Федеральное государственное автономное образовательное учреждение высшего образования
«РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ ИМЕНИ ПАТРИСА ЛУМУМБЫ»

На правах рукописи

Кущазли Анна Ивановна

МОДЕЛИ МАССОВОГО ОБСЛУЖИВАНИЯ ДЛЯ АНАЛИЗА ЭФФЕКТИВНОСТИ МИГРАЦИИ СЕРВИСОВ В ГРАНИЧНЫХ ОБЛАЧНЫХ ВЫЧИСЛЕНИЯХ

Специальность 1.2.3 – Теоретическая информатика, кибернетика
(по физико-математическим наукам)

Диссертация

на соискание ученой степени
кандидата физико-математических наук

Научный руководитель
доктор физико-математических наук, доцент
Кочеткова Ирина Андреевна

Москва - 2026

Оглавление

Введение

Актуальность темы исследования.

Облачные вычисления опираются на централизованные дата-центры, в которых вычислительные ресурсы предоставляются в виде виртуальных машин, размещенных на физических серверах. Быстрый рост объемов трафика и требований к задержке обусловил появление гранично-облачной архитектуры, в частности технологии периферийных вычислений с множественным доступом (англ. multi-access edge computing, MEC), предполагающей размещение сервисов в непосредственной близости к пользователю на вычислительных узлах в сети радиодоступа. Обе архитектуры – облачная и гранично-облачная – являются базовыми для предоставления иммерсивных сервисов дополненной и виртуальной реальности, облачного гейминга в сетях пятого и шестого поколений (5G/6G), где критически важно обеспечение качества обслуживания (англ. quality of service, QoS).

В облачной инфраструктуре миграция – это перенос виртуальной машины вместе с выполняемыми задачами между физическими серверами; основными показателями эффективности являются вероятность миграции и высвобождаемая пропускная способность сервера. Алгоритмы миграции различаются, в частности, моментом оценки занятости ресурсов – до или после размещения поступившей задачи, – что существенно влияет на результат перераспределения. В гранично-облачной архитектуре миграция – это перемещение сервиса и перераспределение пользователей между MEC-узлом и облачным сервером; основным показателем является суммарная межконцевая задержка (англ. end-to-end delay, E2E-delay) по всем пользователям. Ограниченность ресурсов MEC-узла при одновременном наличии нескольких сервисов приводит к конкуренции за возможность обслуживания на MEC-узле: размещение пользователей одного сервиса уменьшает доступные ресурсы для остальных и вынуждает их обслуживать пользователей в облачном сервере с большей задержкой. В обоих случаях для анализа показателей эффективности миграции используются модели массового обслуживания.

Политика миграции определяется тремя компонентами: целевой функцией – критерием оптимальности, например, минимизация занятой пропускной способности или суммарной межконцевой задержки, алгоритмом принятия решения – правилом перемещения виртуальных машин или пользователей сервиса, и моментом принятия

решения – событием, при котором система пересматривает текущее размещение. Выбор момента принятия решения играет ключевую роль: некорректный выбор момента приводит к осцилляциям – циклическим перемещениям виртуальных машин / пользователей между серверами.

Существующие модели миграции в облачных системах не формализуют ограничения на момент принятия решения в рамках модели массового обслуживания, а проблема осцилляций решается двойными порогами загрузки, задержками между миграциями, предсказанием нагрузки. Модели миграции в MEC-системах формулировались для одного сервиса с пороговой политикой или как марковский процесс принятия решений и решались численно без получения аналитического решения. Коррелированный характер входного потока заявок, описываемый марковски-модулированным пуассоновским процессом (англ. Markov-modulated poisson process, MMPP), в существующих моделях миграции не учитывался, а политика миграции не адаптировалась к фазе потока – не предусматривались различные правила принятия решений в фазах высокой и низкой интенсивности поступления заявок. При этом практическое применение MMPP требует оценивания его параметров по реальному сетевому трафику, что также не рассматривалось в контексте моделей миграции.

В результате возникает научная проблема – разработка моделей массового обслуживания для анализа и расчета показателей эффективности миграции виртуальных машин в облачной инфраструктуре и пользователей сервисов в гранично-облачной архитектуре с учетом оптимальных политик миграции, формализованных механизмов предотвращения осцилляций, коррелированного характера входного потока заявок.

Степень разработанности темы исследования.

Значительный вклад в развитие данной тематики внесли следующие российские и зарубежные ученые и исследователи. В области методов анализа показателей эффективности сетей связи, включая облачные вычисления и граничных облачных вычислений: Андреев С.Д., Барабанова Е.А., Бегишев В.О., Волков А.Н., Вытовтов К.А., Гольдштейн Б.С., Киричек Р.В., Крук Е.А., Кучерявый А.Е., Кучерявый Е.А., Кулябов Д.С., Ляхов А.И., Маколкина М.А., Молчанов Д.А., Мутханна А.С.А., Нетес В.А., Орлов Ю.Н., Парамонов А.И., Пшеничников А.П., Росляков А.В., Смелянский Р.Л., Хакимов А.А., Хоров Е.М., Яновский Г.Г., Ateya A.A., Buyya R., Dustdar S., Shi W., Taleb T., Wang S. и др.

Отметим ученых, которые внесли значительный вклад в развитие методов математической теории телетрафика и теории массового обслуживания: Башарин Г.П., Бочаров П.П., Гайдамака Ю.В., Горцев А.М., Ефросинин Д.В., Зейфман А.И., Зорин А.В., Ибрагимов Б.Г., Ивницкий В.А., Карташевский В.Г., Лапатин И.Л., Меликов А.З., Пауль С.В., Печинкин А.В., Разумчик Р.В., Рыков В.В., Самуйлов К.Е., Сатин Я.А., Соколов Н.А., Сопин Э.С., Степанов С.Н., Терпугов А.Ф., Тюрликов А.М., Фархадов М.П., Федоткин М.А., Цитович И.И., Цициашвили Г.Ш., Шнепс М.А. и др., включая по моделям с коррелированными входными потоками: Вишневский В.М., Дудин А.Н., Клименок В.И., Меликов А.З., Моисеев А.Н., Моисеева С.П., Морозов Е.В., Назаров А.А., Наумов В.А., Нежельская Л.А., Пауль С.В., Румянцев А.С., Семенова О.В., Chakravarthy S.R., Fischer W., Latouche G., Lucantoni D.M., Meier-Hellstern K.S., Neuts M.F., Ramaswami V. и др.

Цель исследования состоит в разработке моделей миграции виртуальных машин в облачной инфраструктуре и сервисов в гранично-облачной архитектуре для анализа и расчета вероятностно-временных показателей качества обслуживания пользователей и эффективности миграции.

Достижение сформулированной цели достигается путем решения следующих **задач исследования:**

1. Разработка моделей миграции виртуальных машин в облачной инфраструктуре и сервисов между граничным и облачными серверами в виде систем массового обслуживания с политиками миграции на основе критериев занятой пропускной способности и суммарной межконцевой задержке и принятием решения в моменты изменения состояний системы.
2. Разработка алгоритмов для анализа и расчета показателей эффективности миграции, в том числе вероятности миграции, средней высвобождаемой пропускной способности сервера, средней суммарной межконцевой задержки, с учетом коррелированного характера входного потока заявок

Научная новизна результатов исследования состоит в следующем:

1. Модель миграции виртуальных машин в облачной инфраструктуре реализует перемещение всех заявок класса между группами приборов по алгоритмам, минимизирующим занятую пропускную способность серверов, с оценкой занятости приборов до и после размещения поступившей заявки. Для

предотвращения осцилляций решение о миграции принимается только в момент поступления новой заявки соответствующего класса. Ранее основным критерием являлась загрузка процессора и энергопотребление, а проблема осцилляций решалась без формализации ограничений на момент принятия решения в модели массового обслуживания – двойными порогами загрузки, задержками между миграциями, предсказанием нагрузки.

2. Модель миграции сервисов в гранично-облачной архитектуре реализует перемещение заявок между общей группой приборов с эксклюзивным обслуживанием одного класса и индивидуальными приборами по политике, минимизирующей суммарную межконцевую задержку. Оптимальное распределение заявок получено аналитически в виде функции от числа заявок в системе, а стационарное распределение вероятностей состояний имеет мультипликативный вид. Ранее задача миграции сервисов в MEC-системах формулировалась для одного сервиса с пороговой политикой по числу пользователей или как марковский процесс принятия решений и решалась численно методом итерации без получения аналитического решения в мультипликативном виде.
3. Модель миграции сервиса в гранично-облачной архитектуре учитывает коррелированный характер входного потока заявок пользователей в виде MMPP и реализует адаптивную политику миграции, зависящую от фазы потока: в фазе высокой интенсивности решение пересматривается при каждом событии, в фазе низкой интенсивности – только при поступлении заявки или смене фазы. Ранее модели миграции сервисов в MEC-системах исследовались с пуассоновским входным потоком и без адаптации политики миграции к фазе потока.

Теоретическая значимость работы. Теоретическая значимость результатов работы обоснована тем, что доказаны утверждения, позволяющие вычислять стационарные распределения вероятностей состояний моделей миграции виртуальных машин и сервисов, оптимальные политики миграции и показатели эффективности обслуживания в облачной инфраструктуре и гранично-облачной архитектуре; применительно к проблематике диссертации результативно использован комплекс методов математической теории телетрафика, теории массового обслуживания, марковских случайных процессов, матричных аналитических методов и статистического

оценивания параметров входного потока; изложены различные схемы миграции виртуальных машин и сервисов, критерии принятия решения о миграции – по занятой пропускной способности и суммарной межконцевой задержке, а также адаптивная политика, зависящая от фазы входного потока; изучено влияние на эффективность миграции коррелированного характера входного потока заявок в виде MMPP.

Практическая значимость работы. Заключается в том, что разработанные формулы и алгоритмы могут быть применены сотовыми операторами и провайдерами облачных и граничных сервисов при развертывании и работе беспроводных сетей следующих поколений. Они позволят наиболее эффективно использовать доступную пропускную способность и вычислительные ресурсы граничных узлов, а также учитывать требования к обслуживанию (QoS) трафика. Таким образом, работа может предоставить рекомендации для выбора оптимальной политики миграции с целью минимизации задержки в граничных облачных вычислениях.

Методология и методы исследования. Для решения поставленных в работе задач исследования использовались методы математической теории телетрафика, теории массового обслуживания, марковских случайных процессов, теории вероятностей, математической статистики.

Положения, выносимые на защиту.

1. Модель миграции виртуальных машин в облачной инфраструктуре по алгоритмам, минимизирующим занятую пропускную способность серверов, позволяет рассчитать показатели эффективности миграции, такие как вероятность миграции виртуальной машины и среднюю высвобождаемую пропускную способность сервера.
2. Модель миграции сервисов в гранично-облачной архитектуре по политике, минимизирующей суммарную межконцевую задержку по всем пользователям, и стационарное распределение в мультипликативном виде применимы для расчета показателей эффективности миграции, которые зависят от распределения пользователей между граничным и облачным серверами и размещенного на граничном узле сервиса.
3. Модель миграции сервиса в гранично-облачной архитектуре по политике, зависящей от фазы входного MMPP-потока, и матричный алгоритм расчета стационарного распределения позволяют рассчитать среднюю суммарную

межконцевую задержку по всем пользователям при коррелированном входном потоке заявок.

Степень достоверности результатов работы. Достоверность полученных результатов подтверждается тем, что теория построена на известных методах теории массового обслуживания, марковских случайных процессов и матричных аналитических методов, используемых при доказательствах утверждений; идея базируется на анализе и обобщении передового опыта в области миграции виртуальных машин в облачной инфраструктуре и миграции сервисов в гранично-облачной архитектуре; установлено качественное совпадение частных случаев разработанных автором моделей с известными моделями теории массового обслуживания.

Апробация результатов работы.

Основные положения работы были апробированы на следующих конференциях: International Conference on Next Generation Wired/Wireless Networks and Systems, NEW2AN (2025: Абу-Даби, ОАЭ), International Conference on Information Technologies and Mathematical Modelling, ITMM (2024: Карши, Узбекистан), International Conference on Information, Control, and Communication Technologies, ICCT (2024: Владикавказ), International Conference on Computer-Aided Technologies in Applied Mathematics, ICAM (2024: Катунь), International Conference on Distributed Computer and Communication Networks, DCCN (2024: Москва, РУДН), международный молодежный научный форум «Ломоносов» (2025: Москва, МГУ), всероссийская конференция с международным участием «Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем» ИТТММ (2022-2025: Москва, РУДН). 24 апреля 2026 г. результаты диссертации будут представлены на научном межвузовском семинаре «Современные телекоммуникации и математическая теория телетрафика», проводимом РУДН, МТУСИ, ТГУ, ИПМ РАН.

Автор является победителем конкурсного отбора на назначение стипендии Президента РФ для аспирантов и адъюнктов, обучающихся по очной форме обучения в российских организациях, осуществляющих образовательную деятельность, и проводящих научные исследования в рамках реализации приоритетов НТР РФ, определенных в СНТР РФ (2024-2025), победителем конкурсного отбора программы РУДН «Аспирантура полного дня» (2022-2025). В 2024 г. как стипендиат Президента РФ

автор была приглашена и участвовала в Конгрессе молодых ученых на федеральной территории «Сириус».

Реализация результатов работы. Автор является исполнителем грантов системы грантовой поддержки научных проектов РУДН «Разработка моделей и алгоритмов нарезки радиоресурсов и приоритетного доступа в беспроводной сети 6G» (2023-2024), «Модели математической теории телетрафика для анализа приоритетного обслуживания потокового и эластичного трафика в сетях новых поколений» (2025-2026).

Публикации. Результаты исследования представлены в 12 публикациях, в том числе в 3 работах, опубликованных в изданиях, индексируемых в международных базах цитирования Web of Science и Scopus. По теме диссертации автором получены 3 свидетельства о государственной регистрации программ для ЭВМ.

Соответствие паспорту специальности. Работа соответствует следующим пунктам паспорта научной специальности 1.2.3 «Теоретическая информатика, кибернетика»:

– п. 9 «Математическая теория исследования операций» в части моделей массового обслуживания для исследования политик управления миграцией виртуальных машин между облачными серверами и миграцией пользователей сервисов между граничным и облачными серверами;
– п. 11 «Распределенные многопользовательские системы» в части моделирования многосерверной облачной системы выполнения задач пользователей на виртуальных машинах и системы предоставления сервисов пользователям в распределенной беспроводной гранично-облачной архитектуре;
– п. 12 «Модели информационных процессов и структур» в части моделирования процесса выполнения задач пользователей в облачной инфраструктуре и процесса передачи коррелированного трафика в виде MMPP-потока в гранично-облачной архитектуре.

Личный вклад автора. Разработанные в научно-квалификационной работе модели, программные средства и анализ моделей выполнены автором самостоятельно.

Объем и структура работы. Текст диссертации включает в себя введение, основную часть из трех глав и заключение. Диссертация включает в себя список литературы из 145 библиографических ссылок. Работа изложена на 101 странице текста, содержит 37 рисунков и 13 таблиц.

Глава 1 Анализ миграции сервисов и моделирование трафика в граничных облачных вычислениях

1.1 Особенности миграции сервисов в граничных облачных вычислениях

Формирование современных распределенных вычислительных инфраструктур вызвано эволюцией системы облачных вычислений в направлении децентрализации [1]. Исторически развитие шло от централизованных вычислительных систем (мейнфреймов) [2] и клиент-серверной архитектуры к промышленной виртуализации, обеспечившей стандартизируемое управление ресурсами. Ключевую роль в этом переходе сыграла компания VMware, основанная в 1998 году. Продукты компании показали возможность эффективного подхода и привели к массовому распространению серверной виртуализации в корпоративных центрах обработки данных (ЦОД) к середине 2000-х годов [3]. Коммерциализация модели инфраструктуры как услуги (англ. Infrastructure-as-a-Service, IaaS) сделала облачные ресурсы массово доступными и обеспечила эластичность и программируемое управление. К примеру, экосистема Google Cloud – сервисы Google Compute Engine и Google Cloud Storage реализуют уровень IaaS, Google App Engine и Cloud Run относятся к платформе как услуге (англ. Platform-as-a-Service, PaaS), а Google Workspace (Gmail, Диск, Документы) представляет программное обеспечение как услугу (англ. Software-as-a-Service, SaaS). Трехуровневая модель обслуживания является общепринятой классификацией облачных сервисов; приведенная экосистема Google Cloud представляет собой один из примеров практической реализации [4, 5, 6].

Массовое внедрение облачных вычислений получило импульс с середины 2000-х (появление Amazon Elastic Compute Cloud, Amazon EC2, в 2006 году) [7, 8] и оказалось эффективным в рамках реализации ИТ-проектов, требующих эластичного масштабирования ресурсов и унифицированного управления виртуальной инфраструктурой. Однако по мере расширения класса приложений, чувствительных к задержке и контексту, к примеру интернета вещей, ограниченность строго

централизованной облачной парадигмы стала очевидной. Значительная длина сетевого пути, переменность транспортных условий и стоимость транзита данных приводят к ухудшению метрик качества обслуживания (QoS) и росту риска локальных перегрузок при пиковых нагрузках [9, 10].

Ответом на указанные вызовы стало поэтапное развертывание промежуточных форм размещения вычислений. Эту эволюцию можно представить как переход от крупных центров к иерархическим архитектурам, включающим локализованные микро-ЦОД и туманные вычисления (англ. fog computing) [11], где «туманные» узлы формируют иерархию от периферии до региональных площадок. В отличие от строго централизованного облачного хранилища, архитектура fog computing распределяет вычислительные и сетевые ресурсы по множеству уровней, от оконечных устройств до промежуточных серверов и далее до уровня облачных вычислений. Такая идея обеспечивает локальную обработку и агрегацию данных ближе к источникам трафика, то есть к пользователям. Эта концепция, продвигавшаяся Cisco и формализованная OpenFog Consortium, носит общий характер и не привязана к конкретному типу сети доступа, что отличает ее от стандартизированных граничных архитектур, рассматриваемых далее [12, 13, 14].

