бы заблокированы.

Результаты для сценария моделирования (B) представлены на рисунках 3.2.3.6 и 3.2.3.7. На рисунке 3.2.3.6 показана средняя задержка обработки каждой задачи в каждом рассматриваемом случае сценария (B), а на рисунке 3.2.3.7 представлен уровень выгрузки для каждой задачи в каждом случае, который указывает, где выполняется каждая задача. В этом сценарии всем БПЛА

назначаются задачи, и это снижает вероятность того, что ближайший БПЛА имеет доступные ресурсы для поддержки выгрузки с воздуха. Задачи 1 - 5 и 10 распределяются между пятью дронами одновременно, таким образом, у каждого БПЛА есть задача для выполнения. Следовательно, эти задачи либо выполняются локально (т. е. нулевой уровень выгрузки), либо выгружаются на земле (т. е. второй или третий уровни выгрузки).



(a)



(b)

Рисунок 3.2.3.6 - Средняя задержка обработки задач для разных случаев сценария (B)

Результаты для сценария моделирования (B) представлены на рисунках

3.2.3.6 и 3.2.3.7. На рисунке 3.2.3.6 показана средняя задержка обработки каждой задачи в каждом рассматриваемом случае сценария (B), а на рисунке 3.2.3.7 представлен уровень выгрузки для каждой задачи в каждом случае, который указывает, где выполняется каждая задача. В этом сценарии всем БПЛА назначаются задачи, и это снижает вероятность того, что ближайший БПЛА имеет доступные ресурсы для поддержки выгрузки с воздуха. Задачи 1-5 и 10 распределяются между пятью дронами одновременно, таким образом, у каждого БПЛА есть задача для выполнения. Следовательно, эти задачи либо выполняются локально (т. е. нулевой уровень выгрузки), либо выгружаются на землю (т. е. второй или третий уровни выгрузки).

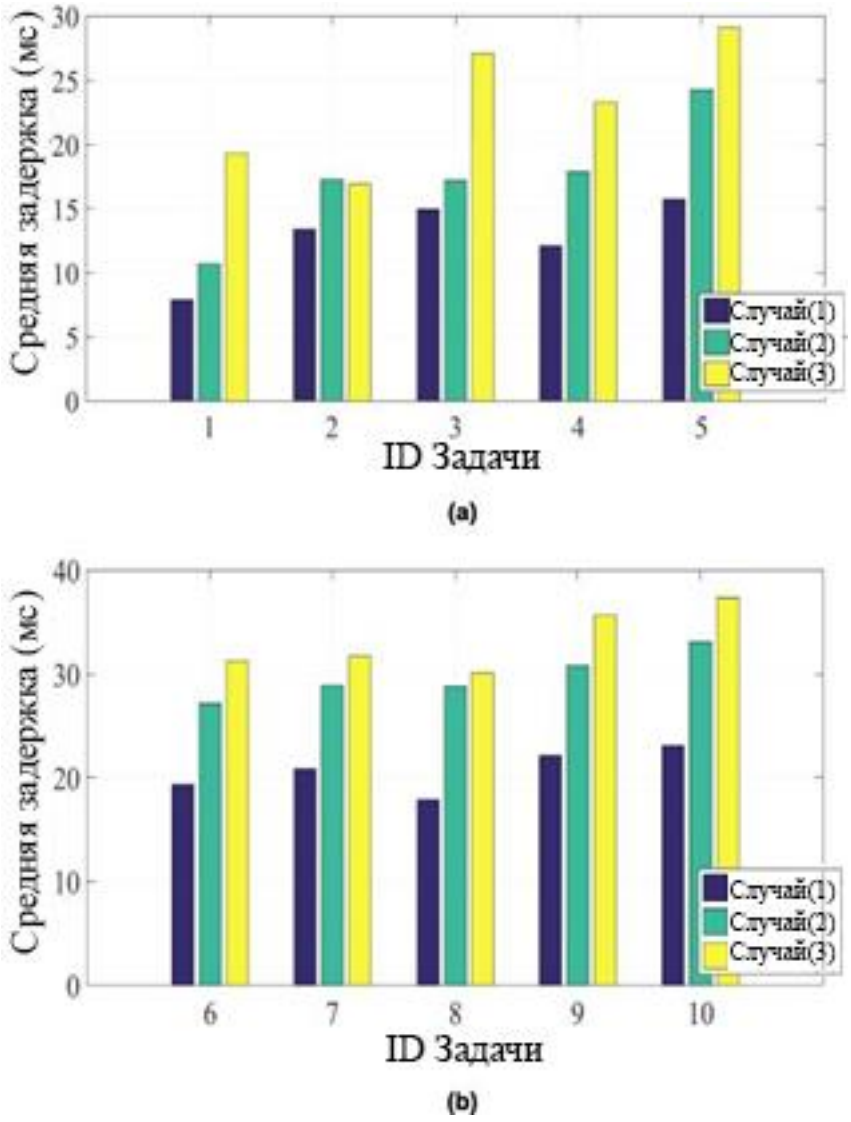


Рисунок 3.2.3.7 Уровни выгрузки каждой задачи для разных случаев сценария (B).

Для оценки важности и эффективности каждого уровня выгрузки рассматривается сценарий моделирования (C). На рисунке 3.2.3.8 показано сравнение пяти рассмотренных случаев сценария моделирования (C) с точки зрения процента заблокированных задач. Система сначала разворачивается без выгрузки, а обработка задач происходит последовательно. Некоторые задачи решаются локально, а другие задачи блокируются из-за недоступности ресурсов

(например, доступные ресурсы обрабатывают задачу за время, превышающее время задержки QoS Ƈ). Заблокированные задачи были определены, и процент заблокированных задач указан на рисунке 3.2.11. Это повторяется три раза, каждый раз соответствует значению пороговой задержки Ƈ, указанному в таблице 2.2.5. Для случая (2) система работает с фиксацией локального выполнения и выгрузки по воздуху, а также с общим количеством заблокированных задач. Как показано на рисунке 3.2.3.7, количество заблокированных задач уменьшается при использовании выгрузки по воздуху помимо локальной обработки. В третьем случае рассмотрим выгрузку на землю, кроме местной выгрузки, и предположим, что выгрузка по воздуху отсутствует. В этом случае количество заблокированных задач уменьшается по сравнению с двумя предыдущими случаями, но некоторые заблокированные задачи все же есть. Это означает, что выгрузки на наземный сегмент, помимо локальной обработки, недостаточно, а также все еще существует блокировка. Таким образом, БПЛА нуждаются в выгрузке по воздуху и на землю для достижения наилучшей производительности. Это пятый случай, который указывает на нулевую блокировку, как показано на рисунке 3.2.11.

Чтобы выяснить значение каждого уровня выгрузки и влияние добавления этих уровней на производительность, процент улучшения блокировки задач рассчитывается для каждого уровня выгрузки по отношению к локальному выполнению. В таблице 3.2.7 показано процентное улучшение блокировки задач, достигнутое каждым уровнем выгрузки по сравнению с локальной обработкой для предыдущего рассмотренного сценария для трех рассмотренных значений задержки QoS, а также среднее значение процентного улучшения для трех представленных к рассмотренных случаев латентности. Результаты показывают, что использование как наземной, так и воздушной выгрузки обеспечивает

наилучшее улучшение производительности системы с точки зрения заблокированных задач.

Таблица 3.2.3.4 – Процент улучшения блокировок задач для каждого уровня разгрузки

Уровень выгрузки	**Выгрузка первого уровня**	**Выгрузка второго уровня**	**2nd и 3rd уровни Выгрузки**	**Все уровни Выгрузки**
	Выгрузка по воздуху	Выгрузка на микро-облако	Выгрузка на наземный сегмент	Выгрузка через воздушный и наземный сегмент
C_1	67%	83.3%	100%	100%
C_2	57%	71.4%	71.4%	100%
C_3	37%	75%	87.5%	100%
Средний	53.8%	76.6%	86.3%	100%

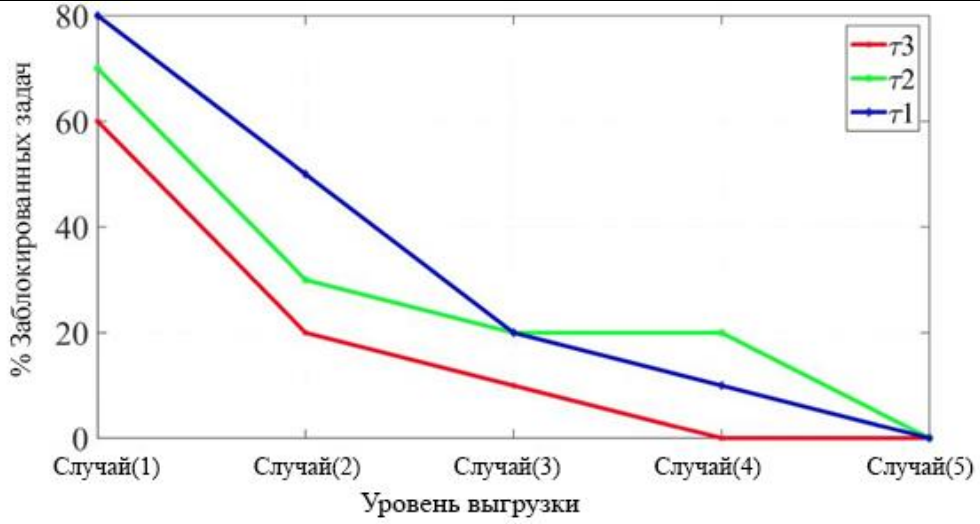


Рисунок 3.2.3.8 - Процент заблокированных задач для каждого случая сценария

(C)

В этом подразделе был представлен алгоритм выгрузки БПЛА с учетом энергозатрат и задержки для приложений, критичных ко времени. Алгоритм включает три основные процедуры:

- первая отвечает за определение места выполнения задачи или метода выгрузки, если это необходимо. Механизм принятия решений основан на наличии доступных ресурсов и требуемых ограничений QoS;

- вторая отвечает за процесс выгрузки по воздуху, который позволяет БПЛА выгрузить свои вычислительные задачи ближайшим БПЛА с доступными вычислительными и энергетическими ресурсами;

- третья процедура отвечает за выгрузку с земли, которая запускается, если выгрузка с воздуха не удалась. Наземная выгрузка основана на многоуровневой системе MEC, которая обеспечивает гетерогенные вычислительные возможности на границе RAN. Предложенный алгоритм достигает более высокой эффективности с точки зрения задержки и вероятности блокировки.

В последующем целесообразно будет решить проблему выгрузки на воздушные и наземные устройства, например, в пространства с поддержкой Интернета вещей (IoT) [243] или в городскую инфраструктуру [244]. Еще одним направлением исследований является изучение безопасности сети БПЛА, особенно в отношении аутентификации. Многие облегченные методы аутентификации, разработанные специально для IoT, например, [245], теоретически могут быть применены к сетям БПЛА.

Таким образом, предложен трехуровневый метод выгрузки, причем на оконечных устройствах используется программный профилировщик, который

определяет сложность вычисляемой задач и по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов о выгрузке трафика на сервер другого БПЛА.

Для решения научной задачи оптимизации структуры сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности в отличии от известных решений для минимизации задержки и энергопотребления при выгрузке трафика с наземной сети на серверы граничных вычислений БПЛА разработан метаэвристический алгоритм на основе хаотического роя сальп.

Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений.

3.3 Энергоэффективный алгоритм выгрузки трафика на основе роя БПЛА

Ожидается, что к 2025 году рыночная выручка от беспилотных летательных аппаратов (БПЛА) достигнет 52 миллиардов долларов, что значительно больше по сравнению с сегодняшним рынком беспилотных летательных аппаратов [248,249]. К тому времени беспилотники обеспечат улучшение качества работы коммерческого, гражданского и государственного секторов за счет внедрения широкого спектра разнородных приложений [250,251]. Эти приложения будут варьироваться от простого мониторинга окружающей среды до сложных приложений с высоким уровнем безопасности.

Основываясь на стандартах версии 14 3GPP и технических отчетах МСЭ-Р, беспилотные летательные аппараты считаются одними из наиболее важных аспектов сотовой системы пятого (5G/IMT2020) и последующих поколений [252-255]. Беспилотники будут активно использоваться для различных целей, таких как дистанционное зондирование, дистанционная локализация, обнаружение целей, мониторинг транспорта и оценка маршрута и т.д. [256-258].

В последнее время было предложено несколько системных архитектур, основанных на граничных вычислениях, для использования в сетях связи беспилотных летательных аппаратов, в которых интенсивные вычислительные задачи перенсятся с мобильного пользователя на более производительный информационно-вычислительный ресурс, расположенный на границе сети радиодоступа (RAN) [259-264]. В нескольких исследованиях рассматривается выгрузка многоуровневых вычислений [267], однако большинство из этих архитектур учитывают только два уровня выгрузки вычислений в системах UAV-MEC [265,266]. Основная часть из этих архитектур преследуют общую цель минимизации энергопотребления, эффективного распределения радио- и вычислительных ресурсов и удовлетворения требований мобильных пользователей к задержке. Однако для поддержки сети, в которой работают беспилотники, необходимо учитывать ряд требований, таких как низкое энергопотребление и обработка информации. Кроме того, в крупномасштабной среде количество подключенных серверов в граничной сети ограничено вычислительной мощностью, что может приводить к большим задержкам, и делает эти устройства непригодными для задач реального времени. Поэтому, поиск наилучшего решения для выгрузки в многопользовательской беспроводной среде с несколькими беспилотниками считается достаточно сложной задачей.

Для решения этих проблем в диссертационной работе предложено построить энергоэффективную модель для многопользовательской сети граничных облачных вычислений с поддержкой нескольких беспилотных летательных аппаратов. В дополнение к этому комбинированная модель распределения ресурсов и выгрузки сформулирована как задача, направленная на минимизацию энергопотребления всей системы с ограничением по задержке. Более того, энергоэффективный алгоритм выгрузки задач для нескольких беспилотных летательных аппаратов разработан для получения решения о выгрузке задач для всех пользователей устройства. Основные положения этого подраздела включают:

- Энергоэффективную модель и метод построения сети для многопользовательских мобильных облачных вычислений с поддержкой нескольких беспилотных летательных аппаратов. Разработанный метод позволяет построить сеть, которая является масштабируемой и может поддерживать увеличенный сетевой трафик без снижения производительности;
- Созданная сеть беспилотных летательных аппаратов обеспечивает покрытие мертвых зон и зон с высокой плотностью сети. Кроме того, беспилотники могут предоставлять другие приложения через сеть умного города, в которой развернута многоуровневая технология MEC для обеспечения как вычислительных возможностей граничной сети, так и базовой сети на основе технологии SDN для управления сетевым трафиком и предоставления инновационного интерфейса сетевым операторам;

- Комбинированная модель и метод распределения ресурсов и выгрузки сформулированы как задача, направленная на минимизацию энергопотребления всей системы с помощью ограничения задержки;
- Моделирование подтверждает эффективность предложенных модели и метода и демонстрирует, что энергопотребление системы может быть значительно снижено.

Существующие исследования по подходам к выгрузке вычислений представлены в подразделе 3.3.1. В подразделе 3.3.2 представлена модель системы, за которой следует формулировка задачи оптимизации. Затем в подразделе 3.3.3 оцениваются результаты имитационного моделирования, чтобы подтвердить факт повышения производительности при использовании предложенной модели и метода.

3.3.1. Существующие исследования в предметной области

В последние годы появилось несколько системных архитектур и оптимизационных моделей на базе БПЛА-МЕС, разработанных для решения основных задач пользователей, где применяется концепция выгрузки вычислений. В большинстве этих подходов рассматривались только сети МЕС с одним [268,269] или несколькими граничными серверами [270,271], и только в некоторых рассматривались сети МЕС с несколькими граничными серверами и централизованным облаком [264,267,272]. В этом разделе представлен краткий обзор существующих подходов, основанных на сетях МЕС с одиночными и многоуровневыми серверами с централизованным облаком и без него.

Одиночный граничный сервер

You C. и др. [267] предложили энергоэффективную платформу для беспроводных мобильных облачных вычислений, которая объединяет передачу энергии микроволновой трансляции с мобильными облачными вычислениями.

Эта структура предназначена только для однопользовательской системы с одним граничным сервером, например, базовая станция (BS) либо передает мощность, либо перегружает вычисления с мобильного устройства в облако, а мобильное устройство в свою очередь использует собранную энергию для вычисления заданных данных локально либо путем выгрузки. Кроме того, задача оптимизации формулируется в виде невыпуклой функции, целью которой является минимизация потребления энергии мобильным пользователем для сбора данных о локальных вычислениях при максимальной экономии энергии мобильными пользователями для выгрузки вычислений. Между тем, в [268] предложена эффективная онлайновая платформа выгрузки, основанная на глубоком обучении с подкреплением. В частности, глубокая нейронная сеть используется для изучения и определения решения о выгрузке для мобильных пользователей таким образом, чтобы они могли либо выполнять вычислительные задачи локально, используя ресурсы мобильного устройства, либо выгружать их на граничный сервер для удаленного выполнения. Кроме того, разработана адаптивная процедура для снижения сложности за счет автоматической настройки гиперпараметров модели глубокого обучения. Наконец, численные и экспериментальные результаты показывают, что эта структура может обеспечить близкое к оптимальному решение.

В [273] был предложен эффективный алгоритм, основанный на глубоком обучении с мета-подкреплением для IoT при использовании облачных систем с целью снижения вычислительной нагрузки и улучшения характеристик обработки задач. В частности, различное количество деконволюционных сетей (DNN) объединяются с Q-обучением для принятия эффективного решения о выгрузке и повышения обучаемости системы. Кроме того, результаты показали, что предложенный алгоритм превосходит схемы бинарной и частичной

выгрузки. Между тем, в [274] оптимизировано качество восприятия (QoE) в среде Интернета вещей с поддержкой граничных вычислений, в которой учитываются задержка передачи и вычислений, вероятность успешного выполнения каждой задачи и потребление энергии. Для решения предложенной модели используется глубокое обучение с подкреплением, где алгоритмы на основе двойного Q-обучения и дуэльных сетей разработаны для ускорения сходимомти и повышения стабильности. Аналогичным образом ограничения вычислительных возможностей устройств Интернета вещей и энергопотребление рассматриваются в [274], где предлагается интегрированная структура для выгрузки задач и распределения ресурсов с целью максимизации энергоэффективности при одновременном снижении ограничений по качеству обслуживания (QoS). Однако одним из распространенных недостатков [274] является то, что передаваемые данные подвержены различным атакам.

Несколько граничных серверов

Минимизация энергопотребления мобильных устройств и сокращение количества мобильных приложений, а также задержка выполнения в многопользовательской системе является основной целью работы [270], в которой скорость вычислений и мощность передачи для мобильных устройств совместно оптимизируются с коэффициентом выгрузки. В частности, авторы сформулировали энергопотребление и задержку выполнения как невыпуклые задачи. Затем они использовали методы подстановки переменных и одномерного поиска, чтобы получить оптимальное решение. Их результаты демонстрируют, что предлагаемая модель может значительно снизить энергопотребление и сократить время ожидания по сравнению с существующими методами выгрузки. Многопользовательская модель выгрузки с несколькими ребрами для статических и динамических каналов предложена в

[270]. Во-первых, на статическом канале решение о выгрузке вычислений формулируется как некооперативная потенциальная игра, в которой каждое мобильное устройство может эгоистично максимизировать циклы процессора и минимизировать потребляемую энергию. На динамическом канале (т.е. при изменяющемся во времени и неизвестном состоянии канала), сформулирован алгоритм выгрузки распределенных вычислений на основе Q-обучения для определения оптимального решения, при котором может быть достигнуто равновесие по Нэшу. Наконец, численные результаты показали, что эта модель не только может превосходить показатели энергопотребления по сравнению с локальной обработкой и случайным назначением, но также может улучшить среднюю долгосрочную выгрузку до 87,87% по сравнению с идеальным вариантом получения информации о состоянии канала.

Янг и др. [275] предложили энергоэффективный метод для многозадачной выгрузки данных на основе кластеров беспилотных летательных аппаратов для игр виртуальной и дополненной реальности в транспортных сетях. Гибридная игровая сцена VR/AR должна позволить водителям и пассажирам пользоваться различными визуальными эффектами в режиме реального времени. Однако из-за динамических изменений в распределении пользователей в режиме реального времени фиксированные граничные узлы не справляются с изменяющейся нагрузкой. Таким образом, использование беспилотных летательных аппаратов вместе с фиксированными узлами в качестве граничных узлов помогает справиться с вычислительной нагрузкой и удовлетворить требования автомобильных сетей. Развертывание большого количества беспилотных летательных аппаратов нецелесообразно из-за взаимных помех и высокой стоимости беспилотных летательных аппаратов. Для решения этих проблем в статье представлена система со следующими характеристиками: многозадачная

выгрузка (беспилотники могут работать над несколькими задачами и результаты одной задачи могут быть использованы для выполнения других задач в последующем); кластеры беспилотных летательных аппаратов, работающие над одной задачей; оптимизация вычислений; кэширование и коммуникационные ресурсы, а также искусственный интеллект – и принятие решений, основанных на этом методе. Кроме того, предлагается архитектура для кластеров беспилотных летательных аппаратов с различными реализациями задач, в которой рассматриваются все крупномасштабные приложения для VR/AR игр и транспортных сетей. Также предлагается система принятия решений на основе искусственного интеллекта для облегчения взаимодействия между беспилотными летательными аппаратами и совместной оптимизации вычислений, кэширования и коммуникационных ресурсов. Предлагаемая архитектура, по сравнению с традиционными структурами MEC (как с беспилотниками, так и без них), имеет преимущества в объеме собираемой информации, производительности в режиме реального времени, принятии решений, а также работоспособности и эффективности. Были рассмотрены два алгоритма: предварительное развертывание, основанное на анализе хранящихся исторических данных, и развертывание в реальном времени, основанное на восприятии в реальном времени. Кроме того, в этой работе был проведен эксперимент по прогнозированию загрузки нескольких беспилотных летательных аппаратов на основе долгой краткосрочной памяти (LSTM).

Чжан, У. и др. [276] представили модель выгрузки много серверных задач децентрализованным игровым способом с целью максимизации полезности системы. Предложен децентрализованный и эффективный алгоритм для получения оптимальной выгрузки, в котором подходы к дифференциальным нейронным вычислениям и модели обучения с глубоким подкреплением

комбинируются без необходимости предварительного знания предпочтений пользователя и пропускной способности сети. Между тем, в [277] беспилотные наземные транспортные средства (UGVS) представлены в качестве эффективного решения оптимизации вычислительных ресурсов и ограничения ресурсов батареи беспилотных летательных аппаратов, при которых интенсивные задачи могут быть выгружены с беспилотных летательных аппаратов для выполнения с помощью UGVS. Более того, разработан новый алгоритм стабильного сопоставления для преобразования задачи выгрузки в эквивалентную форму двустороннего сопоставления. Далее вводится итеративный алгоритм для сопоставления беспилотных летательных аппаратов с соответствующими UGVS. Тем не менее, несмотря на улучшения, зафиксированные в [278,279], передаваемые данные недостаточно защищены от различных типов атак во время передачи.

Несколько граничных серверов и удаленное облако

Бэйцин и др. [267] предложили энергоэффективную платформу граница-облако для выгрузки вычислений системы с несколькими беспилотниками. Они сформулировали задачу оптимизации как трехуровневое взаимодействие устройств, в котором решение о выгрузке могло быть определено децентрализованным способом. Кроме того, разрабатываются эффективные децентрализованные алгоритмы, в которых производительность взаимодействия анализируется с помощью коэффициента ее эффективности и может быть достигнуто равновесие по Нэшу. Наконец, экспериментальные результаты демонстрируют, что эти алгоритмы могут значительно снизить потребление энергии на 30% при потере производительности менее чем на 10% по сравнению с другими эталонными решениями. Лю Ф. и др. [272] представили энергоэффективную облачную платформу граничных вычислений для

совместной выгрузки вычислений задач в датчиках интернета вещей. Эта структура состоит из трехуровневой сети с датчиками интернета вещей на первом уровне и граничным сервером и облачным сервером на двух других уровнях. Кроме того, стратегия выгрузки была сформулирована как задача смешанного целого числа, в которой минимизируется потребление энергии. Главной целью было использование датчиков интернета вещей. Для решения этой проблемы и получения оптимального решения по выгрузке был разработан энергоэффективный алгоритм совместной выгрузки задач, основанный на полуопределенного ослабления. Наконец, результаты моделирования показали, что этот алгоритм может превзойти существующий алгоритм с точки зрения энергопотребления. Кроме того, он может эффективно адаптироваться к различным параметрам системы.

Стандартная структура MEC с фиксированными узлами не способна удовлетворить потребности пользователей в вычислительных ресурсах и соединениях с быстро меняющимися характеристиками во времени и пространстве. Чтобы решить эту проблему, в статье [280] обсуждается использование системы "воздух-земля". MEC – интегрированная среда, состоящая из развернутых беспилотных летательных аппаратов и наземных транспортных средств в качестве дополнительных граничных узлов. Беспилотные летательные аппараты позволяют обеспечить покрытие сети в сложных географических районах и предоставлять более широкий выбор услуг. Предлагаемая сеть состоит из трех уровней: уровень облачных вычислений, уровень граничных вычислений и уровень устройств.

Интегрированная структура предполагает использование основных преимуществ, предоставляемых программно-конфигурируемыми сетями: централизация управления, отделение пользовательских данных от

управляющих и большая абстракция базовых структур. Гипервизор в этой сети отвечает за виртуализацию ресурсов пользовательских устройств, а также за вычислительные ресурсы и ресурсы хранения верхнего уровня (относительно уровня устройства). Варианты предлагаемой сетевой структуры соответствуют основным сценариям в сетях: поддержка сверхнизкой задержки – URLLC, распределенной обработки и анализа больших данных наряду с кэшированием контента и доставкой на мобильные устройства – eMBB, а также масштабируемых массивов и управления мобильными коммуникациями – mMTC. Преимущества, предоставляемые структурой, включают адаптивное распределение ресурсов, дифференцированные гарантии качества обслуживания, улучшенную связь между узлами сети и эффективное управление мобильностью.

Задержка при передаче данных с мобильных устройств в облако удаленных вычислений приводит к большой нагрузке в существующих системах выгрузки трафика. Чтобы решить эту проблему, в [281] разработан код для системы выгрузки трафика, ориентированной на границу сети, под названием Echo. Сетевая структура для кода Echo состоит из трех уровней: мобильные устройства, граница и облако. Echo-код использует централизованный механизм, в котором решение о выгрузке трафика принимается на граничных узлах. Код для выгрузки трафика реализован на Android устройствах. Код Echo решает проблему того, какой метод и какая платформа используются – облачная платформа или граничная. Для более эффективного использования ресурсов граничных узлов используется алгоритм планирования с ограничением по времени с наименьшим оставшимся временем (PC-SRTF), позволяющий минимизировать среднее время решения задачи. Предлагаемый код обеспечивает гарантию качества обслуживания при выгрузке трафика на

граничных узлах. Для оптимизации передачи данных используются алгоритмы отложенной передачи и дифференциального обновления. В первом алгоритме, когда трафик выгружается, создается прокси-сервер для передачи связанных объектов. Во втором способе обновляются только те части объектов, которые отличаются при выгрузке нескольких методов на мобильное устройство. Было обнаружено, что предлагаемый Echo-код имеет меньшее среднее время выполнения задачи и энергопотребление для мобильных устройств по сравнению с существующими системами выгрузки трафика.

В последнее время в [282,283] была представлена многопользовательская модель с поддержкой нескольких беспилотных летательных аппаратов и облачных серверов. В частности, в [282] предложена новая модель взвешенных затрат с целью минимизации энергопотребления и времени выполнения. Кроме того, разработаны эффективный алгоритм размещения, позволяющий эффективно распределять задачи на наиболее подходящий сервер и облегченный алгоритм восстановления после сбоя, для оптимизации параллельного выполнения задач. Ся и др. [283], однако, предложили модель интеллектуальной выгрузки задачи многоуровневой сети MEC, в которой основными целями являются минимизация задержки выполнения задач и удовлетворение вычислительных возможностей беспилотных летательных аппаратов. Более того, с использованием метода двухслойной оптимизации задана эквивалентная и выпуклая форма сформулированной задачи. На первом уровне используется подход дифференциального эволюционного обучения, в то время как на другом уровне используется распределенная глубокая нейронная сеть. Однако измерение времени выполнения не рассматривается в [283], что является одним из основных недостатков этой работы.

Из-за высокой динамики в городских сетях, которые приводят к перегрузке соединения между БПЛА и граничным сервером (что приводит к увеличению времени, затрачиваемого на выполнение задач), а также из-за больших очередей, возникающих на граничных серверах при обработке нескольких задач, общее время выполнения передачи задачи на граничный сервер и время обработки задачи могут превышать время, затраченное на анализ задачи локально на беспилотном летательном аппарате. Здесь представлен процесс оптимизации, который позволяет пользователю выбирать, где будет обработано задание. При этом рассматриваются коммуникационные, вычислительные и энергетические аспекты предлагаемого процесса.

В таблице 3.3.1 представлена краткая информация о соответствующих работах, рассмотренных с точки зрения количества пользователей, граничных и удаленных облачных серверов, и выделены основные слабые стороны существующих исследований.

Таблица 3.3.1.1 - Обзор исследований MEC

Ссылка	Цель	Предложенное решение	Пользователь		Сервер БПЛА	Удаленное облако		Методы оценки	Недостаток
			Одиночный	Множественный	Одиночный	Множественный			
[267]	Свести к минимуму потребление энергии	Энергоэффективная платформа для беспроводных мобильных облачных вычислений	✓		✓		✓	Моделирование	Учитывать в модели только одного пользователя интернета вещей нереально.
[268]	Максимально увеличить взвешенную сумму скорости вычислений	Онлайн-платформа для обучения с глубоким подкреплением для мобильных граничных вычислений		✓	✓			Численные результаты, моделирование	Время выполнения не учитывается.

[271]	Свести к минимуму время и потребление энергии	Эффективный алгоритм, основанный на глубоком обучении с мета-подкреплением		✓	✓		✓	Моделирование	Ограничение по времени выполнения вычислительных задач не учитывается
[272]	Оптимизировать качество работы	Качественно новая модель с учетом опыта		✓	✓			Моделирование	Взаимодействие с ресурсами не рассматривается.
[273]	Максимальная энергоэффективность при минимальных ограничениях на качество обслуживания	Предложена интегрированная модель выгрузки задач и распределения ресурсов		✓	✓		✓	Моделирование	Защита личных данных и взаимодействие с ресурсами не рассматриваются.
[269]	Свести к минимуму потребление энергии и задержку выполнения	Модель частичной выгрузки задач	✓			✓		Моделирование	Ограничение по времени выполнения вычислительных задач не учитывается, учитывать в модели только одного пользователя Интернета вещей нереально.

[270]	Сокращение потребления энергии	Многопользовательская модель с многоуровневой выгрузкой задач	✓			✓		Моделирование	Учитывать в модели только одного пользователя интернета вещей нереально.
[274]		Энергоэффективный метод выгрузки задач		✓		✓		Численные результаты, моделирование	Ограничения по времени выполнения вычислительных задач и конфиденциальность данных не учитываются.
[275]	Максимально увеличить полезность системы	Модель выгрузки многосерверных задач		✓		✓		Моделирование	Вопрос о конфиденциальности данных не рассматривается.
[276]	Максимально увеличить полезность системы	Модель выгрузки задач для аварийно-спасательных работ после катастрофы		✓		✓		Моделирование	Вопрос о конфиденциальности данных не рассматривается.

[266]	Свести к минимуму общее потребление энергии	Энергоэффективная архитектура граничного облака для интеллектуальных многофункциональных БПЛА		✓		✓	✓	Моделирование	Взаимодействие с ресурсами не рассматривается.
[271]	Свести к минимуму потребление энергии	Энергоэффективная модель выгрузки совместных задач для граничной вычислительной сети с поддержкой облачных технологий		✓		✓	✓	Моделирование	Вопрос о конфиденциальности данных не рассматривается.
[277]	Свести к минимуму потребление энергии	Технология интеграции "воздух–земля" для МЕС		✓		✓	✓	Численные результаты, моделирование	Ограничение по времени выполнения вычислительных задач и конфиденциальность данных не учитываются.
[278]		Модель выгрузки кода, ориентированная на граничные ресурсы		✓		✓	✓	Моделирование	Вопрос о конфиденциальности данных не рассматривается.

[279]	Свести к минимуму потребление энергии и время выполнения	Новая модель взвешенных затрат для среды с несколькими беспилотниками		✓		✓	✓	Моделирование	Ограничения по времени выполнения задачи не учитывается.
[230]	Свести к минимуму задержку обработки задач	Интеллектуальная модель выгрузки задач		✓		✓	✓	Численные результаты, моделирование	Время выполнения не измеряется.
Предложенная модель	Минимизировать потребление энергии с учетом задержек	Эффективная модель распределения ресурсов и выгрузки задач		✓		✓	✓	Моделирование	Мобильность и безопасность не рассматриваются.

3.3.2 Модель сети

Как показано на рисунке 3.3.2.1, предлагается многопользовательская модель и метод построения сети MEC с поддержкой нескольких беспилотных летательных аппаратов. В этой сети рассматривается набор небольших беспилотных летательных аппаратов, которые оснащены серверами граничных вычислений, подключенных к централизованным облачным вычислительным ресурсам через магистральный маршрутизатор. Пусть N - набор мобильных устройств, которые связаны с беспилотными летательными аппаратами по беспроводному каналу. Кроме того, каждое мобильное устройство задействовано в высокоинтенсивных вычислительных задачах, которые необходимо обрабатывать локально или передавать и обрабатывать ближайшим беспилотным летательным аппаратам или облачным серверам по беспроводному каналу. При моделировании используется квазистатический сценарий, при котором беспилотный летательный аппарат не перемещается в течение периода выгрузки вычислений, хотя он может быть перемещен в другое место в разные иные периоды времени [274-276].

Рисунок 3.3.2.1 - Модель сети

В следующих подразделах сначала более подробно обсуждаются коммуникационные и вычислительные модели. Затем представлена формулировка оптимизационной задачи.

В таблице 3.3.2.1 обобщены основные обозначения для этого исследования.

Таблица3.3.2.1 – Обозначения

Обозначение	**Описание**
M	Набор небольших БПЛА
N	Набор пользовательских устройств
i	i-й БПЛА
j	Пользователь j-го устройства
$b_{i,j}$	Объем данных для задачи мобильного устройства j, которое подключено к беспилотному летательному аппарату i

$c_{i,j}$	Циклы процессора для выполнения задачи мобильного устройства j, которое подключено к БПЛА i
$\tau_{i,j}$	Требование по ограничению времени выполнения задачи мобильного устройства j, которое подключено к БПЛА i
$\phi_{i,j,a}$	Решение о выгрузке трафика мобильного устройства
$r_{i,j}$	Скорость передачи данных по восходящей линии связи для мобильного устройства j, которое подключено к БПЛА i.
B_i	Пропускная способность системы БПЛА i
$p_{i,j}$	Мощность передачи мобильного устройства j, которое соединено с БПЛА i.
G_i	Коэффициент усиления канала БПЛА i
ω_i	Плотность мощности шума канала, который связан с БПЛА i
$\zeta_{i,j}$	Энергия, потребляемая за цикл процессора мобильного устройства j, подключенного к БПЛА i
δ	Задержка распространения между БПЛА и облаком
$f^{l}_{i,j}$	Вычислительные возможности мобильного устройства j, которое подключено к БПЛА i.
$f^{e}_{i,j}$	Ресурс процессора БПЛА i, назначенный мобильному устройству j
$f^{c}_{i,j}$	Ресурс процессора облака, назначенный мобильному устройству j, которое подключено к БПЛА i

$T^{l}_{i,j}$	Время локального выполнения задачи мобильного устройства j, которое подключено к БПЛА i.
$E^{l}_{i,j}$	Энергия для локального выполнения задачи мобильного устройства j, связанного с БПЛА i
$T^{off}_{i,j}$	Время передачи задания мобильного устройства j БПЛА i.
$T^{e\_exec}_{i,j}$	Время выполнения задачи мобильного устройства j на БПЛА i
$T^{c\_exec}_{i,j}$	Время выполнения задачи мобильного устройства j на облачном сервере

Модель взаимодействия

Сначала представим модель взаимодействия для граничных облачных систем, где в среде имеется несколько небольших беспилотных летательных аппаратов, подключенных к централизованным облачным серверам через магистральный маршрутизатор. Кроме того, каждый беспилотный летательный аппарат подключен к ряду N мобильных устройств, которые выполняют высокоинтенсивные вычислительные задачи. Обозначим набор небольших беспилотных летательных аппаратов и мобильных устройств как M = {1, 2, ..., M} и N = {1, 2, ..., N} соответственно.

Далее обозначим $\phi_{i,j,a}$ как выгрузку решения, где задача мобильного устройства j, которое подключено к беспилотному летательному аппарату i, назначается для исполнения на сервере, и может быть локальным (т.е. a = 0), облачным (т.е. a = M + 1), или в исполнении БПЛА (т.е., $a \in [1 \ldots M]$). В частности, ($\phi_{i,j,a}$ = 1, где a ∈ [1..M]) обозначает, что вычислительная задача устройства j,

которое подключено к беспилотному летательному аппарату i, будет передаваться и выполняться удаленно на одном из доступных беспилотных летательных аппаратов, в то время как ($\phi_{i,j,0} = 1$) и ($\phi_{i,j,M+1} = 1$) обозначают, что задача мобильного устройства j будет обрабатываться локально сама по себе или на облачном сервере, соответственно. В целом, каждое мобильное устройство j должно выполнить свою вычислительную задачу только один раз, будь то локально (a = 0) или удаленно ($a \in \{1, 2, \ldots, M+1\}$), т.е $\Sigma_{a=0}^{M+1} \phi_{i,j,a} = 1$

Следуя закону Шеннона, максимальная скорость передачи данных по восходящей линии связи между каждым мобильным устройством j и подключенным беспилотным летательным аппаратом для передачи данных своей задачи по каналу связи моет быть определена как [277]:

$$r_{i,j} = B_i \log_2 \left(1 + \frac{p_{i,j} g_i^2}{\omega_i B_i}\right)$$

(1)

где B_i обозначает полосу пропускания канала между беспилотным летательным аппаратом i и подключенным к нему мобильным устройством $p_{i,j}$, а w_i и g_i обозначают мощность передачи мобильного устройства j, плотность мощности и коэффициент усиления канала между мобильным устройством и беспилотным летательным аппаратом. В соответствии с [278] технология TDMA используется для уменьшения межканальных помех между пользователями устройств.

В этом подразделе общими накладными расходами с точки зрения энергопотребления и времени на возврат объема выходных данных пренебрегается из-за того, что их размер меньше размера входных данных. Скорость скачивания данных с сервера больше, чем скорость загрузки на сервер [40-42].