Параллельно с эволюцией вычислительных архитектур развивались телекоммуникационные сети, которые прошли путь от проводных инфраструктур к беспроводным сетям мобильного доступа [15]. Смена поколений беспроводных сетей (англ. 3rd Generation, 3G; 4G/Long Term Evolution (LTE)) обеспечила рост пропускной способности и снижение задержек, создав предпосылки для массового распространения мобильных приложений и облачных сервисов на мобильных устройствах [16]. На этом фоне идет переход к сетям пятого (5G) и шестого (6G) поколений. В этом контексте возрастает роль распределения вычислительных ресурсов между централизованными облачными и периферийными вычислениями [17]. В беспроводных сетях 5G/6G вычислительные узлы становятся ключевым элементом инфраструктуры, поскольку сценарии сверхнадежной передачи данных с малой задержкой (англ. Ultra-Reliable Low-Latency Communication, URLLC) и широкополосного доступа (англ. enhanced Mobile Broadband, eMBB) нуждаются в соблюдении жестких требованиях к задержкам и доступности ресурсов. Для выполнения этих требований вычисления необходимо переносить ближе к точкам сетевого доступа [18, 19, 20].

Тенденция к децентрализации привела к появлению концепции граничных вычислений и, в частности, мобильных граничных вычислений (англ. Mobile Edge Computing, MEC). В 2014 году Европейский институт телекоммуникационных стандартов (англ. European Telecommunications Standards Institute, ETSI) учредил группу отраслевых спецификаций ISG MEC для стандартизации размещения вычислительных ресурсов на узлах мобильных сетей доступа [21, 22]. Далее в 2017 году концепция была переименована в Multi-access Edge Computing, что отразило расширение области применения за пределы сугубо мобильных сетей [23]. В архитектуре MEC, определенной ETSI, приложения и сервисы размещаются на узлах сети доступа, из-за чего снижается задержка и повышается предсказуемость характеристик обслуживания. Если fog computing представляет собой обобщенную иерархическую парадигму, не привязанную к конкретному типу сети, то MEC является стандартизированной архитектурой, ориентированной на телекоммуникационную инфраструктуру и размещение вычислений напрямую на узлах сети радиодоступа (англ. Radio Access Network, RAN) [23, 24, 25, 26, 27].

Облачные и граничные архитектуры предоставляют сопоставимые по типам сервисы, но отличаются эксплуатационными свойствами. Централизованные облачные вычисления предлагают высокую мощность, развитые механизмы оркестрации и сильную агрегацию ресурсов. Однако добавляют сетевую задержку и накладные издержки из-за передачи данных на большие расстояния [28]. Граничная архитектура MEC, наоборот, располагает более ограниченными вычислительными и сетевыми ресурсами, но находится ближе к источникам трафика и может обеспечивать более строгие требования по задержкам [29, 30]. При этом ключевым показателем эффективности миграции в MEC выступает суммарная межконцевая задержка (англ. end-to-end delay, E2E-delay) по всем пользователям. Ограниченность ресурсов MEC-узла приводит к конкуренции за обслуживание: размещение одного сервиса уменьшает доступные ресурсы для остальных.

В гибридной облачно-граничной архитектуре функции распределяются следующим образом: задачи, чувствительные к задержке, обрабатываются на границе сети, а ресурсоемкие вычисления и долговременное хранение данных выносятся в облачные вычисления. Такая типовая схема интеграции граничной облачной архитектуры и пользователей представлена на рисунке 1.1. [31, 32].

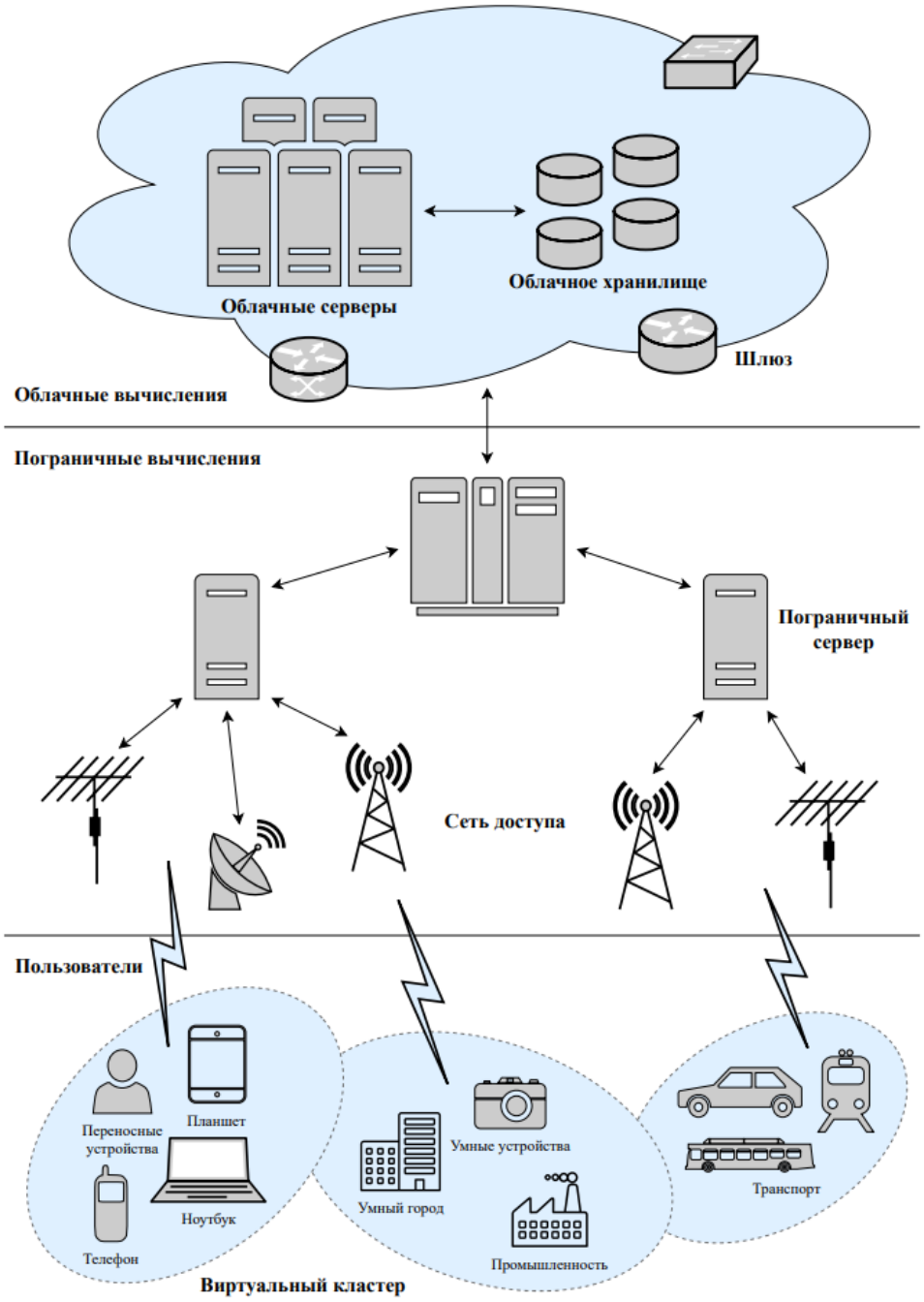

Рисунок 1.1 – Схема архитектуры граничных облачных вычислений

Во взаимодействии облачных и граничных архитектур ключевую роль играет то, где именно в каждый момент времени выполняются приложения и хранятся данные. При изменении нагрузки, появлении новых сервисов или перегрузке отдельных узлов возникает необходимость перенести часть вычислений в другое место инфраструктуры. Такой перенос, называемый миграцией виртуальных ресурсов, выполняется не вручную, а за счет специальных механизмов, обеспечивающих перемещение виртуальных машин (англ. Virtual Machine, VM), контейнеров и сервисов между серверами и площадками внутри облачно-граничной среды без заметной остановки их работы для пользователей.

Миграция позволяет перераспределять нагрузку между облачными и граничными вычислениями, освобождать ресурсы для обслуживания оборудования, реагировать на изменения структуры трафика и поддерживать требуемый уровень качества обслуживания. Таким образом, миграция выступает основным инструментом динамического управления ресурсами в гибридной облачной граничной архитектуре [33, 34, 35].

В контексте централизованных облачных центров обработки данных в настоящей работе миграция виртуальных машин (ВМ) рассматривается через три ключевых вопроса [36, 37, 38]. Когда инициировать миграцию? Какие ВМ выбирать для переноса? На какие целевые серверы их размещать? Критериями инициирования миграции обычно служат признаки перегрузки или неэффективного использования ресурсов, в том числе высокая загрузка процессора и оперативной памяти, превышение пропускной способности сетевых интерфейсов, повышенное энергопотребление отдельных узлов [39, 40], а также статистические показатели отклонения нагрузки от нормального режима. В ряде подходов используются прогнозные оценки нагрузки, позволяющие инициировать миграцию до наступления фактической перегрузки [41].

Выбор мигрируемых ВМ в облачной среде, как правило, нацелен на снижение стоимости миграции при сохранении эффекта разгрузки узла [40]. В качестве кандидатов чаще всего выбираются ВМ с меньшим объемом оперативной памяти или с низкой ожидаемой задержкой миграции, ВМ с наибольшим вкладом в энергопотребление, а также ВМ, существенно ухудшающие показатели качества обслуживания при текущем размещении [40, 41, 42, 43]. Выбор целевого сервера определяется тем, какую целевую функцию оптимизации реализует система, например минимизация суммарного энергопотребления и числа миграций, улучшение балансировки нагрузки между узлами, снижение вероятности нарушений соглашений об уровне обслуживания (англ. Service Level Agreement, SLA) или уменьшение средней миграционной задержки [44, 45, 46]. Конкретные алгоритмы миграции в облачных ЦОД различаются по используемым критериям и эвристикам [47]. В совокупности они решают одну задачу, а именно найти компромисс между эффективным использованием ресурсов и устойчивым качеством обслуживания при динамически изменяющейся нагрузке [48, 49].

Процедуры миграции в облачных и в граничных вычислениях, по сути, решают одни и те же задачи. В граничных они дополнительно учитывают мобильность

пользователей и ограничения по пропускной способности. В архитектуре MEC миграция касается прежде всего сервисов, размещенных рядом с пользователями. Поэтому здесь также можно выделить три вопроса [50, 51]. Когда инициировать перенос сервиса? Какой сервис переносить? На какой граничный или облачный сервер его размещать? Политика миграции определяется тремя компонентами: целевой функцией (например, минимизация E2E-delay), алгоритмом принятия решения (правилом перемещения сервисов) и моментом принятия решения (событием для пересмотра размещения). Некорректный выбор момента может привести к осцилляциям, т.е. циклическим перемещениям сервисов между узлами, снижающим эффективность системы. Инициирование миграции в MEC обычно связано с изменением положения пользователя, перегрузкой вычислительных ресурсов, а также ухудшением задержки и пропускной способности на маршруте до пользователя, при этом нередко используются прогнозы движения и трафика для упреждающего переноса [50, 51, 52].

Выбор мигрируемого сервиса нацелен на сохранение непрерывности обслуживания при минимальных накладных расходах. Как правило, рассматриваются сервисы, обслуживающие мобильных пользователей с жесткими требованиями к задержке или потребляющие значительную долю ресурсов текущего MEC-узла. Широкое применение контейнерных технологий в MEC позволяет сократить длительность миграции по сравнению с переносом полноценных ВМ [51, 53].

Выбор места назначения для миграции сервиса определяется компромиссом между близостью к пользователю, доступными вычислительными и сетевыми ресурсами. В работах по миграции в MEC в качестве целевых функций обычно используются совместная минимизация задержки и энергопотребления. Также рассматривается и ограничение числа миграций при сохранении требуемого качества обслуживания и обеспечение масштабируемости при большом числе MEC-узлов [53, 54].

В условиях граничных облачных систем описанные пороговые и локальные подходы оказываются недостаточными. Нагрузка в узлах MEC характеризуется высокой динамичностью, выраженной асимметрией uplink/downlink-трафика (восходящего и нисходящего), коррелированностью и разнообразием классов сервисов. Кроме того, в реальной системе наблюдается значительная неоднородность типов сервисов, что усложняет использование фиксированных порогов и статических подборов [55].

Управление миграцией в современных распределенных архитектурах переходит к парадигме «Миграции, основанной на данных» [56, 57]. Решения о переносе ВМ и сервисов должны опираться на количественные характеристики реальной нагрузки и состава услуг, включая долю различных классов приложений, распределение объемов трафика между направлениями восходящего и нисходящего потоков, суточную и недельную сезонность, интенсивность всплесков, а также прогнозные оценки будущей нагрузки [58]. Для этого необходимо последовательно исследовать свойства реальных данных и построить набор взаимодополняющих моделей, описывающих трафик на разных уровнях детализации [59].

Каждый из этих подходов обладает собственной областью применимости. К примеру, классификация и имитационная модель трафика необходима для генерации разнообразных сценариев и оценки устойчивости политик миграции [60, 61, 62, 63, 64]. Далее модели временных рядов для интеграции прогнозирования в контур принятия решений и выбора момента миграции [65, 66, 67, 68]. А уже MAP-модели (англ. Markovian Arrival Process, MAP) для построения аналитических СМО с коррелированными потоками и расчета показателей эффективности обслуживания при различных стратегиях миграции [69, 70]. В совокупности они задают рамку для последующих разделов 1.2-1.4 диссертации и подводят к формулировке постановки задачи исследования уже в разделе 1.5.

1.2 Классификация сетевого трафика по типам сервисов на основе номеров портов

Анализ реальных сетевых данных показывает, что поведение нагрузки в распределенных облачных граничных системах определяется структурой представленных сервисов, их долями в общем объеме трафика и характером активности пользователей. Сетевая нагрузка формируется множеством приложений с различной интенсивностью, объемами трафика, временными паттернами и чувствительностью к задержкам, поэтому даже при одинаковом числе активных пользователей профиль нагрузки заметно меняется за счет перераспределения долей классов сервисов.

Исследование характеристик сетевого трафика и структуры услуг было проведено автором на основе реальных данных [71, 72, 73, 74, 75], предоставленных профессором Университета Лиссабона Луишем М. Коррейя (англ. Luis M. Correia). Типовое процентное распределение сервисов мобильного оператора приведено в таблице 1.2.

Таблица 1.1 - Структура сетевого трафика по типам сервисов

Сервис	Доля	Сервис	Доля
Веб-приложения	23,24%	Безопасность	4,02%
Другие приложения	15,58%	Потоковая передача музыки	1,90%
Приложения для обмена мгновенными сообщениями	13,36%	Сетевые службы	1,53%
Игры	12,54%	Одноранговые приложения	0,66%
Передача файлов	10,27%	Мобильные терминалы	0,29%
Электронная почта	6,32%	Файловые системы	0,02%
Потоковые приложения	5,84%	Транзакции баз данных	0,01%
IP-телефония	4,40%	Устаревшие протоколы	< 0,01%

Полноценный анализ реального сетевого трафика при этом осложняется ограничениями наблюдаемости. Распределение трафика по классам сервисов является нестационарным и существенно зависит от сценариев использования, что необходимо учитывать при моделировании нагрузки и разработке механизмов управления ресурсами. Для части наборов данных (например, агрегированного по времени с шагом 1 час трафика оператора Vodafone с явным указанием класса приложения) доступны сведения о типе сервиса и объемах переданного и полученного трафика; прогнозирование объемов рассматривается в разделе 1.3.

Другой распространенный тип операторских данных содержит необработанные потоковые записи систем захвата трафика, включающие только низкоуровневые атрибуты (момент регистрации пакета, IP-адреса (англ. Internet Protocol, IP), номера портов, объем переданных байтов) без информации о прикладном назначении соединения. Для определения типа приложения применяются словари соответствий «порт – тип сервиса», которые для части фиксированных портов позволяют восстановить лишь тип протокола. Современные приложения широко используют динамические порты и нестандартные диапазоны значений. Динамические порты назначаются операционной системой произвольно для исходящих соединений и не несут информации о типе приложения. Кроме того, ряд современных приложений намеренно использует нестандартные порты, а часть трафика передается через шифрованные каналы, маскирующие реальный протокол. Такой тип трафика представлен в виде

последовательности отдельных пакетов, а множество используемых портов включает как зарезервированные значения, регламентируемые Администрацией адресного пространства Интернета (англ. Internet Assigned Numbers Authority, IANA), так и динамические порты [76, 77]. Во втором рассматриваемом автором наборе данных сопоставление сетевых соединений с классами сервисов на основе справочников IANA покрывает лишь порядка половины записей (около 52%), что делает невозможным устойчивое использование прямого соответствия «порт–класс» для анализа всего трафика [78, 79].

Стандартные словари портов, таким образом, дают лишь грубое и неполное сопоставление сетевых соединений с типами протоколов и не позволяют восстановить прикладные классы в терминах, согласованных с эталонным набором данных оператора Vodafone. Задача формулируется в виде многоклассовой классификации: по вектору признаков сетевого соединения требуется определить класс сервиса [80]. Использованы методы машинного обучения, автоматически относящие поток к одному из прикладных классов, что характерно для систем граничных вычислений, где оперативная идентификация класса используется для управления ресурсами и обеспечения качества обслуживания [81]. Знание класса сервиса позволяет оценить долю трафика, чувствительного к задержкам, полосе пропускания и надежности, и учитывать это при выборе политики миграции и распределении ресурсов между облачным сервером и MEC-узлом.

Для применения методов машинного обучения необработанные сетевые данные преобразуются в числовой вектор признаков. IP-адреса источника и назначения декомпозируются и агрегируются в числовые характеристики, отражающие структуру адресного пространства; номера портов включаются с учетом наличия фиксированных и динамических значений, дополнительно формируются агрегированные и бинарные признаки, отражающие функциональную роль портов. Вектор признаков также включает показатели объема переданных данных и другие параметры соединения [82]. В результате для каждого соединения формируется компактный вектор фиксированной размерности. В данном исследовании использовалось 13 признаков, пригодных для обучения и сравнения моделей; на вход моделей подается числовой вектор, на выходе формируется предсказанный класс сервиса.