Вычислительная модель

В этом подразделе будет представлена модель выгрузки из небольших беспилотных летательных аппаратов, подключенных к централизованным облачным вычислениям через магистральный маршрутизатор, в которой беспилотный летательный аппарат имеет номер M. Кроме того, каждый беспилотный летательный аппарат подключен к ряду мобильных устройств N, которые выполняют высокоинтенсивные вычислительные задачи, которые должны быть выполнены. Для каждого мобильного устройства j кортеж $\{b_{i,j}, c_{i,j}, t_{i,j}\}$ представляет требования задачи, где $b_{i,j}$ указывает размер набора данных, который необходимо передать с мобильного устройства j на беспилотный летательный аппарат i, тогда как $c_{i,j}$ и $t_{i,j}$ обозначают количество циклов процессора и требуемые ограничения по времени выполнения задачи мобильного устройства j. Значения $b_{i,j}$ и $c_{i,j}$ могут быть получены с помощью профилирования выполнения задачи [40,43].

Локальное исполнение

Для метода локального выполнения вычислительная задача мобильного устройства j будет выполняться локально, и общее время выполнения и потребление энергии могут быть выражены соответственно как:

$$T_{i,j}^{l} = \frac{c_{i,j}}{f_{i,j}^{l}} \qquad (2)$$

$$E_{i,j}^{l} = \zeta_{i,j} c_{i,j} \qquad (3)$$

где $f_{i,j}^{l}$ определяет вычислительную мощность (процессор/циклы в секунду) мобильного устройства, а $z_{i,j}$ - коэффициент, который определяет

энергию, потребляемую за цикл процессора мобильным устройством *j*, которое подключено к беспилотному летательному аппарату *i*. Пусть $z_{i,j}=10^{-11}(f^{l}_{i,j})^2$, а потребление энергии является линейной функцией частоты мобильного устройства [279,280].

Удаленное выполнение

Для метода удаленного выполнения вычислительная задача мобильного устройства *j*, которое подключено к беспилотному летательному аппарату *i*, будет передана и выполнена на одном из беспилотных летательных аппаратов или облачном сервере, и общее время выполнения может быть выражено как:

$$T_{i,j}^{e} = T_{i,j}^{off} + T_{i,j}^{e\_exec}$$

$$T_{i,j}^{c} = T_{i,j}^{off} + \delta + T_{i,j}^{c\_exec} \qquad (4), (5)$$

где δ - константа, относящаяся к задержке распространения для передачи задачи между беспилотным летательным аппаратом и облаком. $T^{off}_{i,j}$, $T^{e\_exec}_{i,j}$ и $T^{c\_exec}_{i,j}$ определяют время передачи и обработки для выполнения задачи мобильного устройства *j* на беспилотном летательном аппарате и облачном сервере соответственно, которое может быть рассчитано следующим образом:

$$T_{i,j}^{off} = \frac{b_{i,j}}{r_{i,j}}$$

$$T_{i,j}^{e\_exec} = \frac{c_{i,j}}{f_{i,j}^{e}}$$

$$T_{i,j}^{c\_exec} = \frac{c_{i,j}}{f_{i,j}^{c}}$$

(6), (7), (8)

где $f^{e}_{i,j}$ и $f^{c}_{i,j}$ обозначают вычислительные возможности для беспилотного летательного аппарата и облачного сервера, выделенные мобильному устройству j. Как следствие, потребление энергии для передачи и обработки задачи мобильного устройства j удаленно на беспилотном летательном аппарате и облачном сервере может быть рассчитано следующим образом:

$$E^{r}_{i,j} = p_{i,j} T^{off}_{i,j} \qquad (9)$$

С учетом приведенных выше моделей затраты энергии и времени на обработку вычислительной задачи мобильного устройства j могут быть выражены соответственно как:

$$E_{i,j} = \left[\varphi_{i,j,0} E^{l}_{i,j} + \sum_{a=1}^{M+1} \varphi_{i,j,a} E^{r}_{i,j} \right]$$

$$T_{i,j} = \left[\varphi_{i,j,0} T^{l}_{i,j} + \varphi_{i,j,M} + 1^{T^{c}_{i}} + \sum_{a=1}^{M} \varphi_{i,j,a} T^{e}_{i,j} \right]$$

(10), (11)

Формулировка задачи оптимизации

В этом подразделе будет представлена формулировка задачи оптимизации, для которой рассматривается выгрузка вычислений и распределение ресурсов для многопользовательской мобильной системы пограничных облачных вычислений с поддержкой нескольких беспилотных летательных аппаратов. Основная проблема заключается в снижении потребления энергии системой с ограничением задержки. Эта формулировка выглядит следующим образом:

$$\min_{\varphi} \quad \left[\sum_{i=1}^{M} \sum_{j=1}^{N} E_{i,j} \right]$$

Пусть $\left[E_{i,j} - E_{i,j}^{l}\right] \leq 0, \quad \forall_{i,j} \quad C1$

$T_{i,j} \leq \tau_{i,j}, \quad \forall_{i,j} \quad C2$

$\sum_{j=1}^{N} \sum_{a=1}^{M+1} \varphi_{i,j,a} r_{i,j} \leq B_i, \quad \forall_i \in [1..M] \quad C3$

$\sum_{j=1}^{N} \sum_{a=1}^{M} \varphi_{i,j,a} f_{i,j}^{e} \leq F_i, \forall_i \in [1..M] \quad C4$

$\sum_{a=0}^{M+1} \varphi_{i,j,a} = 1, \quad \forall_i \quad C5$

$\varphi_{i,j,a} \in \{0,1\}, \quad \forall_{i,j} \quad C6$

(12)

Ограничение C1 – это верхняя граница потребления энергии. Ограничение C2 определяет ограничение выполнения по времени требований для каждой задачи. Ограничение C3 управляет пропускной способностью полосы пропускания. Ограничение C4 представляет верхний предел пропускной способности граничного сервера. Ограничение C5 гарантирует, что каждая задача должна быть обработана только один раз. Наконец, ограничение C6 гарантирует, что переменная выгрузки решения является двоичной.

Решение о выгрузке может быть получено путем решения этой задачи. Однако эта задача классифицируется как NP-трудная, в которой переменные решения о выгрузке являются двоичными [281]. Следовательно, чтобы эффективно решить эту проблему, используется подход бинарного ослабления. В частности, двоичные переменные преобразуются до новых вещественных переменных, таких как $0 \geq \phi_{i,j,a} \leq 1$ [282]. Таким образом, формулировка задачи после преобразования двоичных переменных выглядит следующим образом (13):

$$\min_{\varphi} \quad \left[\sum_{i=1}^{M}\sum_{j=1}^{N} E_{i,j}\right]$$

$$\text{s.t} \quad \left[E_{i,j} - E_{i,j}^{l}\right] \le 0, \qquad \forall_{i,j} \qquad C1$$

$$T_{i,j} \le \tau_{i,j}, \qquad \forall_{i,j} \qquad C2$$

$$\sum_{j=1}^{N}\sum_{a=1}^{M+1} \varphi_{i,j,a} r_{i,j} \le B_i, \qquad \forall_i \in [1..M] \quad C3$$

$$\sum_{j=1}^{N}\sum_{a=1}^{M} \varphi_{i,j,a} f_{i,j}^{e} \le F_i, \forall_i \in [1..M] \quad C4$$

$$\sum_{a=0}^{M+1} \varphi_{i,j,a} = 1, \qquad \forall_i \qquad C5$$

$$\varphi_{i,j,a} \in [0,1], \qquad \forall_{i,j} \qquad C6$$

(13)

Поскольку задача рассматривается как линейная задача, где целевая функция и ограничения линейны, почти оптимальное решение может быть получено с использованием хорошо изученных линейных подходов.

Многоуровневый энергоэффективный алгоритм выгрузки задач

В этом разделе представлен многоуровневый энергоэффективный алгоритм выгрузки задач, в котором вывод достаточно близкого к оптимальному решения изложен следующим образом (алгоритм 1).

Алгоритм 1 Многоуровневый энергоэффективный алгоритм выгрузки вычислений.

1: **Initialization**: Each device user *j* initializes the offloading decision of its' task

with *j*i,j,0 = 1, 8i, j

2: **for each** UAV *i* and at given time slot *t* **do**

3: **Upload** all the available computational capabilities and bandwidth to the backbone router.

4: **for each** device user *j* **do**

5: **Upload** the required information for their task, $\{b_{i,j}, ci, j, \tau_{i,j}, \zeta_{i,j}, p_{i,j}\}$, as well as their local capabilities $f^{l}_{i,j}$ to the backbone router.

6: **end for**

7: **end for**

8: **Compute** the available data rate *ri,j* for each device user regrading Equation (1).

9: **Solve** the formulated problem in Equation (12) and derive the offloading decision values $\phi_{i,j,a}$ for each task.

10: **Send** the offloading decision values *a*i,j,k to each device user.

Во-первых, решение о выгрузке для всех вычислительных задач мобильных устройств инициализируется с помощью $\phi_{i,j,a} = 1$, что указывает на локальное выполнение. Затем магистральный маршрутизатор выполняет итерацию по беспилотным летательным аппаратам и собирает их доступные возможности, включая память и вычислительную мощность, а также доступную полосу пропускания. Далее он получает необходимую информацию для выполнения задачи пользователя, включая $\{b_{i,j}, ci, j, \tau_{i,j}, \zeta_{i,j}, p_{i,j}\}$, а также локальные возможности для каждого устройства $f^{l}_{i,j}$, через подключенный беспилотник. Далее, максимально допустимая скорость передачи данных для каждого мобильного устройства рассчитывается в соответствии с уравнением (1). Затем значения решения для выгрузки для каждой вычислительной задачи вычисляются и выводятся путем решения задачи, сформулированной в

уравнении (12), а после этого вычисляются указанные значения для каждого пользователя устройства. При этом основной целью является снижение потребления энергии для системы с ограничением задержки.

Алгоритм 1 описывает комплексные этапы принятия решения о выгрузке задачи, для которой вычислительная сложность равна O(MN), где M обозначает количество беспилотных летательных аппаратов, а N обозначает общее количество мобильных устройств.

3.3.3 Результаты моделирования и оценка их эффективности

В этом подразделе главы представлены параметры и результаты моделирования для оценки эффективности разработанных модели и метола, а также алгоритм управления трафиком.

Моделирование проводилось с использованием персонального компьютера, оснащенного процессором Intel® Core(TM) i7-10750H с частотой 2,6 ГГц и объемом оперативной памяти 16 ГБ, работающий на 64-разрядной платформе Windows 10 Home с предустановленной средой Python для разработки. Рассматривается многопользовательская среда с пятью БПЛА на площади 100 x 100 $м^2$, взаимодействующими и связанными с одним облачным сервером через основную сеть, которая также оснащена сервером граничных вычислений. Высота для беспилотника установлена 20 м. Кроме того, есть 30 мобильных устройств, распределенных среди беспилотных летательных аппаратов. У каждого пользовательского устройства имеется сложная ресурсоемкая задача, которую необходимо выполнить. Размер данных распределен равномерно в диапазоне [0, 10] МБ, в то время как циклы процессора, необходимые для обработки каждого бита данных, установлены равными 1000 циклам/бит. Вычислительные возможности процессора облачного сервера определены на частоте 20 ГГц, а для серверов беспилотных летательных

аппаратов распределены равномерно в пределах диапазона [3.5, 5] ГГц, тогда как вычислительная мощность процессора для каждого устройства распределена равномерно в диапазоне {0.5, 0.6, …, 1.0}. Потребление энергии для каждого мобильного устройства равномерно распределено в диапазоне ($0{,}20 \times 10^{-11}$) Дж/цикл [283]. Остальные параметры приведены в таблице 3.3.3.1. Задача оптимизации решается с помощью пакета GEKKO на базе Python [284].

Таблица 3.3.3.1 - Параметры моделирования

Параметр	**Значение**
Количество БПЛА	5
Количество мобильных устройств	30
Высота полета БПЛА	20 м
Пропускная способность системы	20 МГц
Мощность передачи	0,1 Ватт
Размер входных данных	[0, 10] MB Unif-Dist
Циклы процессора для выполнения задачи	1000 циклов/бит
Вычислительные возможности мобильного устройства	{0.5, 0.6,…, 1.0}
Потребление энергии	$(0, 20 \times 10^{-11})$Дж/цикл
Возможности сервера БПЛА	[3,5, 5] Однодиапазонный ГГц
Возможности облачного сервера	20 ГГц
Задержка распространения для облаков	15 мс

Результаты моделирования

В этом подразделе оцениваются результаты для разработанной модели и метода в сравнении с четырьмя сценариями:

- **Локальное выполнение:** выгрузки не существует. Вычислительные задачи всех мобильных устройств будут обрабатываться локально $\phi_{i,j,a}$ = 1;

- **Выполнение сервером беспилотных летательных аппаратов:** Вычислительные задачи всех мобильных устройств будут обрабатываться удаленно на серверах $\Sigma_{a=1}^{M}\ \phi_{i,j,a}$ = 1

- **Облачное выполнение:** Вычислительные задачи всех мобильных устройств будут обрабатываться удаленно на облачном сервере $\phi_{i,j,\mathrm{M+1}}$ = 1;

- **Выполнение выгрузки вычислений в [275]:** Вычислительные задачи всех мобильных устройств будут обрабатываться на одном из доступных серверов, включая локальное выполнение на основе модели, предложенной в [275].

Потребление энергии на выполнение задач для упомянутых сценариев применительно к разному числу пользователей показано на рисунке 3.3.3.1. Как видно, предложенные модель и метод обеспечивают наилучший результат по сравнению с другими сценариями. Кроме того, сценарии граничного и облачного выполнения превосходят сценарий локального выполнения по мере увеличения числа пользователей. Более того, политика граничного выполнения также превосходит облачную с более чем 140 пользователями устройств. Этот результат обусловлен увеличением времени взаимодействия, когда каналы связи являются общими для пользователей, а также более высоким потреблением энергии в процессе выгрузки. Кроме того, серверных ресурсов беспилотника недостаточно для обслуживания всех пользователей устройств, где облачный сервер более ресурсоемкий, тем самым улучшая производительность беспилотника для большого числа пользователей. Аналогично, на рисунке 3.3.3.2 показано время отклика на выполнение вычислительных задач для

разного числа пользователей. Этот рисунок показывает, что время отклика для предложенных модели и метода близко или равно времени отклика для выполнения задач на беспилотных летательных аппаратах, облаках и модели из [275], хотя и меньше, чем при локальном выполнении. Однако по мере роста числа мобильных пользователей производительность беспилотных летательных аппаратов и облачных политик выполнения улучшается по сравнению с локальным выполнением, в то время как разработанные модель и метод по-прежнему превосходят локальную политику и превосходит модель, представленную в [275]. Кроме того, политика облачного исполнения превосходит серверное выполнение беспилотника. Это связано с тем, что каналы связи, совместно используемые пользователями, перегружены, что увеличивает время, в то время как предлагаемые модель и метод могут быть разумно адаптированы к изменениям окружающей среды.

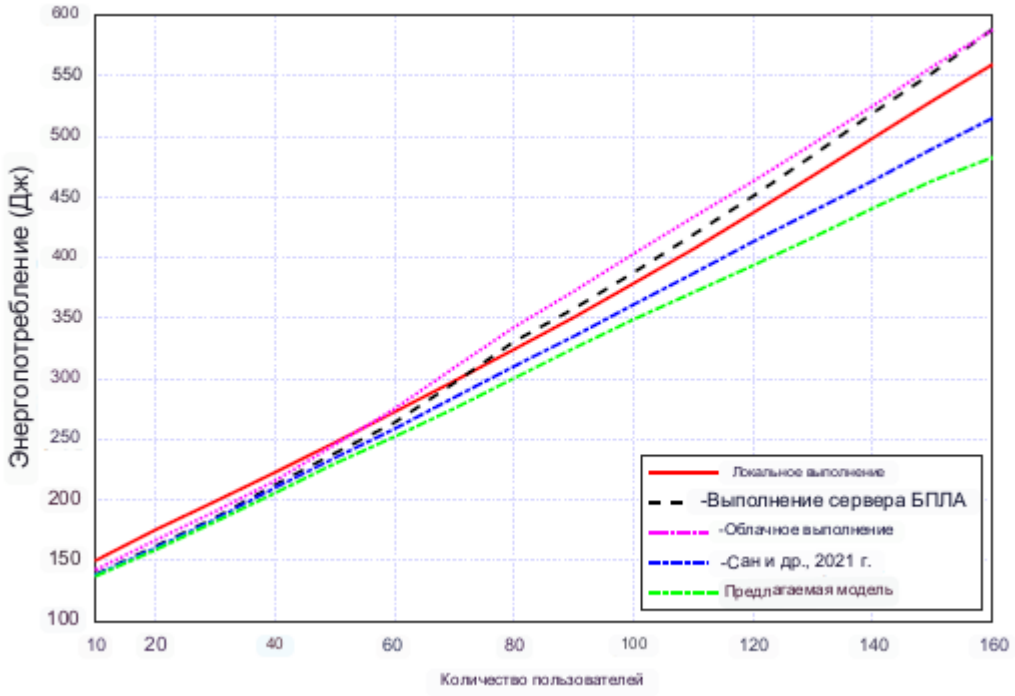


Рисунок 3.3.3.1 - Энергопотребление в зависимости от количества пользователей

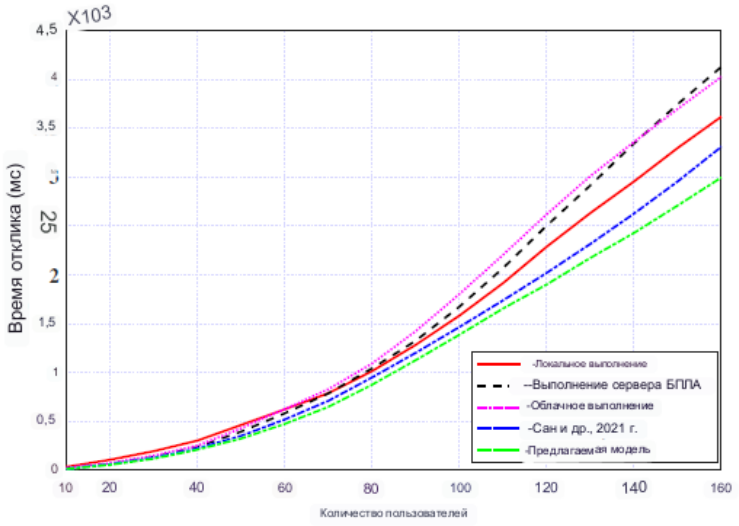


Рисунок 3.3.3.2 - Время отклика в зависимости от количества пользователей

Потребление энергии для выполнения задач для упомянутых сценариев в зависимости от различных размеров данных показано на рисунке 3.3.3.3. Из этого графика можно сделать вывод, что энергопотребление предложенных модели и метода при размерах данных менее 20 МБ близко к энергопотреблению беспилотных летательных аппаратов, облака и модели в сценариях [275], и меньше энергопотребления сценария локального выполнения. Тем не менее, с увеличением размера данных производительность сервера беспилотных летательных аппаратов и облачных сценариев снижается по сравнению со сценарием локального исполнения, тогда как производительность предложенных модели и метода остается ниже, чем у локальных политик и

политики модели в [275]. Более того, политика облачного выполнения превосходит политику выполнения беспилотных летательных аппаратов при больших размерах данных (т.е. более 45 МБ). Время отклика для обработки вычислительных задач в отношении различных объемов данных показано на рисунке 3.3.3.4. Этот рисунок показывает, что время постепенно увеличивается по мере увеличения размера данных. Также, существует значительный разрыв между разработанными моделью и методом и другими политиками, который увеличивается по мере увеличения объема данных. Заметим, что политика выполнения на сервере беспилотника почти соответствует политике выполнения в облаке при размере данных 45 МБ, а затем превысит это значение для больших объемов данных.

Этот вариант объясняется следующим образом. Увеличение объема данных приводит к увеличению времени передачи данных, что затем приводит к увеличению потребления энергии. Тем не менее, разработанные модель и метод могут быть адаптированы для обработки выполнения задачи в различных местах с помощью беспилотных летательных аппаратов, в облаке или локально, тем самым оптимизируя энергию и время.

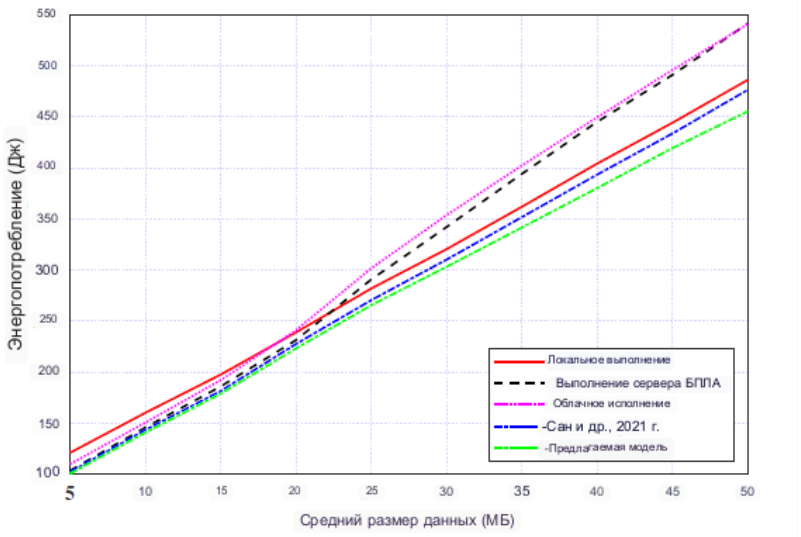


Рисунок 3.3.3.3 - Энергозатраты по отношению к различным значениям объема входных данных

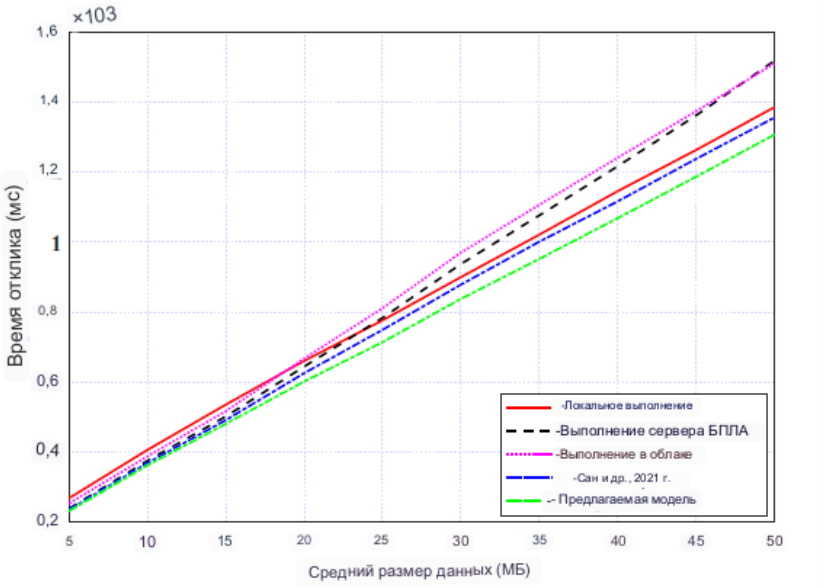


Рисунок 3.3.3.4 - Время отклика в зависимости от значений объема входных данных

Потребление энергии и время отклика на обработку задач применительно к различному количеству беспилотных летательных аппаратов представлены на рисунках 3.3.3.5 и 3.3.3.6 соответственно. Из этих двух рисунков видно, что количество БПЛА не влияет на сценарии локального и облачного выполнения, в то время как потребление энергии и время отклика трех других сценариев (т.е. сервера БПЛА, модели представленной в работе [275] и предлагаемой модели) постепенно уменьшаются по мере увеличения количества БПЛА. Кроме того, разработанные модель и метод позволяют добиться более низкого энергопотребления и периода ответа по сравнению с двумя другими сценариями. Это связано с тем, что затраты с точки зрения энергопотребления и времени отклика снижаются по мере того, как мобильным пользователям выделяется больше ресурсов (т.е. за счет увеличения количества беспилотных летательных

аппаратов), в то время как сценарии локального выполнения и облачного выполнения не зависят от ресурсов беспилотных летательных аппаратов.

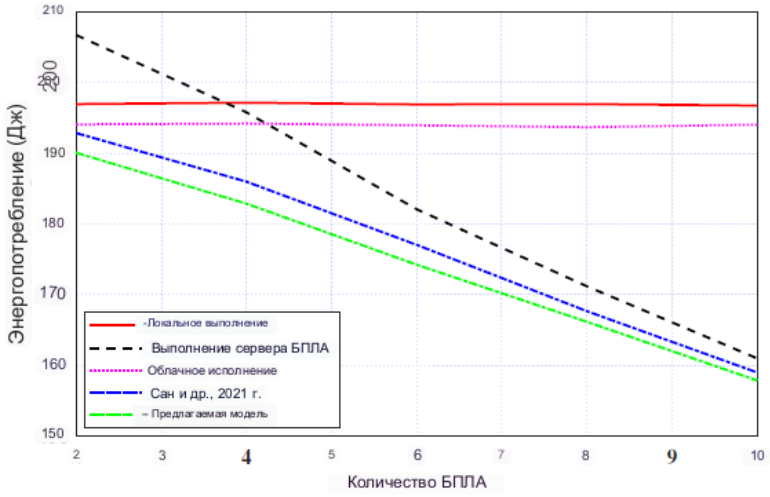


Рисунок 3.3.3.5 - Потребление энергии в зависимости от количества БПЛА

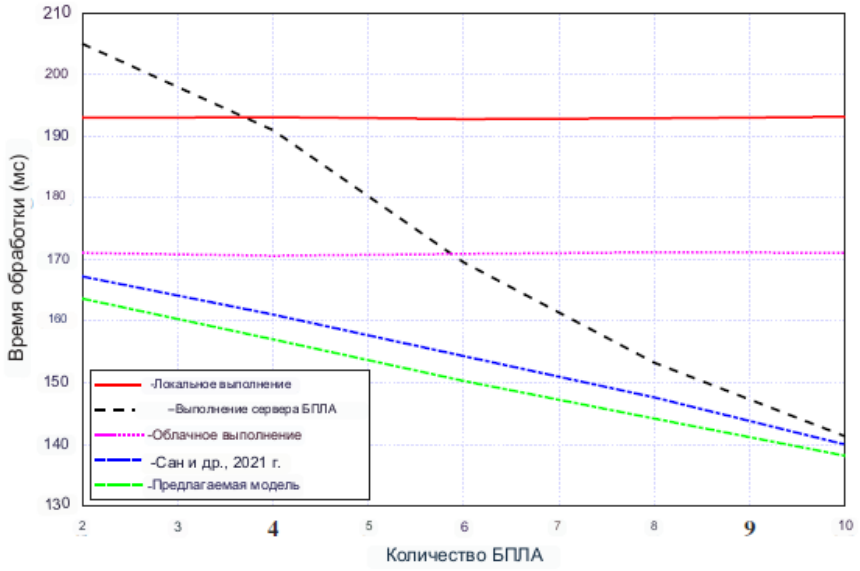


Рисунок 3.3.3.6 - Время отклика в зависимости от количества БПЛА

3.4 Выводы по главе 3

В диссертационной работе были предложены модель и метод энергоэффективной многопользовательской системы, основанной на использовании технологии граничных вычислений мультисервисного доступа с поддержкой беспилотных летательных аппаратов. Эта сеть масштабируема и может поддерживать увеличение сетевого трафика без снижения производительности. При этом сеть беспилотных летательных аппаратов может быть использована для покрытия мертвых зон и зон с высокой плотностью сети. Кроме этого, беспилотные летательные аппараты могут предоставлять другие приложения в сети "умного города", в которой технология MEC развернута на нескольких уровнях для обеспечения вычислительных возможностей на границе сети. Базовая сеть основана на технологии SDN для возможности гибкого управления сетевым трафиком и предоставления инновационного интерфейса сетевым операторам. Наконец, имитационные эксперименты демонстрируют, что предложенные в диссертационной работе модель и метод могут значительно снизить энергопотребление всей системы.

1. Решена научная проблема, отличающаяся от известных тем, что предложены модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности, основанные на интегральном решении задач по размещению граничных серверов на беспилотных летательных аппаратах и оптимизации структуры сети с использованием метаэвристического алгоритма роя сальп.
2. Для решения научной задачи интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности разработана модель сети, отличающаяся от

известных тем, что предложено для уменьшения задержки и энергопотребления в такой сети использовать мобильные серверы граничных вычислений на БПЛА, а связь между наземным и летающим сегментом обеспечивать с использованием NOMA.

3. Для решения задачи минимизации задержки и энергопотребления в разработанной модели предложен метод выгрузки трафика с наземной сети на мобильные серверы граничных вычислений на БПЛА, отличающийся от известных тем, что процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, который определяет сложность вычисляемой задач и по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА.

4. Для решения научной задачи оптимизации структуры сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности в отличии от известных решений для минимизации задержки и энергопотребления при выгрузке трафика с наземной сети на серверы граничных вычислений БПЛА разработан метаэвристический алгоритм на основе хаотического роя сальп.

5. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений и на 30-40% по

сравнению с сетью с использованием только наземных граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш около 10% по сравнению с использованием неоптимизированного алгоритма.

6. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение энергопотребления до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш в 5-10% по сравнению с использованием неоптимизированного алгоритма.
7. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение доли заблокированных задач по выгрузке трафика в десятки раз по сравнению с сетью без использования технологий граничных вычислений, в разы по сравнению с сетью с использованием только наземных граничных вычислений. Использование оптимизации на основе метаэвристического хаотического роя сальп не дает практически значимого эффекта по сравнению с неоптимизированным алгоритмом.
8. Определены зависимости значений задержки, энергопотребления и доли заблокированных задач по выгрузке трафика от плотности устройств сети.

ГЛАВА 4. ИНТЕЛЛЕКТУАЛЬНАЯ РАСПРЕДЕЛЕННАЯ АРХИТЕКТУРА СЕТИ СВЯЗИ ДЛЯ ПОДДЕРЖКИ БЕСПИЛОТНЫХ АВТОМОБИЛЕЙИ

Последние достижения и развитие автономных систем управления автомобилями делают планирование надежной инфраструктуры сети связи еще более востребованными. Сети связи пятого поколения (5G) обеспечивают скачок в развитии, однако к планированию автономных транспортных сетей предъявляются дополнительные требования. Это связано с высокой мобильностью и плотностью трафика таких сетей, а также требованиями к задержке и надежности приложений, работающих в таких сетях. В главе предложена архитектура сети для поддержки беспилотных автомобилей. Разработанная архитектура использует технологии мобильных граничных вычислений (MEC) и программно-конфигурируемых сетей (SDN) для повышения общей надежности сети и ее масштабируемости в условиях высокой плотности трафика. Рассматривается также метод кластеризации для устройств D2D, чтобы обеспечить покрытие сети для централизованно недоступных узлов. Данная архитектура предлагает надежную структуру для поддержки приложений беспилотного транспорта с ультра малой задержкой. Результаты оценки с использованием реалистичных условий для различных сетевых сценариев показывают, что предложенная архитектура с поддержкой MEC/SDN обеспечивает прирост производительности сети на 74% с точки зрения блокировки сетевых задач по сравнению с базовой архитектурой.

Роль телекоммуникационных сетей в эволюции общества в целом продолжает увеличиваться. Телекоммуникационные сети пятого поколения, которые в настоящее время создаются, и сети последующих поколений [296]

будут играть решающую роль в создании эффективной цифровой экономики. Однако требования к новым сетям очень сложны и нетривиальны. Традиционные существующие сети не могут удовлетворить эти требования, и для сетей связи пятого и последующих поколений необходимы новые подходы для их планирования. В настоящее время основным направлением развития сетей связи является сеть 5G [297],[298]. Первоначально сети 5G основывались на концепции Интернета Вещей (IoT), для которой требовалось создавать сети связи сверхвысокой плотности с одним миллионом устройств на один квадратный километр. Эта проблема недавно была осложнена концепцией Тактильного Интернета, которая потребовала ограничения круговой задержки до 1 мс для передачи тактильных ощущений. Все это приводит к пересмотру базовой архитектуры сетей связи и к децентрализации сети. Требования по ультра малым задержкам относятся не только к тактильной информации, но также и к беспилотным автомобилям, приложениям дополненной реальности и т. д. Поэтому, в настоящее время и появилась новая концепция сетей связи с ультра малыми задержками.

Увеличение числа транспортных средств порождает серьезные проблемы, такие, например, как пробки на дорогах и нехватка мест стоянки для автомобилей [299]. Транспортные сети возникли как решение многих современных транспортных проблем. Однако это привело к новым проблемам безопасности и надежности [300]. Эффективная связь — это ключевая особенность автомобильных сетей, которая позволяет транспортным средствам обмениваться информацией для достижения общих целей системы [301]. Поэтому, современные транспортные средства часто имеют различные встроенные технологии связи и локализации, а также системы когнитивного управления, которые поддерживают автономную адаптацию к внешним

условиям [302]. Текущие технологии беспроводной связи, сети широкополосного доступа и другте сети, такие как специализированные автомобильные сети, могут использоваться для обеспечения среды связи и инфраструктуры для транспортных средств, для связи и координации их деятельности [303,304]. В таких сетях следует учитывать условия окружающей среды, поскольку плотность узлов и узлы потока транспортных средств, в основном, зависят от окружающей среды. Новая архитектура сети для поддержки беспилотных автомобилей, основанная на технологиях мобильных граничных вычислений (MEC) и программно-конфигурируемых сетях (SDN), предназначена для решения проблемы высокой плотности трафика и обеспечения масштабируемой и надежной сети транспортных средств. В качестве одного из основных параметров будем использовать плотность движения, которая определяется как количество транспортных средств, занимающих указанную длину дороги в полосе движения. Она измеряется в числе транспортных средствах на один километр. Глава организована следующим образом: краткий обзор проблем планирования сети для беспилотных автомобилей, предлагаемая архитектуры MEC/SDN, кластеризация сети на основе взаимодействия D2D для решения проблем, связанных с покрытием сети, в частности, проблемой пустых зон в кластере, оценка эффективности предложенных решений.

4.1 Состояние исследований в предметной области и постановка задачи.

В последние годы сфера беспилотных автомобилей привлекает всё больше внимания. На разработку умных транспортных средств идет большое количество инвестиций. У общества растет интерес к данному направлению. Разработчики работают над тем, чтобы автомобиль мог видеть и понимать, что происходит вокруг него. Такие автомобили уже ездят по дорогам, получают огромное

количество информации и учатся. Большие исследования в области беспилотных автомобилей проводят такие компании как Waymo, Uber, Tesla, Яндекс и другие.

Автомобили с автономным управлением являются автономными системами принятия решений, которые обрабатывают потоки наблюдений, поступающих от различных бортовых датчиков. Эти наблюдения используются системой автономного вождения автомобиля для принятия решения на дороге [317]. На сегодняшний день производители Waymo, Tesla, Nvidia/Audi достигли третьего уровня автоматизации автономного транспорта [318].

Программное обеспечение автономного автомобиля управляет работой всех систем, приводящих его в движение. Различные датчики и сенсоры собирают информацию об окружающей обстановке, которая ложится в основу действий автомобиля. Датчики и сенсоры устанавливаются на беспилотный автомобиль, в том числе:

- лидары, составляющие часть чувствительных устройств, которые являются устройствами дистанционного зондирования, используемыми для проверки окружения транспортного средства с высокой точностью;
- радары, используемые для обнаружения окружающих автомобиль объектов, и определения их размеров, скорости и дальности нахождения;
- камеры;
- система глобального позиционирования (GPS, ГЛОНАСС);
- датчики одометрии, используемые как средство оценки перемещения при движении приводов.

Программное обеспечение беспилотного автомобиля может включать машинное зрение и глубокие нейронные сети. Для их реализации в сетях связи были предложены несколько архитектур в рамках сети пятого поколения. В статье [314] были рассмотрены особенности функционирования сетей VANET

на основе трехуровневой архитектуры. Авторы пишут о возможностях, которые предоставляет технология граничных вычислений. Рассматривается вариант внедрения SDN и функциональность его сетевых элементов при внедрении в сеть VANET. Авторы подробно описывают основные компоненты автомобилей и необходимые поддерживаемые технологии сетевой инфраструктуры для организации современной интеллектуальной сети. Также в статье большое внимание уделяется решению проблемы мобильности, предлагаются два маршрутных подхода: собственно маршрутный и SDN подход.

В статье [315] была изучена проблема безопасности для автономных транспортных средств внутри автомобильной сети, построенной на архитектуре граничных вычислений. Рассмотрены какие угрозы существуют для различных категорий автомобильных устройств, обеспечивающих автономность: сенсоров, датчиков, систем управления и т.д, а также предложены способы защиты от этих угроз. Статья помогает понять все аспекты построения автономных систем вождения: аспекты безопасности, аспекты проблем, связанных с вычислительной мощностью и энергопотреблением автомобильных устройств, и аспекты взаимодействия автономных транспортных средств.

В статье [316] был проведен математический анализ модели автомобильной сети, обеспечивающий эффективную связь между транспортными средствами V2V. Статья наглядно показывает, какие существуют проблемы для создания сверхнадежных сетей с ультра малыми задержками URLLC. Авторы предлагают использовать для их построения теорию экстремальных значений (EVT) и оптимизацию Ляпунова, как математические методы для моделирования экстремальных длин очередей и распределения ресурсов.

В статье [317] предлагается архитектура автомобильной сети на основе SDN и MEC с оптимальной возможностью распределения радиоресурсов для уменьшения задержек при передаче информации. Авторы за основу сети берут периферийную вычислительную сеть с множественным доступом (SDMEV), в которой реализованы распределенные вычисления на узле eNB. Архитектура представлена в виде четырех логических уровней: уровня передачи, уровня управления, уровня множественного доступа и уровня доступа. В статье рассмотрен пример сценария взаимодействия логических уровней при обнаружении не сопоставленных в Open-Flow таблиц пакетов.

Существует растущий спрос на надежные системы для управления большим автомобильным сетевым трафиком, особенно в густонаселенных городах. Резкое увеличение числа транспортных средств накладывает большие ограничения на планирование сети и, поэтому, необходимы новые технологии, такие как MEC [298] и SDN [309]. Плотность трафика автомобилей приводит к перегрузке трафика данных, в частности, при использовании услуг широкополосного доступа, таких как передача видео с транспортного средства или придорожного блока (RSU) [310]. Еще одна критическая проблема, связанная с автомобильными сетями, заключается в пустотах в зоне покрытия сети, в результате чего некоторые части сети изолированы от сети в какой-то период времени, то есть не имеют зоны покрытия сети. Чтобы решить эту проблему, используются взаимодействия D2D [299]. Использование D2D позволяет выгрузить трафик из проблемной зоны (рис.4.1).

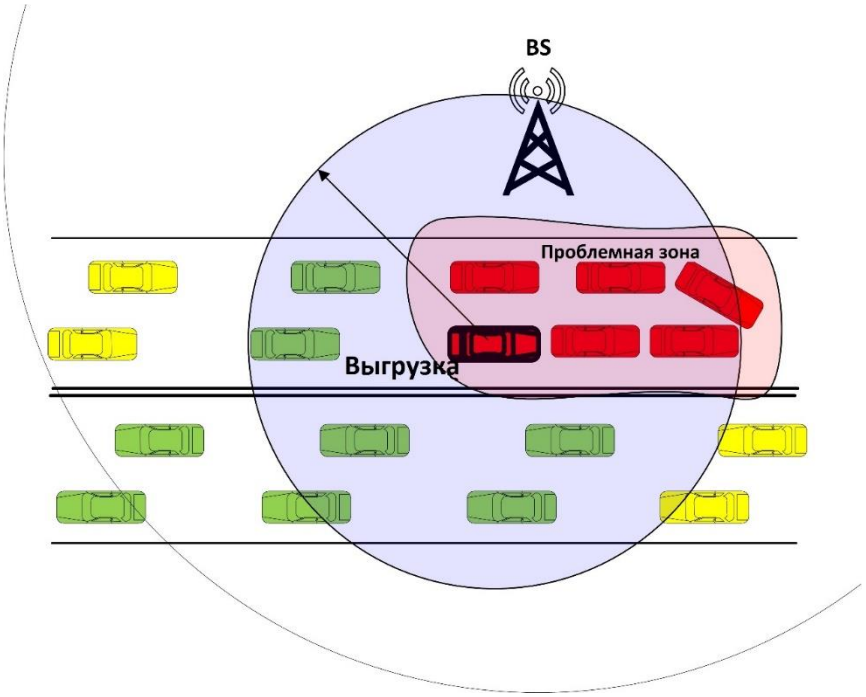


Рисунок 4.1 – Выгрузка трафика

4.2 Метод построения сети MEC/SDN для поддержки приложений беспилотных автомобилей.

Предлагаемый метод построения сети MEC/SDN. Архитектура такой сети изображена на рис.4.2.1. В этой архитектуре используются гетерогенные граничные облачные серверы на границе сети доступа. Каждый стационарный RSU подключается либо к граничному серверу, либо к граничному серверу микро-облака, либо к граничному серверу мини-облака на основе многоуровневой граничной структуры для приложений с ультра малой задержкой, представленной в [311]. Ядро сети построено как мультиконтроллерное. Распределенные коммутаторы OpenFlows с ограниченными вычислительными возможностями развернуты в соответствии с [312]. Развертывание SDN с несколькими контроллерами позволяет сети

обслуживать огромный объем трафика без какого-либо снижения производительности, поскольку контроллеры SDN могут переходить в активные или спящие режимы на основе состояния сети [313]. Кроме того, такая конфигурация системы обеспечивает требуемую масштабируемость сети за счет введения новых контроллеров и граничных серверов в ответ на увеличение сетевого трафика.

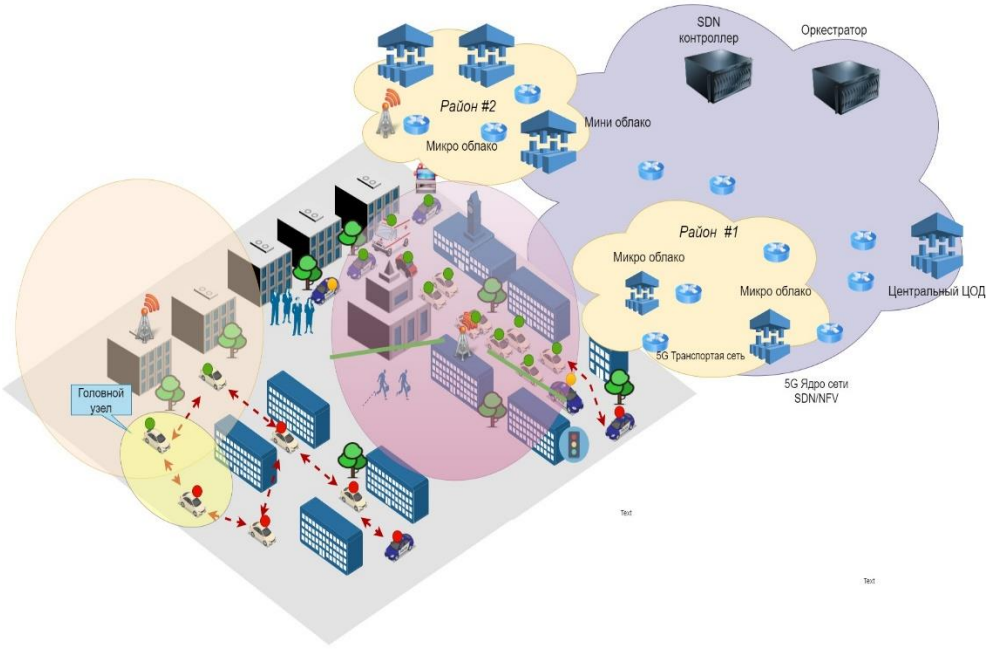


Рисунок 4.2.1 – Разработанная архитектура сети.

Одним из способов обеспечения сетевого подключения для пустот и выгрузки сетевого трафика является развертывание внеполосной связи D2D вместе с эффективной схемой кластеризации. Внедрение кластеризации позволяет добиться более высоких показателей производительности, связанных с задержкой и покрытием [315]. Одним из примеров развертывания эффективной схемы кластеризации является применение алгоритма FOREL, изображенного

на рисунке 4.2.2 Позиции машин обозначены точками, а позиции центров кластеров определены перекрестиями. В [316] была разработана внеполосная схема кластеризации на основе D2D для интерфейса связи WiFi-direct. Эта схема принята в архитектуре MEC/SDN для обеспечения возможности подключения к сети для непокрытых зон и для выгрузки трафика сети. Такая кластеризация позволяет транзитным узлам обрабатывать данные и решать, как передавать трафик. Поскольку выбор транзитных узлов определяет производительность системы, принятый метод кластеризации учитывает возможности узлов и их роль в системе управления транспортом. Выбор сетевого элемента, подходящего для выполнения роли транзитного узла или узла обработки данных, может учитывать другой сетевой критерий. В [317] была определена и решена проблема оптимизации для достижения оптимального выбора транзитного узла при выполнении кластеризации.

Данный алгоритм позволяет получить близкое к оптимальному решение задачи минимизации, которую можно представить в общем виде как

$$\{(x_j, y_j)\} = \arg\max_{x,y}\left\{\sum_{j=1}^{m}\sum_{i=1}^{k_j} q(e_i, C_j)\right\}, \quad j = 1\ldots n \qquad (1)$$

где $q(e_i, C_j)$ - функция, выражающая зависимость качества канала связи между *i*-м узлом сети e_i в *j*-м кластере и центром этого кластера;

C_j; (x_j, y_j) - координаты центра j-го кластера.

Решение дает точки, расположение в которых транзитных узлов или граничных серверов (edge) дает возможность выгрузить трафик большего числа элементов сети.

Вид функции *q*() может быть различным и зависит от применяемой технологии связи канального уровня. Если для выгрузки трафика или

организации граничных вычислений применяется внеполосная кластеризация на основе технологии WiFi – direct, то эта функция может быть найдена в соответствии с [317]. В этом случае она отражает зависимость пропускной способности канала от удаленности элемента сети от центра кластера. В ней также можно учесть влияние интерференции, порождаемой соседними передающими устройствами (сетевыми элементами). Иллюстрация мощности сигналов интерференции приведена на рис.4.2.2, светлые участки соответствуют большей мощности сигналов.

Создание кластеров сетевых элементов и использование внеполосных D2D взаимодействий позволяет существенно разгрузить каналы автомобильной сети и улучшить качество связи между сетевыми элементами.

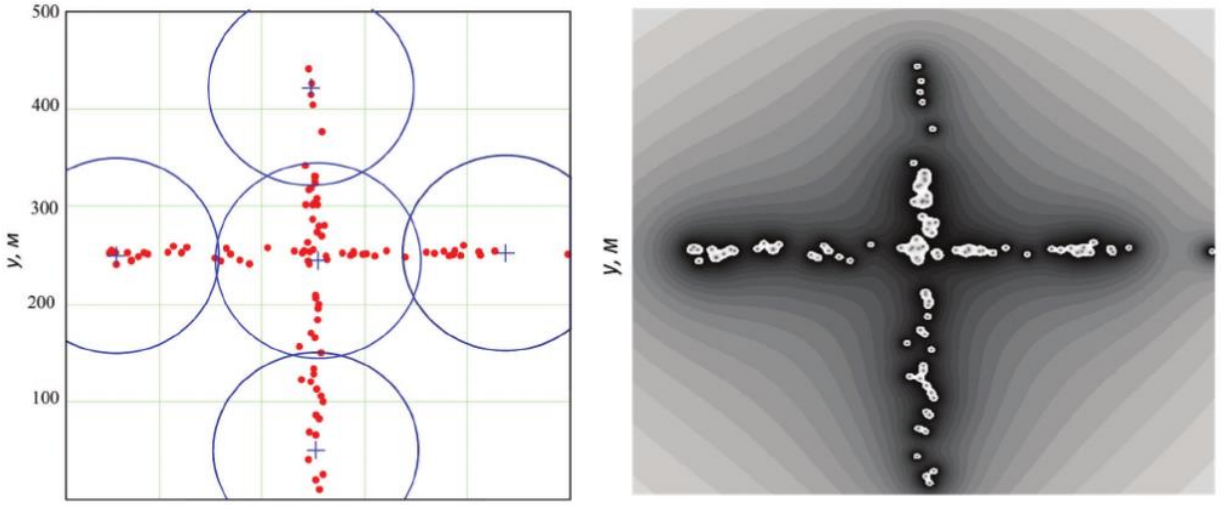


Рисунок 4.2.2 – Метод кластеризации элементов сети с использованием алгоритма FOREL

4.3 Оценка эффективности

Для оценки производительности архитектуры MEC/SDN были промоделированы сетевые архитектуры для реальных условий. Различные

сценарии моделирования были реализованы в Matlab для беспилотных автомобилей с разнородными возможностями. Используемые параметры моделирования приведены в таблице 4.3. Показатель для оценки производительности представляет собой вероятность блокировки задачи, то есть вероятность того, что задача не будет выполнена. Предполагается, что каждое транспортное средство имеет 10 гетерогенных вычислительных задач, которые должны поступить для обработки одновременно. Было смоделировано три сценария, каждый из которых представляет свою конфигурацию системы. Первый сценарий (A) представляет предлагаемую систему с развертыванием многоуровневой MEC, SDN и кластеризацией на основе D2D. Второй сценарий (B) представляет систему беспилотных автомобилей с развертыванием только технологий MEC и SDN. Сценарий (B) рассматривает развертывание серверов MEC с однородными возможностями, но он не моделирует развертывание D2D и кластеризацию. Последний сценарий (C) представляет собой систему без использования технологий MEC, D2D или SDN. На рис.4.3 представлен процент блокировки задач для каждого сценария моделирования. Результаты моделирования показывают, что предлагаемая архитектура MEC/SDN обеспечивает повышение производительности на 74% по сравнению с решением, в котором не используются ни граничные серверы, ни технологии SDN.

Таблица 4.3 – Параметры моделирования

Параметр	Значение
число автомобилей	20
размер поля	1 км x 1км
число микро-облаков	6
число мини-облаков	3

задачи на автомобиль	20

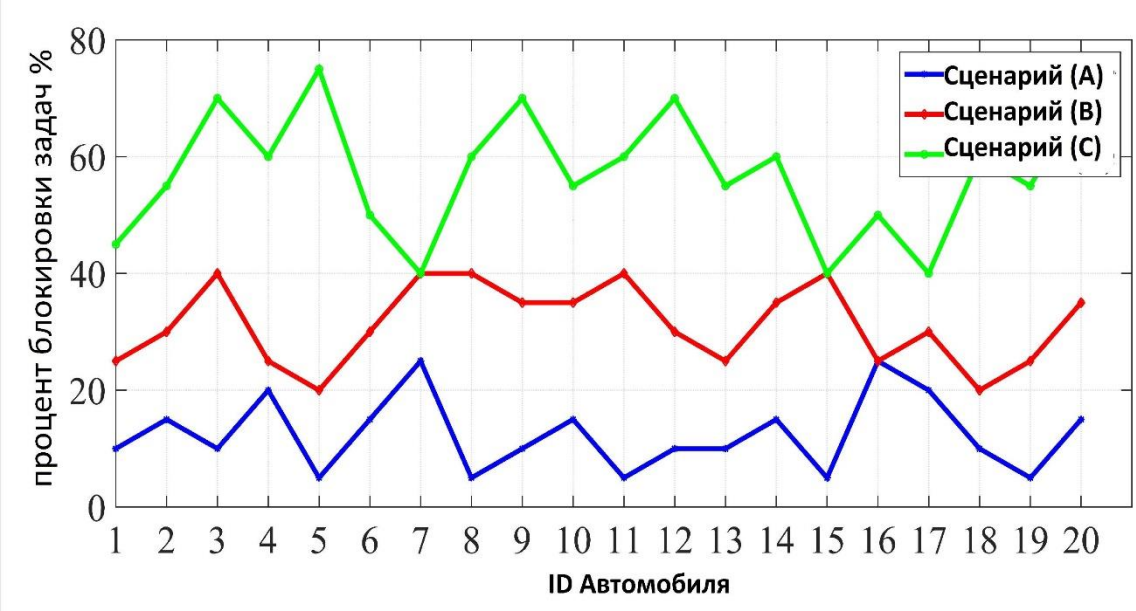


Рисунок 4.3 – Процент заблокированных задач для трех смоделированных сценариев

На рисунке 4.4 представлены полученные результаты по средней задержке для трех смоделированных сценариев с различным количеством подключенных транспортных средств. По мере увеличения количества транспортных средств средняя задержка, связанная с выполнением вычислительных задач, увеличивается во всех трех сценариях. Это обусловлено увеличением нагрузки на вычислительные и сетевые ресурсы, что приводит к росту временных затрат на обработку данных, формированию очереди и пересылке пакетов. Однако в разработанном методе задержка существенно снизилась по сравнению с двумя

другими сценариями благодаря развертыванию сети SDN, которая обеспечивает необходимое управление ресурсами.

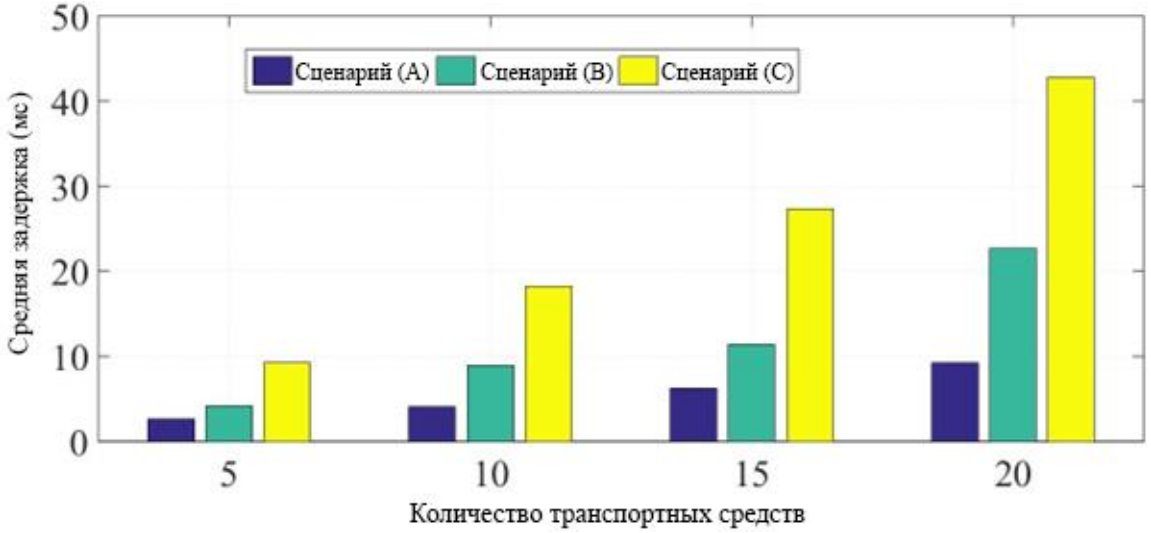


Рисунок 4.4. Средняя задержка для трех смоделированных сценариев с различным количеством подключенных транспортных средств

Кроме того, на рисунке 4.5. представлены полученные результаты по средней требуемой продолжительности обработки задач по трем сценариям с различным количеством назначенных к выполнению задач. Задержка увеличивается с ростом числа задач из-за длительного времени ожидания в очереди и нехватки ресурсов. Однако предложенный метод построения сети на основе SDN превосходит все другие системы благодаря своей гибкости в управлении ресурсами.

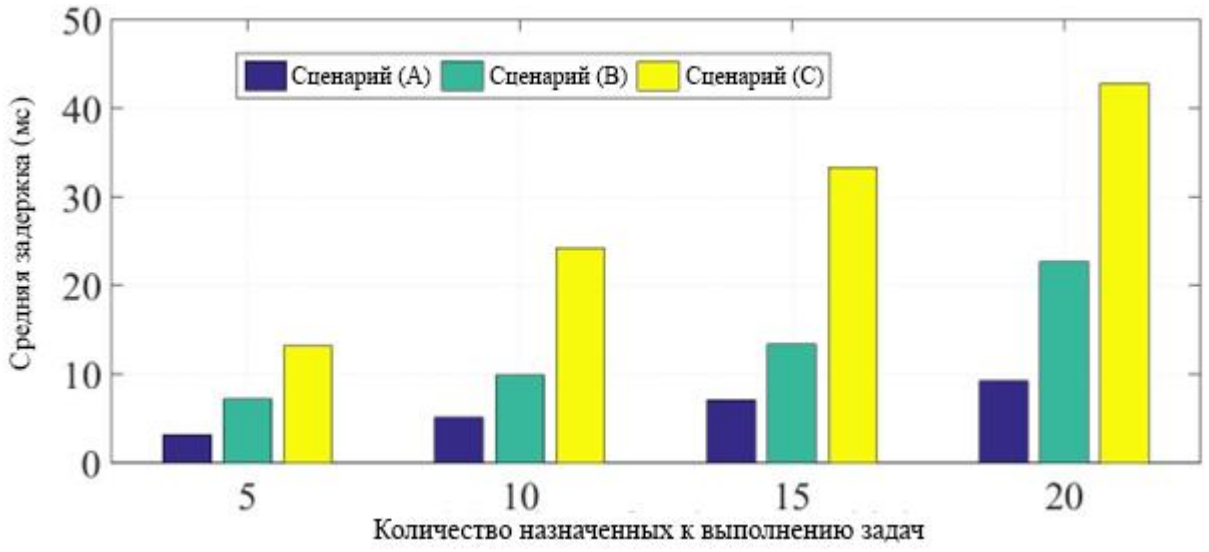


Рисунок 4.5.. Средняя требуемая продолжительность обработки задач по трем сценариям с различным количеством назначенных к выполнению задач

На рисунке 4.6. представлены показатели затрат на передачу данных по трем смоделированным сценариям с различным количеством развернутых транспортных средств. По мере увеличения количества транспортных средств затраты на передачу данных увеличиваются во всех трех сценариях. Это связано с отсутствием управления при увеличении числа конечных устройств. Однако предложенный метод позволил значительно снизить эти затраты по сравнению с двумя другими сценариями благодаря развертыванию SDN, которая обеспечивает необходимое управление.

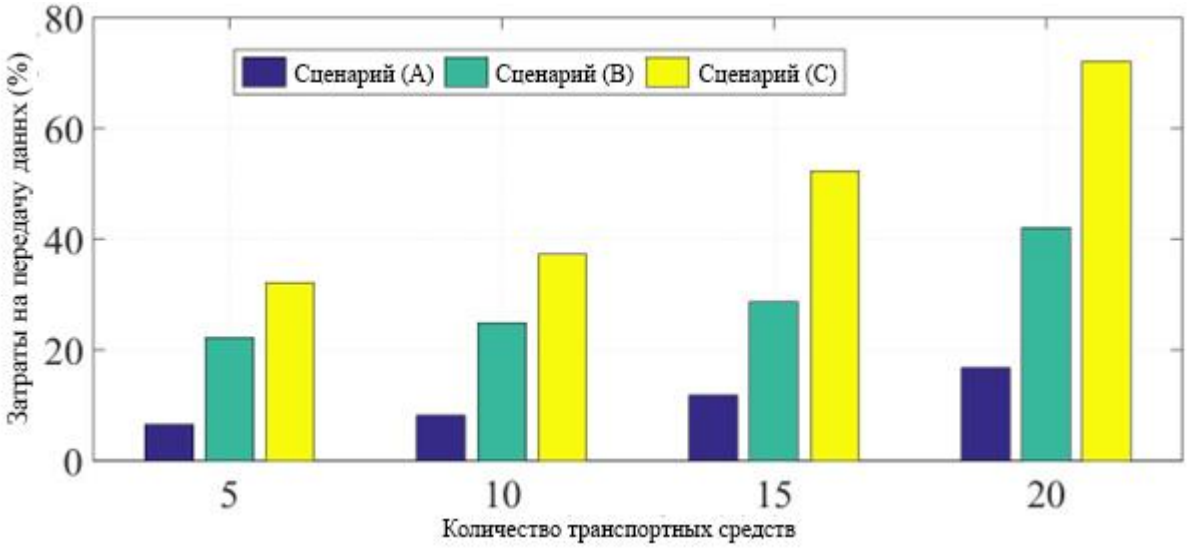


Рисунок 4.6. Затраты на передачу данных по трем смоделированным сценариям с различным количеством развернутых транспортных средств

На рисунке 4.7. представлены результаты затрат на передачу данных по трем сценариям с различным количеством назначенных к выполнению задач. С увеличением числа задач возрастают и затраты из-за длительного ожидания в очереди и нехватки ресурсов. Однако предложенный метод на основе SDN превосходит другие системы благодаря своей гибкости.

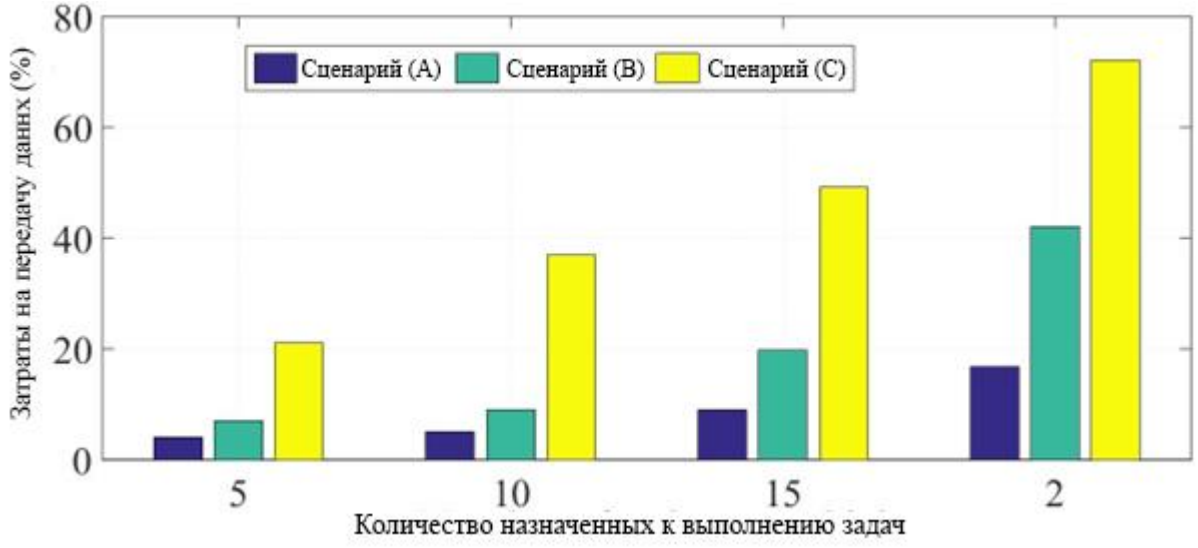


Рисунок 4.7. Затраты на передачу данных по трем сценариям с различным количеством назначенных к выполнению задач

4.4. Выводы по главе 4.

В отличии от известных решений предложен новый метод построения сети с использованием технологий MEC, SDN и D2D для поддержки приложений беспилотных автомобилей. Предлагаемая архитектура направлена на преодоление двух основных проблем автомобильных сетей – высокой плотности трафика и наличия непокрытых зон в автомобильной сети связи. При этом, разработан также алгоритм кластеризации на основе взаимодействий D2D для транспортных средств в непокрытых зонах и для выгрузки трафика сети в регионах с интенсивным движением. Результаты моделирования показывают, что предложенная архитектура дает 74% прироста производительности системы в терминах вероятности блокировки задач.

ГЛАВА 5. МЕТОД РАЗМЕЩЕНИЯ SDN-КОНТРОЛЛЕРОВ НА МОБИЛЬНЫХ УЗЛАХ СЕТЕЙ VANET ДЛЯ ВЫСОКОПЛОТНЫХ И СВЕРХПЛОТНЫХ СЕТЕЙ 6G

В главе исследуется научная проблема создания мобильной SDN, в которые контроллеры SDN устанавливаются на движущиеся объекты, например, автобусы, для обеспечения связи в высокоплотных и сверхплотных сетях 6G. Мобильный контроллер SDN имеет интерфейс с виртуальной машинной версией контроллера SDN на серверах MEC, подключенных к eNB. Система была протестирована на гетерогенных реальных сценариях, полученные результаты подтвердили её дееспособность. Предложенный метод размещения контроллеров позволяет уменьшить задержку на величину до 60% по сравнению с традиционными моделями MEC. Более того, предложенный метод уменьшает потребление энергии на 72% по сравнению с моделью Fog-MEC.

5.1. Международная деятельность в области исследований

В современном быстро меняющемся мире оставаться на связи во время движения стало необходимым условием для большинства людей. Появление мобильных технологий произвело революцию в способах общения и доступа к информации. Мотивированная высокоскоростным развитием мобильного интернета и растущим ее спрсом, система сотовой связи шестого поколения (6G) должна быть недорогой, с низким энергопотреблением, безопасной и надежной [320].

Новые требования и запросы поддерживаются системами 6G для обеспечения новых вариантов использования и новых услуг. Скорость передачи данных предполагается увеличить в 100 раз по сравнению с последней версией сотовых систем пятого поколения (5G), а пиковая скорость передачи данных должна достичь 1 Тбит/с [321, 322]. Одним из основных требований 6G является

уменьшение сетевой задержки и поддержка сверхнадежных сценариев работы с ультрамалой задержкой в таких приложениях, как автомобильные самоорганизующиеся сети VANET (Vehicular Ad Hoc Network), дополненная и виртуальная реальность (AR, VR) и тактильный интернет [323, 324]. Кроме того, ожидается, что 6G позволит повысить эффективность использования спектра в 20 раз, а качество восприятия пользователем новых услуг будет гарантировано на скорости 1000 км/ч. При этом сети 6G будут сталкиваться с множеством проблем, которые включают в себя огромный планируемый трафик, который, как ожидается, будет в 1000 раз больше, чем в существующих системах [325, 326].

Одним из основных требований к системам 6G является улучшение качества обслуживания оконечных пользователей путем предоставления текущих услуг и внедрения новых услуг и приложений с ультрамалой задержкой, высокой скоростью передачи данных, сверхвысокой надежностью и сверхвысокой доступностью [327]. Сети связи для автомобилей VANET являются одним из таких вариантов использования 6G, который предназначен для обеспечения инфраструктуры и покрытия сетью транспортных средств с гетерогенной мобильностью [328]. Однако поддержка сверхвысокой мобильности при высокоплотном развертывании сети является сложной задачей.

Программно-конфигурируемые сети (software defined networks, SDN) – это инновационный подход к управлению сетью, который отделяет плоскость управления от плоскости данных, обеспечивая централизованное управление сетью и программирование ее работы. В отличие от традиционных сетевых архитектур, где сетевые устройства принимают независимые решения, сети SDN имеют один или несколько централизованных контроллеров, которые регулируют поток трафика в сети. Отделяя логику управления от аппаратного

обеспечения, технология SDN предлагает более гибкое, масштабируемое и эффективное управление сетью.

В этой главе будут рассмотрены возможности создания мобильных SDN сетей и важный пример их использования с установкой контроллера SDN на автобус общественного транспорта.

5.2. Структура мобильной SDN сети

Предлагаемый метод построения сети предназначен для решения двух основных проблем сетей 6G: доступность сети в сценариях с высокой и сверхвысокой плотностью развертывания и возможность подключения к сети при высокой мобильности пользователя. Предложенный метод превращает традиционную SDN из стационарной в мобильную SDN.

Эволюция технологии SDN идет от централизованной схемы контроллера SDN к распределенной схеме с несколькими контроллерами. Впоследствии появилось и еще одна архитектура такой сети – распределенное управление сетью через узлы граничных вычислений. В таких сетях функции контроллера SDN распределяются по граничным вычислительным узлам для реализации централизованной схемы управления. Каждый виртуальный контроллер имеет интерфейс с централизованной схемой управления, которая поддерживает один или несколько SDN-контроллеров в зависимости от масштаба сети.

Новый метод развития технологии SDN направлен на обеспечение мобильности контроллера SDN для поддержки сценариев высокоплотных и сверхплотных сетей. Модель сети с мобильным контроллером SDN представлена на рис. 5.2.1. При этом сеть SDN можно рассматривать на трех основных уровнях: централизованные контроллеры SDN, распределенные стационарные контроллеры SDN и распределенные мобильные контроллеры SDN.

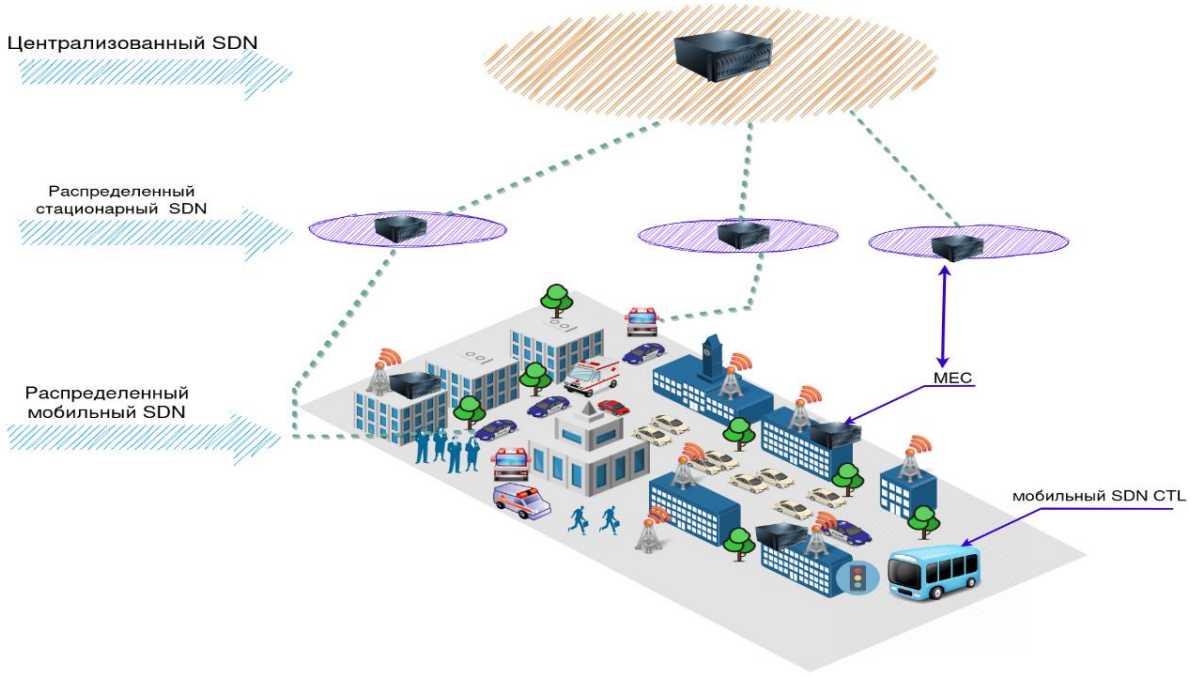


Рисунок 5.2.1. Модель SDN сети

А) Централизованные контроллеры SDN. В этом варианте построения сети SDN используются несколько централизованных контроллеров SDN для обеспечения управления сетью. Этот метод управления определяет основной уровень управления сетью и имеет необходимые интерфейсы с операторами сети. Для решения проблемы размещения и распределения контроллеров в предлагаемой сети используются ранее предложенные алгоритмы хаотической оптимизации роя сальп. Архитектура сети при этом состоит из следующих шести основных частей.

1. Плоскость управления – это набор сетевых приложений, которые управляют логикой SDN сети. Программные средства используются для

обеспечения гибкости и простоты внедрения новых приложений и услуг, для маршрутизации, балансировки нагрузки, применения политик или пользовательских приложений от поставщика услуг. Они также позволяют оркестровать и автоматизировать работу сети с помощью существующих интерфейсов прикладного программирования (API).

2. Контроллеры – это наиболее интеллектуальный и важный уровень архитектуры SDN, который содержит один или несколько контроллеров, передающих различные типы правил и политик на уровень инфраструктуры через южный интерфейс (southbound).

3. Плоскость данных (также известна как инфраструктурный уровень) представляет собой набор устройств передачи данных в сети (маршрутизаторы, коммутаторы, балансировщики нагрузки и т.д.). Она использует южные API/южный мост (southbound API) для взаимодействия с плоскостью управления, получая правила передачи данных и политики, чтобы применять их к соответствующим устройствам.

4. Северные интерфейсы (northbound) – интерфейсы, обеспечивающие связь между уровнем контроля и уровнем управления, в основном представляют собой набор API с открытым исходным кодом.

5. Интерфейсы восток-запад (пока еще не стандартизированы) обеспечивают связь между несколькими контроллерами. Они используют систему уведомлений и обмена сообщениями или распределенные протоколы маршрутизации, такие как BGP (Border Gateway Protocol) и OSPF (Open Shortest Path First). Эти интерфейсы также используются для связи централизованных контроллеров с другими устройствами сети.

6. Южные интерфейсы обеспечивают взаимодействие между плоскостью управления и плоскостью данных, которые можно обобщенно определить как протоколы, позволяющие контроллеру передавать политики на

плоскость передачи данных. Протокол OpenFlow является наиболее широко признанным и распространенным для сетей с поддержкой SDN.

OpenFlow стандартизован Open Networking Foundation (ONF), который поддержан лидерами ИТ-индустрии, Cisco, Google, HP и другими. По этой причине знание архитектуры OpenFlow важно для понимания концепции SDN. Необходимо отметить, что OpenFlow – это только протокол SDN, поскольку существует множество существующих и разрабатываемых других протоколов, например: протокол OpFlex, который перераспределяет часть задач управления сетью на уровень инфраструктуры для улучшения масштабируемости; протокол ForCES, который предлагает гибкий метод для улучшения управления традиционными сетями без использования логически централизованного контроллера; кроме того, есть библиотека ROFL (Revised OpenFlow Library), которая предоставляет API для разработчиков программного обеспечения для эффективной разработки новых приложений.

B) Стационарные распределенные контроллеры SDN. Этот вариант SDN включает интегрированный контроллер SDN на границе сети радиодоступа (Radio Access Network, RAN). RAN состоит из распределенных гетерогенных сотовых ячеек с базовыми станциями eNB. Каждая eNB подключена к серверу MEC (Multi-Access Edge Computing)с разработанной платформой MEC, представленной ниже. В этом варианте MEC представляет собой многоуровневую структуру MEC, поскольку используемые серверы MEC неоднородны с точки зрения вычислительных возможностей и имеют иерархическую структуру из трех основных облачных уровней, соединенных высокоскоростными оптоволоконными линиями.

Для существующей RAN на основе MEC для сетей 6G, представленной на рис. 5.2.2, разработана новая архитектура с применением многоуровневой системы взаимодействий граничныйх вычислительных систем, которая

представлена на рис.1. Первая основная часть платформы граничных вычислений MEC – это аппаратные или физические ресурсы, которые содержат процессор, блоки хранения данных и аппаратные средства, используемые для обеспечения, передачи и учета энергии. Эта часть отличается на разных уровнях облака. Аппаратные ресурсы используются или разделяются между локальными и выгруженными вычислительными задачами. Для эффективного использования ресурсов для граничных вычислений внедрен планировщик ресурсов для управления и распределения ресурсов между различными вычислительными задачами.

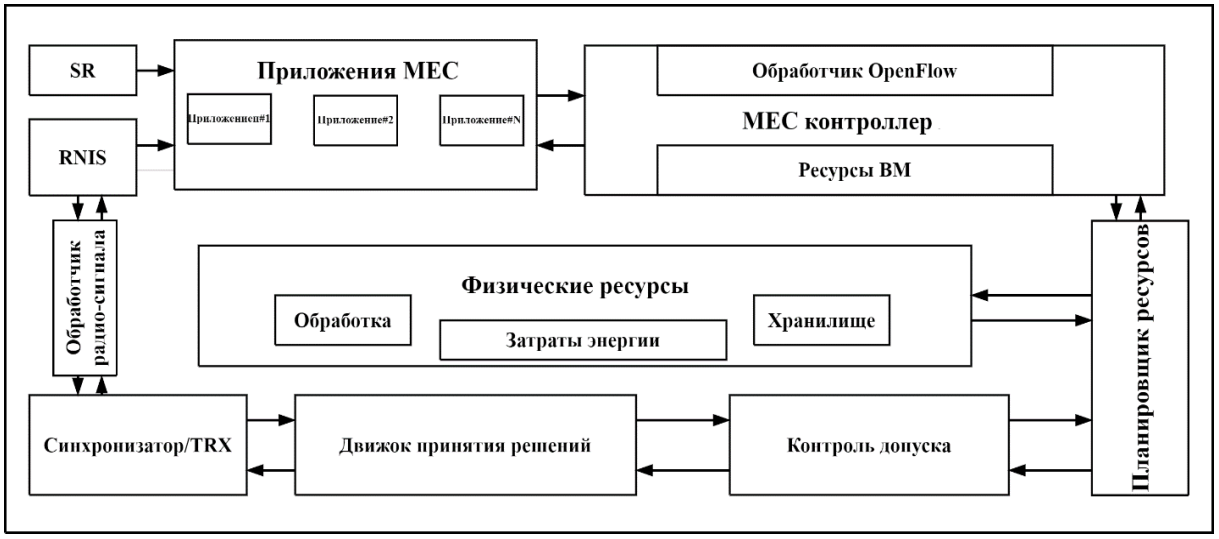


Рисунок 5.2.2. Платформа MEC для разработанного метода размещения SDN-контроллеров на мобильных узлах сетей VANET для высокоплотных и сверхплотных сетей 6G

Вычислительные задачи выгружаются на сервер MEC, например, микро- или мини- MEC, и устройство должно решить, обрабатывать ли запрошенную задачу выгрузки или нет, на основе имеющихся ресурсов и других

предварительно определенных критериев, связанных с качеством обслуживания (QoS). Таким образом, платформа MEC должна содержать механизм принятия решений, который отвечает на запросы о выгрузке, принимая или отклоняя запросы о выгрузке приложений. Механизм принятия решений – это часть платформы MEC, которая реализует алгоритмы выгрузки приложений. Для того чтобы принять решение о выгрузке задачи, механизм принятия решений должен иметь данные о текущих доступных ресурсах на сервере, которые могут быть использованы для обработки запрошенной задачи. Эти метрики ресурсов передаются в механизм принятия решений через систему управления допуском.