Для решения задачи многоклассовой классификации рассматривается набор моделей машинного обучения, включающий как базовые, так и ансамблевые подходы [83]. В качестве простой эталонной модели используется метод k ближайших соседей (англ. k-Nearest Neighbors, kNN) [84], основанный на сравнении объектов в пространстве признаков и позволяющий оценить локальную разделимость классов. Логистическая регрессия (англ. Logistic Regression, LR) [85, 86] применяется как линейная интерпретируемая модель, традиционно используемая в качестве базовой точки сравнения. Для учета более сложных границ между классами используется метод опорных векторов (англ. Support Vector Machine, SVM) [78]. Вероятностный подход реализуется с помощью наивного байесовского классификатора (англ. Naive Bayes, NB) [85], который при своей простоте демонстрирует устойчивые результаты в ряде задач анализа сетевых признаков. Кроме того, рассматриваются ансамблевые методы, включая случайный лес (англ. Random Forest, RF) [87] и методы градиентного бустинга на решающих деревьях: легкий градиентный бустинг (англ. Light Gradient Boosting Machine, LightGBM), категориальный бустинг (англ. Categorical Boosting, CatBoost) и экстремальный градиентный бустинг (Extreme Gradient Boosting, XGBoost) [88], которые широко применяются для анализа табличных данных и, как правило, обеспечивают высокое качество классификации при наличии нелинейных зависимостей между признаками.

Оценка качества классификации проводится с использованием макро-усредненной F1-меры (англ. Macro-averaged F1 Score, F1-macro) и метрик точности классификации (англ. Classification Accuracy, Accuracy) [89, 90]. При этом F1-мера вычисляется отдельно для каждого класса сервиса на основе элементов матрицы ошибок: числа истинно положительных (англ. True Positives, TP), ложно положительных (англ. False Positives, FP) и ложно отрицательных (англ. False Negatives, FN) решений и задается формулой

$$F1_k = \frac{2 \cdot TP_k}{2 \cdot TP_k + FP_k + FN_k}, \tag{1.1}$$

где TP_k число верно классифицированных соединений класса k, FP_k число соединений других классов, ошибочно отнесенных к классу k, а FN_k число соединений класса k, ошибочно отнесенных к другим классам. Следовательно, итоговая макро-оценка формируется путем усреднения по всем K классам:

$$F1_{macro} = \frac{1}{K}\sum_{k=1}^{K} F\,1_k\,. \tag{1.2}$$

Метрика точности классификации характеризует долю сетевых соединений, для которых предсказанный класс совпадает с истинным, и определяется выражением

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i), \quad (1.3)$$

где y_iи $\hat{y}_i$ истинная и предсказанная метки класса сервиса для i-го соединения, а N общее число наблюдений. Следует отметить, что метрика Accuracy учитывает вклад каждого класса сервиса и является более устойчивой к неоднородности распределения классов в сетевом трафике. Сводные результаты по значению F1-меры и по качеству классификации по классам сервиса для различных моделей машинного обучения приведены в таблице 1.2.

Таблица 1.2 - Значение F1-мер в задачи классификации сетевого трафика по типам сервисов на основе номеров портов

Сервис \ Алгоритм	kNN	LR	SVM	NB	RF	Light GBM	CatBoost	XGboost
Транзакции баз данных	0.74	0.00	0.00	0.14	0.82	0.69	0.56	0.80
Файловые системы	0.97	0.70	0.67	0.54	0.98	0.97	0.95	0.98
Передача файлов	0.95	0.67	0.68	0.58	0.98	0.96	0.93	0.98
Игры	0.33	0.00	0.00	0.00	0.40	0.19	0.01	0.53
Приложения для обмена мгновенными сообщениями	0.68	0.00	0.00	0.02	0.74	0.62	0.56	0.76
Устаревшие протоколы	0.51	0.00	0.00	0.04	0.68	0.53	0.30	0.69
Электронная почта	0.94	0.09	0.08	0.43	0.95	0.92	0.86	0.96
Потоковая передача музыки	0.41	0.00	0.00	0.03	0.43	0.25	0.18	0.48
Сетевые службы	0.96	0.40	0.35	0.12	0.97	0.95	0.94	0.97
Другие приложения	0.88	0.19	0.00	0.01	0.90	0.87	0.81	0.90
Одноранговые приложения	0.78	0.00	0.00	0.22	0.82	0.67	0.57	0.82
Безопасность	0.56	0.00	0.00	0.01	0.83	0.56	0.43	0.85
Потоковые приложения	0.81	0.00	0.00	0.00	0.91	0.80	0.85	0.93
Мобильные терминалы	0.91	0.00	0.00	0.00	0.94	0.91	0.81	0.95
IP-телефония	0.89	0.00	0.00	0.01	0.93	0.86	0.81	0.94
Веб-приложения	0.98	0.72	0.70	0.03	0.98	0.96	0.95	0.98
Макро-усредненная F1-мера	0.77	0.17	0.16	0.14	0.83	0.73	0.66	0.84
Точность	0.95	0.63	0.61	0.32	0.96	0.93	0.91	0.97

По полученным данным была также разработана имитационная модель для данных оператора Vodafone. Генерация начинается с задания начального момента времени, а интервалы между последовательными событиями моделируются как независимые случайные величины с экспоненциальным распределением [91, 92]. Для каждого сгенерированного события выбирается класс сервиса в соответствии с наблюдаемым

распределением долей трафика, а также временной интервал суток, определяющий характер нагрузки. Далее формируются сетевые атрибуты соединения, включая IP-адреса и номера портов с учетом как зарезервированных, так и динамических значений, а объемы переданных данных генерируются на основе эмпирических распределений, характерных для соответствующего класса сервиса [93]. Результат работы представлен на рисунке 1.2. Подробно задача и само исследование были представлены в рамках доклада на конференции и в тезисах [74].

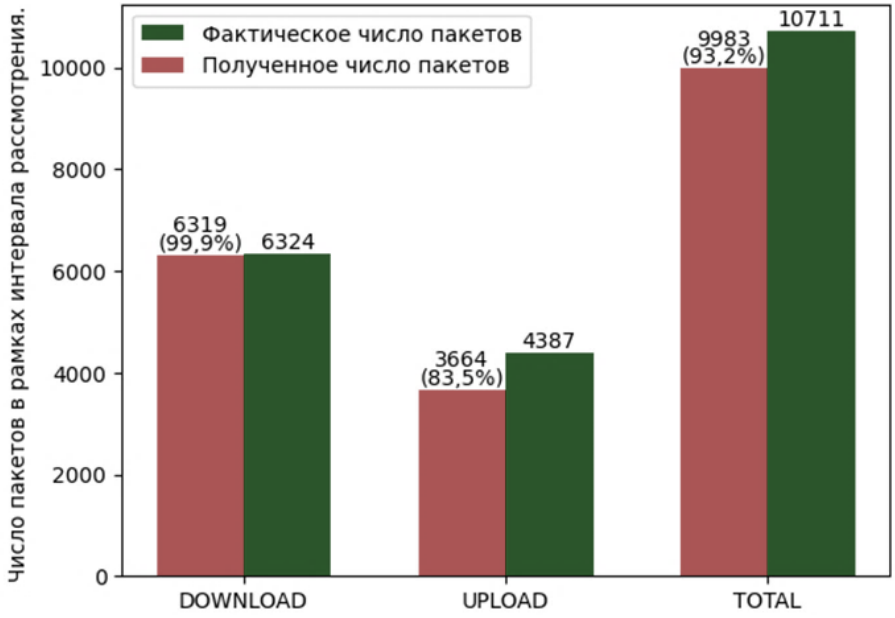

Рисунок 1.2 - Число пакетов сетевого трафика

Результат работы алгоритма представлен на рисунке 1.2. В экспериментах достигается в среднем 83,5% соответствия объемам отправленного трафика и 99,9% соответствия объемам полученного трафика. Сформированный таким образом синтетический трафик может быть использован для моделирования нагрузки и анализа механизмов управления ресурсами в среде граничных вычислений.

1.3 Прогнозирование профиля трафика на основе моделей временных рядов

Как было упомянуто выше, рассматриваемые данные представляют собой агрегированные измерения объемов трафика в восходящем и нисходящем направлениях с фиксированным часовым шагом. Исследование трафика строилось на использовании статистических моделей. Были использованы две классические модели для временных

рядов: сезонная модель SARIMA (англ. Seasonal Autoregressive Integrated Moving Average, SARIMA) [94] и модель Хольта-Уинтерса, также известная как тройное экспоненциальное сглаживание (англ. Holt-Winters) [95]. Модель SARIMA обеспечивает совместный учет авторегрессионной составляющей, компоненты скользящего среднего и сезонности. Модель Хольта-Уинтерса описывает уровень ряда, тренд и сезонную компоненту посредством сглаживания и может применяться в аддитивной или мультипликативной форме в зависимости от характера сезонных колебаний [96, 97, 98].

Параметры и спецификации моделей подбирались с использованием стандартных критериев качества. Для модели SARIMA настраивались параметры не сезонной и сезонной частей так, чтобы при минимальной сложности обеспечивалось наилучшее согласование с обучающими данными; в качестве критерия использовалась информационная мера Акаике (англ. Akaike Information Criterion, AIC) [95]. Для модели Хольта-Уинтерса тип тренда и сезонности выбирался на основе перебора допустимых вариантов и визуально-метрического соответствия данным. В результате для рассматриваемого трафика была принята конфигурация, обеспечивающая устойчивое воспроизведение суточного профиля. При сравнении результатов использовались метрики ошибок прогноза: среднеквадратичная ошибка (англ. Mean Squared Error, MSE), средняя абсолютная ошибка (англ. Mean Absolute Error, MAE) и средняя абсолютная процентная ошибка (англ. Mean Absolute Percentage Error, MAPE) [94, 95, 96, 97, 98].

Прогноз приведен для модели Хольта-Уинтерса представлен на рисунке 1.3. Для нисходящего направления наилучшее согласование с тестовыми данными демонстрирует модель Хольта-Уинтерса. Она лучше воспроизводит суточную сезонность, корректно отражая дневные и вечерние пики, ночные минимумы и переходы между локальными экстремумами, что подтверждается сравнением процентных ошибок. Значение MAPE составляет порядка 11,2% для модели Хольта-Уинтерса против примерно 15% для модели SARIMA, то есть относительная ошибка у модели Хольта-Уинтерса ниже [71].

Далее перейдем к модели SARIMA. Соответствующий прогноз приведен на рисунке 1.4. Для восходящего направления, напротив, характерны более резкие локальные всплески нагрузки, и в этом случае критичной становится способность модели корректно отражать быстрые изменения. Более точные результаты здесь демонстрирует модель SARIMA. По значениям метрики MAPE она дает порядка 4,17% против примерно

9,9% у модели Хольта-Уинтерса, то есть обеспечивает существенно меньшую относительную ошибку [71].

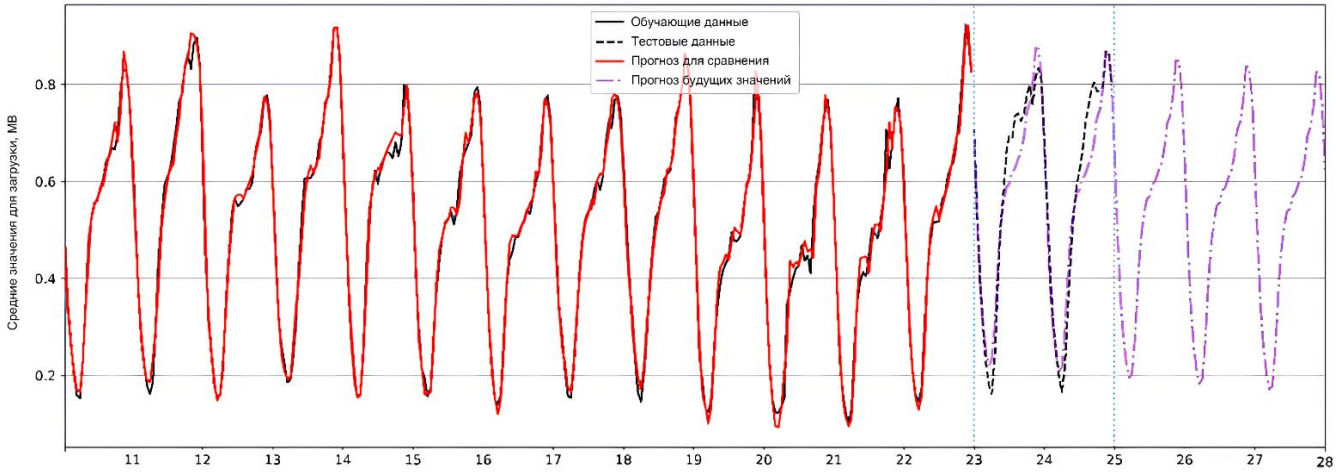

Рисунок 1.3 -Прогнозирование трафика для нисходящего потока трафика методом Хольта-Уинтерса

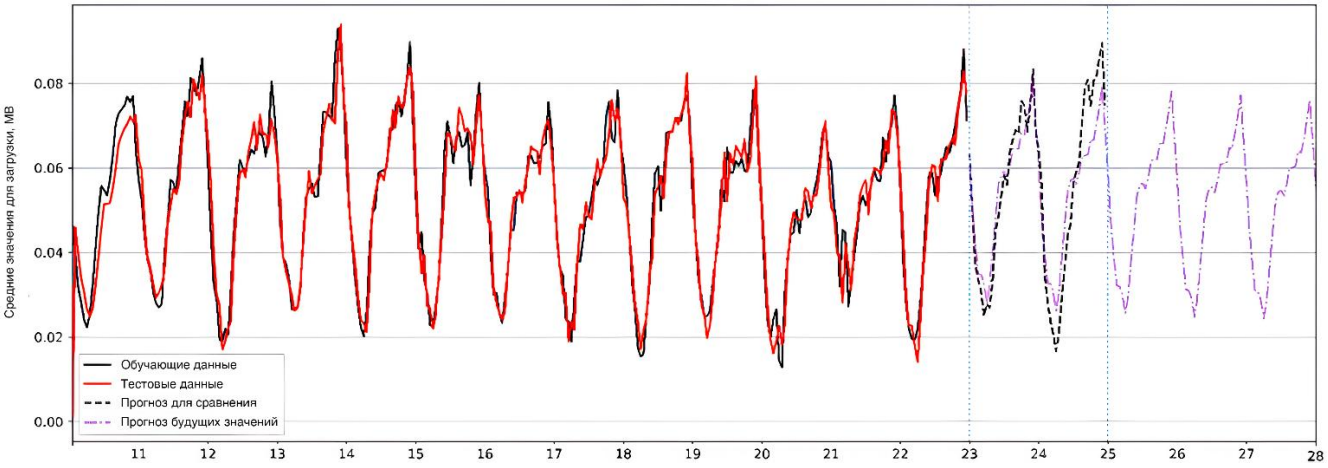

Рисунок 1.4 -Прогнозирование трафика для восходящего потока трафика методом SARIMA

Далее перейдем к рассмотрению этого же типа трафика, но для построения матриц $\boldsymbol{D}_0$ и $\boldsymbol{D}_1$ в виде марковского потока.

1.4 Моделирование сетевого трафика в виде марковского потока

В отличие от пуассоновского процесса MAP-поток допускает зависимость между последовательными событиями. Данная модель предполагает, что есть зависимость между временами прихода соседних запросов. Это делает MAP-поток адекватным инструментом для описания мобильного трафика, характеризующегося неоднородным

поведением, периодическими перегрузками и переходами между фазами фоновой и активной нагрузки [99].

Исторически MAP-модель была предложена Лукантони и соавторами как развитие подхода Нейтса и включает в себя ряд широко используемых частных случаев, таких как PH-распределения (англ. Phase-Type distributions), марковско-модулированный пуассоновский процесс (англ. Markov Modulated Poisson Process, MMPP), дискретный поток MAP-типа (англ. Discrete-time Markovian Arrival Process, DMAP) и пакетный MAP-поток (англ. Batch Markovian Arrival Process, BMAP) [100, 101]. Существенным свойством MAP/BMAP является замкнутость относительно суперпозиции: сумма независимых MAP-потоков также является MAP-потоком, что делает эти модели удобными для анализа нагрузок в комплексных вычислительных системах, включая MEC-архитектуры.

Коррелированные входные потоки широко используются в современных исследованиях систем массового обслуживания и являются предметом активного изучения в русскоязычном научном сообществе. Существенный вклад в развитие данного направления внесен в работах В.М. Вишневского, А.Н. Дудина, О.С. Дудиной, С.А. Дудина, А М. Горцева, В.И. Клименок, И.Л. Лапатина, А.Н. Моисеева, С.П Моисеевой, А.А. Назарова, В.А. Наумова, Л.А. Нежельской, А.С. Румянцева, К.Е. Самуйлова, О.В. Семеновой, Э.С. Сопина.

Базовой работой в данной области является монография В.М. Вишневского, А.Н. Дудина и В.И. Клименок, в которой систематизированы модели СМО с коррелированными потоками, включая MAP-потоки, системы с полумарковским обслуживанием, тандемные структуры и модели с ненадежными приборами, а также предложены алгоритмические методы расчета их характеристик [102, 103, 104]. Далее в работах В.А. Наумова, К.Е. Самуйлова и Э.С. Сопина показано, что коррелированность потоков в современных системах массового обслуживания обусловлена зависимостью интенсивностей поступления и обслуживания от состояния системы и ресурсных ограничений, что делает применение пуассоновских моделей недостаточным [105].

В исследованиях телетрафика и самоподобных потоков установлено, что пачечные и долговременно коррелированные потоки формируют очереди с выраженной циклической структурой, а характеристики ковариационной функции числа заявок существенно влияют на средние длины очередей и задержки [106, 107]. Дальнейшее

развитие данных результатов представлено в работах, посвященных анализу MAP-, марковских и дважды стохастических потоков в системах с повторными вызовами, тандемами и одноканальными приборами, где показано влияние внутренней марковской структуры и внешней стохастической интенсивности на показатели эффективности [108, 109, 110]. При этом отдельное направление составляют исследования устойчивости и предельного поведения систем с коррелированными потоками и повторными обращениями, характерных для телекоммуникационных и вычислительных сетей, где корреляция повторных вызовов существенно влияет на устойчивость и распределения длины очереди [111].

Актуальность MAP-потоков подтверждается их применением при оценке показателей возраста информации (англ. Age of Information, AoI) и задержек [112], моделировании облачных и гибридных вычислительных систем со стохастически изменяющейся нагрузкой [113], пользовательских потоков в сотовых сетях с учетом мобильности и процедур хэндовера [114], а также в задачах управления ресурсами и миграции сервисов в системах мобильных граничных вычислений [115].