Другой частью платформы является синхронизатор/TRX (Transceiver), который представляет собой программнно-аппаратную систему, отвечающую за соединение сервера MEC с другими устройствами, например, с eNB (базовыми станциями). Таким образом, синхронизатор/TRX – это часть, которая принимает и передает данные от и к платформе MEC.

Каждый сервер MEC имеет контроллер, который реализует виртуальную машину (ВМ) метода управления сетью SDN, используемого в опорной сети. Этот контроллер MEC применяется для облегчения интеграции и взаимодействия с опорной сетью. Кроме того, внедрение контроллера MEC позволяет управлять и контролировать работу сервера MEC таким образом, чтобы достигалась доступность, гибкость и меньшая задержка. Контроллер MEC получает обновления от контроллера SDN в опорной сети по протоколу OpenFlow и обновляет необходимые части платформы MEC в соответствии с полученными данными. Связь серверов MEC с контроллером SDN по протоколу OpenFlow обеспечивает эффективное и более быстрое взаимодействие между контроллером SDN и RAN. Еще одной частью платформы MEC являются

приложения MEC, которые предоставляют операторам сети новые способы добавления услуг на основе MEC.

Распределенные стационарные контроллеры SDN на основе MEC имеют интерфейсы с централизованными и мобильными контроллерами SDN.

C) Мобильные распределенные контроллеры SDN. Этот предлагаемый метод включает в себя распределенные (возможно, упрощенные) SDN-контроллеры, установленные на движущихся объектах. Для мобильных контроллеров SDN есть много вариантов использования. Мобильные упрощенные контроллеры SDN могут быть установлены на транспортных средствах для поддержки взаимодействия в сверхплотных сетях и обеспечения сверхвысокой доступности системы. Мобильный SDN-контроллер имеет один или несколько интерфейсов с распределенными стационарными SDN-контроллерами. Такая интеграция отвечает требованиям к связности и доступности для сетей 6G. Для реализации мобильных контроллеров используется облегченная модель платформы MEC, представленная на рисунке 5.2.2.

5.3. Моделирование предложенных решений

Построим мобильную мультиконтроллерную модель сети SDN на основе теории графов. В качестве основного способа связи в SDN обычно используется внутриполосная связь. Топология сети контроллеров представлена неориентированным графом G = (V,E), где V и E – множество узлов и связей соответственно.

Набор развернутых SDN-контроллеров в опорной сети обозначается N_C и определяется следующим образом:

$$N_C = \{C_1, C_2, C_3, \dots\dots\dots\dots, C_N\},\ \forall N \in ¡\ ,$$

где N – общее количество развернутых SDN-контроллеров в ядре сети. Набор развернутых контроллеров на границе сети обозначается E_C и определяется как

$$E_C = \{C_1, C_2, C_3, \ldots\ldots\ldots\ldots, C_K\}, \forall K \in \mathrm{R},$$

где K – общее количество развернутых SDN-контроллеров на границе RAN. Каждая группа граничных контроллеров имеет интерфейс с контроллером опорной сети. Набор граничных контроллеров, принадлежащих контроллеру основной сети C_i, определяется следующим образом:

$$E_C{}^{Ci} = \{C_1{}^{ci}, C_2{}^{ci}, C_3{}^{ci}, \ldots\ldots\ldots\ldots, C_{Kci}{}^{ci}\}, \forall K_{ci} \in \mathrm{R}, K_{ci} < k, E_C{}^{Ci} \subset E_C,$$

где K_{ci} – общее количество граничных контроллеров с интерфейсом с контроллером основной сети C_i, k – число коммутаторов OF. Набор развернутых мобильных контроллеров определяется как

$$M_C = \{C_1, C_2, C_3, \ldots\ldots\ldots\ldots, C_O\}, \forall O \in ¡ ,$$

где O – общее количество развернутых мобильных SDN-контроллеров. Каждая группа мобильных контроллеров имеет интерфейс с граничным контроллером. Набор мобильных контроллеров, принадлежащих граничному контроллеру C_j, определяется следующим образом:

$$M_C{}^{Cj} = \{C_1{}^{cj}, C_2{}^{cj}, C_3{}^{cj}, \ldots\ldots\ldots\ldots, C_{Ocj}{}^{cj}\}, \forall O_{cj} \in \mathrm{R}, O_{cj} < O, M_C{}^{Cj} \subset M_C, \text{где}$$

O_{ci} – общее количество мобильных контроллеров с интерфейсом с граничным контроллером C_j.

В плоскости данных развернутых коммутаторов, распределенных между контроллерами, каждый коммутатор имеет соединение с контроллером SDN, определенным с помощью разработанного алгоритма размещения контроллеров. Набор развернутых коммутаторов определяется следующим образом:

$$S = \{S_1, S_2, S_3, \ldots\ldots\ldots\ldots, S_x\} \forall x \in ¡ .$$

Одним из способов проверки производительности контроллера является оценка его временного отклика, на который в основном влияет задержка в очереди. Контроллеры могут быть смоделированы с помощью многосерверной модели очередей M/M/s, где s – число обслуживающих устройств. Среднее время отклика T_i контроллера C_i является суммой времени ожидания в очереди и времени обработки и может быть рассчитано по формуле Эрланга как функция интенсивности поступления λ_i и интенсивности обслуживания μ.

$$T_i(\lambda) = \frac{C\left(s, \frac{\lambda_i}{\mu}\right)}{s\mu_i - \lambda_i} + \frac{1}{\mu},$$

где $C(s, \lambda/\mu)$ – это вероятность того, что все серверы системы используются и любой прибывающий пакет будет поставлен в очередь, которая может быть рассчитана по формуле:

$$C\left(s, \frac{\lambda}{\mu}\right) = \frac{\left(\frac{(s\rho)^c}{s!}\right)\left(\frac{1}{1-\rho}\right)}{\sum_{k=0}^{s-1}\frac{(s\rho)^k}{k!} + \left(\frac{(s\rho)^c}{s!}\right)\left(\frac{1}{1-\rho}\right)} = \frac{1}{1 + \left(\frac{1}{1-\rho}\right)\left(\frac{s!}{(s\rho)^c}\right)\sum_{k=0}^{s-1}\frac{(s\rho)^k}{k!}}$$

Где, $\rho = \frac{\lambda_i}{s\mu}$,

где ρ представляет собой коэффициент использования сервера, который является показателем стабильности системы. Интенсивность поступления λ_i контроллера C_i может быть рассчитана как сумма средних интенсивностей поступления заявок на обслуживание от коммутаторов, подключенных к контроллеру: $\lambda_i = \sum_{k_i} \lambda_s$, где λ_s – интенсивность поступления заявок, k_i – Число коммутаторов.

1) Стоимость взаимодействия между коммутаторами и контроллерами.(CAPEX и OPEX) Когда таблица потоков, например, новая таблица потоков, должна быть установлена по требованию, коммутатор должен отправить пакеты на контроллер, который рассчитывает путь передачи данных и устанавливает соответствующую метку потока на коммутатор. Затем коммутатор пересылает пакеты в соответствии с таблицей потоков. В этом процессе для контроллера C_i Общая задержка пакета состоит из суммы времени доставки информации пакета в контроллер и обратного ответа контроллера коммутаторам. Стоимости взаимодействия между коммутатором (S_j) и контроллером в сети OpenFlow (C_i) могут быть определены следующим образом:

$$C_{C_i-S_j} = 2p_{r-S_j} \sum_{i \in N_C} \sum_{j \in K_S} \left[\frac{\lambda_{S_j}^t}{p_r} d_{C_i-S_j} B_{C_i-S_j} \right],$$

где $C_{C_i-S_j}$ – стоимость связи между коммутатором S_j и контроллером C_i; p_{r-S_j} – средняя скорость опроса коммутатора S_j; λ_{S_j} – интенсивность заявок от коммутатора S_j; t – временной интервал; p_r – удельная скорость опроса коммутатора; $d_{C_i-S_j}$ – расстояние между контроллером C_i и коммутатором S_j; $B_{C_i-S_j}$ – двоичное решение коммутатора S_j о подключении к контроллеру C_i, полученное с помощью алгоритма хаотического роя; N_C –число контроллеров в ядре; K_S – число коммутаторов OF. Стоимость взаимодействия определяется суммарной задержкой.

2) Стоимость взаимодействия между контроллерами. В среде с несколькими контроллерами требуется синхронизация передачи информации между ними, чтобы каждый контроллер мог поддерживать глобальную информацию о состоянии сети. Стоимость синхронизации состояния в основном относится к затратам взаимодействия, определяемым интерактивной

информацией о состоянии между контроллерами трех вышеуказанных основных уровней. Эта задержка определяется как задержка взаимодействия и синхронизации между участниками в плоскости управления. Стоимость синхронизации рассчитывается следующим образом:

$$C_{C_i-S_j} = \alpha_c \sum_{i\in N_c}\sum_{j\in N_c} d_{C_{i,j}} + \beta_e \sum_{i\in N_c}\sum_{o\in E_C} d_{C_{i,o}} + \gamma_m \sum_{o\in E_c}\sum_{q\in M_C} d_{C_{o,q}}$$

$$\forall \quad \gamma_m < \beta_e < \alpha_c < p_r,$$

где α_c – средняя скорость передачи информации о состоянии контроллеров в основной сети; β_e – средняя скорость передачи информации о состоянии граничных контроллеров; γ_m – средняя скорость передачи информации о состоянии мобильных контроллеров; $d_{C_{i,j}}$ – расстояние между контроллерами C_i и C_j, размещенными в основной сети; $d_{C_{i,o}}$ – расстояние между контроллером C_i в основной сети и граничным контроллером C_o; $d_{C_{o,q}}$ – расстояние между граничным контроллером C_o и мобильным контроллером C_q.

3) Оптимизация задержки взаимодействия в сети SDN. Общая стоимость взаимодействия в сети представляет собой стоимость взаимодействия между коммутаторами и контроллерами, а также между контроллерами одного и разных уровней в определенный момент времени. Для заданной топологии сети стоимость взаимодействия между коммутаторами и контроллерами уменьшается с увеличением числа контроллеров, в то время как стоимость взаимодействия между контроллерами увеличивается. Конечной целью нашей задачи является разделение сети на кластеры управления и определение коммутаторов для каждого кластера таким образом, чтобы минимизировать общую задержку. Таким образом, для достижения этой цели ставится следующая оптимизационная задача:

$$\min(\delta \sum C_{C_i-S_j} + \varepsilon \sum C_{C_i-S_j})$$

при следующих ограничениях:

1. $B_{C_i - S_j} \in \{0,1\}$;

2. $T_{C_i} \wedge T_{C_j} \wedge T_{C_o} \leq \tau \ \forall i \in N_C, j \in E_C, o \in M_C$;

3. $U_{LB-C} \leq U_{C_i} \leq U_{UB-C} \ \forall i \in N_C$;

4. $U_{LB-E} \leq U_{C_j} \leq U_{UB-E} \ \forall j \in E_C$;

5. $U_{LB-M} \leq U_{C_o} \leq U_{UB-M} \ \forall o \in M_C$,

где δ – вес стоимости взаимодействия между коммутаторами и контроллерами; ε – вес стоимости взаимодействия между контроллерами; T_{C_i} – среднее время отклика контроллеров базовой сети; T_{C_j} – среднее время отклика Edge узла; T_{C_o} – Среднее время отклика мобильного контроллера ; $U_{LB\text{-}C}$ – индекс использования SDN контроллеров в основной сети; $U_{UB\text{-}C}$ – индекс использования контроллеров; $U_{LB\text{-}E}$ – индекс использования контроллеров в основной сети; $U_{UB\text{-}E}$ – индекс использования граничных контроллеров; $U_{LB\text{-}M}$ – индекс использования мобильных SDN контроллеров; $U_{UB\text{-}M}$ – индекс использования мобильных SDN контроллеров; M_C – набор контроллеров (вектор).

Второе ограничение указывает, что среднее время отклика контроллеров основной сети, граничных и мобильных контроллеров (T_{C_i}, T_{C_j}, T_{C_o}) не должно превышать порогового значения τ-(время отклика каждого индивидуально приложения), которое предопределено. Это происходит для всех контроллеров в наборе доступных в основной сети, граничных контроллеров и мобильных контроллеров SDN. τ предопределено таким образом, чтобы обеспечить определенное качество обслуживания (QoS). Ограничения с третьего по пятое касаются индекса использования каждого контроллера в сети, который должен

находиться между нижним и верхним пределами индекса использования. Оба значения U_{UB} и U_{LB} для контроллеров предопределены таким образом, чтобы поддерживать QoS системы. Индекс использования контроллера используется для сопоставления с утилизацией энергии, хранения и обработки данных. При этом рассматривались различные пределы использования для контроллеров различного уровня, поскольку такие контроллеры имеют разные возможности. Для поддержки же требований QoS рассматривался один и тот же пороговый предел для отклика контроллера.

5.4. Оценка эффективности предложенного метода

Разработанный метод размещения SDN-контроллеров на мобильных узлах сетей VANET для высокоплотных и сверхплотных сетей 6G с использованием мобильного SDN-контроллера был смоделирован для оценки производительности. Теперь рассмотрим схему моделирования с основными этапами и параметрами и обсудим результаты моделирования.

Среда моделирования. Предложенная схема мобильного контроллера была оценена в среде симулятора сети NS-3 с симулятором сети Cloudsim. В процессе оценки рассматривалась сотовая сеть с топологией, представленной в табл. 5.4.1. Сеть оценивалась для разного количества оконечных устройств, чтобы проверить производительность для различных сценариев развертывания. Рассматриваемые вычислительные задачи относятся к разным категориям приложений, чтобы оценить производительность в разных вариантах использования.

Таблица 5.4.1 Параметры моделирования

Параметр	**Значение**

Размер области сети	5×5 км^2
Количество узлов MEC	2
Количество туманных узлов	5
Количество подключенных мобильных устройств	20, 40, 60, 80, 100
Количество выделенных задач на одного пользователя	10, 20, 30, 40, 50
Число мобильных контроллеров SDN	1
B, Полоса пропускания	20 МГц
σ^2 Уровень принимаемого сигнала.	−60 дБм
γ_0 Принимаемый сигнал на эталонном расстоянии 1 метр	−30 дБ
h_{Di}Высота i-го (Di)	30 м
t_s Длительность временного слота	45 мкс
V_{max} Максимальная скорость БПЛА	50 м/с
V_{Di} Диапазон скорости БПЛА	1–50 м/с
$E_{Cyc\text{-}mob}$ Энергопотребление конечного устройства	1J/GHz
$E_{Cyc\text{-}fog}$ Энергопотребление FOG узла	1J/GHz
$E_{Cyc\text{-}MEC}$ Энергопотребление MEC-узла	1J/GHz
ω_{fog} Рабочая частота процессора FOG	1–3 ГГц
ω_{MEC} Рабочая частота процессора MEC	1,5–7,0 ГГц
ω_{mob} Рабочая частота процессора мобильного контроллера	0,5–1,0 ГГц
$\tau_{app\text{-}I}$ Средняя задержка отклика приложения I	20 мкс

$\tau_{app\text{-}II}$ Средняя задержка отклика приложения II	15 мкс
$\tau_{app\text{-}III}$ Средняя задержка отклика приложения III	10 мкс
$\tau_{app\text{-}IV}$ Средняя задержка отклика приложения IV	7 мкс

В процессе оценки рассматривались рабочие нагрузки четырех гетерогенных типов приложений. Первая категория рассматриваемых приложений – приложения, включающие рабочие нагрузки простых задач, например, необходимых для обработки простых веб-страниц. Вторая категория – приложения на основе изображений, включающие в себя рабочие нагрузки простых задач по обработке изображений. Третья категория приложений – простые видеоприложения, включающие рабочие нагрузки по обработке простого видео. Четвертый тип – приложения, включающие обработку 360-градусных изображений и видео. При переходе от первой категории к четвертой, задачи требуют всё более объемных ресурсов, что снижает вероятность возможности локального выполнения. Таким образом, для обработки задач более высоких категорий требуется больше энергии, чем для задач более низких категорий.

Для того чтобы продемонстрировать эффективность разработанного метода, были рассмотрены следующие пять систем:

1) *традиционная многоконтроллерная SDN* представляет собой сотовую сеть, в которой развернута схема с несколькими контроллерами SDN в основной сети и без какой-либо другой схемы управления;

2) *SDN-сеть на основе туманных вычислений* – SDN централизованной топологии с распределенными граничными контроллерами SDN, интегрированными в туманные узлы;

3) *SDN-сеть на основе MEC* – SDN централизованной топологии с распределенными граничными контроллерами SDN, интегрированными в узлы MEC;

4) SDN-сеть на основе технологий туманных и граничных вычислений (fog-MEC) – SDN централизованной топологии с распределенными граничными контроллерами SDN, интегрированными в туманные и MEC-узлы;

5) *сеть на основе мобильной SDN (MSDN)* – сеть с мобильными SDN-контроллерами, установленными на мобильных узлах, например, на автобусах, с поддержкой вычислительных блоков fog-MEC.

Эти различные системы были рассмотрены, чтобы показать эффективность каждого из методов построения сети SDN в сравнении с предложенным. В качестве метрик производительности сети использовались потребление энергии, задержка и доступность.

5.5. Результаты моделирования

На рис. 5.5.1. представлены полученные результаты вычисления средней задержки обработки вычислительных задач для рассматриваемых пяти систем. В каждом сценарии оконечным пользователям назначалось разное количество вычислительных задач. С увеличением количества назначенных задач средняя задержка, необходимая для их обработки, увеличивалась во всех пяти системах. Это связано с накладными расходами, возникающими из-за увеличения количества доступных вычислительных задач. Однако предложенный метод

превзошел все другие и достиг более высокой эффективности, особенно для большего числа доступных задач.

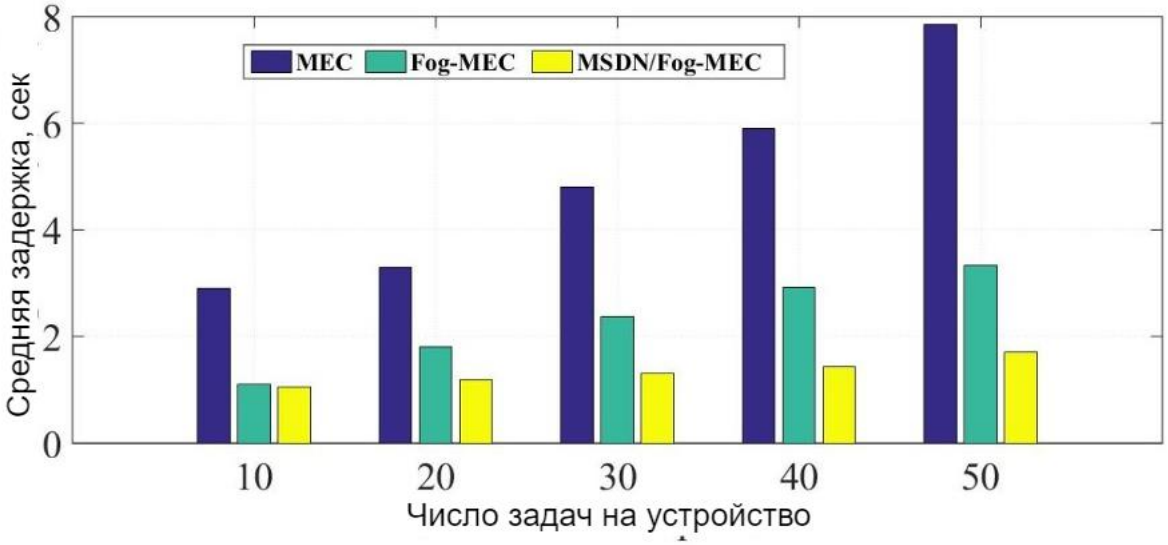


Рисунок 5.5.1 Зависимость средней задержки от количества доступных задач

На рис. 5.5.2. представлены результаты определения средней задержки, необходимой для обработки вычислительных задач, для рассматриваемых пяти систем. Средняя задержка, необходимая для обработки задач, была установлена для ранее упомянутых четырех видов приложений. Развертывание мобильного SDN-контроллера значительно сократило среднее время, необходимое для обработки вычислительных задач различных приложений, особенно для сложных задач с более высокой рабочей нагрузкой.

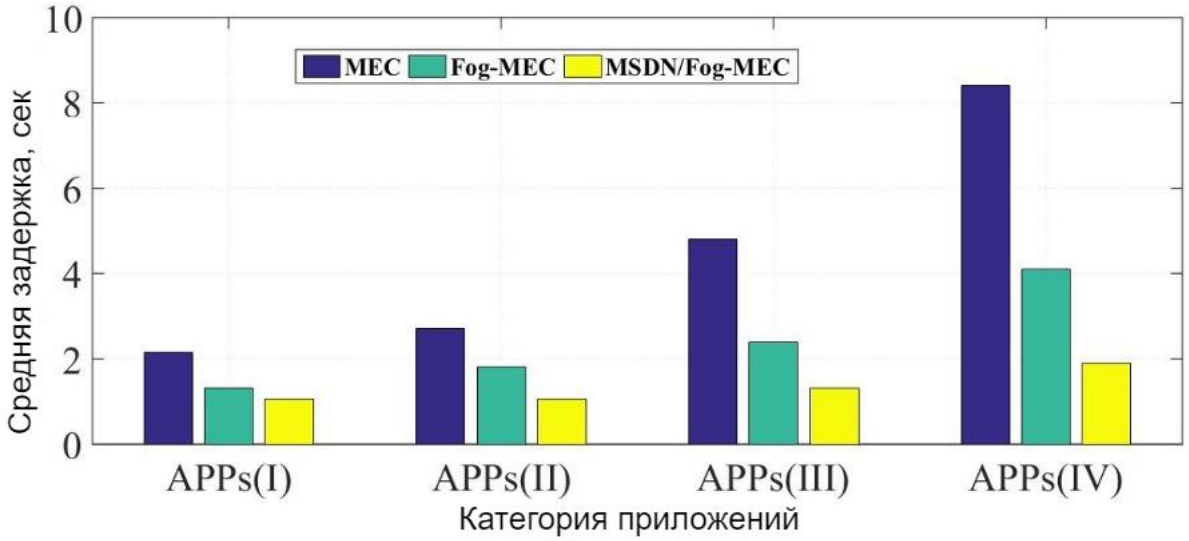


Рисунок 5.5.2 Средняя задержка для различных категорий приложений

Кроме того, была определена задержка для пяти систем при различном количестве развернутых конечных устройств. Результат этого эксперимента представлен на рис. 5.5.3.. Средняя задержка увеличивается с ростом числа развернутых узлов из-за увеличения времени ожидания в очереди и коммуникационных накладных расходов. Однако это увеличение минимально для предложенного метода благодаря высокой гибкости сети, обеспечиваемой разработанной архитектурой SDN, которая помогает управлять ресурсами таким образом, чтобы уменьшить накладные расходы и время ожидания в очереди. Подводя итог, можно сказать, что предложенный метод размещения контроллеров в сети SDN снижает среднюю задержку, необходимую для обработки вычислительных задач сети, в среднем на 60% по сравнению с традиционной топологией SDN.

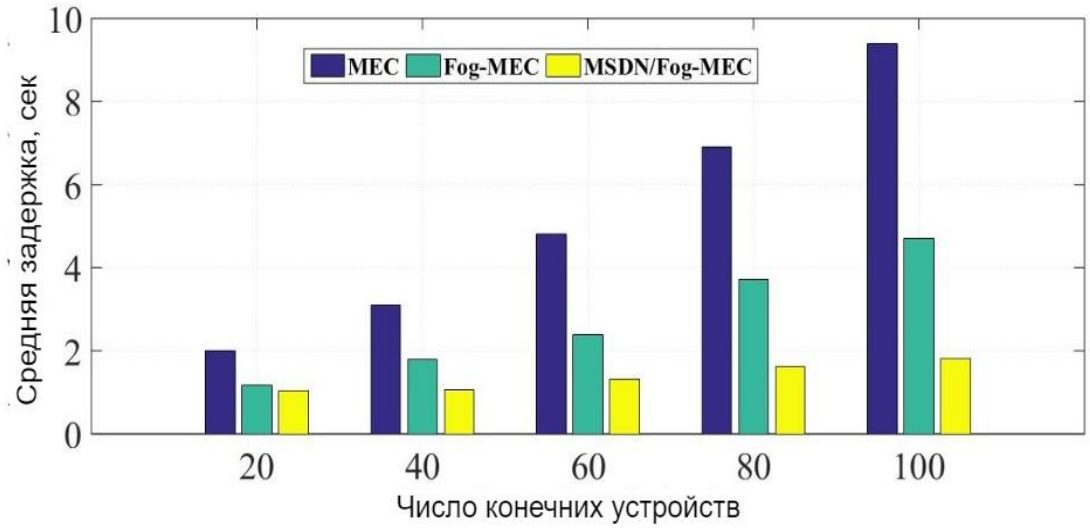


Рисунок 5.5.3. Зависимость средней задержки от количества оконечных устройств

На рис. 5.5.4. представлен процент потребляемой энергии для рассматриваемых пяти систем при различном количестве доступных задач. Видно, что с увеличением числа доступных задач потребление энергии значительно возрастает. Однако предложенный метод, использующий технологию мобильной SDN, управляет потреблением энергии более эффективно и поэтому превосходит по энергетическим показателям другие топологии.

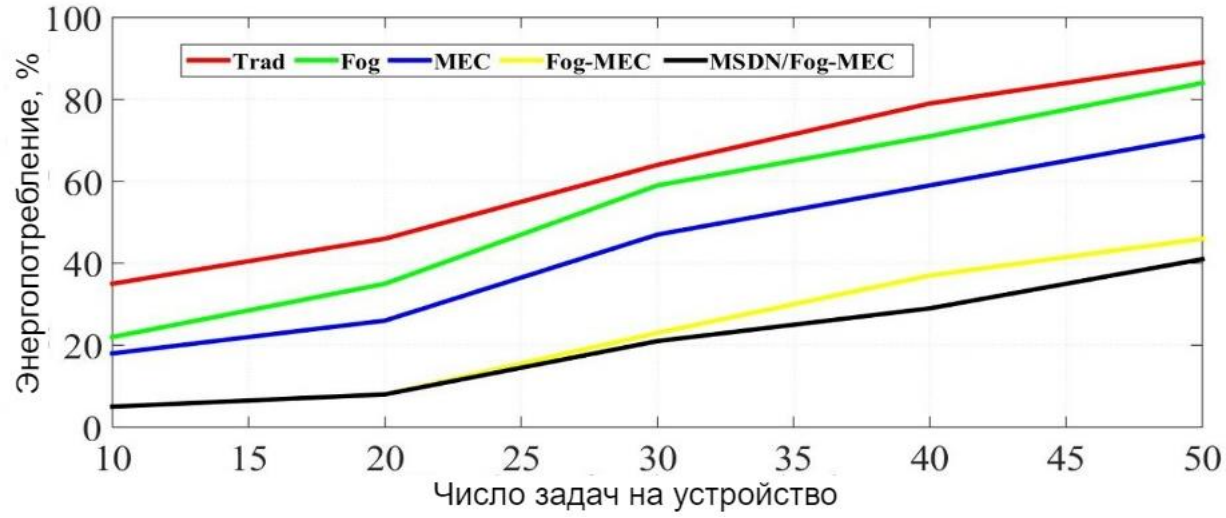


Рисунок 5.5.4. Относительное энергопотребление при разных задач на усстройствах

На рис. 5.5.5. показано потребление энергии в каждой системе для ранее упомянутых четырех видов приложений: использование мобильного контроллера SDN снизило потребление энергии на значительную величину, особенно для сложных задач с более высокой рабочей нагрузкой.

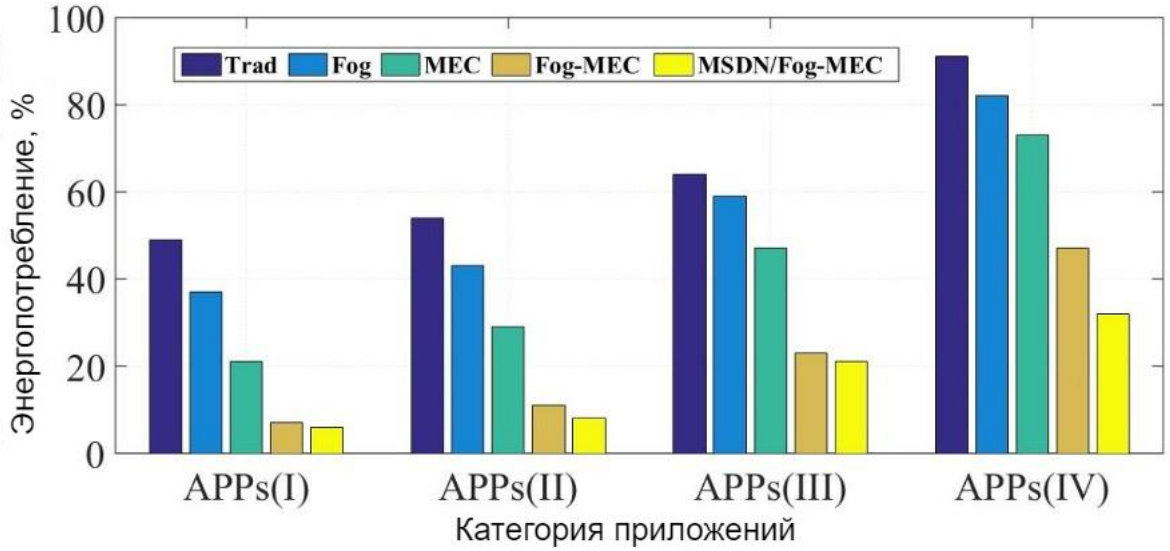


Рисунок 5.5.5. Потребление энергии для различных категорий приложений

Кроме того, было определено потребление энергии для пяти систем при разном количестве развернутых оконечных устройств. Результат –представлен на рис. 5.5.6., где видно, что предложенный метод построения сети SDN снижает среднее энергопотребление в среднем на 72% по сравнению с традиционной топологией SDN.

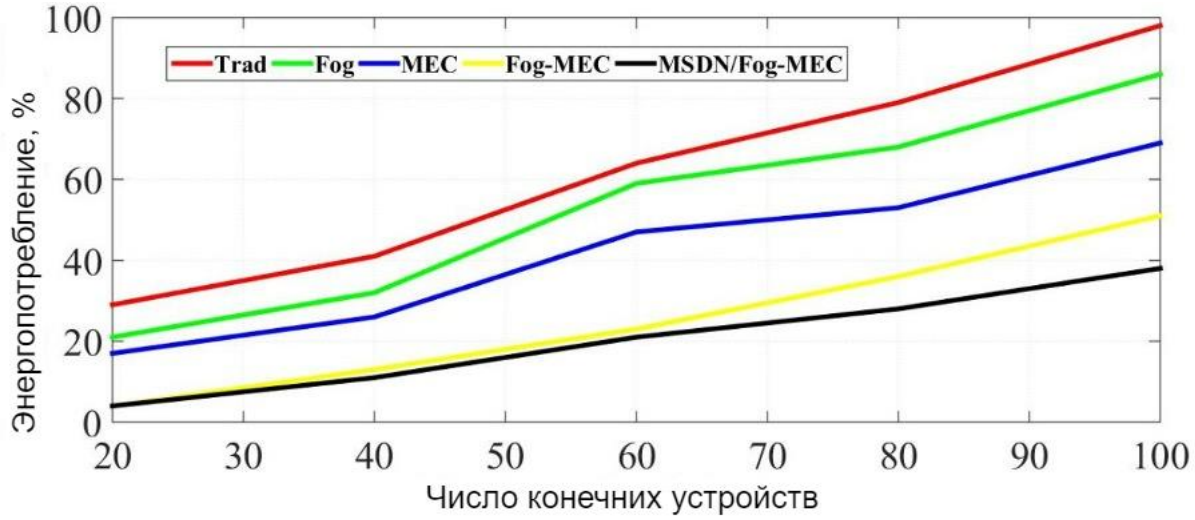


Рисунок 5.5.6. Потребление энергии при различном количестве конечных устройств

Также была определена доля заблокированных задач при разном количестве доступных задач (рис. 5.5.7.) и для различных категорий приложений (рис. 5.5.8). Правда, информация о доле заблокированных задач была получена только для четырех систем. Традиционная система отличается высоким процентом блокировок и поэтому в данном сравнении она не рассматривалась. Результаты показали, что с увеличением числа доступных задач вероятность блокировки задачи возрастает из-за нехватки ресурсов, а также длительного времени ожидания в очереди. Задача блокируется, если превышается время, требуемое для ее обработки с заданными характеристиками качества обслуживания. Предложенный метод уменьшает вероятность блокировки в основном при увеличении количества доступных задач и при высокой плотности расположения узлов. При этом вероятность блокировки уменьшается в среднем на 35% по сравнению с другими методами.

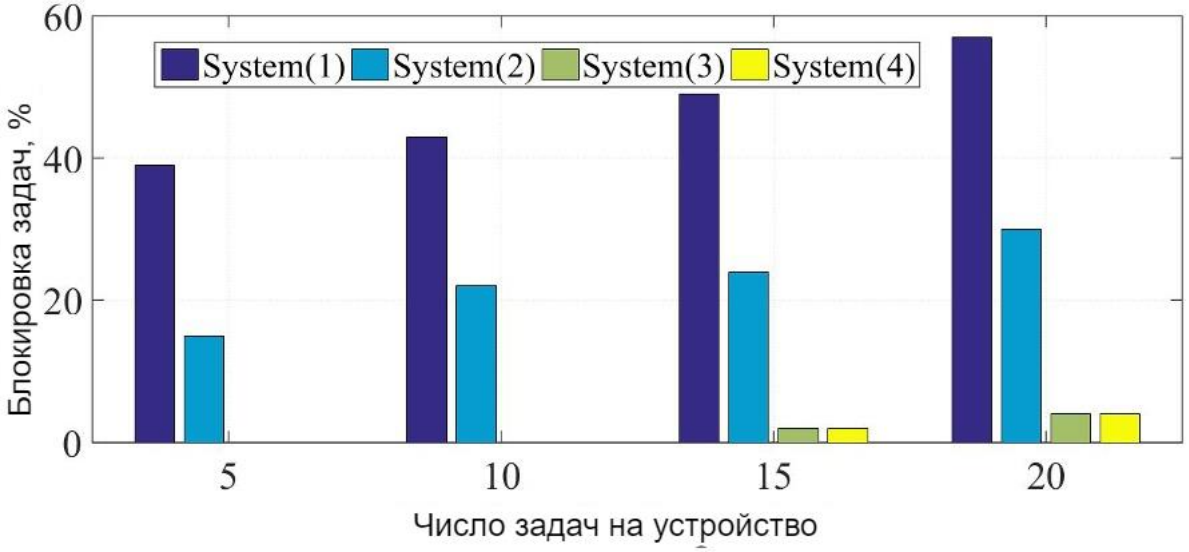


Рисунок 5.5.7 Доля заблокированных задач при различном количестве доступных задач

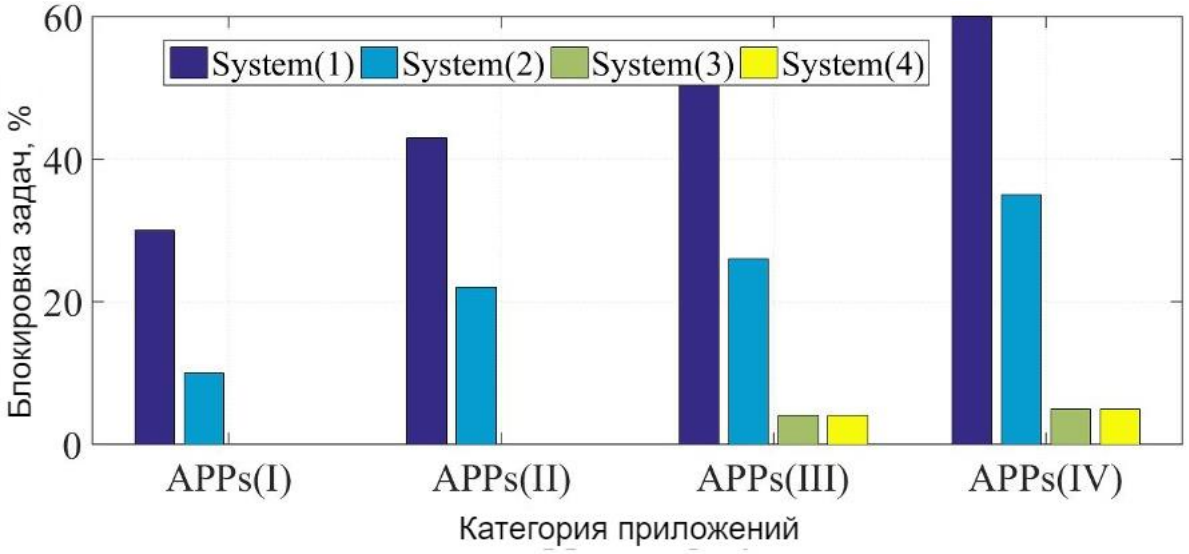


Рисунок 5.5.8. Доля заблокированных задач для различных категорий приложений

5.6. Вывод по главе 5

В главе представлен новый метод размещения SDN-контроллеров в мультиконтроллерных сетях, отличающийся тем, что контроллеры могут располагаться на мобильных узлах сетей VANET, например, в автобусах, для обеспечения связи в высокоплотных и сверхплотных сетях 6G и с туманной средой устройств сети, что позволяет уменьшить задержку на 60% по сравнению с традиционными моделями граничных вычислений на базе SDN, а также снизить потребляемую энергию на 72% по сравнению с методом fog-MEC на базе SDN.

ГЛАВА 6. МОДЕЛЬ ПРОГНОЗИРОВАНИЯ ТРАФИКА ДЛЯ СЕТЕЙ ИНТЕРНЕТА ВЕЩЕЙ ВЫСОКОЙ И СВЕРХВЫСОКОЙ ПЛОТНОСТИ НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ И ТУМАННЫХ ВЫЧИСЛЕНИЙ

Интернет вещей (IoT) — это одна из самых перспективных концепций, которая применяется в системах связи для внедрения огромного количества приложений и услуг во все сферы жизни [329, 330]. Данная концепция представляет собой третью эволюцию традиционного Интернета, и позволяет реализовать парадигму взаимодействия "машина-машина" (M2M) [331].

Не так много времени прошло с тех пор как Интернет вещей был объявлен одним из вариантов использования сотовой связи пятого поколения (5G). Однако интеграция данной концепции в сотовую систему связи привела к множеству проблем. Основной проблемой, с которой столкнулись сотовые системы связи на базе IoT, явился гетерогенный трафик [332, 333].