Отдельно хочется отметить и использование MMPP-потока. В основном изучают системы с повторными вызовами (англ. Retrial Queue, RQ-системы) MMPP|M|1 и MMPP|GI|1 [108, 116, 117]), тандемы и ресурсные системы с параллельным обслуживанием [118], а также модели с бесконечным числом приборов типа MMPP|GI|∞ [119]. В этих задачах MMPP-поток используется для описания «пульсирующего» трафика и анализа загрузки, распределений числа занятых приборов, вероятностей потерь и асимптотик при большой нагрузке. Таким образом, использование коррелированного входящего потока в виде MAP-потока в настоящей работе является естественным и согласуется с современными подходами к моделированию систем массового обслуживания.

Формально MAP-поток задается парой матриц $\boldsymbol{D_0}$ и $\boldsymbol{D_1}$ размерности $n \times n$. Матрица $\boldsymbol{D_0}$ описывает переходы управляющей непрерывной цепи Маркова, не сопровождающиеся поступлением заявки. Диагональные элементы $\boldsymbol{D_0}$ задают параметры экспоненциальных распределений времени пребывания в состояниях, а внедиагональные элементы определяют интенсивности переходов между состояниями без генерации заявок. Матрица $\boldsymbol{D_1}$ описывает переходы управляющей цепи, при которых происходит поступление заявки. Генератор соответствующего марковского процесса имеет вид

$\boldsymbol{Q} = \boldsymbol{D_0} + \boldsymbol{D_1}$ и удовлетворяет условию $\boldsymbol{Q1^T} = \boldsymbol{0}$, где $\boldsymbol{1}$ вектор-столбец единиц. Такая параметризация позволяет одновременно описывать скрытую фазовую структуру трафика.

Для решения задачи параметризации MAP и MMPP на основе рассмотренных ранее данных использована библиотека mapfit для языка программирования R, разработанная Х. Окамурой, Т. Дохи и К. С. Триведи. Данная библиотека позволяет оценивать параметры MAP, BMAP, MMPP и PH-распределений методом максимального правдоподобия. Функционал библиотеки поддерживает обработку как последовательностей временных меток событий, так и агрегированных счетчиков событий за фиксированные интервалы времени, что обеспечивает гибкость при работе с различными типами исходных данных [120]. В результате параметризации были получены двухфазные, формулы (1.4) и (1.5) и трехфазные, формулы (1.6) и (1.7), MAP-модели восходящего и нисходящего трафика [74]:

$$\boldsymbol{D_0^{UP}} = \begin{bmatrix} -14{,}05 & 0{,}14 \\ 0{,}11 & -87{,}5 \end{bmatrix}, \boldsymbol{D_1^{UP}} = \begin{bmatrix} 13{,}31 & 0{,}6 \\ 0{,}31 & 87{,}08 \end{bmatrix}, \tag{1.4}$$

$$\boldsymbol{D_0^{DN}} = \begin{bmatrix} -2{,}17 & 0{,}02 \\ 0{,}01 & -7{,}56 \end{bmatrix}, \boldsymbol{D_1^{DN}} = \begin{bmatrix} 2{,}02 & 0{,}12 \\ 0{,}06 & 7{,}49 \end{bmatrix}, \tag{1.5}$$

$$\boldsymbol{D_0^{UP}} = \begin{bmatrix} -187{,}58 & 14{,}18 & 0{,}003 \\ 12{,}32 & -25{,}35 & 0{,}01 \\ 0{,}008 & 0{,}03 & -14{,}55 \end{bmatrix}, \boldsymbol{D_1^{UP}} = \begin{bmatrix} 150{,}85 & 22{,}49 & 0{,}05 \\ 12{,}46 & 0{,}47 & 0.08 \\ 0{,}04 & 0{,}15 & 14{,}34 \end{bmatrix}, \tag{1.6}$$

$$\boldsymbol{D_0^{DN}} = \begin{bmatrix} -2{,}24 & 0{,}12 & 0{,}02 \\ 0{,}002 & -15{,}28 & 15{,}27 \\ 0{,}13 & 1{,}59 \cdot 10^{-21} & -15{,}27 \end{bmatrix},$$

$$\boldsymbol{D_1^{DN}} = \begin{bmatrix} 2{,}09 & 5{,}22 \cdot 10^{-9} & 2{,}91 \cdot 10^{-53} \\ 1{,}67 \cdot 10^{-48} & 1{,}35 \cdot 10^{-19} & 6{,}31 \cdot 10^{-63} \\ 5{,}15 \cdot 10^{-5} & 15{,}14 & 1{,}1 \cdot 10^{-20} \end{bmatrix}. \tag{1.7}$$

Для нисходящего трафика были построены матриц $\boldsymbol{D_0}$ и $\boldsymbol{D_1}$ для MMPP-потока, формула (1.8):

$$\boldsymbol{D_0} = \begin{bmatrix} -17{,}94265 & 9{,}14739 \\ 2{,}82997 & -11{,}6255 \end{bmatrix}, \boldsymbol{D_1} = \begin{bmatrix} 8{,}79526 & 0{,}00 \\ 0{,}00 & 8{,}79553 \end{bmatrix}. \tag{1.8}$$

Таким образом, построенные MAP-модели позволяют включать реальные характеристики нагрузки в аналитические системы массового обслуживания. В главе 2 входные потоки задач на виртуальные машины рассматриваются в пуассоновском приближении, тогда как в главе 3, в разделах 3.4–3.6, используется частный случай MAP-модели в виде MMPP-потока.

1.5 Постановка задачи исследования

Таким образом, в разделах 1.1–1.4 выполнен анализ разных моделей данных и задач миграции в облачной и граничной средах. Показано, что реальные потоки обладают сезонностью и коррелированностью. Граничные и облачные архитектуры могут рассматриваться как отдельно, так и вместе. В настоящей работе под облачными вычислениями рассматривается выполнение задач на виртуальных машинах, развернутых на облачных серверах. Такие ЦОДы дают высокий запас ресурсов. Под граничными вычислениями – обслуживание части запросов пользователей сервисов на узле MEC. Это, в свою очередь, уменьшает задержку, но ограничено пропускной способностью.

Миграция может быть применена к любой архитектуре. Она снижает вероятность перегрузок, но сопровождается издержками и при частых переключениях может ухудшать задержку и устойчивость системы. Поэтому используются разные политики миграции, т.е. алгоритмы переноса и перераспределения нагрузки [121, 122, 123, 124]. В облачной архитектуре миграция означает перенос виртуальной машины вместе с задачами и пользователями между серверами. С точки зрения граничной архитектуры миграция означает перераспределение пользователей сервиса между облачным сервером и узлом MEC. Может быть ситуация, что узел MEC один и общий для нескольких сервисов, при этом дополнительно требуется выбирать сервис, размещение которого на MEC в данный момент наиболее целесообразно.

Решение об инициации миграции допускается в любом состоянии системы и может приниматься при каждом событии, что повышает гибкость, но увеличивает число переключений. Также миграция может выполняться только при соблюдении специальных алгоритмов, что стабилизирует управление. Для этого применяются, в частности, штрафы за миграцию и гистерезисные пороговые правила, снижающие осцилляцию, рассмотренные в диссертационном исследовании Д.С. Полуэктова [125]. При этом могут достигаться и разные требования, т.е. целевые функции. В одной задаче целью может являться минимизация числа миграций, а в другой минимизация перегрузки системы [121, 122, 123, 124]. Тем самым возникает проблема обоснования алгоритма миграции, который учитывает ограничения системы, моменты миграции, издержки переключений и требования по задержке.

Также хочется отметить, что исследования по облачным и граничным вычислениям охватывают как аналитическое моделирование, так и практико-ориентированные архитектуры и алгоритмы. Решаются проблемы разных уровней, в том числе взаимодействие архитектуры «мобильное устройство – «туманный уровень» – «обалчный» [129, 130, 131]. Дополнительно подход переноса вычислений ближе к пользователю рассматривается и в [126, 127, 128]. Акцент делается на оптимизации вычислительных ресурсов и offloading'а в интеллектуальных транспортных системах и V2X-сценариях.

В связи с этим, цель исследования состоит в разработке и анализе моделей, позволяющих оценивать эффект миграции. В работе используются модели массового обслуживания, где динамика задается непрерывными марковскими моделями, а управление описывается выбором допустимых действий в соответствии с заданной политикой. Однако существующие подходы не учитывают коррелированный характер трафика и приводят к осцилляциям и циклическим перемещениям ВМ/сервисов между узлами [121, 123, 126, 127]. Научная проблема состоит в разработке моделей массового обслуживания с учетом MMPP и механизмов предотвращения осцилляций для повышения устойчивости QoS в гибридных системах [128].

Для достижения цели вводятся три взаимосвязанные модели миграции, согласованные со структурой работы. Первая модель, глава 2, описывает миграцию виртуальных машин между облачными серверами, при этом виртуальная машина переносится вместе со всеми пользователями и их задачами, а критерий связан с минимизацией требуемой пропускной способности серверов. Вторая модель, глава 3, разделы 3.1–3.3, описывает миграцию сервисов в граничной облачной архитектуре, где у каждого сервиса есть собственный облачный сервер, а при размещении сервиса на MEC допускается частичное перераспределение пользователей. Третья модель, глава 3, разделы 3.4–3.6, сохраняет тот же принцип распределения пользователей, но рассматривает один сервис, что позволяет ввести упрощения и исследовать влияние смены фаз на интенсивности поступления запросов.

Сводная характеристика результатов, политик, целевых функций и инициирующих событий для трех моделей приводится в таблице 1.3.

Таблица 1.3 – Задачи диссертационной работы

	Политика	Целевая функция	Инициализирующие события	Входящий поток
Задача 1	Распределение ВМ со всеми задачами по серверам	Минимизация пропускной способности серверов	Поступление задачи	Пуассоновский
Задача 2	Распределение сервисов и их пользователей между МЕС-узлом и облачными серверами	Минимизация суммарной межконцевой задержки	- Поступление запроса пользователей на предоставление сервиса; - Завершение обслуживания пользователя	Пуассоновский
Задача 3	Распределение пользователей одного сервиса между МЕС-узлом и облачным сервером	Минимизация суммарной межконцевой задержки	- Поступление запроса пользователей на предоставление сервиса; - Смена интенсивности поступления запроса пользователя между двумя фазами; - Завершение обслуживания пользователя	MMPP-поток

Глава 2 Модель миграции виртуальных машин в облачной инфраструктуре

2.1 Система массового обслуживания с перемещением заявок между группами приборов

В главе 2 получен результат № 1, связанный с разработкой и анализом модели миграции виртуальных машин с обслуживаемыми задачами между облачными серверами, представляемой в виде системы массового обслуживания с перемещением классов заявок между группами приборов. Целью главы является формализация такой модели в условиях ресурсных ограничений серверов и формулировка алгоритмов миграции, минимизирующих занятую пропускную способность облачной инфраструктуры. Формализованы два алгоритма миграции, направленные на минимизацию занятой пропускной способности серверов в облачной инфраструктуре и различающиеся моментом оценки занятости приборов: до и после принятия поступившей заявки. Сначала рассматривается модель с порогами по числу задач на сервере [28, 135, 136, 137], которая далее обобщается на случай ограничений по пропускной способности серверов; соответствующая постановка и развитие модели представлены в работе [135].

Рассматривается облачная вычислительная система, предназначенная для обслуживания потоков задач, поступающих от пользователей и выполняемых на виртуальных машинах. Система включает конечное множество серверов и виртуальных машин, между которыми допускается динамическая миграция с целью предотвращения перегрузок и обеспечения требуемого качества обслуживания.

Пусть $\mathcal{S} = \{1, \dots, S\}$ множество серверов. Каждый сервер $s \in \mathcal{S}$ обладает фиксированной пропускной способностью C_s бит/с и трактуется как группа приборов, обслуживающих задачи виртуальных машин, размещенных на нем. В системе функционирует множество виртуальных машин $\mathcal{V} = \{1, \dots, V\}$, каждая из которых обслуживает задачи определенного типа. Для виртуальной машины $v \in \mathcal{V}$ задачи поступают в виде пуассоновского потока с интенсивностью λ_v задача/с. Время обслуживания одной задачи на виртуальной машине v является случайной величиной,

имеющей экспоненциальное распределение с параметром μ_v 1/с. Каждая задача, поступающая на виртуальную машину v, требует выделения ресурса в объеме b_v бит/с. Под требованием к ресурсу понимаем пропускную способность канала.

Виртуальные машины не закреплены жестко за серверами и могут мигрировать между ними в процессе функционирования системы. Предполагается, что все задачи, относящиеся к одной виртуальной машине, в каждый момент времени выполняются на одном и том же сервере. Миграция виртуальной машины в модели трактуется как управляющее воздействие, осуществляемое в момент поступления новой задачи и приводящее к изменению компоненты размещения вектора состояния. Схема рассматриваемой модели представлена на рисунке 2.1.

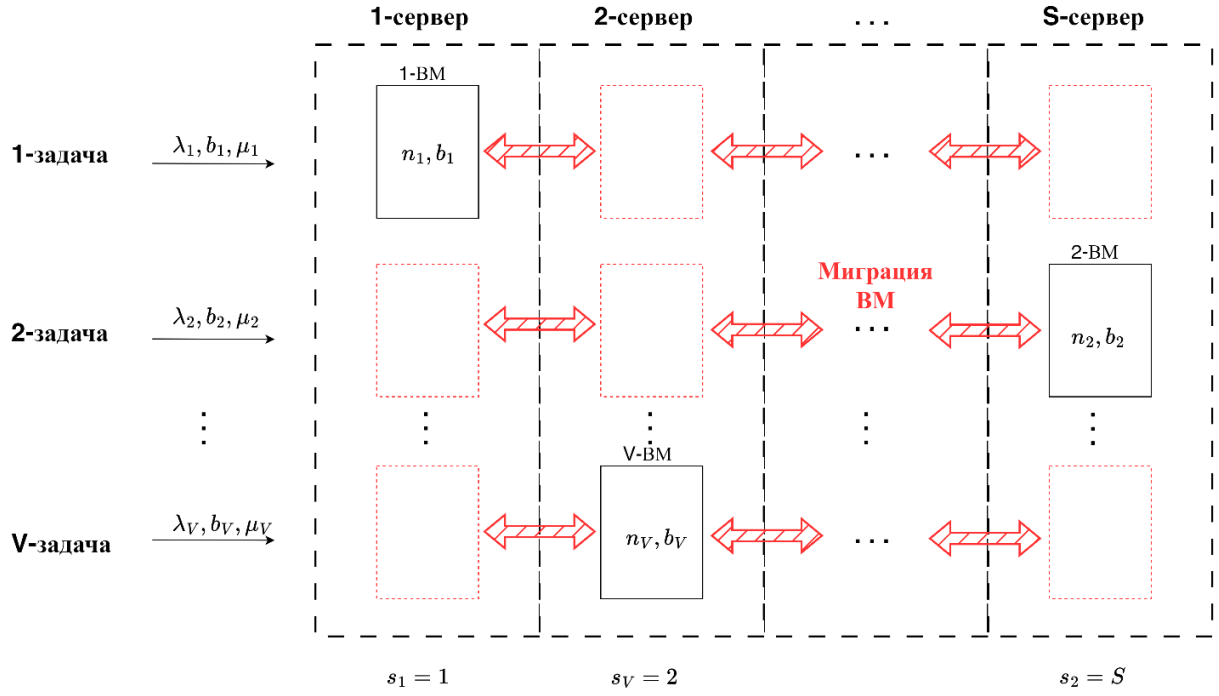

Рисунок 2.1 – Схема модели с перемещением заявок между группами приборов

Динамика системы описывается непрерывным марковским случайным процессом $\boldsymbol{X}(t), t \geq 0$. Состояние системы в момент времени t задается вектором $\boldsymbol{X}(t) = \big(\boldsymbol{N}(t), \boldsymbol{S}(t)\big)$, где $\boldsymbol{N}(t) = \big(N_1(t), \dots, N_V(t)\big)$ вектор, компоненты которого определяют количество активных задач, обслуживаемых соответствующими виртуальными машинами, а $\boldsymbol{S}(t) = \big(S_1(t), \dots, S_V(t)\big)$ вектор размещения виртуальных машин по серверам.

Компонента $S_v(t)$ принимает значения из множества серверов $\mathcal{S}$ и указывает номер сервера, на котором в момент времени t выполняются задачи виртуальной машины v. Значение $S_v(t) = 0$ является служебным и соответствует ситуации отсутствия активных

задач на виртуальной машине v, то есть случаю $N_v(t) = 0$. В этом случае виртуальная машина считается не привязанной ни к одному серверу.

Для произвольного состояния $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$ занятый объем ресурса сервера $s \in \mathcal{S}$ определяется как сумма занятой пропускной списобности всех виртуальных машин, размещенных на данном сервере, и вычисляется по формуле

$$c_s(\boldsymbol{x}) = \sum_{v \in \mathcal{V}} n_v \; b_v \, \mathbf{1}\{s_v = s\}, s \in \mathcal{S}. \tag{2.1}$$

Пространство состояний процесса формируется с учетом условий корректности размещения виртуальных машин и ограничений на ресурсы серверов. Состояние $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$ считается допустимым, если для каждой виртуальной машины выполняется условие согласованности компонент n_v и s_v, а для каждого сервера выполнено ограничение на суммарную нагрузку, определяемую величиной $c_s(\boldsymbol{x})$. Совокупность всех допустимых состояний образует пространство состояний процесса, которое задается в виде

$$\mathcal{X} = \{\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s}) \colon (n_v > 0, s_v \in \mathcal{S}) \vee \; (n_v = 0, s_v = 0), v \in \mathcal{V}; \; c_s(\boldsymbol{x}) \leq C_s, s \in \mathcal{S}\}. \tag{2.2}$$

Таким образом, в каждый момент времени суммарная потребность задач, размещенных на сервере s, не превышает его ресурсных ограничений. Для каждой виртуальной машины $v \in \mathcal{V}$ введем подмножество состояний

$$\mathcal{X}_v = \{\boldsymbol{x} \in \mathcal{X} \colon c_{s_v}(\boldsymbol{x}) + b_v > C_{s_v}\}, \tag{2.3}$$

характеризующее ситуации, в которых поступление новой задачи на виртуальную машину v приводит к превышению допустимой пропускной способности на текущем сервере s_v. Важно отметить, что именно состояния множества $\mathcal{X}_v$ требуют принятия управляющего решения: либо миграции виртуальной машины на другой сервер, либо отказа в обслуживании поступившей задачи. Если множество серверов, допустимых для размещения виртуальной машины в состоянии $\boldsymbol{x}$, оказывается пустым, поступившая задача считается заблокированной.