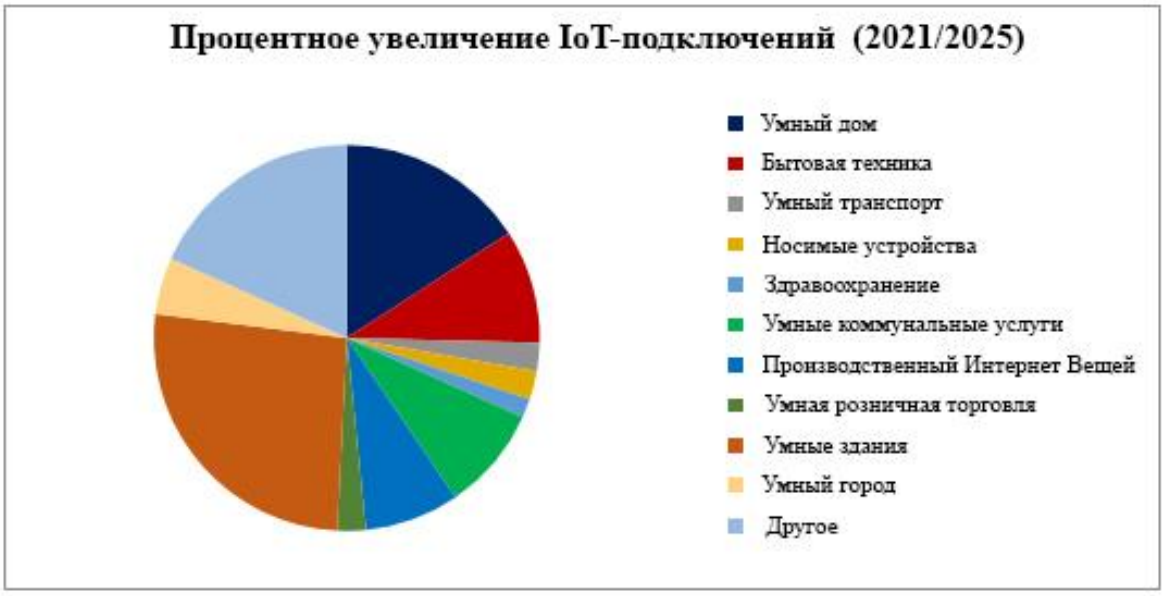


Рисунок 6 – Процентное увеличение количества IoT-подключений для каждого приложения в 2025 году по сравнению с 2021 годом.

Ожидается, что к 2025 году количество устройств, подключенных к Интернету вещей, в десять раз будет превышать число устройств в 2021 году [334]. Такое увеличение количества подключенных устройств приведет к появлению IoT-соединений во многих приложениях. Рис.6 иллюстрирует процентное увеличение подключений Интернета вещей в 2025 году по сравнению с текущим существующим числом подключений в 2021 году [335].

Такое огромное количество приложений Интернета вещей создает огромный гетерогенный сетевой трафик, который будет проблемой при проектировании сетей Интернета вещей, особенно для систем, которые базируются на сотовой связи (например, узкополосной IoT (NB-IoT)) [336, 337]. Одним из основных сценариев развертывания, поддерживаемых системами 5G, является высокоплотное и сверхплотное развертывание, а IoT должна поддерживать сценарии такого развертывания и высокую масштабируемость [338]. Это связано с недавними инновациями в производстве сенсорных устройств, которые привели к появлению огромного количества доступных датчиков.

На данный момент существует множество инструментов и решений для уменьшения влияния разнородности и огромного объема сетевого трафика IoT.

Одним из наиболее эффективных инструментов для минимизации влияния на такой трафик и поддержания производительности сети на высоком уровне является машинное обучение (ML).

ML используется для классификации сетевого трафика IoT и прогнозирования сетевого трафика на определенный период [339 - 341]. Для облегчения управления сетью модели классификации и прогнозирования на основе машинного обучения реализуются в основной сети и на сервере приложений. Однако в последних научных работах рассматривается возможность разработки и внедрения таких моделей на границе сети доступа.

Распределенные граничные вычисления, например, туманные вычисления, являются еще одной эффективной парадигмой, используемой в сетях IoT для повышения масштабируемости сети и обеспечения высокоплотного и сверхплотного развертывания [342]. Недавно туманные вычисления были внедрены в сетях IoT, чтобы предоставить вычислительный ресурс конечным устройствам, работающим от батарей. Это обеспечило огромное количество преимуществ для сетей IoT, которые можно обобщить следующим образом [343 - 343]:

- Уменьшение задержки связи,
- Снижение перегрузки сети,
- Улучшение операций управления сетью,
- Обеспечение возможностей для выгрузки данных,
- Повышение общей надежности сети,
- Повышение общей масштабируемости сети,
- Повышение энергоэффективности и, таким образом, увеличение времени автономной работы оконечных устройств.

Внедрение туманных вычислений облегчает реализацию и интеграцию алгоритмов машинного обучения, разработанных для сетей IoT, реализованных на границе сети доступа [345]. Однако такая интеграция имеет определенный уровень сложности из-за ограниченных вычислительных ресурсов туманных узлов, например, при хранении и обработке данных [346]. Целью исследований диссертационной работы в рассматриваемом направлении является разработка нового алгоритма ML на основе сверточной нейронной сети (CNN) для прогнозирования сетевого трафика в высокоплотных и сверхплотных сетях IoT. Алгоритм является достаточно простым и может быть реализован на туманных узлах.

Основной вклад может быть обобщен в следующих пунктах:

- Проектирование и разработка новой архитектуры сети IoT на основе распределенных туманных вычислений,
- Проектирование и разработка достаточно простого алгоритма ML на основе CNN для прогнозирования сетевого трафика за определенный период,
- Обучение разработанного ML-алгоритма с использованием двух наборов данных,
- Оценка производительности разработанной архитектуры.

Остальная часть главы организована следующим образом:

- В разделе II представлены существующие решения, связанные с разработкой новых решений.
- В разделе III представлены предлагаемые решения на основе ML для прогнозирования трафика IoT.
- В разделе IV представлена оценка эффективности предложенного решения.

6.1 Существующие решения

Традиционная облачная архитектура IoT не может удовлетворить высокие требования 5G систем, учитывая рост количества беспроводных «умных» устройств и коммуникационных технологий. В последнее время распространение сетевых приложений привело к взрывному росту сетевого трафика. Еще более критичными для сетей IoT являются огромное количество подключенных устройств и функционирование многих каналов связи в режиме реального времени. Поэтому в IoT, чтобы увеличить пропускную способность сети и максимально использовать пропускную способности канала, должно использоваться прогнозирование трафика.

Для управление такой сетью требуется применение технологии для классификации сетевого трафика без вмешательства оператора. Из-за актуальности данной задачи многочисленные исследования сосредоточены на точной классификации сетевого трафика.

Многие существующие исследования рассматривают разработку алгоритмов машинного обучения для управления трафиком в сетях IoT [347, 348]. Часть существующей литературы рассматривает машинное обучение для классификации трафика IoT [349]. Также во многих существующих работах рассматривается разработка алгоритмов ML для классификации трафика IoT на шлюзе IoT, что позволяет поставщикам сетевых услуг находить несанкционированный трафик и облегчает процесс управления сетью.

Другая часть существующих работ на основе ML для трафика IoT рассматривает прогнозирование сетевого трафика [350]. Такие решения могут предсказать будущее состояние сети IoT за определенный промежуток времени, основываясь на определенных параметрах, например, на основе предыдущего состояния сети. В этом разделе рассматривается недавняя литература, посвященная разработке алгоритмов ML для прогнозирования сетевого трафика IoT. Были рассмотрены работы, которые наиболее близки к предлагаемому в диссертационной работе решению.

В [351] авторы представили новую чувствительную к затратам модель CNN для классификации сетевого трафика, которая облегчает процесс извлечения признаков из сетевого трафика. В работе в основном рассматривалась проблема несбалансированности данных в процессе обучения глубокой нейронной сети.

В данной работе была использована матрица себестоимости для присвоения более высоких значений классам с низкой вероятностью появления и низких значений классам с высокой вероятностью. Эта функция для

определения стоимости позволила достичь высокую точность в процессе классификации. Разработанная модель достигла 98% точности для двухклассовой классификации, при этом только 2% сетевого трафика были неправильно классифицированы. Авторы не рассмотрели оценку реализации предложенной модели и эффективность для высокоплотных и сверхплотных сетей с неоднородным трафиком.

В [352] авторы рассмотрели возможность зондирования трафика из социальных сетей путем извлечения связанных с трафиком микроблогов из платформы SinaWeibo, которая представляет собой наиболее удобный способ извлечения подробной информации о трафике, например, местоположение дорожного инцидента. Данная задача была представлена как классификация коротких текстов, представленная ML-моделями. Авторы разработали модель глубокой нейронной сети для классификации микроблогов на два класса: трафик с высокой значимостью и другие, не имеющие критической значимости. Подход «мешка слов» (CBOW) был применен для обучения представлений встраивания слов из используемого набора данных для трех миллиардов немаркированных микроблогов. Эксперименты показали, что разработанная модель глубокой нейронной сети превосходит как метод опорных векторов (SVM), так и метод многослойного перцептрона (MLP).

В [353] авторы представили наборы данных сетевых пакетов для обучения пяти рассматриваемых глубоких нейронных сетей с использованием сверточных нейронных сетей (CNN) и остаточных сетей (ResNets). Пакеты были превращены в изображения, используемые в качестве входных данных для нейронных сетей. Рассмотренные модели глубокого обучения были использованы для классификации сетевого трафика на основе CNN и ResNets. Кроме того, для получения оптимальных гиперпараметров, обеспечивающих оптимальную производительность модели глубокого обучения, была использована модель

перекрестной валидированной сетки. Результаты показали, что разработанная модель классифицирует сетевой трафик с высокой производительностью.

В [354] авторы исследовали способность нейронной сети классифицировать данные временного ряда с долгосрочной зависимостью от Интернета, используя коэффициент Херста в качестве меры самоподобия. Синтетические данные были сгенерированы с использованием дробного гауссовского шума для моделирования реальных данных. Обученная модель классифицирует как синтетические данные, полученные на основе распределения Парето, так и реальные данные о трафике. При оценке для оптимизации сети использовались различные целевые функции и различное количество сверточных слоев. Результаты показали, что отдельные оптимизаторы достигли сопоставимой производительности; таким образом, обучение моделей с несколькими типами оптимизаторов является хорошей практикой для определения модели с наивысшей точностью.

В [355] авторы разработали метод на основе преобразования данных и CNN для прогнозирования трафика IoT. CNN - это нейронная сеть, которая формирует карту признаков на основе пространственно-временной модели. Кроме того, была разработана еще одна нейронная сеть для сокращения вычислений, необходимых для прогнозирования трафика. Эта нейронная сеть была внедрена для минимизации средних ошибок и повышения точности прогнозирования. Сначала данные обрабатываются для получения необходимых пространственно-временных характеристик. Затем вводится облегченный алгоритм прогнозирования для предсказания трафика в сетях IoT. Он основан на глубоком обучении, и сначала оптимизирует необходимые параметры. Алгоритм был обучен на наборе данных, собранных у реального поставщика услуг, т.е. Telecom Italia, причем 90% данных были использованы для обучения. Оставшаяся часть, 10 %, была использована для тестирования алгоритма.

Данные, представленные в наборе данных, были собраны в течение 50 дней, с интервалом между данными 10 минут. Данная работа отличается от разработанной в диссертации модели, поскольку предложенная в диссертации модель имеет другую структуру CNN с другим методом предварительной обработки данных. Кроме того, предложенный в диссертации метод прогнозирования обучается на двух наборах данных с временным интервалом в 30 минут. Кроме того, предложенный в диссертационной работе метод предусматривает реализацию разработанного алгоритма на туманных распределенных узлах, что увеличивает производительность и эффект процесса прогнозирования.

В [356] авторы разработали модель CNN для классификации Tor трафика. Заголовки пакетов предварительно обрабатывались для преобразования части пакета, т.е. первые 54 байта переводились в десятичный формат и были представлены в качестве входных данных для CNN. Разработанная модель была обучена с использованием набора данных UNB-CIC по трафику Tor, причем 80% данных были использованы для обучения, а 20% - для тестирования. При тестировании система достигла точности 99,3%.

В [357] авторы разработали контролируемый алгоритм глубокого обучения для прогнозирования трафика в сетях IoT. Разработанная модель представляет собой адаптивный градиентный бустинг (GB), основанный на блоках обучения. Такая нейросетевая модель выполняла процесс прогнозирования быстрее и проще благодаря использованию GB. Эта модель была обучена от начала и до конца с использованием набора данных реального трафика, записанного у мобильного оператора. Модель глубокого обучения была обучена с использованием набора данных, собранных от 6214 мобильных пользователей с разнородной активностью в течение 26 дней. Разработанная модель прогнозирует на шесть часов, т.е. шесть выходных прогнозируемых

значений, основанных на временном интервале в один час. К основным преимуществам разработанной модели относятся легкость обучения, скорость прогнозирования и точность предсказанных результатов. Однако в работе не рассматривалась реализация разработанной модели и стоимость вычислений.

В [358] авторы рассмотрели проблему изменения сетевого трафика с помощью CNN. Процесс прогнозирования сетевого трафика превратился в процесс классификации, и для решения подобных задач классификации была введена разработанная модель глубокого обучения. Выходом классификатора являлся один из классов сетевого трафика. CNN была обучена на наборе данных EDU1, причем 80% данных использовалось для обучения, а остальные 20% - для тестирования. Разработанный алгоритм был оценен и валидирован с точностью 92,6%.

В [359] авторы разработали интеллектуальную модель прогнозирования трафика с когнитивным кэшированием для обеспечения прогнозирования в реальном времени в сетях радиодоступа на основе тумана (RAN). Рассматриваемая сетевая структура представляет собой туманную RAN с подключением к опорной сети в режиме реального времени. Модель прогнозирования транспортных потоков была разработана с использованием долговременной краткосрочной памяти (LSTM) и стратегии когнитивного кэширования на основе коллаборативной фильтрации. Разработанная модель была обучена и протестирована на построенном наборе данных, содержащем 5000 исторических журналов, собранных с туманных узлов доступа. Данные состояли из текстовых данных, изображений и видеоданных. Система была оценена, и результаты показали, что разработанная стратегия может точно предсказать тип транспортного потока и эффективно снизить задержку в сети.

В [360] авторы рассмотрели проблему прогнозирования трафика сети IoT с помощью ML. Процесс прогнозирования был основан на трансферном

обучении. В работе предлагалось решение для прогнозирования, основанное на алгоритмах обучения временных рядов, включая рекуррентный блок с управлением (GRU-NN) и LSTM. Разработанный GRU-NN сохранял характеристики трафика сети IoT в течение длительного периода, что позволило системе прогнозировать будущий трафик на основе сохраненного трафика. В работе рассматривалась модель обучения с градиентным усилением для повышения точности прогнозирования трафика IoT и трансферное обучение для устранения барьеров, связанных с низким трафиком зафиксированных данных. Результаты показали, что модель превосходит другие существующие методы прогнозирования трафика по статистическим метрикам оценки эффективности.

Чтобы показать новизну предложенного в диссертационной работе метода по сравнению с существующими, в таблице 6.1 приведено сравнение существующих предложений по классификации и прогнозированию трафика IoT. Новизна разработанного метода заключается в использовании туманных вычислений для реализации модели CNN. Предложенная CNN является достаточно простой и выполняется на распределенных узлах тумана. Насколько известно, это первая работа, в которой рассматривается реализация алгоритма прогнозирования трафика на границе сети доступа, т.е. на туманных узлах.

Таблица 6.1 – Сравнение существующих методов прогнозирования трафика.

Ref.	**year**	**Prediction**	**Classification**	**Traffic**	**Fog**	**Lightweight**	**Imp.**	**KPIs**
[24]	2017	X	√	Social media	X	X	X	Recall, Precision
[28]	2018	X	√	Tor	X	X	X	Recall, Precision
[25]	2019	X	√	General	X	X	X	Recall, Precision

[29]	2019	√	X	IoT	X	X	X	Mean squared error
[33]	2019	√	X	IoT	X	X	X	Packet loss
[30]	2020	√	√	General	X	X	X	Accuracy
[31]	2020	√	X	Cellular	√	X	√	Delay
[26]	2020	X	√	Internet	X	X	X	Accuracy
[23]	2021	X	√	General	X	X	X	Accuracy
[32]	2021	√	X	IoT	X	X	X	Mean squared error
[27]	2021	√	X	IoT	X	√	X	Mean absolute error
Proposed model		√	X	IoT	√	√	√	Accuracy, Mean squared error

6.2 Предлагаемое решение

Предложенная архитектура построения сети IoT представлена на рис. 2. Сеть состоит из четырех уровней: уровень устройств, уровень распределенных граничных вычислений, уровень сети доступа и уровень приложений. Уровень распределенных граничных вычислений введен между оконечными устройствами IoT и сетью доступа для предоставления вычислительных возможностей и энергетических ресурсов в пределах одного транзитного участка. Рассматриваемая технология граничных вычислений представляет собой парадигму туманных вычислений с высокой гибкостью развертывания и уровнем мобильности. Туманные узлы распределяются рядом с оконечными устройствами IoT с ограниченными вычислительными ресурсами и низкой мобильностью.

Слой туманных узлов очень востребован в сценариях высокоплотного и сверхплотного развертывания для достижения необходимой доступности и

масштабируемости сети. Внедрение туманных узлов облегчает реализацию сетевых алгоритмов и методов, включая методы на основе машинного обучения. Методы и алгоритмы управления сетевым трафиком, обработки данных и сбора энергии могут быть реализованы и выполняться туманными узлами вместо оконечных устройств IoT. Это позволяет повысить эффективность сети и сэкономить энергию батареи оконечных устройств. Далее рассматриваются проблемы, связанные с массивным сетевым трафиком и перегрузками в высокоплотных и сверхплотных сетях IoT.

Для прогнозирования сетевого трафика в высокоплотных и сверхплотных сетях IoT разработана модель на основе ИИ. Разработанный алгоритм является достаточно простым и может быть реализован на уровне тумана.

Рисунок 6.2.1 – Предложенная архитектура сети IoT.

Разработанный метод для прогнозирования трафика основан на сверточной нейронной сети (CNN), которая широко используется в приложениях, основанных на извлечении признаков. Поэтому CNN хорошо подходит для приложений прогнозирования трафика. Разработанная сверточная

нейронная сеть CNN - достаточно простая нейронная сеть, которая может быть реализована на распределенных граничных вычислительных узлах, т.е. туманных узлах. Разработанная облегченная CNN для прогнозирования трафика называется LTP-CNN и прогнозирует сетевой трафик в соответствии с текущим состоянием сетевого трафика для заранее определенного интервала времени. На рис. 6.2.1 представлена архитектура сети IoT с распределенными граничными вычислительными узлами с поддержкой LTP-CNN.

LTP-CNN является одномерной CNN, поскольку процессы необходимо проводить с одномерными входными данными, т.е. сегментом пакета. В большинстве существующих литературных источников доказано, что одномерные CNN имеют наивысшую производительность при прогнозировании сетевого трафика. Процесс прогнозирования осуществляется на основе процесса классификации. Сетевой трафик классифицируется, и прогнозирование трафика зависит от количества обнаруженных классов трафика.

Рисунок 6.2.2 – Блок-схема разработанного метода прогнозирования трафика в сети.

На рисунке 6.2.2 представлена блок-схема разработанного метода прогнозирования трафика. Полученные пакеты предварительно обрабатываются и направляются на разработанную CNN, которая извлекает особенности трафика для классификации сетевого трафика по классам. Если происходит совпадение, сетевой трафик классифицируется, а затем делается прогноз на основе количества обнаруженных классов.

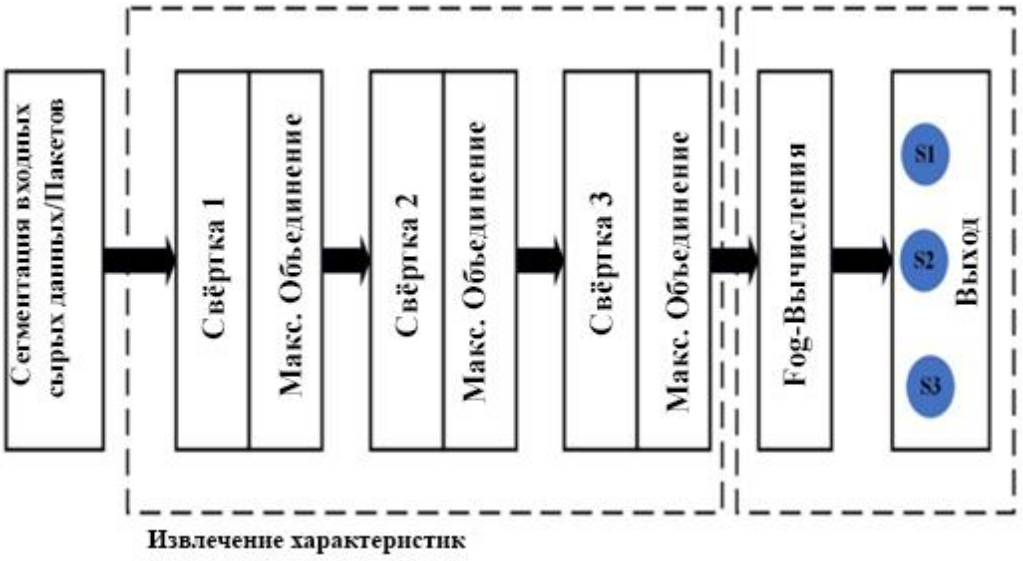


Рисунок 6.2.3. Структура разработанной нейронной сети LTP-CNN.

Разработанная нейронная сеть LTP-CNN классифицирует входной сетевой трафик на основе признаков, которые опираются на исходные данные, т.е. сегмент пакета. Предложенная LTP-CNN состоит из входного слоя, сверточных слоев, слоев объединения, полностью связанных скрытых слоев и выходного слоя. На рисунке 6.2.3 представлена блок-схема разработанной трехслойной LTP-CNN, а в таблице 6.2.1 представлена структура разработанной трехслойной LTP-CNN с указанием размера каждого рассматриваемого сверточного слоя, шага и размера фильтра.

Таблица 6.2.1 – Основные слои разработанной LTP-CNN.

LTP-CNN layer	**Filter**	**Size**	**Stride**

Conv.1	16	3 X 1	2.
Max. Pooling	-		-
Conv.2	32	3 X 1	2.
Max. Pooling	-	-	-
Conv.3	64	3 X 1	2
Max. Pooling	-	-	-
FC	-	-	-

Слой пакетной нормализации вводится для нормализации и предотвращения проблемы переобучения. Процесс пакетной нормализации используется для уменьшения количества эпох, необходимых для обучения глубокой нейронной сети и, таким образом, делает процесс более стабильным. После нормализации данных сети следующий слой становится независимым.

Сверточный слой является инструментом извлечения признаков, который извлекает различные признаки входных данных. Это, в основном, зависит от размера ядра, поскольку общее количество обнаруженных признаков соответствует размеру ядра. Рассматриваемый сверточный слой имеет функцию активации, которая представляет собой выпрямленную линейную единицу (ReLU). ReLU используется для достижения нелинейности модели CNN. Таким образом, предварительное обучение без учителя не требуется. Использование ReLU в качестве функции активации вместо сигмоидной функции активации позволяет быстрее и эффективнее обучать CNN. Более того, ReLU более эффективна в вычислениях, что делает ее эффективной для предложенной нейронной сети LTP-CNN, поскольку встроенный процессор реализует алгоритм с ограниченными возможностями, т.е. шлюз IoT с туманным узлом.

Объединяющий слой вводится для уменьшения количества объектов и вычислений. Данный слой уменьшает пиксельные окна карты признаков до однопиксельных окон. Используются два распространенных типа процесса объединения; max-pooling и average-pooling. Max-pooling - это дискретный

процесс, который применяется для уменьшения выборки входных данных. Это делается путем выбора максимального значения в каждом окне, что делает max-pooling более эффективным для изображений с тусклым фоном и распределенными светлыми пикселями. Однако при усредненном пулинге берется среднее значение всех пикселей в окне, что делает его полезным для извлечения гладких характеристик. Рассматриваемый слой свертки в разработанной LTP-CNN - это max-pooling с размером (2×2). Последний слой свертки обеспечивает извлечение признаков из представленной нейронной сети и подает их на выходной слой.

6.3 Оценка эффективности

В этом разделе оценивается эффективность предложенного метода. Сначала проводится обучение разработанной LTP-CNN, а затем представлен процесс тестирования. В первой части этого раздела представлены рассматриваемые наборы данных и подготовка этих наборов данных для обучения и тестирования CNN. Во второй части рассматривается процесс оценки полученных результатов.

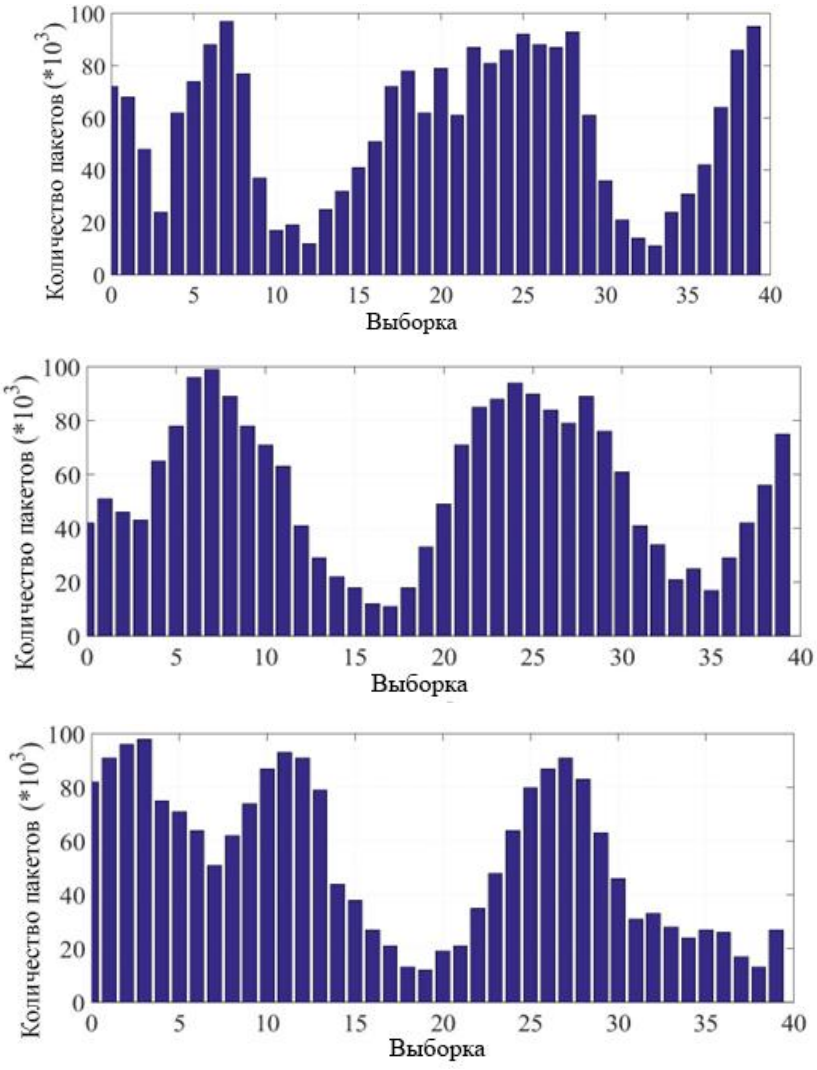
Предварительная обработка набора данных

Были рассмотрены два разнородных набора данных для обучения и тестирования разработанной LTP-CNN. В каждом наборе данных используются программы для анализа пакетов трафика (PCAP). Для обнаружения и сохранения сетевых пакетов в формате PCAP могут быть использованы такие анализаторы пакетов, как TCPDUM и Wireshark.

Первый набор данных (I), представляет собой набор данных, представленный в [362] и доступный в [363]. Он состоит из 1 262 022 захваченных потоков в течение 66 дней. Это содержит более 35 ГБ искусственно созданных пакетов. Второй набор данных (II), представляет собой набор данных

UNI2, представленный в [364] и доступный в [365]. Трассы обоих наборов данных преобразованы в файлы PCAP с временным интервалом в один час.

Для эффективного использования обоих наборов данных для разработанной LTP-CNN файлы PCAP были переведены в массивы данных. Это сделано путем извлечения информации о сетевом трафике из выделенных PCAP-трасс за определенный период и превращения ее в массивы данных. Эти массивы вводятся в LTP-CNN в качестве входных данных. На рис. 6.3 представлены примеры выборки пакетов с интервалом выборки 2 и 4 секунды для обоих рассматриваемых наборов данных.



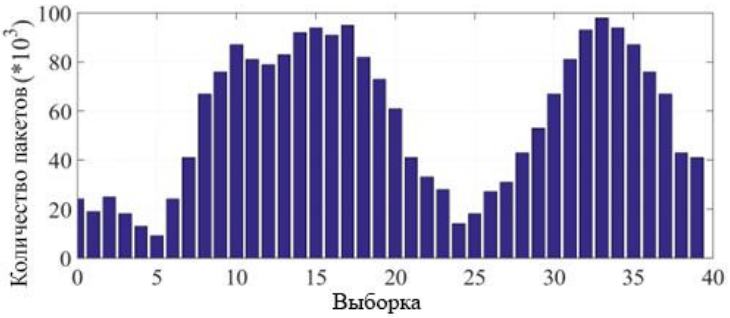


Рисунок 6.3 – Выборочные данные ряда обнаруженных пакетов. (a) и (b) - образцы из набора данных I, а (c) и (d) - образцы из набора данных II.

6.4 Моделирование и результаты

Разработанная LTP-CNN реализована с характеристиками, представленными в таблице параметров моделирования (Таб.3). Экспериментальная оценка проводится с использованием библиотек Keras 2.2.0 в среде Python 3.6. Пакеты наборов данных делятся на 90% для обучения, а остальные 10% - для тестирования. Для обучения используется 3-кратная кросс-валидация, а размер патча представлен в табл. 2. Разработанная LTP-CNN оценивается с помощью двух рассматриваемых наборов данных, а в качестве метрики эффективности используется точность. Точность LTP-CNN рассчитывается как процентное отношение суммы истинно положительных T+, и истинно отрицательных T- к общей совокупности, как представлено в (1).

$$Accuracy = \frac{\sum T^{+}+T^{-}}{\sum T^{+}+T^{-}+F^{+}+F^{-}} \times 100 \qquad (1)$$

Табл. 6.4.1 – Технические характеристики.

Parameter	**Value**
GPU	GeForce RTX 3080
GPU-Accelerated Libraries	NVIDIA CUDA-X
CPU	Intel Core i9
RAM	64GB

На рис. 6.4.1 и рис. 6.4.2 представлены результаты точности I набора данных и II набора данных. Результаты показывают, что точность постепенно увеличивается и стабильно составляет около 90% для обоих наборов данных. Для того чтобы оценить эффективность процесса прогнозирования по сравнению с существующими традиционными CNN, был проведен статистический анализ при сравнении средней квадратической ошибки (MSE) части предсказанных значений LTP-CNN и традиционной CNN. Эти значения представляют собой двадцать реальных предсказанных значений, полученных после применения LTP-CNN и традиционной CNN на туманном узле ранее разработанной тестовой площадки IoT, представленной в [366]. Статистический анализ был проведен с помощью инструмента GraphPad Prism 5 с использованием t-критерия Стьюдента. Рассматривались два основных сценария: в первом сценарии сравнивалась среднеквадратичная ошибка (MSE) для LTP-CNN, обученной один раз на I наборе данных и другой раз на II наборе данных; в другом сценарии сравнивалась среднеквадратичная ошибка (MSE) прогнозирования LTP-CNN с прогнозированием, полученным в соответствии с традиционным методом.

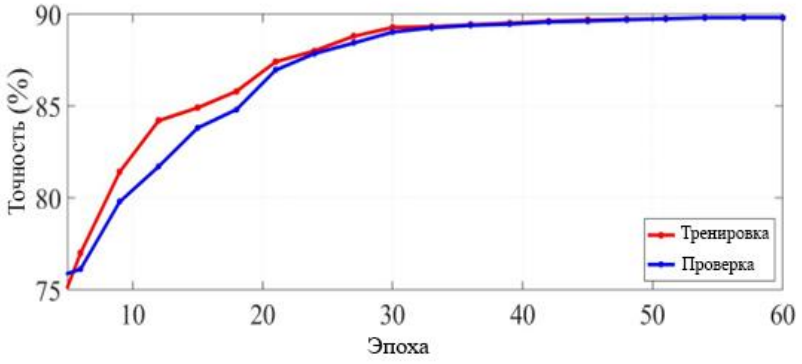


Рисунок 6.4.1 – Точность набора данных (I).

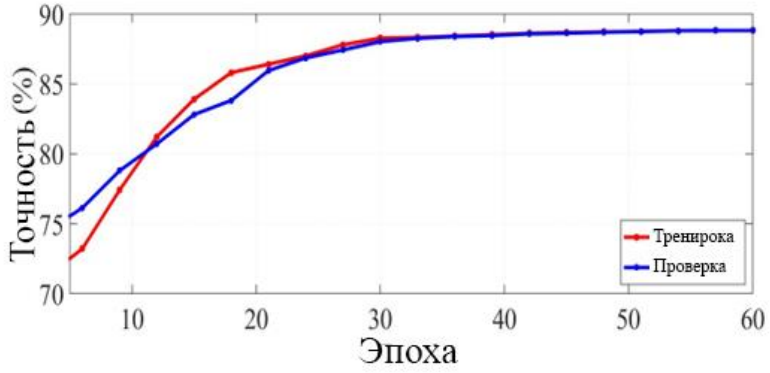


Рисунок 6.4.2 – Точность набора данных (II).

На рис. 6.4.3 и 6.4.4 представлены полученные результаты. В первом сценарии результаты показывают отсутствие существенной разницы между среднеквадратичной ошибки (MSE) для LTP-CNN, обученной на каждом наборе данных, при P-значении 0,1243. Во втором сценарии результаты показывают, что среднеквадратичная ошибка (MSE) предсказанных данных с использованием LTP-CNN значительно отличается от среднеквадратичной ошибки (MSE) для традиционной CNN сети, при P-значении 0,0013. Разработанная LTP-CNN достигает меньшего MSE, и, таким образом, предложенный метод прогнозирует с большей эффективностью, чем существующие.

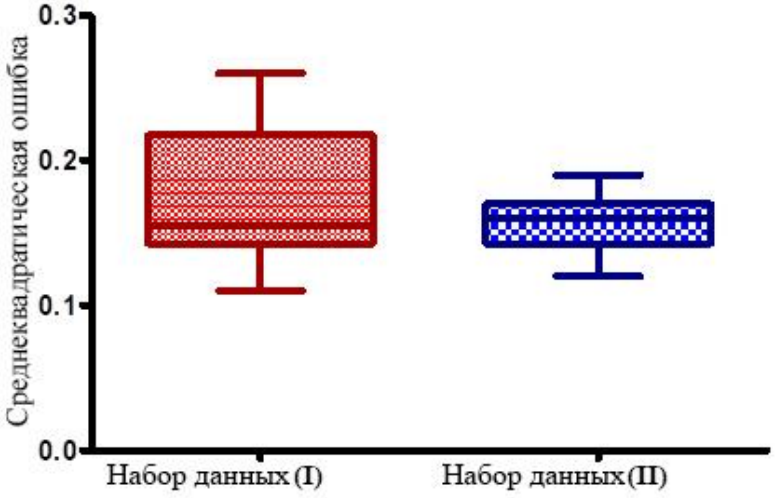


Рисунок 6.4.3 – MSE прогнозных значений с помощью LTP-CNN, обученной на наборах данных I и II.

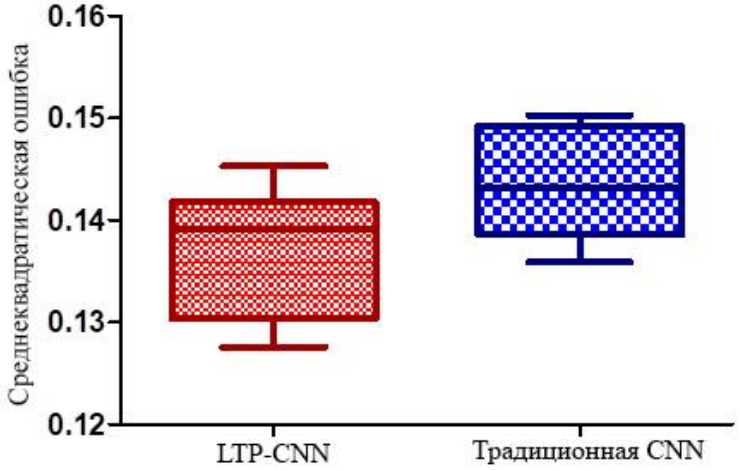


Рисунок 6.4.4 – MSE прогнозных значений с использованием LTP-CNN по сравнению с традиционной CNN.

Для того чтобы оценить эффективность разработанного метода прогнозирования сетевого трафика с точки зрения реализации, был проведен натурный эксперимент. Разработанная LTP-CNN была реализована на туманном узле на ранее разработанной IoT тестовой площадки, представленной в [367]. Кроме того, была использована другая существующая работа, использующая

традиционную нейронную сеть для прогнозирования трафика [21]. Спецификация и параметры натурного эксперимента представлены в таблице 6.4.1.

Таблица 6.4.1 - IoT тестовая площадка и параметры экспериментальной установки.

Параметр	Значение
Число конечных устройств	300
Узел Интернета вещей	Raspberry pi 3
Fog- Processing	Intel®Xeon®CPU E5-2620v4@2.10 GHz, 32 core
Fog-RAM	64 GB
Fog-HDD	500 GB

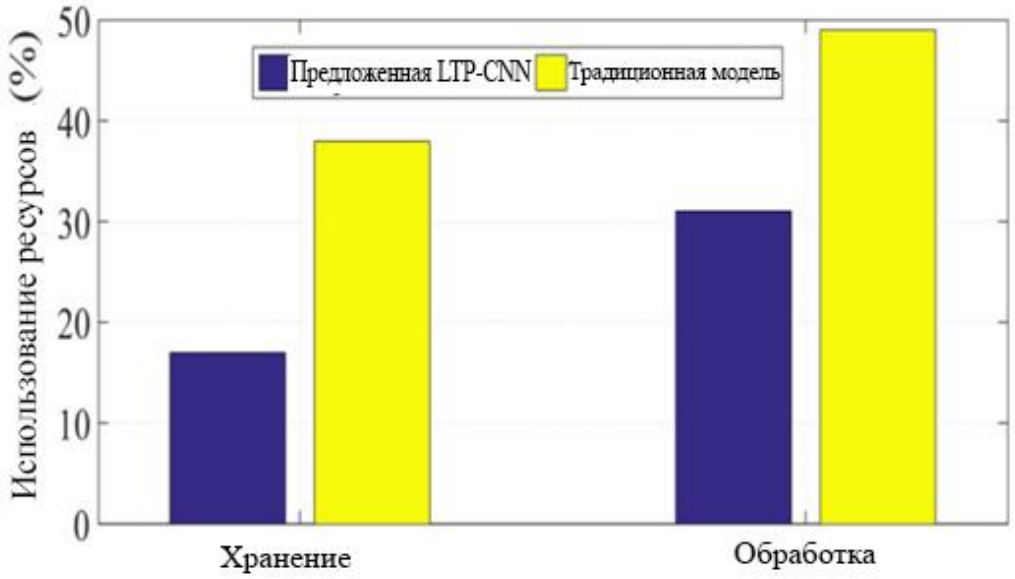


Рисунок 6.4.5 – Процент использования ресурсов LTP-CNN по сравнению с традиционной нейронной сетью CNN.