В рассматриваемой системе миграция понимается как перенос виртуальной машины вместе с обслуживаемыми ею задачами с одного сервера на другой. Миграция инициируется в момент поступления новой задачи, если ее выполнение на текущем сервере невозможно без нарушения ресурсных ограничений.

Пусть $\mathcal{S}_v(\boldsymbol{x}) \subseteq \mathcal{S} \setminus \{s_v\}$ есть множество серверов, допустимых для миграции виртуальной машины v из состояния $\boldsymbol{x}$. Непустота множества $\mathcal{S}_v(\boldsymbol{x})$ означает наличие по крайней мере одного сервера, на который может быть перенесена виртуальная машина

без нарушения ограничений. Для каждого состояния $\boldsymbol{x} \in \mathcal{X}_v$, в котором миграция возможна, определяется целевой сервер $s_v^*(\boldsymbol{x}) \in \mathcal{S}_v(\boldsymbol{x})$, выбираемый в соответствии с заданной политикой миграции. Конкретные правила формирования множества $\mathcal{S}_v(\boldsymbol{x})$ и выбора сервера $s_v^*(\boldsymbol{x})$ определяются политикой и рассматриваются далее.

Миграция виртуальных машин в данной модели трактуется как элемент управления системой массового обслуживания. Управляющие решения принимаются в определенных состояниях и определяют допустимые переходы между состояниями системы, тем самым влияя на характеристики обслуживания входящего потока заявок.

Динамика системы определяется матрицей интенсивностей $\boldsymbol{Q}$, элементы которой соответствуют событиям поступления задач, завершения обслуживания и, при выполнении пороговых условий, изменению размещения виртуальной машины. Далее приведены ненулевые элементы матрицы интенсивностей. Для компактного описания изменений вектора состояния используется обозначение $\boldsymbol{e}_v$, соответствующее базисному вектору с единицей в v-й позиции. Тогда ненулевые элементы матрицы $\boldsymbol{Q}$ имеют вид:

$$\begin{aligned}
&\boldsymbol{Q}[(\boldsymbol{n},\boldsymbol{s}),(\boldsymbol{n}+\boldsymbol{e}_v,\boldsymbol{s})] = \lambda_v, c_{s_v}(\boldsymbol{n},\boldsymbol{s}) + b_v \le C_{s_v}, v \in \mathcal{V},\\
&\boldsymbol{Q}[(\boldsymbol{n},\boldsymbol{s}),(\boldsymbol{n}+\boldsymbol{e}_v,\boldsymbol{s}')] = \lambda_v, c_{s_v}(\boldsymbol{n},\boldsymbol{s}) + b_v > C_{s_v}, \mathcal{S}_v(\boldsymbol{n},\boldsymbol{s}) \neq \emptyset, s_v' = s_v^*(\boldsymbol{n},\boldsymbol{s}), v \in \mathcal{V},\\
&\boldsymbol{Q}[(\boldsymbol{n},\boldsymbol{s}),(\boldsymbol{n}-\boldsymbol{e}_v,\boldsymbol{s})] = n_v\mu_v, n_v > 0, v \in \mathcal{V}.
\end{aligned} \quad (2.4)$$

Таким образом, в разделе 2.1 задан общий механизм функционирования облачной системы, структура пространства состояний и условия возникновения миграции, что позволяет далее формализовать политики миграции.

2.2 Алгоритмы миграции для минимизации занятой пропускной способности серверов

В настоящем разделе вводятся политики миграции виртуальных машин, определяющие правила выбора целевого сервера при поступлении задач. Политики миграции применяются исключительно в моменты поступления новых задач на виртуальные машины. Если размещение поступившей задачи на текущем сервере невозможно без нарушения ресурсных ограничений, система переходит к процедуре выбора целевого сервера для миграции виртуальной машины. В противном случае задача принимается на обслуживание без изменения размещения виртуальных машин, и управляющее воздействие не осуществляется.

В работе рассматриваются две политики миграции, реализующие различные подходы к выбору целевого сервера. Для удобства изложения общая политика выбора целевого сервера $s^*(\boldsymbol{x})$ далее конкретизируется в виде двух политик $s_1^*(\boldsymbol{x})$ и $s_2^*(\boldsymbol{x})$. Общая последовательность действий при поступлении задачи, проверке допустимости ее размещения и выборе целевого сервера в рамках рассматриваемых политик проиллюстрирована на рисунке 2.2.

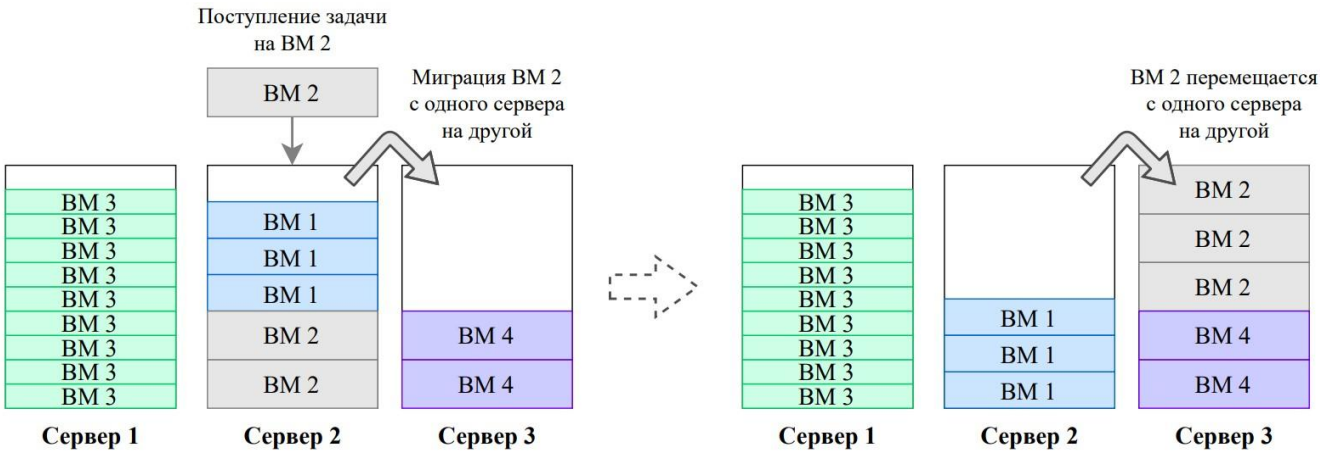

Рисунок 2.2 - Схема миграции виртуальной машины между серверами

Первая политика миграции $(i = 1)$ ориентирована на выравнивание загрузки серверов и предотвращение накопления перегрузок в отдельных группах вычислительных ресурсов. В рамках данной политики целевой сервер выбирается из множества допустимых серверов таким образом, чтобы суммарная нагрузка на сервер после выполнения миграции была минимальной. При формировании множества допустимых серверов учитываются текущая загрузка серверов, требования задач соответствующей виртуальной машины и ограничения по пропускной способности серверов. При высокой интенсивности поступления задач может приводить к увеличению числа миграций. Под прогнозируемой загрузкой в рамках первой политики $(i = 1)$ понимается значение нагрузки серверов непосредственно после размещения виртуальной машины. Введенная политика минимизации прогнозируемого использования ресурсов может быть формализована в виде алгоритма 2.1 принятия решения о миграции виртуальной машины при поступлении задачи.

Алгоритм 2.1. Для модели с перемещением заявок между группами приборов политика миграции, ориентированная на минимизацию прогнозируемого использования ресурса, формализуется следующим образом.

1: *Вход:* состояние $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$, виртуальная машина v, сервер размещения s_v.

2: *Шаг 1.* При поступлении новой задачи на ВМ v в состоянии $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$ проверяется, принадлежит ли состояние множеству критических состояний.

3: *Шаг 2.* Если $\boldsymbol{x} \in \mathcal{X}_v$, формируется множество целевых серверов, минимизирующих прогнозируемое использование ресурса при одновременном выполнении ограничения допустимости:

$$\mathcal{S}_v^1(\boldsymbol{x}) = \Big\{ s \in \mathcal{S}\colon s = \\ = arg \min_{s' \in \mathcal{S} \setminus \{s_v\}} [c_{s'}(\boldsymbol{x}) + (n_v + 1)b_v] \cdot \mathbf{1}\{c_{s'}(\boldsymbol{x}) + (n_v + 1)b_v \le C_{s'}\} \Big\}, \quad \boldsymbol{x} \in \mathcal{X}_v,\ v \in \mathcal{V}. \tag{2.5}$$

4: *Шаг 3.* Целевой сервер выбирается детерминировано:

$$s_v^{*(1)}(\boldsymbol{x}) = \min_{s \in \mathcal{S}_v^1(\boldsymbol{x})} s, \qquad \boldsymbol{x} \in \mathcal{X}_v,\ v \in \mathcal{V}. \tag{2.6}$$

5: *Выход:* целевой сервер $\mathrm{s}_\mathrm{v}^{*(1)}(\boldsymbol{x})$, следовательно выполняется миграция виртуальной машины v на этот сервер.

Вторая политика миграции ($i = 2$) основана на выборе сервера с наименьшей текущей загрузкой и ориентирована на оперативное устранение перегрузок. В рамках данной политики приоритет отдается серверам, которые в момент принятия решения обладают минимальной нагрузкой, без учета возможного изменения состояния системы в будущем. После выбора сервера выполняется проверка допустимости размещения виртуальной машины с учетом поступившей задачи. В случае отсутствия допустимых серверов задача отклоняется, а миграция не осуществляется. Данная политика характеризуется меньшими вычислительными затратами на принятие управляющих решений и может быть эффективно использована в системах с высокой динамикой нагрузки. Аналогично предыдущей политике, процедура выбора целевого сервера и проверки допустимости миграции формализуется в виде алгоритма.

Алгоритм 2.2. Для модели с перемещением заявок между группами приборов политика миграции для минимизации текущей загрузки ресурса записывается следующим образом:

1: *Вход:* состояние $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$, виртуальная машина v, сервер размещения s_v.

2: *Шаг 1.* При поступлении новой задачи на ВМ v в состоянии $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$ проверяется принадлежность состояния множеству критических состояний $\mathcal{X}_v$. Если $\boldsymbol{x} \notin \mathcal{X}_v$, то миграция не инициируется.

3: *Шаг 2.* Если $\boldsymbol{x} \in \mathcal{X}_v$, определяется множество серверов с минимальной текущей загрузкой:

$$\bar{\mathcal{S}}_v^2(\boldsymbol{x}) = \left\{ s \in \mathcal{S} \colon s = arg \min_{s' \in \mathcal{S} \setminus \{s_v\}} c_{s'}(\boldsymbol{x}) \right\}, \quad \boldsymbol{x} \in \mathcal{X}_v,\, v \in \mathcal{V}. \tag{2.7}$$

4: *Шаг 3.* Из множества $\mathcal{S}_{\mathrm{v}}^2(\mathbf{x})$выбирается сервер с наименьшим индексом:

$$s_v^{2*}(\boldsymbol{x}) = \min_{s \in \bar{\mathcal{S}}_v^2(\boldsymbol{x})} s, \quad \boldsymbol{x} \in \mathcal{X}_v,\, v \in \mathcal{V}. \tag{2.8}$$

5: *Шаг 4.* Выполняется проверка допустимости размещения виртуальной машины v с новой задачей на выбранном сервере. В формализованном виде это записывается следующим образом:

$$\mathcal{S}_v^2(\boldsymbol{x}) = \begin{cases} \bar{\mathcal{S}}_v^2(\boldsymbol{x}), & c_{s_v^{2*}(\boldsymbol{x})}(\boldsymbol{x}) + (n_v + 1) b_v \le C_{s_v^{2*}(\boldsymbol{x})} \\ \emptyset, & c_{s_v^{2*}(\boldsymbol{x})}(\boldsymbol{x}) + (n_v + 1) b_v > C_{s_v^{2*}(\boldsymbol{x})} \end{cases}, \quad \boldsymbol{x} \in \mathcal{X}_v, v \in \mathcal{V}. \tag{2.9}$$

6: *Выход:* либо целевой сервер $\mathrm{s}_{\mathrm{v}}^{2*}(\boldsymbol{x}) = \bar{\mathcal{S}}_{\mathrm{v}}^2(\boldsymbol{x})$, при котором выполняется миграция виртуальной машины v, либо отказ в миграции и блокирование поступившей задачи.

Выбор конкретной политики миграции определяется спецификой целевой вычислительной инфраструктуры и характером входной нагрузки. Политика, реализованная алгоритмом 2.1, является предпочтительной для относительно стабильных инфраструктур, где приоритетными являются долгосрочная оптимизация распределения нагрузки и ограничение числа миграций виртуальных машин. Алгоритм 2.2, напротив, ориентирован на функционирование в высокодинамичных облачных и граничных вычислительных средах с эластичной нагрузкой, для которых критическим фактором является скорость принятия управляющих решений.

Таким образом, введенные политики миграции и соответствующие алгоритмы формализуют механизм принятия управляющих решений в рамках марковской модели и служат основой для последующего анализа стационарных характеристик системы и оценки эффективности различных подходов к управлению миграцией виртуальных машин.

2.3 Анализ политики миграции по числу задач на сервере

В данном разделе рассматривается частный случай математической модели, введенной в разделе 2.1, позволяющий упростить анализ системы и получить наглядные аналитические результаты для ограниченной конфигурации вычислительных ресурсов. Рассматриваемый частный случай соответствует системе, состоящей из двух серверов,

между которыми допускается миграция виртуальных машин при выполнении пороговых условий. Объем ресурсов, требуемый для обслуживания одной задачи любой виртуальной машиной, является одинаковым и равен единице, то есть $b_v = 1$. Это предположение позволяет свести ограничение по требованию к ресурсу сервера к ограничению по максимальному числу одновременно обслуживаемых задач и тем самым существенно упростить описание пространства состояний системы. Далее рассматривается система, включающая $V = 3$ виртуальные машины и $S = 2$ сервера. Для каждой виртуальной машины $v \in \{1,2,3\}$ поток поступающих задач является пуассоновским с интенсивностью λ_v, а время обслуживания задач имеет экспоненциальное распределение с параметром μ_v.

Для заданной конфигурации системы множество допустимых целевых серверов для миграции в каждом критическом состоянии либо является пустым, либо содержит единственный элемент, вследствие чего правила выбора целевого сервера, соответствующие различным политикам миграции, приводят к одному и тому же управляющему решению. Таким образом, формируется единый механизм миграции виртуальных машин, полностью определяемый текущим состоянием системы и ресурсными ограничениями. Все последующие вычисления стационарных характеристик и результаты численного анализа, представленные в настоящем разделе, относятся именно к указанному частному случаю модели и не зависят от конкретной формализации политики миграции, рассмотренной в разделе 2.2.

Суммарная нагрузка на сервер s в состоянии $\mathbf{x}$ определяется выражением

$$m_s(\boldsymbol{x}) = \sum_{v \in \mathcal{V}} n_v \ \mathbf{1}\{s_v = s\}, \tag{2.10}$$

где n_v число задач, обслуживаемых виртуальной машиной v. В силу принятого предположения каждая задача занимает одну условную единицу ресурса сервера $m_s(\boldsymbol{x}) = c_s(\boldsymbol{x})$, при $b_v = 1$. Тогда ограничение по пропускной способности сервера принимает вид $m_s(\boldsymbol{x}) \le M_s$, где $M_s = C_s$ и есть максимально допустимое число задач, одновременно обслуживаемых на сервере s. Следовательно, $m_s(\boldsymbol{x}) \le M_s, s \in \{1,2\}$.

Состояние системы описывается вектором $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$, где $\boldsymbol{n} = (n_1, n_2, n_3)$ число задач, обслуживаемых виртуальными машинами, $\boldsymbol{s} = (s_1, s_2, s_3)$ их размещение по серверам.

Пространство состояний марковского процесса формируется в соответствии с введенными ограничениями:

$$\mathcal{X} = \{\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s}): n_v \geq 0,\ s_v \in \{0,1,2\},\ v \in \{1,2,3\},\ m_s(\boldsymbol{x}) \leq M_s,\ s \in \{1,2\}\}. \quad (2.11)$$

Динамика системы описывается матрицей интенсивностей $\boldsymbol{Q}$, ненулевые элементы которой имеют следующий вид:

$$\begin{aligned}
&\boldsymbol{Q}[(\boldsymbol{n}, \boldsymbol{s}), (\boldsymbol{n} + \boldsymbol{e}_v, \boldsymbol{s})] = \lambda_v, s_v = 1,\ m_1(\boldsymbol{x}) + 1 \leq M_1,\ v \in \mathcal{V},\\
&\boldsymbol{Q}[(\boldsymbol{n}, \boldsymbol{s}), (\boldsymbol{n} + \boldsymbol{e}_v, \boldsymbol{s})] = \lambda_v, s_v = 2,\ m_2(\boldsymbol{x}) + 1 \leq M_2,\ v \in \mathcal{V},\\
&\boldsymbol{Q}[(\boldsymbol{n}, \boldsymbol{s}), (\boldsymbol{n} + \boldsymbol{e}_v, \boldsymbol{s} + \boldsymbol{e}_v)] = \lambda_v, s_v = 1,\ m_1(\boldsymbol{x}) + 1 > M_1,\ m_2(\boldsymbol{x}) + 1 \leq M_2,\ v \in \mathcal{V},\\
&\boldsymbol{Q}[(\boldsymbol{n}, \boldsymbol{s}), (\boldsymbol{n} + \boldsymbol{e}_v, \boldsymbol{s} - \boldsymbol{e}_v)] = \lambda_v, s_v = 2,\ m_2(\boldsymbol{x}) + 1 > M_2,\ m_1(\boldsymbol{x}) + 1 \leq M_1,\ v \in \mathcal{V},\\
&\boldsymbol{Q}[(\boldsymbol{n}, \boldsymbol{s}), (\boldsymbol{n} - \boldsymbol{e}_v, \boldsymbol{s})] = n_v\mu_v, n_v > 0,\ v \in \mathcal{V}.
\end{aligned} \quad (2.12)$$

Решение системы уравнений равновесия позволяет получить стационарные вероятности $\pi(\boldsymbol{x})$ и показатели качества обслуживания. В частности, могут быть вычислены следующие характеристики.