На рисунке 6.4.5 представлены экспериментальные показатели использования ресурсов узла fog при реализации разработанной LTP-CNN и традиционной нейронной сетью CNN. Здесь представлен процент использования ресурсов хранения для обеих рассматриваемых моделей. Кроме того, представлен процент использования ресурсов обработки. Разработанная модель использует меньше ресурсов хранения в среднем на 21% и на 18% меньше ресурсов обработки для реализации разработанного предиктора, чем традиционные существующие предикторы.

6.5 Выводы по главе 6

Разработан метод прогнозирования трафика на основе CNN - LTP-CNN, который предсказывает трафик сети IoT на базе состояния сети за предыдущий интервал времени, отличающийся от известных тем, что алгоритм прогнозирования реализован на туманных узлах, которые представляют собой основную часть сетей IoT/5G. Результаты показывают, что разработанный LTP-CNN может предсказывать трафик сети IoT с точностью около 90%.

ЗАКЛЮЧЕНИЕ

В диссертационной работе получены следующие основные результаты:

1. предложен метод построения мультиконтроллерной сети, основанный на интегральном решении задач по размещению контроллеров в таких сетях, базирующийся на метаэвристическом (вследствие сложности решаемых задач) алгоритме и алгоритме балансировки нагрузки, позволяющем обеспечить наилучшее использование ресурсов контроллеров.

2. предложено использовать иерархическую кластеризацию такой сети, включающую в себя кластеры с головными узлами и централизованный контроллер, что обеспечивает балансировку нагрузки в разработанном методе построения сети. В-третьих, разработан модифицированный алгоритм CSSA для использования в иерархических кластерных сетях clus-CSSA.

3. Разработанный метод построения мультиконтроллерной сети позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1 до 10 мс по сравнению как с широко известными метаэвристическими алгоритмами PSO и GWO, так и с предыдущей версией CSSA. При этом для наиболее сложного случая задержки величиной в 1 мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза. В дальнейшем планируется реализовать алгоритм CSSA для сети Интернета Вещей высокой плотности.

4. предложена модель энергоэффективной многопользовательской системы основанной на использовании технологии периферийных вычислений мультисервисного доступа с поддержкой беспилотных летательных аппаратов. Эта система масштабируема и может поддерживать увеличение сетевого трафика без снижения производительности.
5. Развернута сеть беспилотных летательных аппаратов для покрытия мертвых зон и зон с высокой плотностью сети. Кроме этого, беспилотные летательные аппараты могут предоставлять другие приложения в сети "умного города", в которой технология MEC развернута на нескольких уровнях для обеспечения вычислительных возможностей на границе сети. Базовая сеть основана на технологии SDN для возможности гибкого управления сетевым трафиком и предоставления инновационного интерфейса сетевым операторам.
6. Имитационные эксперименты демонстрируют, что предложенная нами модель может значительно снизить энергопотребление всей системы.
7. Решена научная проблема, отличающаяся от известных тем, что предложены модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности, основанные на интегральном решении задач по размещению граничных серверов на беспилотных летательных аппаратах и оптимизации структуры сети с использованием метаэвристического алгоритма роя сальп.
8. Для решения научной задачи интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности разработана модель сети, отличающаяся от

известных тем, что предложено для уменьшения задержки и энергопотребления в такой сети использовать мобильные серверы граничных вычислений на БПЛА, а связь между наземным и летающим сегментом обеспечивать с использованием NOMA.

9. Для решения задачи минимизации задержки и энергопотребления в разработанной модели предложен метод выгрузки трафика с наземной сети на мобильные серверы граничных вычислений на БПЛА, отличающийся от известных тем, что процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, который определяет сложность вычисляемой задач и по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА.

10. Для решения научной задачи оптимизации структуры сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности в отличии от известных решений для минимизации задержки и энергопотребления при выгрузке трафика с наземной сети на серверы граничных вычислений БПЛА разработан метаэвристический алгоритм на основе хаотического роя сальп.

11. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений и на 30-40% по

сравнению с сетью с использованием только наземных граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш около 10% по сравнению с использованием неоптимизированного алгоритма.

12. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение энергопотребления до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш в 5-10% по сравнению с использованием неоптимизированного алгоритма.

13. Результаты моделирования доказали, что разработанные модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение доли заблокированных задач по выгрузке трафика в десятки раз по сравнению с сетью без использования технологий граничных вычислений, в разы по сравнению с сетью с использованием только наземных граничных вычислений. Использование оптимизации на основе метаэвристического хаотического роя сальп не дает практически значимого эффекта по сравнению с неоптимизированным алгоритмом.

14. Определены зависимости значений задержки, энергопотребления и доли заблокированных задач по выгрузке трафика.

15. Метод размещения SDN-контроллеров в мультиконтроллерных сетях на мобильных узлах сетей VANET для обеспечения связи в плотных и сверхплотных сетях 6G и взаимодействия с туманной средой устройств сети

16. метод построения сети с использованием технологий MEC, SDN и D2D для поддержки приложений беспилотных автомобилей. Предлагаемая архитектура направлена на преодоление двух основных проблем автомобильных сетей – высокой плотности трафика и наличия непокрытых зон в автомобильной сети связи. При этом, разработан также алгоритм кластеризации на основе взаимодействий D2D для транспортных средств в непокрытых зонах и для выгрузки трафика сети в регионах с интенсивным движением. Результаты моделирования показывают, что предложенная архитектура дает 74% прироста производительности системы в терминах вероятности блокировки задач.

СПИСОК ЛИТЕРАТУРЫ

1. Ateya, A. A. Energy efficient offloading scheme for MEC-based augmented reality system / Ateya, A. A., Muthanna, A., Koucheryavy, A., Maleh, Y., & El-Latif, A. A. A. // Cluster Computing, 26(1), 2023, 789-806.
2. Schäfer A. A Survey on Synchronous Augmented, Virtual and Mixed Reality Remote Collaboration Systems / A. Schäfer, G. Reis, D. Stricker // ACM Comput. Surv. (CSUR) – 2021.
3. Маколкина М.А. приложения дополненной реальности в "умных городах" / Маколкина М.А., Бородин А.С., Мутханна А.С., Кучерявый А.Е. // Электросвязь. 2019. № 12. С. 44-50..
4. Roopa D. Revolutionizing education system with interactive augmented reality for quality education / D. Roopa, R. Prabha, G. Senthil // Mater. Today Proc. – 2021, № 46. – P. 3860 – 3863.
5. Кучерявый А.Е. Модельная сеть для исследований и обучения в области услуг телеприсутствия / Кучерявый А.Е., Маколкина М.А., Парамонов А.И., Выборнова А.И., Мутханна А.С., Матюхин А.Ю., Дунайцев Р.А., Владимиров С.С., Ворожейкина О.И., Захаров М.В., Фам В.Д., Марочкина А.В., Горбачева Л.С., Паньков Б.О., Анваржонов Б.Н. // Электросвязь. 2022. № 1. С. 14-20.
6. Muthanna A. AR enabled system for cultural heritage monitoring and preservation / A. Muthanna, A.A. Ateya, A. Amelyanovich, M. Shpakov, P. Darya, M. Makolkina // In Internet of Things, Smart Spaces, and Next Generation Networks and Systems // Springer: Berlin/Heidelberg, Germany, – 2018; – P. 560 – 571.
7. Волков А.Н. Перспективные исследования сетей и услуг 2030 в лаборатории 6G MEGANETLAB СПБГУТ / Волков А.Н., Мутханна А.С.А., Кучерявый А.Е., Бородин А.С., Парамонов А.И., Владимиров С.С., Фокин Г.А., Дунайцев Р.А., Захаров М.В., Горбачева Л.С., Паньков Б.О., Анваржонов Б.Н. // Электросвязь. 2023. № 6. С. 5-14.
8. Aliprantis J. A survey of augmented reality applications in cultural heritage / J. Aliprantis, G. Caridakis // Int. J. Comput. Methods Herit. Sci. (IJCMHS) – 2019, № 3. – P. 118–147.
9. Rejeb A. How augmented reality impacts retail marketing: A state-of-the-art review from a consumer perspective / A. Rejeb, K. Rejeb, H. Treiblmaier // J. Strateg. Mark. – 2021, – P. 1–31.

10. Yoo, J. The effects of perceived quality of augmented reality in mobile commerce—An application of the information systems success model. // Informatics. – 2020, № 7. – P. 14.
11. Alshahrani, A. Efficient multi-player computation offloading for VR edge-cloud computing systems / Alshahrani, A., Elgendy, I. A., Muthanna, A., Alghamdi, A. M., & Alshamrani, A // Applied Sciences, 10(16), 2020, 5515.
12. Szajna A. The production quality control process, enhanced with augmented reality glasses and the new generation computing support system / A. Szajna, R. Stryjski, W. Woźniak, N. Chamier-Gliszczyński, T. Królikowski, T. // Procedia Comput. Sci. – 2020, № 176. – P. 3618 – 3625.
13. Manuri F. A survey on applications of augmented reality / F. Manuri, A. Sanna // ACSIJ Adv. Comput. Sci. Int. J. – 2016, № 5. – P. 18 – 27.
14. Tsai T.H. Research study on applying SLAM-Based Augmented Reality technology for gamification history guided tour. In Proceedings of the 2019 / T.H. Tsai, Y.W. Chiang, // IEEE International Conference on Architecture, Construction, Environment and Hydraulics (ICACEH), – Xiamen, China, 20–22 December – 2019. – P. 116 – 119.
15. Mao C.C. Augmented reality of 3D content application in common operational picture training system for army / C.C. Mao, C.H. Chen // Int. J. Hum. Comput. Interact. – 2021, № 37. – P. 1899 – 1915.
16. Zaman F. Investigating a Virtual Reality Based Subterranean Scenario Examining Augmented Reality Implications for Military Operators. In Proceedings of the Human Factors and Ergonomics Society Annual Meeting / F. Zaman, W. Drake, J. Intriligator, A. Gardony, M. Natick, J. Rife // SAGE Publications Sage CA: Los Angeles, CA, USA, – 2021; – Vol. 65, – P. 1129 – 1133.
17. Jones D. Characterising the Digital Twin: A systematic literature review / D. Jones, C. Snider, A. Nassehi, J. Yon, B. Hicks // CIRP Journal of Manufacturing Science and Technology, – 2020, № 29. – P. 36 – 52.
18. Lu Y. Adaptive edge association for wireless digital twin networks in 6G /Y. Lu, S. Maharjan, Y. Zhang // IEEE Internet of Things Journal, – 2021. № 8(22). – P. 16219 – 16230.
19. Ahmadi H. Networked twins and twins of networks: An overview on the relationship between digital twins and 6G / H. Ahmadi, A. Nag, Z. Khar, K. Sayrafian, S. Rahardja // IEEE Communications Standards Magazine, – 2021. № 5(4). – P. 154 – 160.

20. Kuruvatti, N. P., Habibi, M. A., Partani, S., Han, B., Fellan, A., & Schotten, H. D. Empowering 6G Communication Systems With Digital Twin Technology: A Comprehensive Survey / Kuruvatti, N. P., Habibi, M. A., Partani, S., Han, B., Fellan, A., & Schotten, H. D. // IEEE Access, – 2022.
21. Masaracchia A. Digital Twin for 6G: Taxonomy, Research Challenges, and the Road Ahead / A. Masaracchia, V. Sharma, B. Canberk, O.A. Dobre, T.Q. Duong // IEEE Open Journal of the Communications Society, – 2022.
22. Allam Z. Future (post-COVID) digital, smart and sustainable cities in the wake of 6G: Digital twins, immersive realities and new urban economies. / Z. Allam, D.S. Jones // Land use policy – 2021. 101, – P. 105201.
23. Elzanaty A. Towards 6G holographic localization: Enabling technologies and perspectives / A. Elzanaty, A. Guerra, F. Guidi, D. Dardari, M.S. Alouini // arXiv preprint arXiv:2103.12415. – 2021.
24. Long W. A promising technology for 6G wireless networks: Intelligent reflecting surface / W. Long, R. Chen, M. Moretti, W. Zhang, J. Li // Journal of Communications and Information Networks, – 2021. № 6(1). – P. 1 – 16.
25. Kurniawan C. Pengembangan Model Pembelajaran 3D Display System Berbasis Holografi // Sinteks: Jurnal Teknik, – 2016. № 5(2).
26. Huang C. M. Holographic MIMO surfaces for 6G wireless networks: Opportunities, challenges, and trends / C. Huang, S. Hu, G.C. Alexandropoulos, A. Zappone, C. Yuen, R. Zhang, M. Debbah // IEEE Wireless Communications, – 2022. № 27(5). – P. 118-125.”.
27. Huang C. Holographic MIMO surfaces for 6G wireless networks: Opportunities, challenges, and trends. IEEE Wireless Communications / C. Huang, S. Hu, G.C. Alexandropoulos, A. Zappone, C. Yuen, R. Zhang, M. Debbah // – 2020. № 27(5). – P. 118 – 125.
28. Elmeadawy, S., & Shubair, R. M. 6G wireless communications: Future technologies and research challenges. In 2019 international conference on electrical and computing technologies and applications (ICECTA). // IEEE. – 2019. March, – P. 1 – 5.
29. Beniiche A. 5.0: Internet as if People Mattered. IEEE Wireless Communications / A. Beniiche, S. Rostami, M. Maier // – 2022. № 29(6). – P. 160 – 168.
30. Maier, M. Toward 6G: A new era of convergence. In Optical Fiber Communication Conference. // Optica Publishing Group. – 2021. June, – P. F4H-1.

31. Maier M. The Internet of No Things: Making the Internet Disappear and “See the Invisible / M. Maier, A. Ebrahimzadeh, S. Rostami, A. Beniiche // IEEE Communications Magazine, – 2020. № 58(11). – P. 76-82.
32. Maier, M. 6G as if people mattered: From industry 4.0 toward society 5.0. In 2021 International Conference on Computer Communications and Networks (ICCCN). // IEEE. – 2021. July, – P. 1 – 10.
33. Мутханна А.С. Метод размещения SDN-контроллеров на мобильных узлах сетей VANET для высокоплотных и сверхплотных сетей 6G / Мутханна А.С // Электросвязь. 2023. № 8. С. 19-27.
34. Chen, N. Toward 6G internet of things and the convergence with RoF system / N. Chen, M. Okada // IEEE Internet of Things Journal, – 2020. № 8(11). – P. 8719 – 8733.
35. Vaigandla, K. K. Communication Technologies and Challenges on 6G Networks for the Internet: Internet of Things (IoT) Based Analysis. // In 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM) – 2022. February, – P. 27 – 31.
36. Chowdhury M.Z. The role of optical wireless communication technologies in 5G/6G and IoT solutions: Prospects, directions, and challenges / M.Z. Chowdhury, M. Shahjalal, M. K. Hasan, Y. M. Jang // Applied Sciences, – 2019. № 9(20). – P. 4367.
37. Владыко А.Г. выгрузки трафика В V2X/5G сетях на основе системы граничных вычислений / Владыко А.Г., Мутханна А.С., Кучерявый А.Е. Метод // Электросвязь. 2020. № 8. С. 24-30
38. Мутанна М.С. Метод глубокого обучения с подкреплением для систем интернета вещей на базе технологии lora с ограниченными ресурсами и поддержкой QOS / Мутанна М.С., Мутханна А.С., Бородин А.С. // Электросвязь. 2021. № 8. С. 23-26.
39. Hu S. Blockchain and artificial intelligence for dynamic resource sharing in 6G and beyond / S. Hu, Y.C. Liang, Z. Xiong, D. Niyato // IEEE Wireless Communications. – 2021. № 28(4). – P. 145 – 151.
40. Lu Y. 6G: A survey on technologies, scenarios, challenges, and the related issues / Y. Lu, X. Zheng // Journal of Industrial Information Integration, – 2020. 19, 100158.
41. Peng, H. 6G toward Metaverse: Technologies, Applications, and Challenges. / P.C. Chen, P.H. Chen, Y.S. Yang, C.C. Hsia, L.C. Wang // In 2022 IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS) // IEEE. – 2022. August, – P. 6 –10

42. Kharche S. /Interoperability Issues and Challenges in 6G Networks / S. Kharche, P.Dere // Journal of Mobile Multimedia. – 2022. № 18(5). – P. 1445 – 1470.
43. ITU-R Recommendation M.2083-0. IMT Vision, Framework and overall objectives of the future development of IMT for 2020 and beyond: ITU-R, – Sep. 2015.
44. International Congress on Ultra Modern Telecommunications and Control Systems and Workshops / A.A. Ateya, A. Muthanna, A. Koucheryavy, M. Khayyat // Toward Tactile Internet, 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT 2019). – 2019. – P. 8970990.
45. ITU-T Recommendation Y.3104. Architecture of the IMT-2020 network. ITU-T. – Geneva. – December, 2018.
46. Бородин А. С. Искусственный интеллект в сетях связи пятого и последующих поколений / А. Н. Волков, А. С. Мутханна, А. Е. Кучерявый // Электросвязь. – № 1. – 2021. – С. 17-22.
47. Мутханна А.С. Интеллектуальная распределенная архитектура сети связи для поддержки беспилотных автомобилей / Мутханна А.С. // Электросвязь. 2020. № 7. С. 29-34.
48. Patel M. et al. // Contributing Organizations and Authors. – no. 1, – P. 36.
49. Khan A. ur R. A Survey of Mobile Cloud Computing Application Models / A. ur R. Khan, M. Othman, S.A. Madani, S.U. Khan // IEEE Commun. Surv. Tutor., vol. 16, no. 1, – 2014. – P. 393–413.
50. Sanaei Z. Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges / Z. Sanaei, S. Abolfazli, A. Gani, R. Buyya // IEEE Commun. Surv. Tutor., vol. 16, no. 1, – 2014. – P. 369 – 392.
51. Brummett T. Performance Metrics of Local Cloud Computing Architectures / T. Brummett, P. Sheinidashtegol, D. Sarkar, and M. Galloway // in 2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing, New York, NY, USA, – 2015. – P. 25–30.
52. Zhao T. A Cooperative Scheduling Scheme of Local Cloud and Internet Cloud for Delay-Aware Mobile Cloud Computing / T. Zhao, S. Zhou, X. Guo, Y. Zhao, Z. Niu // in 2015 IEEE Globecom Workshops (GC Wkshps), San Diego, CA, USA, – 2015. – P. 1–6.
53. Liu Y Adaptive Multi-Resource Allocation for Cloudlet-Based Mobile Cloud Computing System / Y. Liu, M.J. Lee, Y. Zheng // IEEE Trans. Mob. Comput., vol. 15, no. 10, – Oct., 2016. – P. 2398–2410.

54. El-Barbary A.E.-H.G. A cloudlet architecture using mobile devices / A.E.-H.G. El-Barbary, L.A.A. El-Sayed, H.H. Aly, M.N. El-Derini // in 2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA), Marrakech, – 2015. – P. 1–8.
55. Verbelen T. Cloudlets: bringing the cloud to the mobile user / T. Verbelen, P. Simoens, F. De Turck, B. Dhoedt, // in Proceedings of the third ACM workshop on Mobile cloud computing and services - MCS ’12, Low Wood Bay, Lake District, UK, – 2012. – P. 29.
56. Pang Z. A Survey of Cloudlet Based Mobile Computing / Z. Pang, L. Sun, Z. Wang, E. Tian, S. Yang // in 2015 International Conference on Cloud Computing and Big Data (CCBD), Shanghai, China, – 2015. – P. 268–275.
57. Taherkordi A. Data-Centric IoT Services Provisioning in Fog-Cloud Computing Systems: Poster Abstract / A. Taherkordi, F. Eliassen // in Proceedings of the Second International Conference on Internet-of-Things Design and Implementation - IoTDI ’17, Pittsburgh, PA, USA, – 2017. – P. 317–318.
58. Gandotra P. Green Communication in Next Generation Cellular Networks: A Survey / P. Gandotra, R.K. Jha, S. Jain // IEEE Access, vol. 5, – 2017. – P. 11727 – 11758.
59. Beck M.T. Mobile Edge Computing: A Taxonomy / M. T. Beck, M. Werner, S. Feld, and T. Schimper, // – 2014. – P. 7.
60. Wu J. Cloud radio access network (C-RAN): a primer / J. Wu, Z. Zhang, Y. Hong, Y. Wen // IEEE Netw., vol. 29, no. 1, 2015. – P. 35–41.
61. Luan T.H. Fog Computing: Focusing on Mobile Users at the Edge / T.H. Luan, L. Gao, Z. Li, Y. Xiang, G. We, L. Sun // – P. 11.
62. Mao Y. A Survey on Mobile Edge Computing: The Communication Perspective / Y. Mao, C. You, J. Zhang, K. Huang, K.B. Letaief // IEEE Commun. Surv. Tutor., vol. 19, no. 4, – 2017. – P. 2322 – 2358.
63. Piumsomboon T. User-Defined Gestures for Augmented Reality / T. Piumsomboon, A. Clark, M. Billinghurst, // – P. 6.
64. Azuma R. Recent advances in augmented reality / R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, B. MacIntyre, // IEEE Comput. Graph. Appl., vol. 21, no. 6, – Dec., 2001. – P. 34 – 47.

65. Taleb T. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration / T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, // IEEE Commun. Surv. Tutor., vol. 19, no. 3, – 2017. – P. 1657 – 1681.
66. Taleb T. Anything as a Service' for 5G Mobile Systems / T. Taleb, A. Ksentini, R. Jantti // IEEE Netw., vol. 30, no. 6, – Nov. 2016. – P. 84–91.
67. Liu J. Delay-optimal computation task scheduling for mobile-edge computing systems / J. Liu, Y. Mao, J. Zhang, and K. B. Letaief // in 2016 IEEE International Symposium on Information Theory (ISIT), Barcelona, Spain, – 2016, – P. 1451 – 1455.
68. Chen M. Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network / M. Chen, Y. Hao // IEEE J. Sel. Areas Commun., vol. 36, no. 3, – Mar., 2018. – P. 587 – 597.
69. Liu J. Offloading Schemes in Mobile Edge Computing for Ultra-Reliable Low Latency Communications / J. Liu, Q. Zhang // IEEE Access, vol. 6, – 2018. – P. 12825 – 12837.
70. Yang L. Multi-User Computation Partitioning for Latency Sensitive Mobile Cloud Applications / L. Yang, J. Cao, H. Cheng, Y. Ji // IEEE Trans. Comput., vol. 64, no. 8, – Aug., 2015. – P. 2253 – 2266.
71. Kim Y. Dual-Side Optimization for Cost-Delay Tradeoff in Mobile Edge Computing / Y. Kim, J. Kwak, and S. Chong // IEEE Trans. Veh. Technol., vol. 67, no. 2, – Feb., 2018. –P. 1765 – 1781.
72. Hu X. Wireless Powered Cooperation-Assisted Mobile Edge Computing / X. Hu, K.-K. Wong, K. Yang // IEEE Trans. Wirel. Commun., vol. 17, no. 4, – Apr., 2018. – P. 2375 – 2388.
73. Lorenzo P. D. Joint Optimization of Radio Resources and Code Partitioning in Mobile Edge Computing / P. D. Lorenzo, S. Barbarossa, and S. Sardellitti // – P. 13.
74. Trinh H. et al., "Energy-Aware Mobile Edge Computing and Routing for Low-Latency Visual Data Processing // IEEE Trans. Multimed., vol. 20, no. 10, – Oct., 2018. – P. 2562 – 2577.
75. Yang L. Mobile Edge Computing Empowered Energy Efficient Task Offloading in 5G / L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji // IEEE Trans. Veh. Technol., vol. 67, no. 7, – Jul., 2018. – P. 6398 – 6409.
76. Thinh T.Q. Offloading in Mobile Edge Computing: Task Allocation and Computational Frequency Scaling / T. Q. Thinh, J. Tang, Q. D. La, and T.Q.S. Quek // IEEE Trans. Commun., – 2017. – pp.3571-3584.

77. Мутханна А.С. Модель интеграции граничных вычислений в структуру сети «воздух–земля» и метод выгрузки трафика для сетей Интернета Вещей высокой и сверхвысокой плотности. Труды учебных заведений связи/ Мутханна А.С // 2023;9(3):42-59
78. Chen X. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing / X. Chen, L. Jiao, W. Li, and X. Fu // IEEEACM Trans. Netw., vol. 24, no. 5, – Oct., 2016. – P. 2795 – 2808.
79. Yu S. Computation offloading for mobile edge computing: A deep learning approach / S. Yu, X. Wang R. Langar // in 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, QC, – 2017. – P. 1 – 6.
80. Bi S. Computation Rate Maximization for Wireless Powered Mobile-Edge Computing With Binary Computation Offloading / S. Bi, Y. J. Zhang // IEEE Trans. Wirel. Commun., vol. 17, no. 6, – Jun., 2018. – P. 4177–4190.
81. Zhou T. Joint Cell Activation and Selection for Green Communications in Ultra-Dense Heterogeneous Networks / T. Zhou, N. Jiang, Z. Liu, C. Li // IEEE Access, vol. 6, – 2018. – P. 1894 – 1904.
82. Beyranvand H. Backhaul-Aware User Association in FiWi Enhanced LTE-A Heterogeneous Networks / H. Beyranvand, W. Lim, M. Maier, C. Verikoukis, J. A. Salehi // IEEE Trans. Wirel. Commun., vol. 14, no. 6, – Jun., 2015. – P. 2992 – 3003.
83. Luo X. Delay-Oriented QoS-Aware User Association and Resource Allocation in Heterogeneous Cellular Networks // IEEE Trans. Wirel. Commun., vol. 16, no. 3, – Mar., 2017. – P. 1809 – 1822.
84. F. Wang Joint Optimization of User Association, Subchannel Allocation, and Power Allocation in Multi-Cell Multi-Association OFDMA Heterogeneous Networks / F. Wang, W. Chen, H. Tang, Q. Wu // IEEE Trans. Commun., vol. 65, no. 6, – Jun., 2017. – P. 2672 – 2684.
85. Mukherjee M. Survey of fog computing: Fundamental, network applications, and research challenges[J] / M. Mukherjee, L. Shu, D. Wang // IEEE Communications Surveys & Tutorials, – 2018. 20(3): – P. 1826-1857.
86. Palattella M. Internet of Things in the 5G Era: Enablers, Architecture and Business Models[J]. / M. Palattella, M. Dohler, A. Grieco, et al. // IEEE Journal on Selected Areas in Communications, – 2016. 34(3): – P. 510 – 527.

87. Abolfazli S. Cloud-Based Augmentation for Mobile Devices: Motivation, Taxonomies, and Open Challenges[J] / S. Abolfazli, Z. Sanaei, E. Ahmed, et al. // IEEE Communications Surveys Tutorials, – 2014, 16(1): – P. 337 – 368.
88. Vallina-Rodriguez N. Energy management techniques in modern mobile handsets[J]. / N. Vallina-Rodriguez, J. Crowcroft // IEEE Communications Surveys & Tutorials, – 2012, 15(1): – P. 179 – 198.
89. Yadav R. Adaptive energy-aware algorithms for minimizing energy consumption and SLA violation in cloud computing[J]. / R. Yadav, W. Zhang, O. Kaiwartya, et al. // IEEE Access, – 2018. 6: – P. 55923 – 55936.
90. Alam T. A reliable communication framework and its use in internet of things (IoT)[J]. CSEIT1835111, – 2018. 10: – P. 450 – 456.
91. Kumar K. Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?[J]. / K. Kumar, Y.H. Lu // Computer, – 2010. 43(4): – P. 51 – 56.
92. Othman M. A survey of mobile cloud computing application models[J]. / M. Othman, S.A. Madani, S.U. Khan, et al. // IEEE communications surveys & tutorials, – 2013. 16(1): – P. 393 – 413.
93. Elgendy I. A. An efficient and secured framework for mobile cloud computing[J]. / I.A. Elgendy, W-Z Zhang, C-Y Liu, et al. //IEEE Transactions on Cloud Computing, – 2018. 9(1): – P. 79 – 87.
94. Noor T.H. Mobile cloud computing: Challenges and future research directions[J]. / T.H. Noor, S. Zeadally, A. Alfazi, et al. // Journal of Network and Computer Applications, – 2018. 115: – P. 70-85.
95. Mollah M.B. Security and privacy challenges in mobile cloud computing: Survey and way ahead[J]. / M.B. Mollah, M.A.K. Azad, A. Vasilakos. // Journal of Network and Computer Applications, – 2017. 84: – P. 38-54.
96. Mao Y. A survey on mobile edge computing: The communication perspective[J]. / Y. Mao, C. You, J. Zhang, et al. // IEEE Communications Surveys & Tutorials, – 2017. 19(4): – P. 2322 – 2358.
97. Wang J. Edge cloud offloading algorithms: Issues, methods, and perspectives[J]. / J. Wang, J. Pan, F. Esposito, et al. // ACM Computing Surveys (CSUR), – 2019. 52(1): – P. 1 – 23.

98. Satyanarayanan M. The case for vm-based cloudlets in mobile computing[J]. / M. Satyanarayanan, P. Bahl, R. Caceres, et al. // IEEE pervasive Computing, – 2009. 8(4): – P. 14 – 23.
99. Yaqoob I. Mobile ad hoc cloud: A survey[J]. / I. Yaqoob, E. Ahmed, A. Gani, et al. //Wireless Communications & Mobile Computing, – 2016, 16(16): – P. 2572 – 2589.
100. Cong S. Serendipity: Enabling Remote Computing among Intermittently Connected Mobile Devices[C] / S. Cong, V. Lakafosis, M.H. Ammar, et al. // Acm Mobihoc. – 2012.
101. Mach P. Mobile edge computing: A survey on architecture and computation offloading[J]. / P. Mach, Z. Becvar // IEEE Communications Surveys & Tutorials, – 2017. 19(3): – P. 1628 – 1656.
102. Liu F. A survey on edge computing systems and tools[J]. / F. Liu, G. Tang, Y. Li, et al. // Proceedings of the IEEE, – 2019. 107(8): – P.1537 – 1562.
103. Satria D. Recovery for overloaded mobile edge computing[J]. / D. Satria, D. Park, M. Jo // Future Generation Computer Systems, – 2017. 70: – P. 138 – 147.
104. Georgakopoulos D. Internet of Things and edge cloud computing roadmap for manufacturing[J]. / D. Georgakopoulos, P. P. Jayaraman, M. Fazia, et al. // IEEE Cloud Computing, – 2016. 3(4): – P. 66 – 73.
105. Абделлах А.Р. Применение робастных m-оценок для машинного обучения в сетях VANET / Абделлах А.Р., Мутханна А., Кучерявый А.Е // Электросвязь. 2020. № 5. С. 41-46.
106. Zhang K. Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks[J]. / K. Zhang, Y. Mao, S. Leng, et al. // IEEE access, – 2016. 4: – P. 5896 – 5907.
107. Sheng Z. Energy efficient cooperative computing in mobile wireless sensor networks[J]. / Z. Sheng, C. Mahapatra, V.C. Leung, et al. // IEEE Transactions on Cloud Computing, – 2015. 6(1): – P.114 – 126.
108. Kao Y-H. Hermes: Latency optimal task assignment for resource-constrained mobile computing[J]. / Y-H. Kao, B. Krishnamachari, M-R. Ra, et al. // IEEE Transactions on Mobile Computing, – 2017. 16(11): – P. 3056 – 3069.
109. Ren J. Latency optimization for resource allocation in mobile-edge computation offloading[J]. / J. Ren, G. Yu, Y. Cai, et al. // IEEE Transactions on Wireless Communications, – 2018. 17(8): – P. 5506-5519.

110. Khalili S. Inter-layer per-mobile optimization of cloud mobile computing: a message-passing approach[J]. / S. Khalili, Simeone O. // Transactions on Emerging Telecommunications Technologies, – 2016. 27(6): – P. 814 – 827.
111. Mao Y, Zhang J, Song S, et al. Power-delay tradeoff in multi-user mobile-edge computing systems[C] / Y. Mao, J. Zhang, S. Song, et al. // 2016 IEEE Global Communications Conference (GLOBE- COM). – 2016. – P. 1 – 6.
112. Liu L. Multi-objective optimization for computation offloading in mobile-edge computing[C] / L. Liu, Z. Chang, X. Guo, et al. // 2017 IEEE Symposium on Computers and Communications (ISCC). – 2017. – P. 832 – 837.
113. Xu X. Multi-objective computation offloading for internet of vehicles in cloud-edge computing[J]. / X. Xu, R. Gu, F. Dai, et al. // Wireless Networks, – 2020. 26(3): – p. 1611 – 1629.
114. Zhang X. Multi-objective resource allocation for mobile edge computing systems[C] / X. Zhang, Y. Mao, J. Zhang, et al. // 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). – 2017. – P. 1 – 5.
115. Yu Y. Joint subcarrier and CPU time allocation for mobile edge computing[C] / Y. Yu, J. Zhang, K.B. Letaief // 2016 IEEE Global Communications Conference (GLOBE- COM). – 2016: – P. 1 – 6.
116. Liu J. Delay-optimal computation task scheduling for mobile-edge computing systems[C] / J. Liu, Y. Mao, J. Zhang, et al. // 2016 IEEE International Symposium on Information Theory (ISIT). – 2016: – P. 1451 – 1455.
117. Chen X. Efficient multi-user computation offloading for mobile-edge cloud computing[J]. / X. Chen, L. Jiao, W. Li, et al. // IEEE/ACM Transactions on Networking, – 2016. 24(5): – P. 2795 – 2808.
118. Alam M.G.R. Autonomic computation offloading in mobile edge for IoT applications[J]. / M.G.R. Alam, M.M. Hassan, M.Z. Uddin, et al. // Future Generation Computer Systems, – 2019. 90: – P. 149 – 157.
119. Mahmoodi S.E. Optimal joint scheduling and cloud offloading for mobile applications[J]. / S.E. Mahmoodi, R. Uma, K. Subbalakshmi. // IEEE Transactions on Cloud Computing, – 2016. 7(2): – P. 301 – 313.

120. Wang Y. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling[J]./ Y. Wang, M. Sheng, X. Wang, et al. // IEEE Transactions on Communications, – 2016. 64(10): – P. 4268 – 4282.
121. Saleem U. Latency Minimization for D2D-Enabled Partial Computation Offloading in Mobile Edge Computing[J]. / U. Saleem, Y. Liu, S. Jangsher, et al. // IEEE Transactions on Vehicular Technology, – 2020. 69(4): – P. 4472 – 4486.
122. Wang J. Mobility-aware partial computation offloading in vehicular networks: A deep reinforcement learning based scheme[J]. / J. Wang, T. Lv, P. Huang, et al. // China Communications, – 2020. 17(10): – P. 31 – 49.
123. Mao Y. Dynamic computation offloading for mobile-edge computing with energy harvesting devices[J]. / Y. Mao, J. Zhang, K.B. Letaief // IEEE Journal on Selected Areas in Communications, – 2016. 34(12): – P. 3590 – 3605.
124. Mao Y. Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems[C] / Y. Mao, J. Zhang, K.B. Letaief // 2017 IEEE wireless communications and networking conference (WCNC). – 2017. – P. 1 – 6.
125. Deng M. Fine-granularity based application offloading policy in cloud-enhanced small cell networks[C] / M. Deng, H. Tian, B. Fan // 2016 IEEE International Conference on Communications Workshops (ICC). – 2016. – P. 638 – 643.
126. Ning Z. A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things[J]. / Z. Ning, P. Dong, X. Kong, et al. // IEEE Internet of Things Journal, – 2018. 6(3): – P. 4804 – 4814.
127. Li S. Energy-aware mobile edge computation offloading for IoT over heterogenous networks[J]. / S. Li, Y. Tao, X. Qin, et al. // IEEE Access, – 2019. 7: – P. 13092 – 13105.
128. Dong L. Computation offloading for mobile-edge computing with multi-user[C] /L. Dong L, M.N. Satpute, J. Shan, et al. // 2019 IEEE 39th international conference on distributed computing systems (ICDCS). – 2019. – P. 841 – 850.
129. Huang L. Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing[J]. / L. Huang, X. Feng, C. Zhang, et al. // Digital Communications and Networks, – 2019. 5(1): – P. 10 – 17.

130. Zhao T. A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing[C] / T. Zhao, S. Zhou, X. Guo, et al. // 2015 IEEE Globecom Workshops (GC Wkshps). – 2015. – P. 1 – 6.
131. Di Valerio V. Optimal virtual machines allocation in mobile femto- cloud computing: An MDP approach[C] / V. Di Valerio, F.L. Presti // 2014 IEEE Wireless Communications and Networking Conference Workshops (WCNCW). – 2014. – P. 7 – 11.
132. Guo X. An index-based task assignment policy for achieving optimal power-delay tradeoff in edge cloud systems[C] / X. Guo, R. Singh, T. Zhao, et al. // 2016 IEEE International Conference on Communications (ICC). – 2016. – P. 1 – 7.
133. Zhao T. Energy-optimal and delay-bounded computation offloading in mobile edge computing with heterogeneous clouds[J]. / T. Zhao, S. Zhou, L. Song, et al. // China Communications, – 2020. 17(5): – P. 191 – 210.
134. Gu X. Energy-Optimal Latency-Constrained Application Offloading in Mobile-Edge Computing[J]. / X. Gu, C. Ji, G. Zhang // Sensors, – 2020. 20(11): P. 3064 – 3086.
135. Oueis J. Small cell clustering for efficient distributed cloud computing[C] / J. Oueis, E.C. Strinati, S. Barbarossa // 2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC). – 2014. – P. 1474 – 1479.
136. Oueis J. Small cell clustering for efficient distributed fog computing: A multi-user case[C] / J. Oueis, E.C. Strinati, S. Sardellitti, et al. // 2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall). – 2015. – P. 1 – 5.
137. Tanzil S.S. Femto-cloud formation: A coalitional game-theoretic approach[C] / S.S. Tanzil, O.N. Gharehshiran, V. Krishnamurthy // 2015 IEEE Global Communications Conference (GLOBECOM). – 2015. – P. 1 – 6.
138. Wang S. Online placement of multi-component applications in edge computing environments[J]. / S. Wang, M. Zafer, K.K. Leung //IEEE Access, – 2017. 5: – P. 2514 – 2533.
139. Li K. Exploiting computation replication for mobile edge computing: A fundamental computation-communication tradeoff study[J]. / K. Li, M. Tao, Z. Chen // IEEE Transactions on Wireless Communications, – 2020. 19(7): – P. 4563 – 4578.
140. Bi S. Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading[J]. / S. Bi, Y.J. Zhang // IEEE Transactions on Wireless Communications, – 2018, 17(6).: – P. 4177 – 4190.