Вероятность блокировки задачи:

$$B = B_1 = B_2 = \sum_{\boldsymbol{x} \in \mathcal{B}} \pi(\boldsymbol{x}), \quad (2.13)$$

где $\mathcal{B} = \{\boldsymbol{x} \in \mathcal{X}: m_1(\boldsymbol{x}) + 1 > M_1, m_2(\boldsymbol{x}) + 1 > M_2\}$.

Среднее число задач, обслуживаемых виртуальной машиной v:

$$\bar{N}_v = \sum_{\boldsymbol{x} \in \mathcal{X}} n_v\, \pi(\boldsymbol{x}), v = 1,2,3. \quad (2.14)$$

Среднее число задач, обслуживаемых на сервере s:

$$\bar{M}_s = \sum_{\boldsymbol{x} \in \mathcal{X}} m_s\,(\boldsymbol{x})\, \pi(\boldsymbol{x}), s = 1,2. \quad (2.15)$$

В рамках численного анализа рассматривается сценарий, отражающий функционирование облачных сервисов потокового вещания игр, относящихся к модели программного обеспечения как услуги (SaaS). Подобные сервисы характеризуются высокой чувствительностью к задержкам и отказам в обслуживании, что делает задачи управления нагрузкой и миграции виртуальных машин особенно актуальными. В рассматриваемой модели потоковые игровые приложения используются в качестве типового примера сервисов с интенсивным поступлением задач и ограниченными вычислительными ресурсами [138].

Численный анализ проводится для системы, включающей три виртуальные машины и два сервера, между которыми допускается миграция виртуальных машин при превышении пороговых условий [139]. В качестве основных варьируемых параметров рассматриваются интенсивности поступления задач и средняя продолжительность их выполнения, поскольку именно данные характеристики в наибольшей степени определяют уровень загрузки системы. Для двух рассматриваемых сценариев задаются ограничения на максимальное число задач, одновременно обслуживаемых на серверах.

В первом сценарии исследуется влияние интенсивности поступления задач на характеристики функционирования системы при фиксированном среднем времени обслуживания, включая различные соотношения между входными потоками. Во втором сценарии, напротив, суммарная интенсивность поступления задач фиксируется, а анализируется влияние средней продолжительности выполнения задач и ее неоднородности между виртуальными машинами.

Результаты численного анализа используются для оценки влияния варьируемых параметров на ключевые характеристики системы, включая вероятность блокировки задач, а также уровни загрузки виртуальных машин и серверов. Полученные зависимости демонстрируют рост вероятности блокировки и загрузки ресурсов при увеличении интенсивности поступления задач и средней продолжительности их выполнения, а также чувствительность системы к дисбалансу параметров между виртуальными машинами. Результаты проиллюстрированы на рисунках 2.3-2.6.

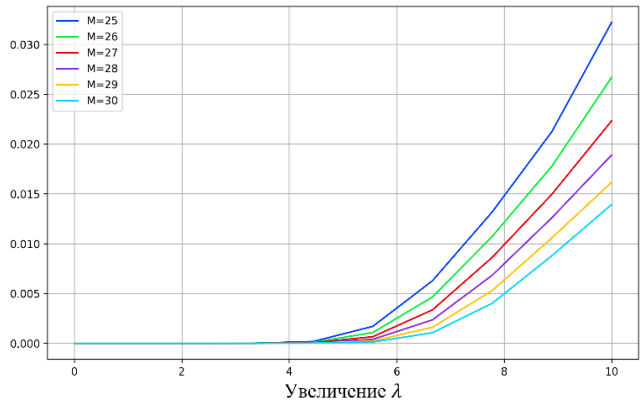

Рисунок 2.3 - Вероятность блокировки (сценарий 1)

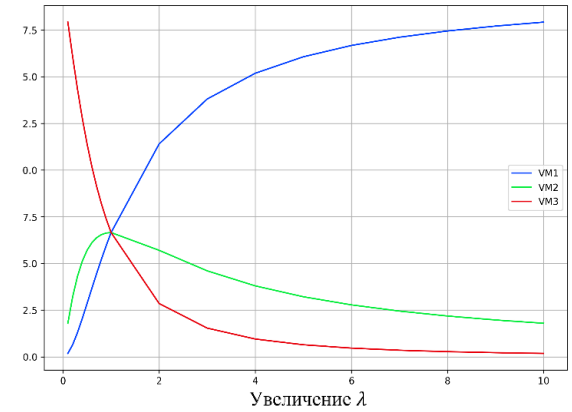

Рисунок 2.4 - Среднее количество задач на v-ВМ (сценарий 1)

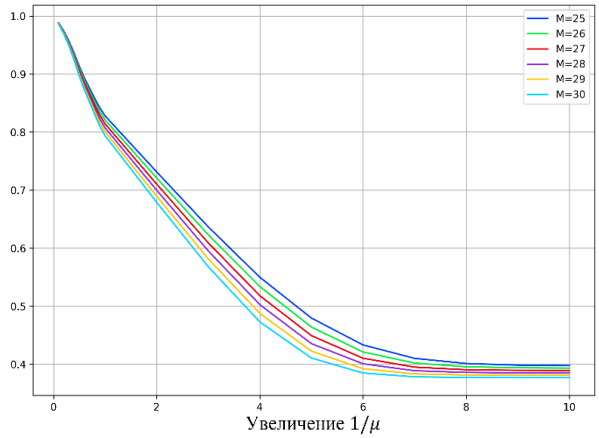

Рисунок 2.5 - Вероятность блокировки (сценарий 2)

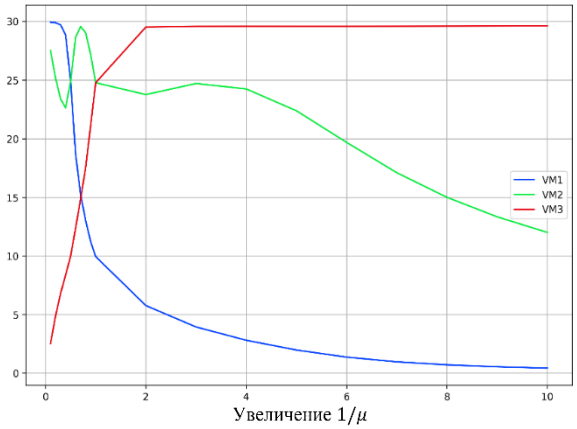

Рисунок 2.6 - Среднее количество задач на v-ВМ (сценарий 2)

2.4 ПОКАЗАТЕЛИ ЭФФЕКТИВНОСТИ МИГРАЦИИ ВИРТУАЛЬНЫХ МАШИНАХ

По стационарному распределению $\pi(\boldsymbol{x})$ марковского процесса вычислим показатели качества обслуживания. В настоящем разделе характеристики QoS рассматриваются на уровне сеанса обслуживания, в частности, анализируется вероятность блокировки задач. Все показатели эффективности целесообразно разделить на две группы. Метрики, не зависящие от механизмов миграции виртуальных машин, и метрики, обусловленные процессами миграции.

К первой группе относятся характеристики, определяемые исключительно текущим состоянием системы и не зависящие от факта переноса виртуальных машин между серверами. Вероятность блокировки задачи для виртуальной машины v определяется как

$$B_v = \sum_{\boldsymbol{x}\in\mathcal{B}_v} \pi(\boldsymbol{x}),\ \ \mathcal{B}_v = \{\boldsymbol{x} \in \mathcal{X}_v : \mathcal{S}_v(\boldsymbol{x}) = \emptyset\}, v \in \mathcal{V}. \tag{2.16}$$

Среднее использование полосы пропускания серверов задается выражением

$$\bar{c}_s = \sum_{\boldsymbol{x}\in\mathcal{X}} c_s\,(\boldsymbol{x})\,\pi(\boldsymbol{x}), s \in \mathcal{S}. \tag{2.17}$$

Среднее использование полосы пропускания виртуальной машиной v определяется

$$\bar{b}_v = \sum_{\boldsymbol{x}\in\mathcal{X}} n_v\, b_v\, \pi(\boldsymbol{x}), v \in \mathcal{V}. \tag{2.18}$$

Среднее число работающих серверов вычисляется по формуле

$$\bar{s} = \sum_{\boldsymbol{x}\in\mathcal{X}} \sum_{s\in\mathcal{S}} \mathbf{1}\{\, c_s(\boldsymbol{x}) > 0\}\, \pi(\boldsymbol{x}), s \in \mathcal{S}. \tag{2.19}$$

В отличие от указанных характеристик, метрики, связанные с миграцией ВМ, зависят не только от текущего состояния системы, но и от вероятностей наступления событий, инициирующих процесс миграции, а также от выбранной политики управления.

Утверждение 2.1. Для модели с перемещением заявок между группами приборов вероятность миграции произвольной ВМ будет выглядеть следующим образом

$$P^{mg} = \sum\nolimits_{\boldsymbol{x}\in\mathcal{X}} P^{mg}\,(\boldsymbol{x}) \cdot \pi(\boldsymbol{x}), \tag{2.20}$$

где

$$P^{mg}(\boldsymbol{x}) = \sum\nolimits_{v\in\mathcal{V}} P_v^{mg}\,(\boldsymbol{x}), \boldsymbol{x} \in \mathcal{X} \tag{2.21}$$

условная вероятность миграции любой виртуальной машины в состоянии $\boldsymbol{x}$, а

$$P_v^{mg}(\boldsymbol{x}) = \frac{\lambda_v}{\sum_{v' \in \mathcal{V}} (\lambda_{v'} + n_{v'}\mu_{v'})} \cdot \mathbf{1}\{\boldsymbol{x} \in M_v\}, v \in \mathcal{V}, \boldsymbol{x} \in \mathcal{X} \tag{2.22}$$

вероятность миграции v -ВМ в состоянии $\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$, при этом

$$\mathcal{M}_v = \{\boldsymbol{x} \in \mathcal{X} : c_{s_v}(\boldsymbol{x}) + b_v > C_{s_v}, \mathcal{S}_v(\boldsymbol{x}) \neq \emptyset\} = \{\boldsymbol{x} \in \mathcal{X}_s : \mathcal{S}_v(\boldsymbol{x}) \neq \emptyset\}, v \in \mathcal{V} \tag{2.23}$$

множество состояний, запускающих миграцию ВМ v после поступления новой задачи:

Доказательство.

Обозначим следующие события:

искомое событие $\mathcal{A}$: «произошла миграция некоторой виртуальной машины»;

событие $\mathcal{B}(\boldsymbol{x})$: «система находится в состоянии $\boldsymbol{x}$»;

событие $\mathcal{C}(v)$: «поступила новая задача на v-виртуальную машину».

По формуле полной вероятности вероятность искомого события $\mathcal{A}$ имеет вид

$$Pr(\mathcal{A}) = \sum_{\boldsymbol{x} \in \mathcal{X}} Pr\big(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x})\big) \cdot Pr\big(\mathcal{B}(\boldsymbol{x})\big), \tag{2.24}$$

где $Pr\big(\mathcal{B}(\boldsymbol{x})\big) = \pi(\boldsymbol{x})$ стационарная вероятность состояния $\boldsymbol{x}$. Условные вероятности $Pr\big(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x})\big)$ равны 0, если система находится в состоянии, при котором миграция невозможна. Следовательно, необходимо учесть, на какую именно ВМ пришла задача, поэтому обозначим через $v \in \mathcal{V}$. Тогда

$$Pr\big(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x})\big) = \sum_{v \in \mathcal{V}} Pr(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x}), \mathcal{C}(v)) \cdot Pr\big(\mathcal{C}(v) \mid \mathcal{B}(\boldsymbol{x})\big). \tag{2.25}$$

При этом

$$Pr(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x}), \mathcal{C}(v)) = \begin{cases} 0, \boldsymbol{x} \notin M_v, \\ 1, \boldsymbol{x} \in M_v, \end{cases} \Rightarrow \mathbf{1}\{\boldsymbol{x} \in M_v\}, v \in \mathcal{V}, \tag{2.26}$$

где $\mathcal{M}_v = \{\boldsymbol{x} \in \mathcal{X} : c_{s_v}(\boldsymbol{x}) + b_v > C_{s_v}, \mathcal{S}_v(\boldsymbol{x}) \neq \emptyset\} = \{\boldsymbol{x} \in \mathcal{X}_s : \mathcal{S}_v(\boldsymbol{x}) \neq \emptyset\}, v \in \mathcal{V}$, множество состояний, запускающих миграцию, так как при поступлении новой задачи суммарная требуемая пропускная спобобность на текущем сервере превышает его емкость.

При фиксированном состоянии $\boldsymbol{x}$ могут произойти два типа событий для каждой ВМ – поступление новой задачи на эту ВМ с интенсивностью λ_v и завершение обслуживания одной из n_v задач на этой ВМ с суммарной интенсивностью $n_v\mu_v$. Вероятность того, что следующим событием будет именно поступление задачи на v-ВМ, равна отношению ее интенсивности к суммарной интенсивности:

$$Pr\big(\mathcal{C}(v) \mid \mathcal{B}(\boldsymbol{x})\big) = \frac{\lambda_v}{\sum_{v' \in \mathcal{V}} (\lambda_{v'} + n_{v'}\mu_{v'})}. \tag{2.27}$$

В таком случае условная вероятность миграции любой виртуальной машины в состоянии $\boldsymbol{x}$ равна

$$Pr\big(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x})\big) = \frac{\lambda_v}{\sum_{v' \in \mathcal{V}} (\lambda_{v'} + n_{v'}\mu_{v'})} \cdot \mathbf{1}\{\boldsymbol{x} \in M_v\} = P_v^{mg}(\boldsymbol{x}), \tag{2.28}$$

так что вероятность того, что именно поступление новой задачи v-ВМ приведет к миграции, при условии, что система только что была в состоянии $\boldsymbol{x}$**,** общая условная вероятность миграции в состоянии $\boldsymbol{x}$ есть сумма по всем ВМ

$$P^{mg}(\boldsymbol{x}) = \sum_{v \in \mathcal{V}} P_v^{mg}(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathcal{X}. \tag{2.29}$$

Следовательно, вероятность миграции произвольной ВМ

$$P^{mg} = \sum_{\boldsymbol{x} \in \mathcal{X}} Pr\big(\mathcal{A} \mid \mathcal{B}(\boldsymbol{x})\big) \cdot Pr\big(\mathcal{B}(\boldsymbol{x})\big) = \sum_{\boldsymbol{x} \in \mathcal{X}} P^{mg}(\boldsymbol{x}) \cdot \pi(\boldsymbol{x}), \quad v \in \mathcal{V}, \boldsymbol{x} \in \mathcal{X}. \tag{2.30}$$

Утверждение доказано. □

Из доказанного Утверждения 2.1 непосредственно вытекает ряд характеристик, описывающих поведение системы.

Следствие 2.1. Для модели с перемещением заявок между группами приборов вероятность миграции виртуальной машины v на сервер s – вероятность того, что виртуальная машина v в результате миграции будет перенесена на сервер $s \in \mathcal{S}$, определяется выражением

$$P_{vs}^{mg2sr} = \sum_{\boldsymbol{x} \in \mathcal{X}} \mathbf{1}\{s = s_v^*(\boldsymbol{x})\}\, p_v^{mg}(\boldsymbol{x})\pi(\boldsymbol{x}), \quad v \in \mathcal{V}, s \in \mathcal{S}, \tag{2.31}$$

где $s_v^*(\boldsymbol{x})$ целевой сервер миграции виртуальной машины v, определяемый принятой политикой управления в состоянии $\boldsymbol{x}$.

Вероятность миграции любой виртуальной машины на сервер s – вероятность того, что миграция в системе осуществляется на сервер s, независимо от номера виртуальной машины:

$$P_s^{mg2sr} = \sum_{v \in \mathcal{V}} P_{vs}^{mg2sr}, \qquad s \in \mathcal{S}. \tag{2.31}$$

Среднее использование полосы пропускания виртуальной машиной $\boldsymbol{v}$ при миграции, т.е. среднее значение ресурса полосы пропускания, потребляемого виртуальной машиной v в процессе миграции, определяется как

$$\bar{b}_v^{mg} = \sum_{\boldsymbol{x} \in \mathcal{X}} n_v\ b_v\, p_v^{mg}(\boldsymbol{x})\pi(\boldsymbol{x}), \qquad v \in \mathcal{V}. \tag{2.33}$$

Среднее использование полосы пропускания при миграции любой виртуальной машины, т.е. агрегированная характеристика среднего использования полосы пропускания при миграции в системе:

$$\bar{b}^{mg} = \frac{1}{V}\sum_{v\in\mathcal{V}} \bar{b}_v^{mg}. \tag{2.34}$$

2.5 Численный анализ модели для сценария обслуживания иммерсивных сервисов

В данном разделе представлены результаты численного анализа марковской модели системы обслуживания с миграцией виртуальных машин, разработанной в главе 2. Цель анализа заключается в исследовании влияния выбора политики миграции на показатели качества обслуживания и характер использования вычислительных ресурсов. Все показатели эффективности вычисляются на основе стационарного распределения вероятностей состояний, полученного из системы уравнений равновесия численно.

В качестве численного примера рассматривается система, состоящая из $S = 3$ серверов и $V = 2$ виртуальных машин. Такая конфигурация позволяет учесть гетерогенность вычислительной среды и проанализировать перераспределение нагрузки между серверами при использовании различных политик миграции. Серверы обладают различной пропускной способностью, что отражает неоднородность вычислительной инфраструктуры и исключает искусственное выравнивание нагрузки.

Первая виртуальная машина моделирует сервис расширенной реальности (Extended Reality, XR), характеризующийся высокой чувствительностью и выраженной вариативностью входной нагрузки. Вторая виртуальная машина соответствует голографическому сервису, для которого характерна более стабильная нагрузка. Однако работа такого сервиса требует высокой пропускной способности сети [140, 141]. Данная постановка позволяет рассматривать XR-сервис в качестве доминирующего источника входного трафика, а голографический сервис, в качестве фоновой компоненты нагрузки. Численные значения параметров системы и режимы их варьирования приведены в таблице 2.1.