141. Huang L. Distributed deep learning-based offloading for mobile edge computing networks[J]. / L. Huang, X. Feng, A. Feng, et al. // Mobile Networks and Applications, – 2018. – P.1 – 8.
142. Huang L. Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks[J]. / L. Huang, S. Bi, Y-J.A. Zhang // IEEE Transactions on Mobile Computing, – 2019. 19(11): – P. 2581 – 2593.
143. Wang F. Optimal Energy Allocation and Task Offloading Policy for Wireless Powered Mobile Edge Computing Systems[J]. / F. Wang, J. Xu, S. Cui // IEEE Transactions on Wireless Communications, – 2020. 19(4): – P. 2443 – 2459.
144. Chen M-H. Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point[C] /M-H. Chen, B. Liang, M. Dong // IEEE INFOCOM 2017-IEEE Conference on Computer Communications. – 2017. – P. 1 – 9.
145. Liu F. Energy-efficient collaborative task computation offloading in cloud-assisted edge computing for IoT sensors[J]. / F. Liu, Z. Huang, L. Wang // Sensors, – 2019. 19(5): – P. 1105.
146. Huang L. Multi-server multi-user multi-task computation offloading for mobile edge computing networks[J]. / L. Huang, Z. Feng, Zhang L, et al. // Sensors, – 2019., 19(6): – P. 1446.
147. Mach P. Cloud-aware power control for cloud-enabled small cells[C] / P. Mach, Z. Becvar // 2014 IEEE Globecom Workshops (GC Wkshps). – 2014: – P. 1038 – 1043.
148. Mach P. Cloud-aware power control for real-time application offloading in mobile edge computing[J]. /P. Mach, Z. Becvar // Transactions on Emerging Telecommunications Technologies, – 2016. 27(5): – P. 648 – 661.
149. Rodrigues T.G. Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration[J]. / T.G. Rodrigues, K. Suto, H. Nishiyama, et al. // IEEE Transactions on Computers, – 2018. 67(9): – P. 1287 – 1300.
150. Wang S. Dynamic service placement for mobile microclouds with predicted future costs[J]. / S. Wang, R. Urgaonkar, T. He, et al. // IEEE Transactions on Parallel and Distributed Systems, – 2016. 28(4): – P. 1002 – 1016.
151. Secci S. Linking virtual machine mobility to user mobility[J]. / S. Secci, P. Raad, P. Gallard // IEEE Transactions on Network and Service Management, – 2016. 13(4): – P. 927 – 940.
152. Nadembega A. Mobility prediction model-based service migration procedure for follow me cloud to support QoS and QoE[C] / A. Nadembega, A.S. Hafid, R. Brisebois // 2016 IEEE International Conference on Communications (ICC). – 2016: – P. 1 – 6.

153. Sun X. PRIMAL: Profit maximization avatar placement for mobile edge computing[C] / X. Sun, N. Ansari // 2016 IEEE International Conference on Communications (ICC). – 2016: – P. 1 – 6.
154. Ali Z. A Deep Learning Approach for Mobility-Aware and Energy-Efficient Resource Allocation in MEC[J]. / Z. Ali, S. Khaf, Z.H. Abbas, et al. // IEEE Access, – 2020, 8: – P. 179530 – 179546.
155. Becvar Z. Path selection using handover in mobile networks with cloud-enabled small cells[C] / Becvar Z, Plachy J, Mach P. // 2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC). – 2014: – P.1480 – 1485.
156. Plachy J. Path selection enabling user mobility and efficient distribution of data for computation at the edge of mobile network[J]. / J. Plachy, Z. Becvar, P. Mach P. // Computer Networks, – 2016. 108: – P. 357 – 370.
157. Plachy J. Dynamic resource allocation exploiting mobility prediction in mobile edge computing[C] / J. Plachy, Z. Becvar, E.C. Strinati // 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). – 2016. – P. 1 – 6.
158. Yu F. DMPO: Dynamic mobility-aware partial offloading in mobile edge computing[J]. / F. Yu, H. Chen, J. Xu // Future Generation Computer Systems, – 2018. 89: – P. 722 – 735.
159. Кучерявый А.Е. Тактильный интернет. Сети связи со сверхмалыми задержками / А.Е. Кучерявый, М.А. Маколкина, Р.В. Киричек // Электросвязь. – 2016. – № 1. – С. 44 – 46.
160. Бородин А.С. Сети связи пятого поколения как основа цифровой экономики / А.С. Бородин А.С., А.Е. Кучерявый //Электросвязь. – 2017. – № 5. – С. 45 – 49.
161. Кучерявый А.Е. Самоорганизующиеся сети. / А.Е. Кучерявый, А.В. Прокопьев, Е.А. Кучерявый. – СПб: Типография «Любавич», 2011. – 312 с.
162. Мухизи С. Модели сегментации и кластеризации ресурсов в программно-конфигурируемых сетях / Мухизи С., Атея А.А., Мутханна А.С., Киричёк Р.В. // Электросвязь. 2019. № 4. С. 26-31.
163. Heller B. The controller placement problem // Proceedings of the Special Interest Group on Data Communication (SIGCOMM ’12, Helsinki, Finland, 13 August−17 August 2012). / B.Heller, R. Sherwood, N. McKeown // Special October issue ACM SIGCOMM Computer Communication Review. New York: ACM, – 2012. Vol. 42. Iss. 4. – P. 473 − 478. DOI:10.1145/2377677. 2377767

164. Yao G. On the Capacitated Controller Placement Problem in Software Defined Networks / G. Yao, J. Bi, Y. Li, L. Guo // IEEE Communications Letters. − 2014. Vol. 18. Iss. 8. − P. 1339 – 1342. DOI:10.1109/LCOMM.2014.2332341

165. Dixit A. Towards an elastic distributed SDN controller // Proceedings of the Special Interest Group on Data Communication (SIGCOMM ’13, Hong Kong, China, 16 August 2013). / A. Dixit, F. Hao, S. Mukherjee, T.V. Lakshmanet, R. Kompella // Special October issue ACM SIGCOMM Computer Communication Review. New York: ACM, − 2013. Vol. 43. Iss. 4. − P. 7 − 12. DOI: 10.1145/2534169. 2491193

166. Ozsoy F.A. An exact algorithm for the capacitated vertex pcenter problem / F.A. Ozsoy, M.C. Pinar // Computers & Operations Research. − 2006. Vol. 33. Iss. 5. − P. 1420 – 1436. DOI:10.1016/j.cor.2004.09.035

167. Sahoo K.S. Metaheuristic Solutions for Solving Controller Placement Problem in SDN-based WAN Architecture /K.S. Sahoo, A. Sarkar, S.K. Mishra, B. Sahoo, D. Puthal, M.S. Obaidat, et al. // Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) and 8th International Conference on Data Communication Networking (DCNET), Madrid, Spain, 15–23 July 2017. SciTePress Digital Library, – 2017. – P. 15 – 23. DOI:10.5220/0006483200150023

168. Rath H.K. Optimal controller placement in Software Defined Networks (SDN) using a non-zero-sum game /H.K. Rath, V. Revoori, S.M. Nadaf, A. Simha // Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks (Sydney, Australia, 19 June 2014). IEEE, – 2014. DOI:10.1109/WoWMoM.2014.6918987

169. Ksentini A. On using bargaining game for Optimal Placement of SDN controllers / A. Ksentini, M. Bagaa, T. Taleb, I. Balasingham.// Proceedings of the International Conference on Communications (ICC, Kuala Lumpur, Malaysia, 22–27 May 2016). IEEE, – 2016. DOI:10.1109/ICC.2016.7511136

170. Ateya A.A. Chaotic salp swarm algorithm for SDN multi-controller networks / A.A. Ateya, A. Muthanna, A. Vybornova, A.D. Algarni, A. Abuarqoub, Y. Koucheryavy, et al.// Engineering Science and Technology, an International Journal. – 2019. Vol. 22. Iss. 4. – P. 1001 – 1012. DOI:10.1016/j.jestch.2018.12.015

171. Killi B.P. Capacitated Next Controller Placement in Software Defined Networks / B.P. Killi, S.V. Rao // IEEE Transactions on Network and Service Management. – 2017. Vol. 14. Iss. 3. – P. 514 – 527. DOI:10.1109/TNSM.2017.2720699

172. Мутханна А.С.А. Интегральное решение проблемы размещения контроллеров и балансировки нагрузки Труды учебных заведений связи. 2023/ Мутханна А.С // Т. 9. № 2. С. 81-93.

173. Wang G. A K-means-based network partition algorithm for controller placement in software defined network / G. Wang, Y. Zhao, J. Huang, Q Duan, J. Li // Proceedings of the International Conference on Communications (ICC, Kuala Lumpur, Malaysia, 22–27 May 2016). IEEE, – 2016. DOI:10.1109/ICC.2016.7511441

174. Мухизи С. Исследование моделей балансировки нагрузки в программно-конфигурируемых сетях / Мухизи С., Мутханна А.С., Киричек Р.В., Кучерявый А.Е. // Электросвязь. 2019. № 1. С. 23-29.

175. Hu Y. BalanceFlow: Controller load balancing for OpenFlow networks /Y. Hu, W. Wang, X. Gong, X. Que, S. Cheng // Proceedings of the 2nd International Conference on Cloud Computing and Intelligent Systems (Hangzhou, China, 30 October 2012–01 November 2012. IEEE, – 2013. – P. 780 – 785. DOI:10.1109/CCIS.2012.6664282

176. Дунайцев Р.А. Интегрированная сеть космос-воздух-земля-море как основа сетей связи шестого поколения / Р.А. Дунайцев, А.С. Бородин, А.Е. Кучерявый //Электросвязь. – 2022. – № 10. – С. 5 – 8.

177. Ateya, A. Study of 5G services standardization: Specifications and requirements / A. Ateya, A. Muthanna, M. Makolkina, A. Koucheryavy // in 2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), – 2018, – P. 1 – 6.

178. Guo F. Enabling massive IoT toward 6G: A comprehensive survey / F. Guo, F.R. Yu, H. Zhang, X. Li, H. Ji, V.C.M. Leung // IEEE Internet Things J. – 2021. – vol. 8, no. 15, pp. – P. 11891 – 11915.

179. Laghari D. A review and state of art of internet of things (IoT) / Laghari, K. Wu, R.A. Laghari, M. Ali, and A. A. Khan // Arch. Comput. Methods Eng. – 2022. – vol. 29, no. 3, – P. 1395 – 1413.

180. Ateya, A. Enabling heterogeneous IoT networks over 5G networks with ultra-dense deployment—using MEC/SDN / A. Ateya, D. Laghari, K. Wu, R.A. Laghari, M. Ali, and A. A. Khan // Electronics (Basel). – 2021. – vol. 10, no. 8, – P. 910.
181. Bhuiyan M. N. Internet of things (IoT): A review of its enabling technologies in healthcare applications, standards protocols, security, and market opportunities / M.N. Bhuiyan, M.M. Rahman, M.M. Billah, D. Saha // IEEE Internet Things J. – 2021. – vol. 8, no. 13, – P. 10474 – 10498.
182. Carvalho G. Edge computing: current trends, research challenges and future directions / G. Carvalho, B. Cabral, V. Pereira, and J. Bernardino / Computing, – 2021. – vol. 103, no. 5, – P. 993 – 1023.
183. Haibeh L.A. A survey on mobile edge computing infrastructure: Design, resource management, and optimization approaches / L.A. Haibeh, M.C.E. Yagoub, A. Jarray // IEEE Access, – 2022. – vol. 10, – P. 27591 – 27610.
184. Cruz P. On the edge of the deployment: A survey on multi-Access Edge Computing / P. Cruz, N. Achir, A. C. Viana // ACM Comput. Surv., – 2022.
185. L. Kong et al. Edge-computing-driven Internet of Things: A survey. ACM Comput. Surv., – 2022.
186. Алзагир А.А. Алгоритмы кластеризации для БПЛА в сетях пятого и последующих поколений / Алзагир А.А., Коваленко В.Н., Бородин А.С., Мутханна А.С.А., Кучерявый А.Е. // Электросвязь. 2022. № 10. С. 9-15.
187. Amarasingam N. A review of UAV platforms, sensors, and applications for monitoring of sugarcane crops / N. Amarasingam, A.S. Ashan Salgadoe, K. Powell, L.F. Gonzalez, S. Natarajan // Remote Sens. Appl. Soc. Environ., – 2022. – vol. 26, no. 100712, – P. 100712.
188. Liu Y. Unmanned aerial vehicle for internet of everything: Opportunities and challenges / Y. Liu, H.-N. Dai, Q. Wang, M. K. Shukla, M. Imran // Comput. Commun., – 2020. – vol. 155, – P. 66 – 83.
189. Pakrooh R. A survey on unmanned aerial vehicles-assisted internet of things: A service-oriented classification / R. Pakrooh, A. Bohlooli // Wirel. Pers. Commun., – 2021. – vol. 119, no. 2, – P. 1541 – 1575.

190. Idrissi. A review of Quadrotor unmanned aerial vehicles: Applications, architectural design and control algorithms / Idrissi, M. Salami, and F. Annaz // J. Intell. Robot. Syst., – 2022. – vol. 104, no. 2.
191. Labib S. The rise of drones in internet of things: A survey on the evolution, prospects and challenges of unmanned aerial vehicles / S. Labib, M. R. Brust, G. Danoy, and P. Bouvry // IEEE Access, – 2021. – vol. 9, – P. 115466 – 115487.
192. Siddharthraju K. A survey on IoE-enabled unmanned aerial vehicles / K. Siddharthraju, R. Dhivyadevi, M. Supriya, B. Jaishankar, T. Shanmugaraja // Unmanned Aerial Vehicles for Internet of Things (IoT). Wiley, – 24-Aug.2021. – P. 173 – 192,
193. Shehzad M.K. Backhaul-aware intelligent positioning of UAVs and association of terrestrial base stations for fronthaul connectivity / M.K. Shehzad, A. Ahmad, S.A. Hassan, H. Jung // IEEE Trans. Netw. Sci. Eng., – 2021. – vol. 8, no. 4, – P. 2742 – 2755.
194. S. H. Alsamhi et al. Computing in the sky: A survey on intelligent ubiquitous computing for UAV-assisted 6G networks and industry 4.0/5.0. // Drones, – 2022. – vol. 6, no. 7, – P. 177.
195. Yazid Y. UAV-enabled mobile edge-computing for IoT based on AI: A comprehensive review / Y. Yazid, I. Ez-Zazi, A. Guerrero-González, A. El Oualkadi, and M. Arioua // Drones, – 2021. – vol. 5, no. 4, – P. 148.
196. Zhang, W. Liu, and N. Ansari, “Joint wireless charging and data collection for UAV-enabled internet of things network / Zhang, W. Liu, and N. Ansari // IEEE Internet Things J., – 2022. – P. 1 – 1.
197. D. C. Nguyen et al. 6G internet of things: A comprehensive survey // IEEE Internet Things J., – 2022.vol. 9, no. 1, – P. 359 – 383.
198. H. Alsamhi et al. Drones’ edge intelligence over smart environments in B5G: Blockchain and federated learning synergy // IEEE Trans. On Green Commun. Netw., – 2022. – vol. 6, no. 1, – P. 295 – 312.
199. Zeb S. Edge intelligence in softwarized 6G: Deep learning-enabled network traffic predictions / S. Zeb, M.A. Rathore, A. Mahmood, S.A. Hassan, J. Kim, and M. Gidlund // in 2021 IEEE Globecom Workshops (GC Wkshps), – 2021, – P. 1 – 6.
200. Qadir Z. Towards 6G Internet of Things: Recent advances, use cases, and open challenges / Z. Qadir, K.N. Le, N. Saeed, H. S. Munawar // ICT Express, – 2022.

201. Beniwal G. A systematic literature review on IoT gateways / G. Beniwal, A. Singhrova //J. King Saud Univ. - Comput. Inf. Sci., – 2021.
202. Jeong S. Mobile cloud computing with a UAV-mounted cloudlet: optimal bit allocation for communication and computation / S. Jeong, O. Simeone, and J. Kang // IET Commun., – 2017. – vol. 11, no. 7, – P. 969 – 974.
203. Jeong S. Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning / S. Jeong, O. Simeone, J. Kang // IEEE Trans. Veh. Technol., – 2018. – vol. 67, no. 3, – P. 2049 – 2063.
204. Ateya Ashraf Energy- and latency-aware hybrid offloading algorithm for UAVs / Ashraf Ateya, A. Muthanna, R. Kirichek, M. Hammoudeh, A. Koucheryavy // IEEE Access, – 2019. – vol. 7, – P. 37587 – 37600.
205. Ateya Latency and energy-efficient multi-hop routing protocol for unmanned aerial vehicle networks / Ateya, A. Muthanna, I. Gudkova, Y. Gaidamaka, and A. D. Algarni // Int. J. Distrib. Sens. Netw., – 2019. – vol. 15, no. 8.
206. Castelli M. Salp Swarm Optimization: A critical review / M. Castelli, L. Manzoni, L. Mariot, M.S. Nobile, A. Tangherloni // Expert Syst. Appl., – 2022. – vol. 189, no. 116029, – P. 116029.
207. Pradhan. A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment / Pradhan, S.K. Bisoy, A. Das // J. King Saud Univ. - Comput. Inf. Sci., – 2022. – vol. 34, no. 8, – P. 4888 – 4901.
208. Parthiban S. Chaotic salp swarm optimization-based energy-aware VMP technique for Cloud Data Centers / S. Parthiban, A. Harshavardhan, S. Neelakandan, V. Prashanthi, A.-R.A. Alhassan Alolo, S. Velmurugan // Comput. Intell. Neurosci., – 2022. vol. 2022, – P. 4343476.
209. Sliwa. Lightweight simulation of hybrid aerial- and ground-based vehicular communication networks / Sliwa, M. Patchou, C. Wietfeld // in 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), – 2019, – P. 1 – 7.
210. Feasibility Study on New Services and Markets Technology Enablers, document 3GPP TR 22.891, ver. 14.2.0, – Sep., 2016.
211. Aldabbas O. Unmanned ground vehicle for data collection in wireless sensor net-works: Mobility-aware sink selection / O. Aldabbas, A. Abuarqoub, M. Hammoudeh, U. Raza, A. Bounceur // Open Autom. Control Syst. J. – Jul., 2016. – vol. 8. № 1.

212. Motlagh N.H. UAV-based IoT platform: A crowd surveillance use case. / N.H. Motlagh, M. Bagaa, T. Taleb // IEEE Commun. Mag. – Feb., 2017. – vol. 55. – № 2. – P. 128 – 134.
213. Shah S.A.A.5G for vehicular Communications / S.A.A. Shah, E. Ahmed, M. Imran, S. Zeadally // IEEE Commun. Mag. – Jan., 2018. – vol. 56. – № 1. – P. 111 – 117.
214. H. Shakhatreh et al. Unmanned aerial vehicles: A survey on civil applications and key research challenges. [Online]. Available: https://arxiv.org/abs/1805.00881. – 2018.
215. Sekander S. Multi-tier drone architecture for 5G/B5G cellular networks: Challenges, trends, and prospects / S. Sekander, H. Tabassum, E. Hossain // IEEE Commun. Mag. – Mar., 2018. – vol. 56, – № 3. – P. 96 – 103.
216. Zahariadis T. Preventive maintenance of critical infrastructures using 5G networks drones / T. Zahariadis, A. Voulkidis, P. Karkazis, P. Trakadas, // in Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS), – Sep., 2017. – P. 1 – 4.
217. Adam, A. Toward Smart Traffic Management With 3D Placement Optimization in UAV-Assisted NOMA IIoT Networks / Adam, A., Muthanna, M. S. A., Muthanna, A., & Nguyen, T. N. // IEEE Transactions on Intelligent Transportation Systems.
218. Zeng Y. Cellular-connected UAV: Potentials, challenges and promising technologies / Y. Zeng, J. Lyu, R. Zhang. // [Online]. – 2018. Available: https://arxiv.org/abs/1804.02217
219. Thibbotuwawa A. Energy consumption in unmanned aerial vehicles:Areviewof energy consumption models and their relation to the UAV routing / A. Thibbotuwawa, P. Nielsen, B. Zbigniew, G. Bocewicz // in Proc. 39th Int. Conf. Inf. Syst. Archit. Technol. Cham, Switzerland: Springer. – 2018. – P. 173 – 184.
220. E. Ahmed et al. Bringing computation closer toward the user network: Is edge computing the solution?" IEEE Commun. Mag. – Nov., 2017. – vol. 55. – № 11. – P. 138 – 144.
221. Khawaja W. A survey of air-to-ground propagation channel modeling for unmanned aerial vehicles / W. Khawaja, I. Guvenc, D. Matolak, U.C. Fiebig, N. Schneckenberger // – 2018. [Online]. Available: https://arxiv.org/abs/1801.01656
222. Valentino R. Opportunistic computational offloading system for clusters of drones / R. Valentino, W.S. Jung, Y.B. Ko // in Proc. 20th Int. Conf. Adv. Commun. Technol. – Feb., 2018. – P. 303 – 306.
223. F. Cheng et al. UAV trajectory optimization for data offloading at the edge of multiple cells," IEEE Trans. Veh. Technol. – Jul., 2018. – vol. 67, – № 7. – P. 6732 – 6736.

224. Management and Orchestration; 5G end to end Key Performance Indicators (KPI), Release 15, document 3GPP TS 28.554, ver. 2.0.0. – Sep., 2018.

225. Ahmed E. Process state synchronization-based application execution management for mobile edge/cloud computing / E. Ahmed, A. Naveed, A. Gani, S.H.A. Hamid, M. Imran, M. Guizani // Future Gener. Comput. Syst. – Feb., 2019. – vol. 91. – № 2. – P. 579 – 589.

226. Kim B. Dynamic computation offloading scheme for drone-based surveillance systems / B. Kim, H. Min, J. Heo, and J. Jung, // Sensors. – 2018. – vol. 18. – № 9. – P. 2982.

227. Хакимов А.А. Разработка интеллектуальной системы для управления граничными вычислениями / Хакимов А.А., Мутханна А.С., Выборнова А.И. //Электросвязь. 2021. № 4. С. 37-42.

228. Messous M.A. Computation offloading game for an UAV network in mobile edge computing / M.A. Messous, H. Sedjelmaci, N. Houari, S. M. Senouci // in Proc. IEEE Int. Conf. Commun. – May., 2017. – P. 1 – 6.

229. M. Narang et al. UAV-assisted edge infrastructure for challenged networks. in Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS). – May., 2017. – P. 60 – 65.

230. Hassan N. The role of edge computing in Internet of Things / N. Hassan, S. Gillani, E. Ahmed, I. Ibrar, M. Imran // IEEE Commun. Mag. doi: 10.1109/MCOM.2018.1700906. – Nov., 2018. – vol. 56. – № 11. – P. 110 – 115.

231. Z. Zhao et al. Software-defined unmanned aerial vehicles networking for video dissemination services. Ad Hoc Netw., – Feb., 2019. – vol. 83. – P. 68 – 77.

232. Ateya A.A. Energy-aware of_oading algorithm for multi-level cloud based 5G system / A.A. Ateya, A. Muthanna, A. Vybornova, P. Darya, A. Koucheryavy // in Proc. Internet of Things, Smart Spaces, Next Gener. Netw. Syst. Cham, Switzerland: Springer. – Aug., 2018. – P. 355 – 370.

233. Ateya A.A. Multilevel cloud based Tactile Internet system / A.A. Ateya, A. Vybornova, R. Kirichek, A. Koucheryavy // in Proc. IEEE 19th Int. Conf. Adv. Commun. Technol. (ICACT), Bongpyeong, South Korea. – Feb., 2017. – P. 105 – 110.

234. Luo C. A UAV-cloud system for disaster sensing applications / C. Luo, J. Nightingale, E. Asemota, C. Grecos // in Proc. IEEE 81st Veh. Technol. Conf. (VTC Spring). – May., 2015. – P. 1 – 5.

235. Lynskey J. Maximizing offloading opportunities for UAV communication. Korean Netw. Oper. Manage., Jeju Nat. Univ., Jeju, South Korea, Tech. Rep., – May., 2018.

236. Zhou Z. An air-ground integration approach for mobile edge computing in IoT / Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong // IEEE Commun. Mag. – Aug., 2018. – vol. 56. – № 8. – P. 40 – 47.

237. Jung W.S. Adaptive offloading with MPTCP for unmanned aerial vehicle surveillance system / W.S. Jung, J. Yim, Y.B. Ko // Ann. Telecommun. – Oct. 2018. – vol. 73. – P. 1 – 14.

238. Shukla R.M. Software-defined network based resource allocation in distributed servers for unmanned aerial vehicles / R.M. Shukla, S. Sengupta, A. N. Patra // in Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf. (CCWC). – Jan., 2018. – P. 796 – 802.

239. Ateya A.A. System model for multi-level cloud based tactile Internet system / A.A. Ateya, A. Vybornova, K. Samouylov, A. Koucheryavy // in Proc. Int. Conf. Wired/Wireless Internet Commun. – Mar., 2017. – P. 77 – 86.

240. Sharma V. LoRaWAN-based energy-ef_cient surveillance by drones for intelligent transportation systems / V. Sharma, I. You, G. Pau, M. Collotta, J.D. Lim, and J. N. Kim, // Energies. – 2018. – vol. 11. – № 3. – P. 573.

241. Krichen L. Communication architecture for unmanned aerial vehicle system / L. Krichen, M. Fourati, L. C. Fourati // in Proc. Int. Conf. Ad-Hoc Netw. Wireless. Cham, Switzerland: Springer. – Sep., 2018. – P. 213 – 225.

242. Secinti G. SDNs in the Sky: Robust end-to-end connectivity for aerial vehicular networks / G. Secinti, P.B. Darian, B. Canberk, K.R. Chowdhury // IEEE Commun. Mag. – Jan., 2018. – vol. 56. – № 1. – P. 16 – 21.

243. Kartun-Giles A. Euclidean matchings in ultra-dense networks / A. Kartun-Giles, S. Jayaprakasam, S. Kim //IEEE Commun. Lett. – Jun., 2018. –vol. 22. – № 6. – P. 1216 – 1219.

244. Sharma V. UAV-assisted heterogeneous networks for capacity enhancement / V. Sharma, M. Bennis, R. Kumar // IEEE Commun. Lett. – Jun. 2016. – vol. 20. – № 6. – P. 1207 – 1210.

245. Coates A. Internet of things for buildings monitoring: Experiences and challenges / A. Coates, M. Hammoudeh, K.G. Holmes // in Proc. Int. Conf. Future Netw. Distrib. Syst., ACM., – Jul., 2017. –P. 38.

246. Jogunola et al. Distributed adaptive primal algorithm for P2P-ETS over unreliable communication links. Energies, – 2018. – vol. 11. – №. 9. – P. 2331.

247. Atwady Y. A survey on authentication techniques for the Internet of Things / Y. Atwady and M. Hammoudeh // in Proc. Int. Conf. Future Netw. Distrib. Syst., – Jul., 2017. – P. 8.
248. Wang, J. Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones / J. Wang. C. Jiang, Z. Han, Y. Ren, R.G. Maunder, L. Hanzo // IEEE Veh. Technol. Mag. – 2017. – № 12, P. 73 – 82.
249. CAAC. Low-Altitude Connected Drone Flight Safety Test Report. Available online: http://www.caac.gov.cn/en/ (accessed on 27 April 2022).
250. Hayat S. Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint / S. Hayat, E. Yanmaz, R. Muzaffar // IEEE Commun. Surv. Tutorials – 2016. – № 18. – P. 2624 – 2661.
251. Zeng Y. Wireless communications with unmanned aerial vehicles: Opportunities and challenges. / Y. Zeng, R. Zhang, T.J. Lim // IEEE Commun. Mag. – 2016. – № 54. – P. 36 – 42.
252. Yaacoub E. A key 6G challenge and opportunity—connecting the base of the pyramid: A survey on rural connectivity. / E. Yaacoub, M.S. Alouini // Proc. IEEE – 2020. – № 108. – P. 533 – 582.
253. Dang S. What should 6G be? / S. Dang, O. Amin, B. Shihada, M.S. Alouini, // Nat. Electron. – 2020. – № 3. – P. 20 – 29.
254. Tikhomirov A. Recommended 5G frequency bands evaluation. / A. Tikhomirov, E. Omelyanchuk, A. Semenova // In Proceedings of the 2018 Systems of Signals Generating and Processing in the Field of on Board Communications, Moscow, Russia, – 14 –15 March 2018. – P. 1 –5.
255. Bargis M. From Standards to Service-the EuropeanWay to 5G. / M. Bargis, G. Romano // IEEE 5G Tech Focus – 2017. – № 1. – P. 15 – 21.
256. Obanawa H. Applications of UAV Remote Sensing to Topographic and Vegetation Surveys. In Unmanned Aerial Vehicle: Applications in Agriculture and Environment; / H. Obanawa, H. Shibata // Springer: Berlin/Heidelberg, Germany, – 2020; – P. 131 – 142.
257. Bithas P.S. A Survey on Machine-Learning Techniques for UAV-Based Communications. / P.S. Bithas, E.T. Michailidis, N. Nomikos, D. Vouyioukas, A.G. Kanatas // Sensors. – 2019. – № 19. – P. 5170.

258. Khayyat M. Multilevel Service-Provisioning-Based Autonomous Vehicle Applications. / M. Khayyat, A. Alshahrani, S. Alharbi, I. Elgendy, A. Paramonov, Koucheryavy, A. // Sustainability. – 2020. – № 12. – P. 2497.
259. Mach P. Mobile edge computing: A survey on architecture and computation offloading. / P. Mach, Z. Becvar //IEEE Commun. Surv. Tutorials. – 2017. – № 19. – P. 1628 – 1656.
260. Cheng N. Air-ground integrated mobile edge networks: Architecture, challenges, and opportunities. / N. Cheng, W. Xu, W. Shi, Y. Zhou, N. Lu, H. Zhou, X. Shen // IEEE Commun. Mag. – 2018. – № 56. – P. 26 – 32. Appl. Sci. 2022, 12, 6566 17 of 18
261. Zhou F. Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems. / F. Zhou, Y. Wu, R.Q. Hu, Y. Qian, Y. // IEEE J. Sel. Areas Commun. – 2018. – № 36. – P. 1927 – 1941.
262. Mao Y. A survey on mobile edge computing: The communication perspective. / Y. Mao, C. You, J. Zhang, K. Huang, K.B. Letaief // IEEE Commun. Surv. Tutorials. – 2017. – № 19. – P. 2322 – 2358.
263. Khayyat M. Advanced Deep Learning-based Computational Offloading for Multilevel Vehicular Edge-Cloud Computing Networks. / M. Khayyat, I.A. Elgendy, A. Muthanna, A. Alshahrani, S. Alharbi, A. Koucheryavy // IEEE Access – 2020. – № 8. – P. 137052 – 137062.
264. Zhang X. Resource allocation for a UAV-enabled mobile-edge computing system: Computation efficiency maximization. / X. Zhang, Y. Zhong, P. Liu, F. Zhou, Y. Wang // IEEE Access – 2019. – № 7. – P. 113345 – 113354.
265. Zhang J. Stochastic computation offloading and trajectory scheduling for UAV-assisted mobile edge computing. / J. Zhang, L. Zhou, Q. Tang, E.C.H. Ngai, X. Hu, H. Zhao, J. Wei // IEEE Internet Things J. – 2018. – № 6. – P. 3688 – 3699.
266. Beiqing C. RESERVE: An Energy-Efficient Edge Cloud Architecture for Intelligent Multi-UAV. / C. Beiqing, Z. Haihang, Y. Jianguo, G. Haibing // IEEE Trans. Serv. Comput. – 2019. – № 15. – P. 819 – 832.
267. You C. Energy efficient mobile cloud computing powered by wireless energy transfer. / C. You, K. Huang, H. Chae // IEEE J. Sel. Areas Commun. – 2016. – № 34. – P. 1757 – 1771.
268. Huang L. Deep reinforcement learning for online computation offloading in wireless powered mobile-edgecomputing networks. / L. Huang, S. Bi, Y.J. Zhang // IEEE Trans. Mob. Comput. – 2019. – № 19. – P. 2581 – 2593.

269. Wang Y. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling / Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li // IEEE Trans. Commun. – 2016. – № 64. – P. 4268 – 4282.
270. Dinh T.Q. Learning for computation offloading in mobile edge computing. /T.Q. Dinh, Q.D. La, Quek, T.Q.; Shin, H. // IEEE Trans. Commun. – 2018. – № 66. – P. 6353 – 6367.
271. Liu F. Energy-efficient collaborative task computation offloading in cloud-assisted edge computing for IoT sensors. / F. Liu, Z. Huang, L. Wang // Sensors, – 2019. – № 19. – P. 1105.
272. Qu G. DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloudcomputing. // G. Qu, H. Wu, R. Li, P. Jiao // IEEE Trans. Netw. Serv. Manag. – 2021. – № 18. – P. 3448 – 3459.
273. Lu H. Edge QoE: Computation offloading with deep reinforcement learning for Internet of Things. / H. Lu, X. He, M. Du, X. Ruan, Y. Sun, K. Wang, K. // IEEE Internet Things J. – 2020. – № 7. – P. 9255 – 9265.
274. Ji L. Energy-efficient cooperative resource allocation in wireless powered mobile edge computing. / L. Ji, S. Guo // IEEE Internet Things J. – 2018. – № 6. – P. 4744 – 4754.
275. Hu L. Ready player one: UAV-clustering-based multi-task offloading for vehicular VR/AR gaming. / L. Hu, Y. Tian, J. Yang, T. Taleb, L. Xiang, Y. Hao, // IEEE Netw. – 2019. –№ 33. – P. 42 – 48.
276. Zhan Y. A deep reinforcement learning based offloading game in edge computing. / Y. Zhan, S. Guo, P. Li, J. Zhang // IEEE Trans. Comput. – 2020. – № 69. – P. 883 –893.
277. Wang Y. Task Offloading for Post-Disaster Rescue in Unmanned Aerial Vehicles Networks. / Y. Wang, W. Chen, T.H. Luan, Z. Su, Q. Xu, R. Li, N. Chen, N. // IEEE/ACM Trans. Netw. – 2022. early access.
278. Zhou Z. An air-ground integration approach for mobile edge computing in IoT. / Z. Zhou, J. Feng, L. Tan, L.; Y. He, J. Gong // IEEE Commun. Mag. – 2018. – №56. – P. 40 – 47.
279. Lin L. Echo: An edge-centric code offloading system with quality of service guarantee. / L. Lin, P. Li, X. Liao, H. Jin, Y. Zhang, Y. // IEEE Access – 2018. – № 7. – P. 5905 – 5917.
280. Goudarzi M. An application placement technique for concurrent IoT applications in edge and fog computing environments. / M. Goudarzi, H. Wu, M. Palaniswami, R. Buyya // IEEE Trans. Mob. Comput. – 2021. – № 20. – P. 1298 – 1311.

281. Xia J. Intelligent task offloading and collaborative computation in multi-UAV-enabled mobile edge computing. /J. Xia, P. Wang, B. Li, Z. Fei / China Commun. – 2022. – № 19. – P. 244 – 256.
282. Bai T. Energy-Efficient Computation Offloading for Secure UAV-Edge-Computing Systems. / T. Bai, J. Wang, Y. Ren, L. Hanzo // IEEE Trans. Veh. Technol. – 2019. – № 68. – P. 6074 – 6087.
283. Yang Z. Energy Efficient Resource Allocation in UAV-Enabled Mobile Edge Computing Networks. / Z. Yang, C. Pan, K. Wang, M. Shikh-Bahaei // arXiv – 2019. arXiv:1902.03158.
284. Elgendy I. An efficient and secured framework for mobile cloud computing. IEEE Trans. / I. Elgendy, W. Zhang, C. Liu, C.H. Hsu // Cloud Comput. – 2018. – № 9. – P. 79 – 87.
285. Deb S. Learning-based uplink interference management in 4G LTE cellular systems. / S. Deb, P. Monogioudis // IEEE/ACM Trans. Netw. – 2015. – № 23. – P. 398 – 411.
286. Sun S. Optimizing multi-UAV deployment in 3-D space to minimize task completion time in UAV-enabled mobile edge computing systems. / S. Sun, G. Zhang, H. Mei, K. Wang, K. Yang, K. // IEEE Commun. Lett. – 2021. – № 25. – P. 579 – 583.
287. Elgendy I.A. Resource allocation and computation offloading with data security for mobile edge computing. / I.A. Elgendy, W. Zhang, Y.C. Tian, K. Li // Future Gener. Comput. Syst. – 2019. – № 100. – P. 531 – 541.
288. Lyu X. Multiuser joint task offloading and resource optimization in proximate clouds. / X. Lyu, H. Tian, C. Sengul, P. Zhang // IEEE Trans. Veh. Technol. – 2016. – № 66. – P. 3435 – 3447.
289. Elgendy I.A. Efficient and Secure Multi-User Multi-Task Computation Offloading for Mobile-Edge Computing in Mobile IoT Networks. /I.A. Elgendy, W.Z. Zhang, Y. Zeng, H. He, Y.C. Tian, Y. Yang // IEEE Trans. Netw. Serv. Manag. – 2020. – № 17. – P. 2410 – 2422. Appl. Sci. 2022, 12, 6566 18 of 18
290. Lyu X. Adaptive receding horizon offloading strategy under dynamic environment. / X. Lyu, H. Tian // IEEE Commun. Lett. – 2016. – № 20. – P. 878 – 881.
291. Chen X. Decentralized Computation Offloading Game for Mobile Cloud Computing. IEEE Trans. Parallel Distrib. Syst. – 2015. – № 26. – P. 974 – 983.
292. Fooladivanda D. Joint resource allocation and user association for heterogeneous wireless cellular networks. / D. Fooladivanda, C. Rosenberg // IEEE Trans. Wirel. Commun. – 2012. – № 12. – P. 248 – 257.

293. Guignard M. Lagrangean relaxation. Top – 2003. – № 11. – P. 151 – 200.
294. Chen X. Efficient multi-user computation offloading for mobile-edge cloud computing. / X. Chen, L. Jiao, W. Li, X. Fu // IEEE/ACM Trans. Netw. – 2015. – № 24. – P. 2795 – 2808.
295. Beal L.D. Gekko optimization suite. / L.D. Beal, D.C. Hill, R.A. Martin,J.D. Hedengren // Processes – 2018. – № 6. – P. 106.
296. Кучерявый А.Е., Бородин А.С., Киричек Р.В. Сети связи 2030 .Электросвязь. 2018. № 11. С. 52-56.
297. Атея А.А. Интеллектуальное ядро для сетей связи 5G и тактильного интернета на базе программно-конфигурируемых сетей. / А.А. Атея, А.С. Мутханна, А.Е. Кучерявый // Электросвязь. – 2019. – № 3. – С. 34 – 40.
298. Маколкина М.А. Метод выгрузки трафика приложений дополненной реальности в многоуровневой системе граничных вычислений. / М.А. Маколкина, А.А. Атея, А.С. Мутханна, А.Е. Кучерявый // Электросвязь. – 2019. – № 6. – С. 36 – 42.
299. Бородин А.С. Особенности использования D2D-технологий в зависимости от плотности пользователей и устройств. / А.С. Бородин, А.Е. Кучерявый, А.И. Парамонов // Электросвязь. – 2018. – № 10. – С. 40 – 45.
300. Carli R. Monitoring traffic congestion in urban areas through probe vehicles: a case study analysis. /R. Carli, M. Dotoli, N. Epicoco // Internet Technol Lett. – 2018;1(4):e5.
301. Fernandes B. Implementation and analysis of IEEE and ETSI security standards for vehicular communications. / B. Fernandes, J. Rufino, M. Alam, J. Ferreira // Mob Netw Appl. – 2018. – № 23(3). – P. 469 – 478.
302. ITU. Feasibility study on new services and markets technology enablers for critical communications. 3GPP TR 22.862. 3GPP, 2016. https://www.itu.int/dmspubrec/itu-r/rec/m/R-REC-M.2083-0-201509-I!!PDF-E.pdf. Accessed – March 30, 2019.
303. Dighriri M. Measurement and classification of smart systems data traffic over 5G mobile networks. In: Mohammad D, Hamid A, Babak A, eds. Technology for Smart Futures. / M. Dighriri, G.M. Lee, T. Baker //Cham, Switzerland: Springer; – 2018. – P. 195 – 217.
304. Jaballah W.B. Survey on software-defined VANETs: Benefits, challenges, and future directions. / W.B. Jaballah, M. Conti, C.A. Lal // arXiv preprint arXiv:1904.04577. – 2019. https://dblp.org/rec/bib/journals/corr/abs-1904-04577

305. Wang J. Networking and communications in autonomous driving: a survey. / J. Wang, J. Liu, N. Kato // IEEE Commun Surv Tutor. – 2019. – № 21(2). – P. 1243 –1274.
306. Joy J. Internet of vehicles: enabling safe, secure, and private vehicular crowdsourcing. / J. Joy, V. Rabsatt, M. Gerla // Internet Technol Lett. – 2018. – № 1(1). – P.16.
307. Al Ridhawi I. A collaborative mobile edge computing and user solution for service composition in 5G systems. / Al Ridhawi I., M. Aloqaily, Y. Kotb, Al Ridhawi Y., Y. Jararweh // Trans Emerg Telecommun Technol. – 2018. –№ 29(11). – P. 3446.
308. Aldabbas O. Unmanned ground vehicle for data collection in wireless sensor networks: mobility-aware sink selection. / O. Aldabbas, A. Abuarqoub, M. Hammoudeh, U. Raza, A. Bounceur // Open Automat Control Syst J. – 2016. – №8(1). – P. 35 – 46.
309. Ateya A.A. Multilevel cloud based tactile inter net system. Paper presented at: 2017 19th International Conference on Advanced Communication Technology (ICACT); / A.A. Ateya, A. Vybornova, R. Kirichek, A. Koucheryavy // IEEE; – 2017. – P. 105 – 110.
310. Muthanna A. Secure and reliable IoT networks using fog computing with software-defined networking and blockchain. / A. Muthanna, A.A. Ateya, A. Khakimov, et al. // J Sensor Actuator Netw. – 2019. – № 8(1). – P.15.
311. Hu T. Multi-controller based software-defined networking: a survey. / T. Hu, Z. Guo, P. Yi, T. Baker, J. Lan // IEEE Access. – 2018. –№ 6. –P.15980 – 15996.
312. Ateya A.A. Multi-level cluster based device-to-device (D2D) communication protocol for the base station failure situation. / A.A. Ateya, A. Muthanna, A. Vybornova, A. Koucheryavy // Internet of Things, Smart Spaces, and Next Generation Networks and Systems. Cham, Switzerland: Springer; – 2017. –P. 755 – 765.
313. Paramonov A. Clustering optimization for out-of-band D2D communications. / A. Paramonov, O. Hussain, K. Samouylov, A. Koucheryavy, R. Kirichek, Y. Koucheryavy // Wirel Commun Mob Comput. – 2017. – P. 1 – 11.
314. Bezoui M. Detecting gaps and voids in WSNs and IoT networks: the angle-based method. / M. Bezoui, A. Bounceur, L. Lagadec, et al. // Paper presented at: Proceedings of the 2nd International Conference on Future Networks and Distributed Systems; – 2018. – P. 7.
315. Raza S. A survey on vehicular edge computing: Architecture, applications, technical issues, and future directions. / S. Raza, S. Wang, M. Ahmed, & M.R. Anwar // Wireless Communications and Mobile Computing 2019. – 2019.

316. Liu, S. et al. Edge Computing for Autonomous Driving: Opportunities and Challenges. Proceedings of the IEEE. – 2019. doi:10.1109/JPROC.2019.291598
317. Nkenyereye L. Software Defined Network-based Multi-access Edge Framework for Vehicular Networks. / L. Nkenyereye, L. Nkenyereye, S.M. Riazul Islam, C.A. Kerrache, A. Alamri. // IEEE Access. – 2019. – P. 99.
318. Grigorescu S. A Survey of Deep Learning Techniques for Autonomous Driving [Электронный ресурс] / S. Grigorescu, B. Trasnea, T. Cocias, G. Macesanu. // arXiv:1910.07738v2. – 2020. 28 с. URL: https://arxiv.org/abs/1910.07738v2 (дата обращения 03.04.2020)
319. Lin S-C. The Architectural Implications of Autonomous Driving: Constraints and Acceleration / S-C. Lin, C-H. Hsu, Y. Zhang, M. Skach, M. E. Haque, L. Tang, J. Mars // ASPLOS '18: Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems. – 2018. – P. 751 – 766.
320. Muthanna A. D2D enabled communication system structure based on software defined networking for 5G network. / A. Muthanna, A.A. Ateya, M. Al Balushi, and R. Kirichek // In 2018 International Symposium on Consumer Technologies (ISCT) IEEE. – May. 2018. – P. 41 – 44.
321. Ateya A.A. Chaotic salp swarm algorithm for SDN multi-controller networks. / A.A. Ateya, A. Muthanna, A. Vybornova, A.D. Algarni, A. Abuarqoub, Y. Koucheryavy, A. Koucheryavy //Engineering Science and Technology, an International Journal, – 2019. – № 22(4). – P. 1001 – 1012.
322. Mahmoud M. Distributed Edge Computing to Assist LPWAN: Fog-MEC Model. / M. Mahmoud, A.A. Ateya, A. Muthanna, A. Zaghloul, R. Kirichek, A. Koucheryavy // In The 5th International Conference on Future Networks & Distributed Systems – December 2021. – P.587 – 594.
323. Qiao L. A survey on 5G/6G, AI, and Robotics. / L. Qiao, Y. Li, D. Chen, S. Serikawa, M. Guizani, Z. Lv // Comput. Electr. Eng. – 2021. –№ 9. – P. 107372.
324. Dogra A. A survey on beyond 5G network with the advent of 6G: Architecture and emerging technologies. / A. Dogra, R.K. Jha, S. Jain // IEEE Access – 2020. – № 9. – P. 67512 – 67547.
325. Long Q. Software defined 5G and 6G networks: A survey. / Q. Long, Y. Chen, H. Zhang, X. Lei // Mob. Netw. – Appl. 2019. – P. 1

326. Ge X. 5G software defined vehicular networks. / X. Ge, Z. Li, S. Li // IEEE Commun. Mag. – 2017. – № 55. – P. 87 – 93.
327. Duo R. SDN-based handover approach in IEEE 802.11 p and LTE hybrid vehicular networks. / R. Duo, C. Wu, T. Yoshinaga, Y. Ji // In Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scal-able Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (Smart-World/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Guangzhou, China, 8–12 October 2018; IEEE: Manhattan, NY, USA, – 2018. –P. 1870 – 1875.
328. Muthanna A. A mobile edge computing/software-definednetworking-enabled architecture for vehicular networks. / A. Muthanna, R. Shamilova, A.A. Ateya, A. Paramonov, M. Hammoudeh / Internet Technol. Lett. – 2020.
329. Stoyanova M. A survey on the internet of things (IoT) forensics: challenges, approaches, and open issues. / M. Stoyanova, Y. Nikoloudakis, S. Panagiotakis, E. Pallis, E.K. Markakis // IEEE Communications Surveys & Tutorials, – 2020. – № 22(2). – P. 1191 – 1221.
330. Nižetić, S. Internet of Things (IoT): Opportunities, issues and challenges towards a smart and sustainable future. / S. Nižetić, P. Šolić, D.L.D.I. González-de, L. Patrono // Journal of Cleaner Production, – 2020. – P. 274, 122877.
331. Prasad R. Internet of Things (IoT) and Machine to Machine (M2M) Communication. / R. Prasad, V. Rohokale //In Cyber Security: The Lifeline of Information and Communication Technology Springer, Cham. – 2020. – P. 125 – 141.
332. Tahaei H. The rise of traffic classification in IoT networks: A survey. / H. Tahaei, F. Afifi, A. Asemi, F. Zaki, N.B. Anuar // Journal of Network and Computer Applications, – 2020. – P. 154, 102538.
333. Popli S. Green IoT: A Short Survey on Technical Evolution & Techniques. / S. Popli, R.K. Jha, S Jain // Wireless Personal Communications, – 2021. – P. 1 – 29.
334. Ateya A. End-to-end system structure for latency sensitive applications of 5G. / A. Ateya, M. Al-Bahri, A. Muthanna, A. Koucheryavy // Электросвязь, – 2018. – № 6. – P. 56 – 61.
335. Polepaka S. Internet of Things and Its Applications: An Overview./ S. Polepaka, M. Swami Das, R.P. Ram Kumar // Advances in Cybernetics, Cognition, and Machine Learning for Communication Technologies, – 2020. – P. 67 – 75.

336. Ugwuanyi S. Survey of IoT for developing countries: performance analysis of LoRaWAN and cellular NB-IoT networks. / S. Ugwuanyi, G. Paul, J. Irvine // Electronics, – 2021. – № 10(18). – P. 2224.
337. Al-Shargabi B. Internet of Things: An exploration study of opportunities and challenges. / B. Al-Shargabi, O. Sabri // In 2017 International Conference on Engineering & MIS (ICEMIS). IEEE. – May, 2017. – P. 1 – 4.
338. Ateya, A.A. Study of 5G services standardization: specifications and requirements. / A.A. Ateya, A. Muthanna, M. Makolkina, A. Koucheryavy // In 2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). IEEE. – Nov., 2018. – P. 1 – 6.
339. Аль-Свейти М.А. Система обнаружения и распознавания движущихся биологических объектов для беспилотных автомобилей на основе интеллектуальных граничных вычислений / Аль-Свейти М.А., Мутханна А.С., Бородин А.С., Кучерявый А.Е. // Электросвязь. 2021. № 9. С. 35-41.
340. Li W. A comprehensive survey on machine learning-based big data analytics for IoT-enabled smart healthcare system. / W. Li, Y. Chai, F. Khan, S.R.U. Jan, S. Verma, V.G. Menon, X. Li // Mobile Networks and Applications, – 2021. – P. 1 – 19.
341. Abbasi M. Deep learning for network traffic monitoring and analysis (ntma): A survey. / M. Abbasi, A. Shahraki, A. Taherkordi // Computer Communications. – 2021.
342. Muthanna A. Enabling M2M communication through MEC and SDN. / A. Muthanna, A. Khakimov, A.A. Ateya, A. Paramonov, A. Koucheryavy // In International Conference on Distributed Computer and Communication Networks. Springer, Cham. – September 2018. – P. 95 – 105.
343. Rafique W. Complementing IoT services through software defined networking and edge computing: A comprehensive survey. /W. Rafique, L. Qi, I. Yaqoob, M. Imran, R.U. Rasool, W. Dou // IEEE Communications Surveys & Tutorials, – 2020. – № 22(3). – P. 1761 – 1804.
344. Magesh S. Concepts and contributions of edge computing in internet of things (IoT): A survey. / S. Magesh, J. Indumathi, R. RamMohan, V. Niveditha, P. Prabha // Int. J. Comput. Netw. Appl, – 2020. – № 7. – P. 146 – 156.

345. Akbar A. SDN-Enabled Adaptive and Reliable Communication in IoT-Fog Environment Using Machine Learning and Multiobjective Optimization. / A. Akbar, M. Ibrar, M.A. Jan, A.K. Bashir, L. Wang // IEEE Internet of Things Journal, – 2020. – № 8(5). – P. 3057 – 3065.
346. Ungurean I. Software Architecture of a Fog Computing Node for Industrial Internet of Things. / I. Ungurean, N.C. Gaitan // Sensors, – 2021. – № 21(11). – P. 3715.
347. Laha S. How Can Machine Learning Impact on Wireless Network and IoT? – A Survey. / S. Laha, N. Chowdhury, R. Karmakar // In 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT) IEEE. – July, 2020. – P. 1 – 7.
348. Li, Y., & Tu, W. (2020, August). Traffic Modelling for IoT Networks: A Survey. In Proceedings of the 2020 10th International Conference on Information Communication and Management (pp. 4-9).
349. Shahraki A. Active Learning for Network Traffic Classification: A Technical Survey./ A. Shahraki, M. Abbasi, A. Taherkordi, A.D. Jurcut // arXiv preprint arXiv:2106.06933. – 2021.
350. Khedkar S.P. Prediction of Traffic Generated by IoT Devices Using Statistical Learning Time Series Algorithms. / S.P. Khedkar, R.A. Canessane, M.L. Najafi // Wireless Communications and Mobile Computing, – 2021.
351. Khodaverdian Z. Combination of Convolutional Neural Network and Gated Recurrent Unit for Energy Aware Resource Allocation. / Z. Khodaverdian, H. Sadr, S.A. Edalatpanah, M.N. Solimandarabi // arXiv preprint arXiv:2106.12178. – 2021.
352. Chen Y. A convolutional neural network for traffic information sensing from social media text. / Y. Chen, Y. Lv, X. Wang, F.Y. Wang // In 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE. – October, 2017. – P. 1 – 6.
353. Lim H.K. Packet-based network traffic classification using deep learning. / H.K. Lim, J.B. Kim, J.S. Heo, K. Kim, Y.G. Hong, Y.H. Han // In 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC). IEEE. – February, 2019. – P. 046 – 051).
354. Filus K. Long-Range Dependent Traffic Classification with Convolutional Neural Networks Based on Hurst Exponent Analysis. / K. Filus, A. Domański, J. Domańska, D. Marek, J. Szyguła // Entropy, – 2020. – № 22(10). – P. 1159.

355. Chien W.C. A lightweight model with spatial–temporal correlation for cellular traffic prediction in Internet of Things. / W.C. Chien, Y.M. Huang // The Journal of Supercomputing. – 2021. – P. 1 – 17.

356. Kim M. Tor traffic classification from raw packet header using convolutional neural network. / M. Kim, A. Anpalagan // In 2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII). IEEE. – July, 2018. – P. 187 – 190.

357. Lopez-Martin M. Neural network architecture based on gradient boosting for IoT traffic prediction. / M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas // Future Generation Computer Systems. – 2019. – P. 100, 656 – 673.

358. Ko T. Network prediction with traffic gradient classification using convolutional neural networks. / T. Ko, S.M. Raza, D.T. Binh, M. Kim, H. Choo // In 2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM). IEEE. – January, 2020. – P. 1 – 4.

359. Hu L. If-rans: intelligent traffic prediction and cognitive caching toward fog-computing-based radio access networks. / L. Hu, Y. Miao, J. Yang, A. Ghoneim, M.S. Hossain, M. Alrashoud // IEEE Wireless Communications. – 2020. – № 27(2). – P. 29 – 35.

360. Patil S.A. Prediction of IoT Traffic Using the Gated Recurrent Unit Neural Network-(GRU-NN-) Based Predictive Model. / S.A. Patil, L.A. Raj, B.K. Singh // Security and Communication Networks. – 2021.

361. Abdellah A.R. IoT traffic prediction using multi-step ahead prediction with neural network. / A.R. Abdellah, O.A.K. Mahmood, A. Paramonov, A. Koucheryavy, A. // In 2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). IEEE. – October, 2019. – P. 1 – 4.

362. Carela-Español, V. Is our ground-truth for traffic classification reliable?. / V. Carela-Español, T. Bujlow, P. Barlet-Ros // In International Conference on Passive and Active Network Measurement. Springer, Cham. – March, 2014. – P. 98-108.

363. Traffic Classification at the Universitat Politècnica de Catalunya (UPC) [Online]. Available: http://www.cba.upc.edu/monitoring/traffic-classification

364. Benson T. Network traffic characteristics of data centers in the wild. /T. Benson, A. Akella, D.A. Maltz. // In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement. – November, 2010. – P. 267 – 280.

365. UNI2 dataset. [Online]. Available: http://pages.cs.wisc.edu/~tbenson/IMC10_Data.html
366. Muthanna A. Secure and reliable IoT networks using fog computing with software-defined networking and blockchain. / A. Muthanna, A. Ateya, A. Khakimov, I. Gudkova, A. Abuarqoub, K. Samouylov, A. Koucheryavy // Journal of Sensor and Actuator Networks. – 2019. – № 8(1). – P. 15.

Приложение А

Достижимая скорость выгрузки, $R_{b\text{-}Di\text{-}nts}$, может быть рассчитана следующим образом.

$$R_{b-Di-nts} = B\log_2\left(1+\frac{\gamma_{nts}{}^{Di}\left(x_{Di}{}^{nts},y_{Di}{}^{nts}\right)p}{\sigma^2}\right) \quad (49)$$

$$R_{b-Di-nts} = B\log_2\left(1+\frac{\frac{\gamma_0}{\left(x_{Di}{}^{nts}-x_{Ij}{}^{nts}\right)^2+\left(y_{Di}{}^{nts}-y_{Ij}{}^{nts}\right)^2}p}{\sigma^2}\right) \quad (50)$$

$$R_{b-Di-nts} = B\log_2\left(1+\frac{SNR}{\left(x_{Di}{}^{nts}-x_{Ij}{}^{nts}\right)^2+\left(y_{Di}{}^{nts}-y_{Ij}{}^{nts}\right)^2\sigma^2}\right), SNR=\frac{\gamma_0 p}{\sigma^2} \quad \forall\, n \in \mathbb{R},\ D_i \in D \quad (51)$$

$$T_{u-tx} = {}^{S}/_{R_{b-Di-nts}} \quad (52)$$

$$T_{dw-tx} = {}^{Z}/_{R_{b-Di-nts}} \quad (53)$$

$$T_{pro} = {}^{\mathbb{C}}/_{L_{I_{i,D_j}}} \quad (54)$$

Приложение Б. Актыреализации диссертационных исследований

МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,
СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ
ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)

Юридический адрес: набережная реки Мойки,
д. 61, литера А, Санкт-Петербург, 191186

Почтовый адрес: пр. Большевиков, д. 22, корп. 1,
Санкт-Петербург, 193232
Тел.(812) 3263156, Факс: (812) 3263159
http://sut.ru
E-mail: rector@sut.ru
ОКПО 01179934 ОГРН 1027809197635
ИНН 7808004760 КПП 784001001
ОКТМО 40909000

31.08.2023 № 2004/54
на №_____ от _____

УТВЕРЖДАЮ

Проректор по научной работе
канд. техн.наук

Брусиловский
Сергей Александрович

Акт

о внедрении научных результатов,

полученных в диссертационной работе Аммара Салеха Али Мутханны “ Разработка и исследовнаие комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений ”

Комиссия в составе декана факультета Инфокоммуникационных сетей и систем к.т.н., Д.В.Окуневой, доцента кафедры сетей связи и передачи данных к.т.н., PhD Р.А.Дунайцевом и заведующей лаборатории кафедры сетей связи и передачи данных О.И.Ворожейкиной составила настоящий акт в том, что научные результаты, полученные Аммаром Салехом Али Мутханной в диссертации "Разработка и исследовнаие комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений”, использованы:

1. При чтении лекций и проведении практических занятий для бакалавров по дисциплине «Интернет Вещей и самоорганизующиеся сети» (Рабочая Программа регистрационный номер № 22.05/307-Д), раздел Программы:

- Архитектура сети
- Сети связи шестого поколения и их отличие от сетей связи пятого поколения
- Использование БПЛА для организации сетей связи

2. При чтении лекций и проведении практических занятий для бакалавров по дисциплине «Современные проблемы науки в области инфокоммуникаций» (Рабочая Программа регистрационный номер № 22.05/469-Д), раздел Программы:

- Самоорганизующиеся сети, услуги и приложения таких сетей
- Концепции развития сетей связи. Текущее состояние развития сетей. Прогнозы развития сетей связи

3. При чтении лекций и проведении практических занятий для бакалавров по дисциплине "Сети связи для цифровой экономики" (Рабочая Программа регистрационный номер № 22.05/405-Д), раздел Программы:

- Тактильный Интернет, сети связи с ультра малыми задержками, сети связи 2030
- Децентрализация сетей связи и сокращения цифрового разрыва
- Искусственный интеллект в сетях связи.

В указанных дисциплинах используются следующие новые научные результаты, полученные Аммаром Салехом Али Мутханной в диссертационной работе:

1. Метод построения мультиконтроллерной сети, основанный на интегральном решении задач по размещению контроллеров в мультиконтроллерных сетях, базирующийся на метаэвристическом алгоритме вследствие сложности решаемых задач, и алгоритме балансировки нагрузки, позволяет обеспечить наилучшее использование ресурсов контроллеров в таких сетях.

2. Модифицированный алгоритм хаотического роя сальп для использования в иерархических кластерных сетях clus-CSSA позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1мс до 10мс по сравнению как с широко известными метаэвристическими алгоритмами роя частиц PSO (Particle Swarm Optimization) и серого волка GWO (Grey Wolf Optimization), так и с предыдущей версией хаотического алгоритма роя сальп CSSA (Chaotic Salp Swarm Algorithm). При этом для наиболее сложного случая задержки величиной в 1мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза.

3. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности, в котором процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, определяющий сложность вычисляемой задачи, а по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры

выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА. При этом результаты моделирования доказали, что обеспечиваются уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений и на 30-40% по сравнению с сетью с использованием только наземных граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш около 10% по сравнению с использованием неоптимизированного алгоритма.

4. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение энергопотребления до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений. Кроме того, использование оптимизации на основе метаэвристического хаотического роя сальп дает дополнительный выигрыш в 5-10% по сравнению с использованием неоптимизированного алгоритма.

5. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение доли заблокированных задач по выгрузке трафика в десятки раз по сравнению с сетью без использования технологий граничных вычислений, в разы по сравнению с сетью с использованием только наземных граничных вычислений. Использование оптимизации на основе метаэвристического хаотического роя сальп не дает практически значимого эффекта по сравнению с неоптимизированным алгоритмом. Кроме того, определены зависимости значений задержки, энергопотребления и доли заблокированных задач по выгрузке трафика от плотности сети.

6. Метод построения сети с интеграцией технологий MEC, SDN и D2D для поддержки приложений беспилотных автомобилей и алгоритм кластеризации на основе взаимодействий D2D для транспортных средств в непокрытых зонах и для выгрузки трафика сети в регионах с интенсивным движением дает 74% прироста производительности системы в терминах вероятности блокировки задач.

7. Метод прогнозирования на основе CNN - LTP-CNN, который предсказывает трафик сети IoT по информации о состояния сети за предыдущий интервал времени, реализующийся на туманных узлах, которые представляют собой основную часть сетей IoT/5G позволяет предсказывать трафик сети IoT с точностью около 90%.

8. Метод размещения SDN-контроллеров в мультиконтроллерных сетях, который предусматривает размещение контроллеров на мобильных узлах сетей VANET, например, автобусах, для обеспечения связи в плотных и сверхплотных сетях 6G и взаимодействия с туманной средой устройств сети, позволяет уменьшить задержку на 60% по сравнению с традиционными моделями граничных вычислений, а также снизить потребляемую энергию на 72% по сравнению с методом Fog-MEC.

Полученные научные результаты использованы при выполнении Соглашения о предоставлении из федерального бюджета гранта в форме субсидий, выделяемого для государственной поддержки научных исследований, проводимых под руководством ведущих ученых в российских образовательных организациях высшего образования, научных учреждениях и государственных научных центрах Российской Федерации от «06» июля 2022 г. № 075-15-2022-1137 по приоритетному направлению научно-технологического развития Российской Федерации 20а - Переход к передовым цифровым, интеллектуальным производственным технологиям, роботизированным системам, новым материалам и способам конструирования, создание систем обработки больших объемов данных, машинного обучения и искусственного интеллекта.

Декан факультета ИКСС,
к.т.н. Д.В.Окунева

Доцент кафедры ССиПД,
к.т.н., PhD Р.А.Дунайцев

Заведующая лабораторией
кафедры ССиПД О.И.Ворожейкина

ГИПРОСВЯЗЬ
ОПЫТ МАСШТАБ ПЕРСПЕКТИВА
Публичное акционерное общество «ГИПРОСВЯЗЬ»

Утверждаю
Заместитель Генерального директора
ПАО «ГИПРОСВЯЗЬ»
А.Б. Васильев

АКТ

о внедрении результатов диссертационной работы Аммара Салеха Али Мутханны на тему «Разработка и исследование комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений»

Настоящим актом подтверждаем, что научные результаты диссертационной работы Аммара Салеха Али Мутханны «Разработка и исследование комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений», представленной на соискание ученой степени доктора технических наук, внедрены в ПАО “ГИПРОСВЯЗЬ” при разработке «Методики планирования мульти контроллерных программно-конфигурируемых сетей SDN».

При разработке были использованы следующие новые научные результаты из диссертации А.С.А.Мутханны:

- Метод построения мульти контроллерной сети, основанный на интегральном решении задач по размещению контроллеров в мульти контроллерных сетях, базирующийся на мета эвристическом алгоритме

вследствие сложности решаемых задач, и алгоритме балансировки нагрузки, позволяет обеспечить наилучшее использование ресурсов контроллеров в таких сетях.

- Модифицированный алгоритм хаотического роя сальп для использования в иерархических кластерных сетях clus-CSSA позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1мс до 10мс по сравнению как с широко известными мета эвристическими алгоритмами роя частиц PSO (Particle Swarm Optimization) и серого волка GWO (Grey Wolf Optimization), так и с предыдущей версией хаотического алгоритма роя сальп CSSA (Chaotic Salp Swarm Algorithm). При этом для наиболее сложного случая задержки величиной в 1мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза.

Председатель комиссии:

Директор департамента _____ А.А. Иванов

Члены комиссии:

Главный специалист _____ Ю.А. Нопина

Экз. _ из 3

МИНИСТЕРСТВО
ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ
И МАССОВЫХ КОММУНИКАЦИЙ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ
«Ордена Трудового Красного Знамени
Российский научно-исследовательский
институт радио имени М.И. Кривошеева»
(ФГБУ НИИР)
Почтовый адрес: Казакова ул., д. 16, Москва, 105064
Телефон: (495) 647-17-77, факс: (499) 261-00-90
E-mail: info@niir.ru, http://www.niir.ru
ОКПО 56622156, ОГРН 1227700388827
ИНН/КПП 9709082715/770901001

_№ _

На № _____от _____

«УТВЕРЖДАЮ»

Первый заместитель генерального директора ФГБУ НИИР, кандидат технических наук

М.Ю. Сподобаев

«___» _____ 2023 г.

АКТ

внедрения результатов диссертационной работы Мутханны Аммара Салеха Али на тему «Разработка и исследование комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений», представленной на соискание ученой степени доктора технических наук по специальности 2.2.15– Системы, сети и устройства телекоммуникаций

Комиссия в составе:

Председатель комиссии – зам. директора НТЦ Анализа ЭМС С.Ю. Пастух, к.т.н.

Члены комиссии – зам. начальника отдела НТЦ Анализа ЭМС Е.В. Тонких, к.т.н.

начальник отдела НТЦ Анализа ЭМС Н.В. Варламов,

установила, что в диссертационной работе Мутханны Аммара Салеха Али, доцента кафедры сетей связи и передачи данных Санкт-Петербургского государственного университета им. проф. М.А. Бонч-Бруевича, к.т.н. на тему «Разработка и исследование комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений» получены новые научные результаты,

которые внедрены в 2022 - 2023 гг. в рамках выполнения государственных контрактов по научно-техническому и методическому обеспечению выполнения Министерством цифрового развития, связи и массовых коммуникаций функций администрации связи Российской Федерации в части, касающейся международно-правовой защиты интересов Российской Федерации в области радиосвязи и электросвязи в виде предложений (вкладов) от имени администрации связи Российской Федерации (Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации) в Сектор стандартизации электросвязи Международного союза электросвязи (МСЭ-Т).

Эти вклады представлены на заседаниях Исследовательской комиссии 11 (ИК11) «Требования к сигнализации, протоколы, спецификации испытаний и борьба с контрафактными устройствами электросвязи/ИКТ» МСЭ-Т (май 2023 года) по обновлению базового текста проекта Рекомендации МСЭ-Т UHD-T «Тестирование трехмерных сетей IoT сверхвысокой плотности (The testing of 3D ultra-high density IoT networks Session-layer network coding protocol for multicast data transmission)» (вклад C189) и Исследовательской комиссии 13 «Сети будущего и появляющиеся сетевые технологии» МСЭ-Т (март 2023 года) по базовому тексту нового направления исследования МСЭ-Т Y.ORCH-DIS «Сетевые архитектуры для сельских сетей с использованием оркестраторов (Network architectures for rural networks using orchestrators)» (вклад C392).

Вышеуказанные вклады определили позицию Администрации связи Российской Федерации в области сетей связи пятого и последующих поколений на заседаниях ИК11 МСЭ-Т по вопросу тестированию сверхплотных сетей в трехмерном пространстве и ИК13 МСЭ-Т по вопросу возможного построения сетей для сельской местности на основе распределенных оркестраторов и являются основой для разработки соответствующих рекомендаций Сектора стандартизации электросвязи Международного союза электросвязи.

Председатель комиссии _____ С.Ю. Пастух

Члены комиссии _____ Е.В. Тонких

_____ Н.В. Варламов

Подписи С.Ю. Пастуха, Тонких Е.В., Н.В. Варламова заверяю.

Начальник отдела кадров ФГБУ НИИР Е.П. Буянова

РӘСӘЙ ФЕДЕРАЦИЯҺЫ ФӘН һӘМ
ЮҒАРЫ БЕЛЕМ БИРЕҮ МИНИСТРЛЫҒЫ
ЮҒАРЫ БЕЛЕМ БИРЕҮ
ФЕДЕРАЛЬ ДӘҮЛӘТ БЮДЖЕТ МӘҒАРИФ
УЧРЕЖДЕНИЕҺЫ

«ӨФӨ ФӘН һӘМ ТЕХНОЛОГИЯЛАР УНИВЕРСИТЕТЫ»
(Өфө университеты)
Зәки Вәлиди урамы, 32, Өфө ҡалаһы, БР, 450076

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО
ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«УФИМСКИЙ УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ»
(Уфимский университет, УУНиТ)
Заки Валиди ул., 32, Уфа, РБ, 450076

тел.: 8 (347) 272-63-70 факс: (347) 273-67-78 e-mail: rector@uust.ru https://uust.ru
ОКПО 79067778 ОГРН 1220200037474 ИНН/КПП 0274975591/027401001

от _____№ _____
на № _____ от _____

«УТВЕРЖДАЮ»
Проректор
по развитию образования
Ю. В. Рахманова

АКТ

внедрения результатов диссертационной работы Аммара Салеха Али Мутханны на тему «Разработка и исследовнаие комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений», представленной на соискание ученой степени доктора технических наук по специальности 2.2.15– Системы, сети и устройства телекоммуникаций

Комиссия в составе и .о. декана факультета информатики и робототехники к.т.н., Доцента А. С. Ковтуненко, и. о. заведующего кафедрой телекоммуникационных систем, к. т. н., Доцента Р. В. Кутлуярова, доцента кафедры телекоммуникационных систем, к.т.н., Доцента Г. С. Воронкова составила настоящий акт в том, что научные результаты, полученные Аммаром Салехом Али Мутханной в диссертации «Разработка и исследовнаие комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений», использованы в учебном процессе Уфимского университета науки и технологий, а именно:

1. При чтении лекций и проведении практических занятий по дисциплине «Коммуникационные технологии для интернета вещей» (для обучающихся по направлению подготовки магистров 11.04.02 «Инфокоммуникационные технологии и системы связи», профиль «Технологии беспроводной связи и интернет вещей»)
2. При чтении лекций и проведении практических занятий по дисциплине «Введение в интернет вещей» (для обучающихся по направлению подготовки магистров 11.04.02 «Инфокоммуникационные технологии и системы связи», профиль «Технологии беспроводной связи и интернет вещей»)
3. При чтении лекций и проведении практических занятий по дисциплине «Проектирование и техническая эксплуатация систем передачи данных» (для

обучающихся по направлению подготовки магистров 11.04.02 «Инфокоммуникационные технологии и системы связи», профиль «Технологии беспроводной связи и интернет вещей»).

4. При чтении лекций и проведении практических занятий по дисциплине «Микропроцессорные устройства и системы управления объектами организационно-технических систем» (для обучающихся по направлению подготовки специалистов 27.05.01 «Специальные-организационно-технические системы», специализация ««Информационно-аналитическая деятельность в специальных организационно-технических системах»)

При проведении занятий по указанным дисциплинам используются следующие новые научные результаты, полученные Аммаром Салехом Али Мутханной в диссертационной работе:

1. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности, в котором процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, определяющий сложность вычисляемой задачи, а по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА, на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА. При этом результаты моделирования доказали, что обеспечиваются уменьшение задержки до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений и на 30-40% по сравнению с сетью с использованием только наземных граничных вычислений. Кроме того, использование оптимизации на основе мета эвристического хаотического роя сальп дает дополнительный выигрыш около 10% по сравнению с использованием неоптимизированного алгоритма.

2. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение энергопотребления до более, чем в 2 раза по сравнению с сетью без использования технологий граничных вычислений. Кроме того, использование оптимизации на основе мета эвристического хаотического роя сальп дает дополнительный выигрыш в 5-10% по сравнению с использованием неоптимизированного алгоритма.

3. Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей высокой и сверхвысокой плотности обеспечивают уменьшение доли заблокированных задач по выгрузке трафика в десятки раз по сравнению с сетью без использования технологий граничных вычислений, в разы по сравнению с сетью с использованием только наземных граничных вычислений. Использование оптимизации на основе мета эвристического хаотического роя сальп не дает практически значимого эффекта по сравнению с неоптимизированным алгоритмом. Кроме того, определены зависимости значений задержки, энергопотребления и доли заблокированных задач по выгрузке трафика от плотности сети.

4. Метод прогнозирования на основе CNN - LTP-CNN, который предсказывает трафик сети IoT по информации о состояния сети за предыдущий интервал времени, реализующийся на туманных узлах, которые представляют собой основную часть сетей IoT/5G позволяет предсказывать трафик сети IoT с точностью около 90%.

и. о. декана факультета
информатики и робототехники,
к.т.н., Доцент  А. С. Ковтуненко

и. о. заведующего кафедрой
телекоммуникационных систем,
к.т.н., Доцент  Р. В. Кутлуяров

доцент кафедры
телекоммуникационных систем,
к. т. н., Доцент  Г. С. Воронков

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Российский университет дружбы народов
имени Патриса Лумумбы» (РУДН)

ул. Миклухо-Маклая, д. 6, Москва, Россия, 117198
ОГРН 1027739189323; ОКПО 02066463; ИНН 7728073720

Телефон: +7495 434 53 00, факс: +7495 433 15 11
www.rudn.ru; rudn@rudn.ru

1 сентября 20 23
№ 0200-54/С3/23-09-01

УТВЕРЖДАЮ
Первый проректор,
проректор по научной работе,
доктор медицинских наук, профессор,
член-корреспондент РАН

Костин Андрей Александрович

«1» сентября 2023 г.

АКТ

внедрения результатов диссертационной работы
Мутханны Аммара Салеха Али на тему
«Разработка и исследование комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений»,
представленной на соискание ученой степени доктора технических наук
по специальности 2.2.15 – Системы, сети и устройства телекоммуникаций

Настоящим актом подтверждаем, что научные результаты диссертационной работы Мутханны Аммара Салеха Али «Разработка и исследование комплекса моделей и методов интеграции граничных вычислений в сетях связи пятого и шестого поколений», представленной на соискание ученой степени доктора технических наук, внедрены в федеральном государственном автономном образовательном учреждении высшего образования «Российский университет дружбы народов имени Патриса

Лумумбы» при выполнении проектов № 025304-0-000 «Исследование и разработка моделей построения сетей связи шестого поколения» (2021-2023) и № 025319-2-000 «Разработка моделей и алгоритмов нарезки радиоресурсов и приоритетного доступа в беспроводной сети 6G» (2023) на базе научного центра моделирования беспроводных сетей новых поколений института компьютерных наук и телекоммуникаций РУДН.

При выполнении проектов № 025304-0-000 и № 025319-2-000 были использованы следующие новые научные результаты из диссертации А.С.А. Мутханны:

- Метод построения мульти контроллерной сети, основанный на интегральном решении задач по размещению контроллеров в мульти контроллерных сетях, базирующийся на мета эвристическом алгоритме вследствие сложности решаемых задач, и алгоритме балансировки нагрузки, позволяет обеспечить наилучшее использование ресурсов контроллеров в таких сетях.
- Модель и метод интеграции граничных вычислений в структуру сети «воздух-земля» для сетей Интернета Вещей (internet of things, IoT) высокой и сверхвысокой плотности, в котором процедура выгрузки трафика является трехуровневой, причем на оконечных устройствах используется программный профилировщик, определяющий сложность вычисляемой задачи, а по результатам его работы механизм принятия решения определяет необходимость выгрузки трафика. Кроме того, на втором уровне процедуры выгрузки трафика сервер БПЛА (беспилотный летательный аппарат), на который выгружается трафик, может принять решение в условиях недостаточного объема ресурсов выгрузить трафик на сервер другого БПЛА.
- Метод построения сети с интеграцией технологий MEC (multi-access edge computing), SDN (software-defined networking) и D2D (device-to-device) для поддержки приложений беспилотных автомобилей и алгоритм кластеризации на основе взаимодействий D2D для транспортных средств в непокрытых зонах и для выгрузки трафика сети в регионах с интенсивным движением.
- Метод прогнозирования на основе сверточной нейронной сети – LTP-CNN (long-term prediction convolutional neural network), который предсказывает трафик сети IoT по информации о состояния сети за предыдущий интервал времени, реализующийся на туманных узлах, которые представляют собой основную часть сетей IoT/5G.

01.09.2023

Директор института компьютерных наук
и телекоммуникаций РУДН,
доктор технических наук, профессор
Самуйлов Константин Евгеньевич