Таблица 2.1 - Исходные данные для численного анализа модели 1

Элемент системы	Параметр	Обозначение	Значение	Режим анализа
ВМ 1 (XR)	Интенсивность поступления	λ_1	1-3 задачи/с	Варьируется (сценарий 1)
	Среднее время обслуживания	μ_1^{-1}	1 с	Фиксировано
	Требуемая полоса на задачу	b_1	700 Мбит/с	Фиксировано
ВМ 2(голография)	Интенсивность поступления	λ_2	2 задачи/с	Фиксировано
	Среднее время обслуживания	μ_2^{-1}	0,5 - 1,5 с	Варьируется (сценарий 2)
	Требуемая полоса на задачу	b_2	1200 Мбит/с	Фиксировано
Сервер 1	Пропускная способность	C_1	4 Гбит/с	Фиксировано
Сервер 2	Пропускная способность	C_2	7 Гбит/с	Фиксировано
Сервер 3	Пропускная способность	C_3	10 Гбит/с	Фиксировано

Выбор варьируемых параметров обусловлен необходимостью раздельной оценки ключевых факторов, определяющих загрузку системы и вероятность отказов. В первой серии экспериментов изучается влияние интенсивности поступления задач λ_1 на виртуальную машину XR-сервиса. Параметр λ_1 изменяется в заданном диапазоне, в то время как параметры обслуживания и второй ВМ фиксированы (Таблица 2.1). Такой подход позволяет оценить чувствительность системы к росту нагрузки и сравнить политики миграции в экстремальных условиях. Результаты представлены на Рис. 2.7-2.14.

Результаты показывают, что при увеличении интенсивности поступления задач вероятность блокировки возрастает для обеих виртуальных машин, что обусловлено ограниченностью пропускной способности серверов. Вместе с тем применение политики прогнозируемой миграции обеспечивает устойчивое снижение вероятности отказов по сравнению со второй политикой. Эффект проявляется для обеих виртуальных машин и в рассматриваемой конфигурации выражен умеренно: для XR-сервиса снижение вероятности блокировки составляет порядка нескольких процентов (около 2%), а для голографического сервиса несколько больше (порядка 4–5%). Такое различие связано с тем, что перераспределение нагрузки, инициируемое миграцией, косвенно влияет на доступность пропускной способности.

Снижение вероятности блокировки сопровождается изменением структуры загрузки серверов. Наиболее заметное перераспределение нагрузки наблюдается для второго сервера: при использовании прогнозируемой политики его средняя загрузка уменьшается (в рассматриваемых расчетах порядка 10–13%), тогда как загрузка первого и третьего серверов возрастает незначительно. Это указывает на более равномерное использование системы при применении первой политики без выраженного роста суммарной нагрузки на систему.

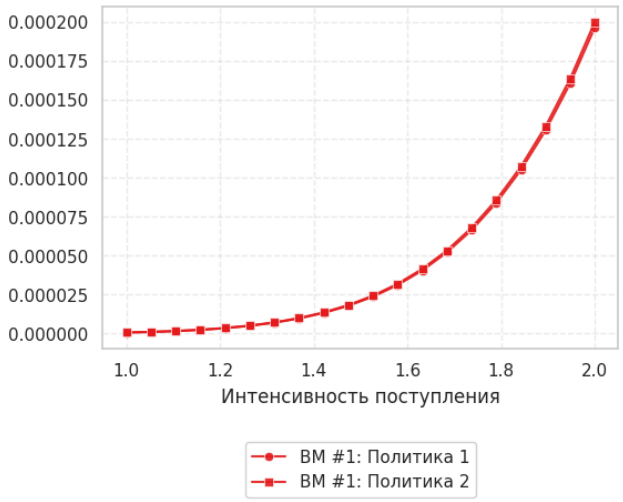

Рисунок 2.7 - Вероятность блокировки задачи ВМ 1

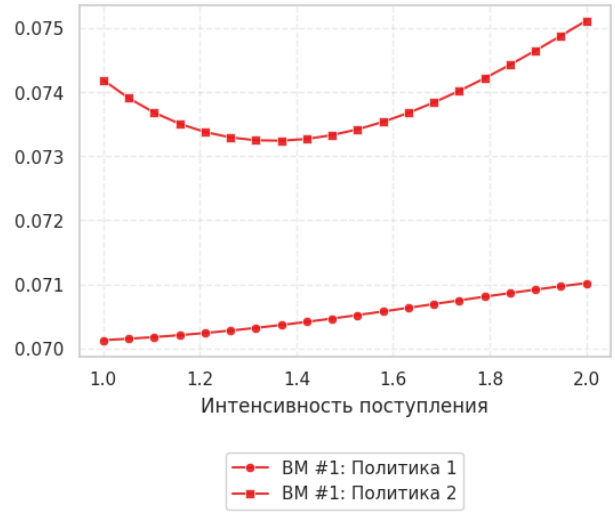

Рисунок 2.8 - Вероятность блокировки задачи ВМ 2

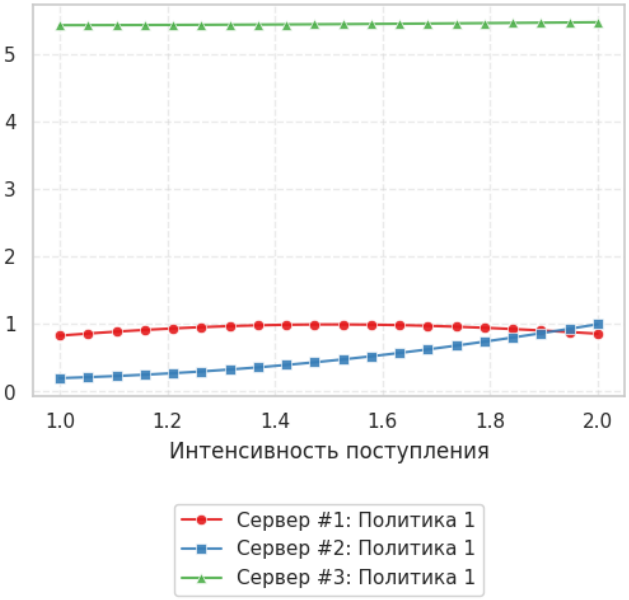

Рисунок 2.9 - Средняя загрузка серверов для политики 1

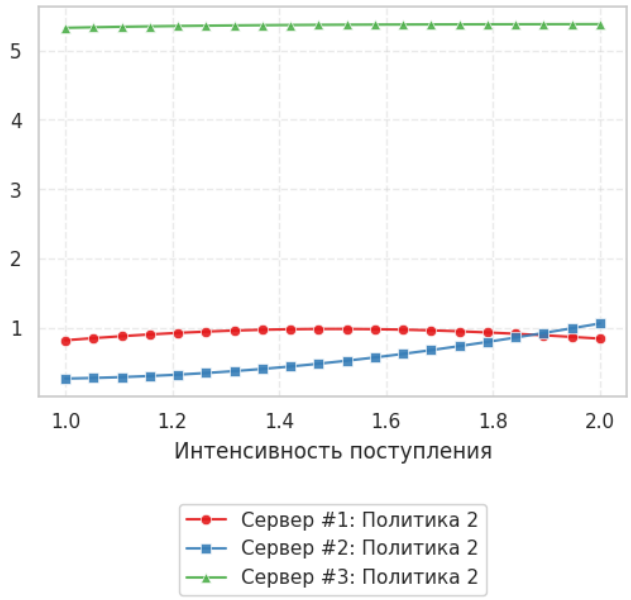

Рисунок 2.10 - Средняя загрузка серверов для политики 2

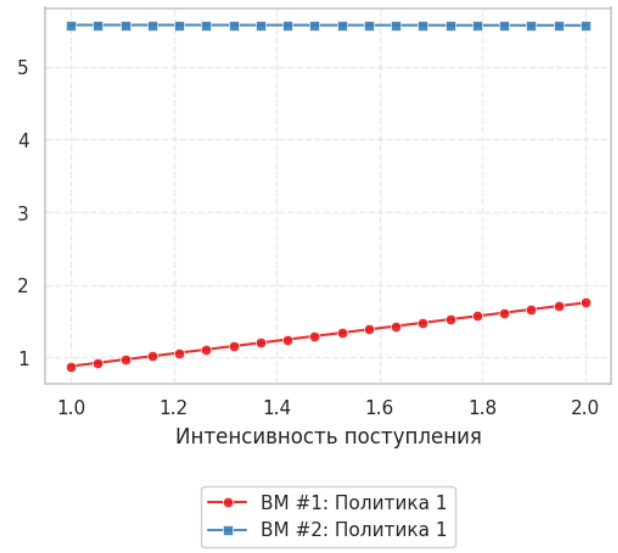

Рисунок 2.11 - Средняя загрузка ВМ для политики 1

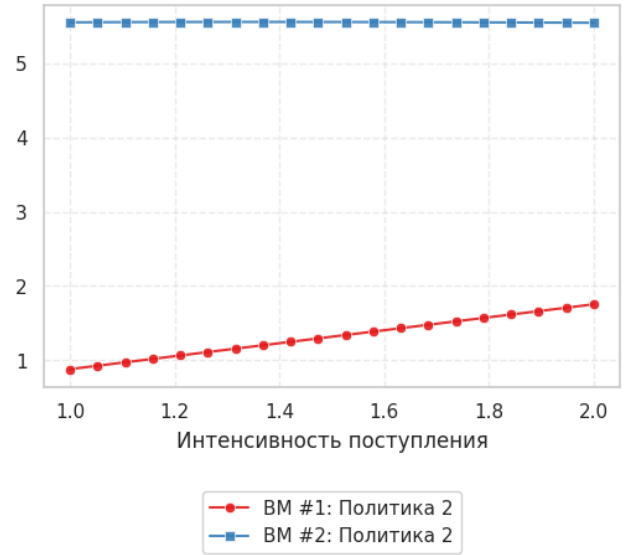

Рисунок 2.12 - Средняя загрузка ВМ для политики 2

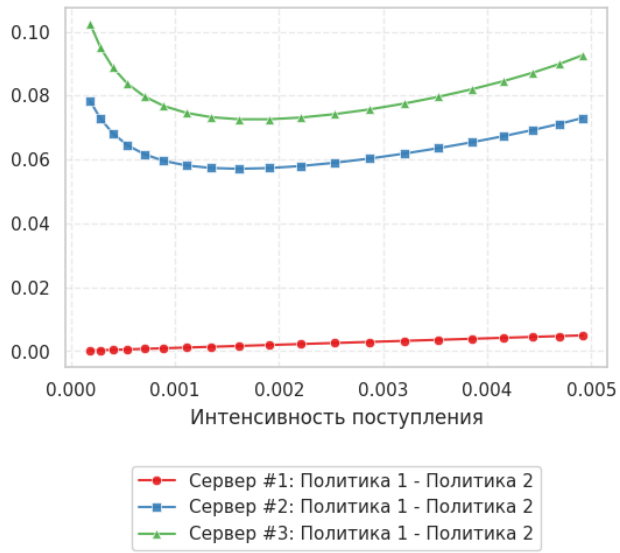

Рисунок 2.13 - Разница между политиками по загрузке серверов

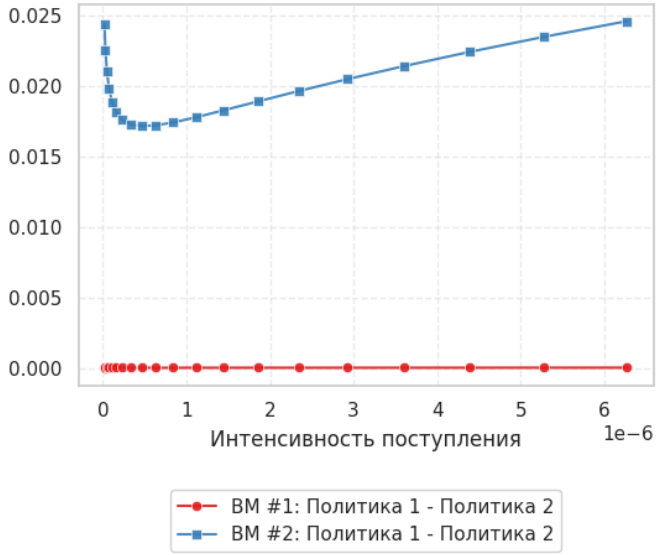

Рисунок 2.14 - Разница между политиками по загрузке ВМ

Во втором режиме вычислительного эксперимента фиксируются интенсивности поступления задач, а исследуется влияние производительности обработки задач XR-сервиса. Интенсивность обслуживания μ_2 варьируется в заданном диапазоне при неизменных параметрах поступления и пропускной способности серверов. Такой режим позволяет оценить, в какой мере увеличение скорости обслуживания способно компенсировать рост нагрузки и как выбор политики миграции проявляется при изменениях в системе. Результаты численного анализа при варьировании интенсивности обслуживания задач виртуальной машины XR-сервиса представлены на рисунках 2.15-2.22.

Полученные зависимости демонстрируют снижение вероятности блокировки и уменьшение средней загрузки при увеличении μ_1, что соответствует ожидаемому поведению системы обслуживания при росте интенсивности обслуживания. При этом различия между политиками сохраняются во всем диапазоне значений параметров: политика прогнозируемой миграции обеспечивает дополнительное снижение вероятности блокировки по сравнению со второй политикой. В рассматриваемых расчетах величина выигрыша остается умеренной и составляет порядка 1–2% для XR-сервиса и порядка 3–4% для голографического сервиса, что согласуется с выводами первого режима эксперимента.

Изменения средней загрузки серверов при переходе к прогнозируемой политике носят умеренный характер: в среднем наблюдается снижение загрузки наиболее нагруженного сервера порядка 9–10% при незначительном увеличении загрузки остальных серверов (как правило, не более 1–2%). Это позволяет заключить, что

улучшение показателей качества обслуживания достигается за счет перераспределения нагрузки, а не за счет существенного роста ресурсопотребления инфраструктуры.

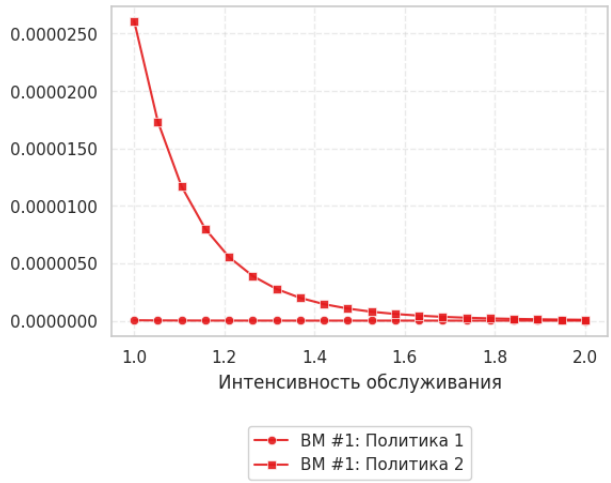

Рисунок 2.15 - Вероятность блокировки задач ВМ 1

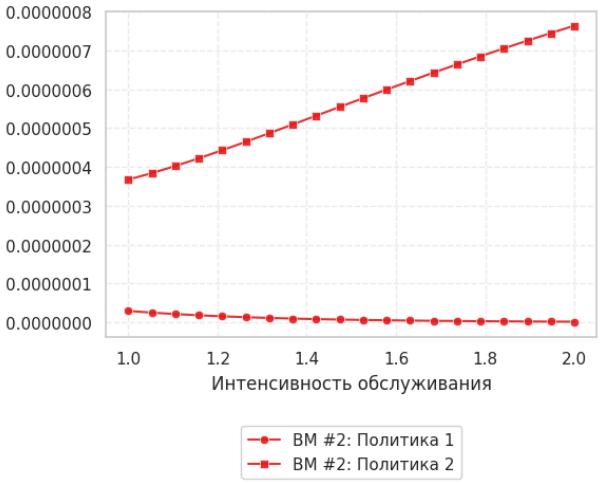

Рисунок 2.16 - Вероятность блокировки задач ВМ 2

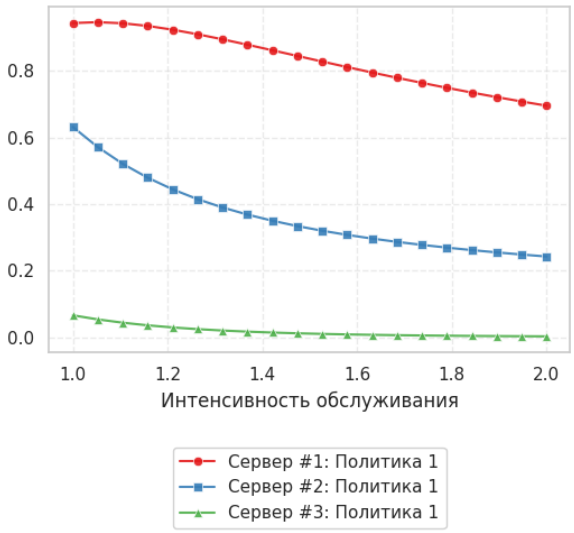

Рисунок 2.17 - Средняя загрузка серверов для политики 1

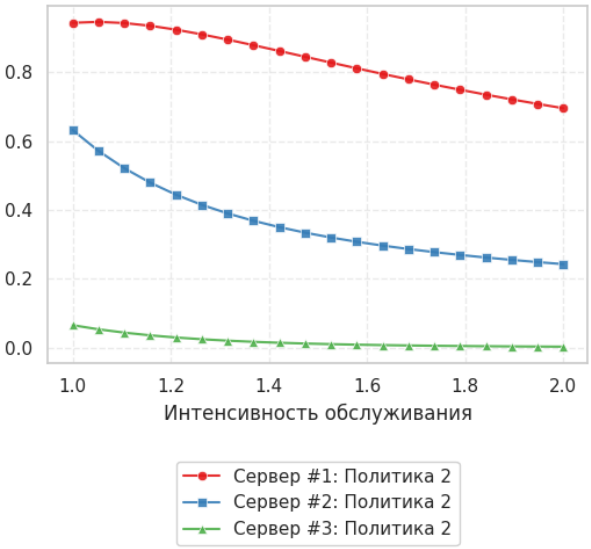

Рисунок 2.18 - Средняя загрузка серверов для политики 2

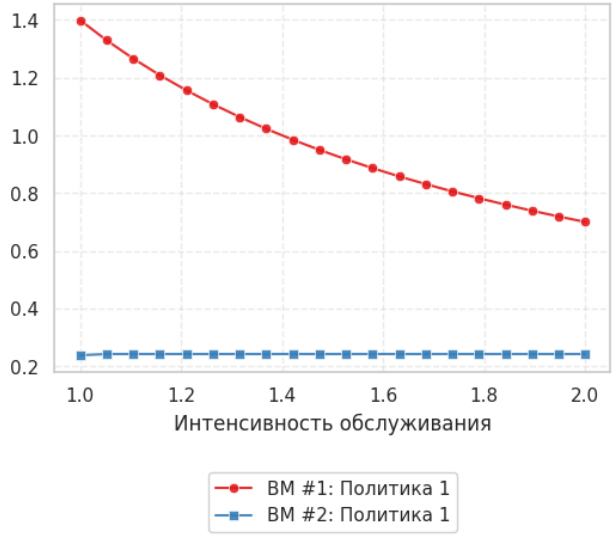

Рисунок 2.19 - Средняя загрузка виртуальных машин для политики 1

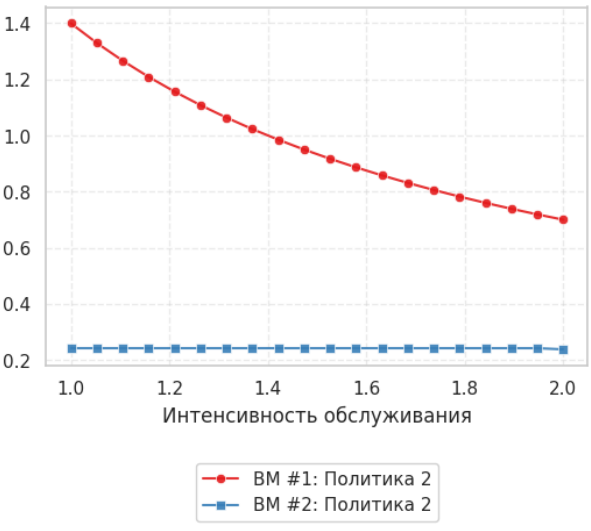

Рисунок 2.20 - Средняя загрузка виртуальных машин для политики 2

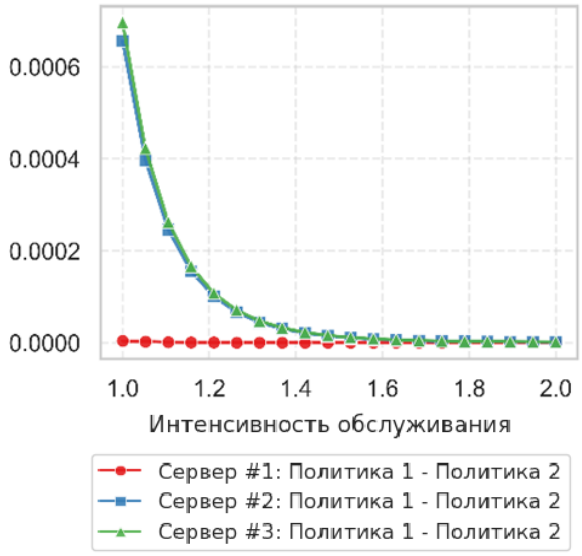

Рисунок 2.21 - Разница между политиками по загрузке серверов

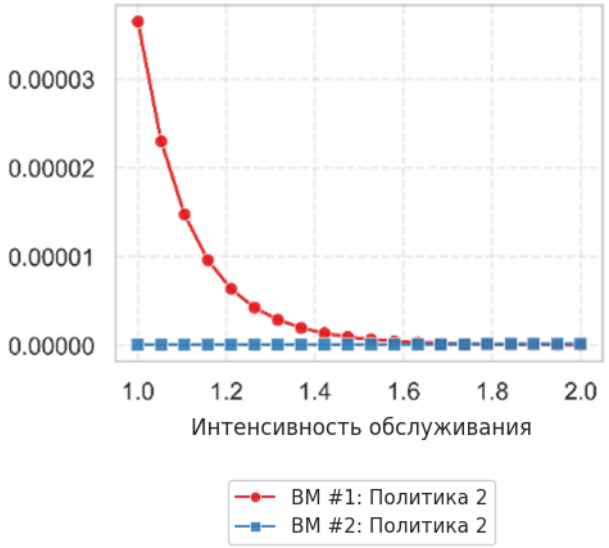

Рисунок 2.22 - Разница между политиками по загрузке ВМ

В целом результаты численного анализа подтверждают, что выбор политики миграции виртуальных машин оказывает заметное влияние на показатели качества обслуживания и распределение вычислительных ресурсов в системе. Учет перспективного состояния системы при принятии решений о миграции позволяет снизить вероятность блокировки задач и обеспечить более сбалансированное использование серверов.

Глава 3 Модели миграции сервисов в граничо-облачной архитектуре

3.1 Система с перемещением заявок между общей и индивидуальными группами приборов

Во третьей главе получены результаты № 2 и № 3, связанные с построением и анализом моделей систем массового обслуживания для описания миграции пользователей сервисов между граничной и облачной инфраструктурами Основной целью главы является разработка математических моделей гибридной вычислительной системы, в которой пользовательские запросы различных сервисов перераспределяются между вычислительными ресурсами, расположенными на узлах и в облачных дата-центрах, с учетом требований к межконцевой задержке и коррелированного характера входного трафика. Постановка задачи представлена в тезисах [142].

Граничная облачная вычислительная система предназначена для обслуживания пользовательских запросов нескольких сервисов. Множество сервисов обозначается как $\mathcal{K} = \{1, \dots, K\}$, что соответствует и количеству облачных серверов. Поток пользовательских запросов сервиса $k \in \mathcal{K}$ является пуассоновским с интенсивностью λ_k. Время обслуживания запросов сервиса k имеет экспоненциальное распределение со средним значением $\frac{1}{\mu_k}$ с. Каждый запрос от пользователя на сервис k требует выделения пропускной способности в объеме b_k.

Канал связи между пользователями и МЕС-узлом характеризуется пропускной способностью C_0 бит/с. Канал связи между пользователями и облачной инфраструктурой для сервиса k характеризуется пропускной способностью C_k бит/с. Задержка обслуживания запроса сервиса k в облачном сервере обозначается как d_k с., а задержка обслуживания запроса сервиса k на МЕС как d_0 с. Предполагается выполнение неравенства $d_0 < d_k, k \in \mathcal{K}$, что отражает преимущество обслуживания на МЕС с точки зрения задержки.

Пусть МЕС-узел в каждый момент времени может обслуживать пользовательские запросы только одного s-сервиса, $s \in \mathcal{K}$. При этом для выбранного сервиса допускается

одновременное обслуживание части запросов s-сервиса на MEC и части запросов в облачном s-сервере, что позволяет перераспределять нагрузку между двумя уровнями системы. Схема модели граничных облачных вычислений с миграцией сервисов представлена на рисунке 3.1.

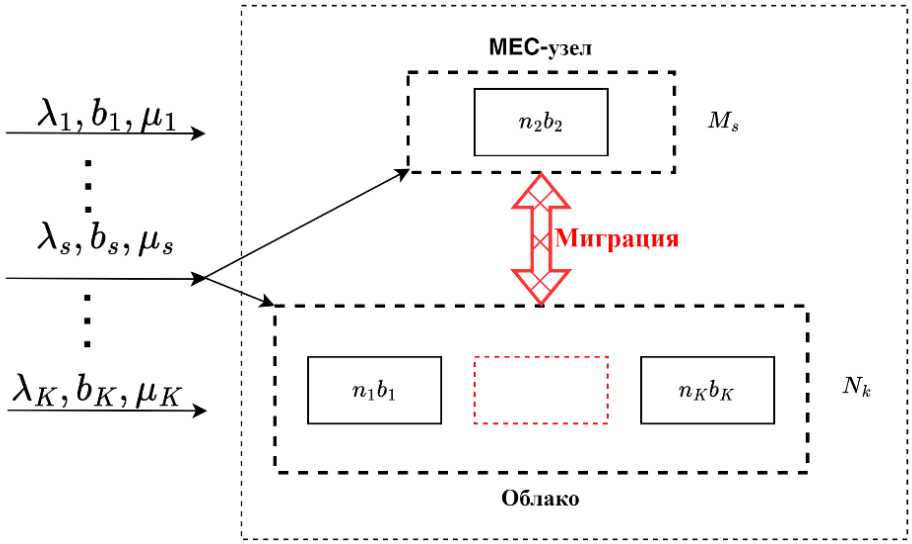

Рисунок 3.1 – Схема модели с перемещением заявок между общей и индивидуальными группами приборов

Динамика рассматриваемой гибридной системы описывается марковским процессом $\boldsymbol{X}(t), t \geq 0$. Состояние системы в момент времени t задается вектором $\boldsymbol{X}(t) = \big(\boldsymbol{N}(t), M(t), S(t)\big)$, где $\boldsymbol{N}(t) = \big(N_1(t), \dots, N_K(t)\big)$ вектор, компоненты которого определяют число пользователей соответствующих сервисов в системе, $S(t) \in \{0,1, \dots, K\}$ индекс сервиса, размещенного на MEC-узле, а $M(t)$ число пользователей сервиса $S(t)$, обслуживаемых на MEC-узле.

Следует уточнить, что при $S(t) = 0$ полагается $M(t) = 0$, что соответствует состоянию, при котором MEC-узел не обслуживает ни один сервис, т.е. система пуста. Для $S(t) = s \in \mathcal{K}$ величина $N_s(t)$ общее число пользователей сервиса s в системе, из которых $M(t)$ обслуживаются на MEC-узле, а $N_s(t) - M(t)$ на облачном сервере этого сервиса. Для сервисов $k \neq s$ все $N_k(t)$ пользователей обслуживаются в соответствующих облачных k-серверах.

Состояние гибридной граничной облачной системы в произвольный момент времени описывается тройкой

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) = (n_1, \dots, n_K, m, s), \tag{3.1}$$

где $\boldsymbol{n} = (n_1, \dots, n_K)$ вектор числа пользователей по сервисам, $s \in \{0,1, \dots, K\}$ индекс сервиса, размещенного на МЕС-узле, m число пользователей сервиса s, обслуживаемых на МЕС-узле.

Пропускная способность МЕС-узла и облачных серверов сервисов ограничена и в состоянии системы $\boldsymbol{x}$ задается следующим образом

$$\begin{gathered} c_0(\boldsymbol{x}) = m\, b_s \le C_0, s \in \mathcal{K}, \\ c_s(\boldsymbol{x}) = (n_s - m)\, b_s \le C_s, \\ c_k(\boldsymbol{x}) = n_k\, b_k \le C_k, k \ne s, k \in \mathcal{K}. \end{gathered} \tag{3.2}$$

Ограничения на допустимые численности пользователей

$$M_s = \left\lfloor \frac{C_0}{b_s} \right\rfloor, \qquad N_k = \left\lfloor \frac{C_k}{b_k} \right\rfloor,$$

где M_s максимальное число пользователей сервиса s, которые могут одновременно обслуживаться на МЕС-узле, а N_k максимальное число пользователей сервиса k, которые могут обслуживаться в облачном сервере.

Пространство $\mathcal{X}$ состояний СП $\boldsymbol{X}(t)$ определяется всеми возможными комбинациями $(\boldsymbol{n}, m, s)$, удовлетворяющими ограничениям по полосе пропускания, а также принятому правилу, согласно которому МЕС-узел в каждый момент времени может размещать не более одного сервиса. Представление в виде объединения трех непересекающихся классов состояний выглядит следующим образом:

1. пустая система: $\boldsymbol{x} = (\boldsymbol{0}, 0,0)$. В системе отсутствуют пользователи;
2. МЕС-узел не заполнен: $0 < m < M_s$ и $s > 0$. В этом случае на МЕС размещен сервис s, причем имеется свободная емкость. Все m пользователей сервиса s обслуживаются на МЕС, то есть $n_s = m$. Для остальных сервисов $k \ne s$ выполняется $0 \le n_k \le N_k$;
3. МЕС-узел заполнен: $m = M_s$ и $s > 0$. МЕС полностью занят обслуживанием M_s пользователей сервиса s. Дополнительные пользователи данного сервиса обслуживаются в облачном сервере, поэтому $n_s \ge M_s$. Для всех $k \ne s$ выполняется $0 \le n_k \le N_k$.

С учетом указанных ограничений множество допустимых состояний системы имеет вид

$$\begin{gathered} \widetilde{\mathcal{X}} = \{(\boldsymbol{0}, 0,0)\} \cup \\ \cup \{(\boldsymbol{n}, m, s) \colon n_s = m,\ 0 < m < M_s,\ 0 \le n_k \le N_k, k \in \mathcal{K} \setminus \{s\},\ s \in \mathcal{K}\} \cup \\ \cup \{(\boldsymbol{n}, M_s, s) \colon M_s \le n_s \le N_s,\ 0 \le n_k \le N_k, k \in \mathcal{K} \setminus \{s\},\ s \in \mathcal{K}\}. \end{gathered} \tag{3.3}$$

Пространство состояний $\mathcal{X} \subseteq \widetilde{\mathcal{X}}$ определяется переходами, разрешенными в рамках политики миграции, минимизирующей задержку, описанной далее в разделе 3.2.

В рамках рассматриваемой модели выбор размещения сервиса на MEC-узле трактуется как элемент управления функционированием системы массового обслуживания. Управляющее решение заключается в выборе допустимого варианта размещения сервиса и распределения пользовательской нагрузки между MEC-узлом и облачной частью системы при заданном состоянии нагрузки. Для формализации политики размещения сервиса вводится суммарная задержка пользователей как показатель качества обслуживания системы. В рамках принятой динамической модели управляющее решение реализуется не напрямую, а через выбор одного из допустимых переходов марковского процесса, инициируемых наступлением соответствующих событий. Поэтому задача минимизации суммарной задержки эквивалентна задаче выбора допустимого перехода, приводящего к состоянию с наименьшей суммарной задержкой. Формализация допустимых переходов и правил выбора оптимального перехода приведена в разделе 3.2.

3.2 Алгоритм миграции сервисов для минимизации суммарной межконцевой задержки

В качестве целевого показателя качества обслуживания используется суммарная межконцевая задержка пользователей (E2E-delay) в состоянии $\boldsymbol{x} = (\boldsymbol{n}, m, s)$. Суммарная задержка представляется как сумма вкладов пользователей, обслуживаемых в облачной части системы, и пользователей, обслуживаемых на MEC-узле. Задержка пользователей, обслуживаемых в облачной части системы, в состоянии $\boldsymbol{x}$ равна

$$d_c(\boldsymbol{n}, m, s) = \sum_{k \in \mathcal{K} \setminus \{s\}} n_k \; d_k + (n_s - m)\, d_s, \tag{3.4}$$

Суммарная задержка для пользователей, обслуживаемых на MEC-узле в состоянии $\boldsymbol{x}$, задается выражением

$$d_0(\boldsymbol{n}, m, s) = d_0(m) = m\, d_0, \tag{3.5}$$

тогда общая задержка, равная сумме задержек на обоих вычислительных уровнях, вычисляется по формуле

$$d(\boldsymbol{n}, m, s) = d_c(\boldsymbol{n}, m, s) + d_0(\boldsymbol{n}, m, s) = \sum_{k \in \mathcal{K}} n_k\, d_k \; + \; m\,(d_0 - d_s). \tag{3.6}$$

Перенос m пользователей текущего s-сервиса на МЕС-узел изменяет суммарную задержку на величину, пропорциональную разности $d_0 - d_s$.

Динамику системы опишем через случайные события A_{ik}, $i \in \{1,2\}$, $k \in \mathcal{K}$, т.е. событие A_{1k} соответствует поступлению новой заявки пользователя сервиса k, а событие A_{2k} завершению обслуживания пользователя сервиса k. После наступления события система принимает управляющее решение, определяющее конфигурацию МЕС-узла: выбирается, какой сервис размещается на МЕС-узле и сколько пользователей этого сервиса обслуживаются на граничном узле. Будем называть действием пару $a = (m', s')$, где s' индекс сервиса после наступления события A_{ik}, а m' число пользователей сервиса s'. Для фиксированного состояния $\boldsymbol{x}$ и события A_{ik} введем множество допустимых действий $\mathcal{A}_{ik}(\boldsymbol{x})$, под которым будем понимать множество всех пар $a = (m', s')$, которые могут быть реализованы после наступления события A_{ik}, при условии, что система находилась в состоянии $\boldsymbol{x}$ и выполняются ограничения на пропускную способность МЕС-узла и облачных серверов.

Лемма 3.1. Для модели с перемещением заявок между общей и индивидуальными группами приборов в состоянии $\boldsymbol{x}$ при наступлении события A_{ik}, $i \in \{1,2\}$, $k \in \mathcal{K}$ множество допустимых действий $\mathcal{A}_{ik}(\boldsymbol{x})$ имеет вид

$$\mathcal{A}_{ik}(\boldsymbol{x}) = \begin{cases} \{(1,k)\}, s = 0, k \in \mathcal{K}, i = 1, \\ \{(min(n_s + 1, M_s), s), (min(n_j, M_j), j)\}, j \in \mathcal{K}\backslash\{s\}, s \neq 0, k = s, i = 1, \\ \{(min(n_k + 1, M_k), k), (min(n_s, M_s), s)\}, k \in \mathcal{K}\backslash\{s\}, s \neq 0, i = 1, \\ \{(min(n_s - 1, M_s), s), (min(n_j, M_j), j)\}, j \in \mathcal{K}\backslash\{s\}, s \neq 0, k = s, m > 0, i = 2, \\ \{(min(n_s, M_s), s)\}, s \neq 0, k = s, n_s > 0, i = 2. \end{cases} \quad (3.7)$$

Доказательство.

Рассмотрим произвольное состояние $\boldsymbol{x} = (\boldsymbol{n}, m, s)$ и событие A_{ik}. По определению, действие $a = (m', s')$ допустимо тогда, когда после его применения выполняются ограничения на пропускную способность МЕС-узла и облачных серверов:

$$m' b_{s'} \leq C_0, \qquad (n_{s'} - m') b_{s'} \leq C_{s'}, \qquad n_j b_j \leq C_j, j \neq s',$$

что эквивалентно условиям $m' \leq M_{s'}$ и $n_j \leq N_j$. Далее последовательно рассмотрим все возможные типы событий

а) Множество действий при поступлении запроса от пользователя

Пусть реализуется событие A_{1k}, соответствующее поступлению новой заявки сервиса k в состоянии $\boldsymbol{x} = (\boldsymbol{n}, m, s)$. Все возможные переходы при этом перечислены в таблице 3.1.

Таблица 3.1 – Поступление запроса от пользователя

№	Условие	Интерпретация	Миграция	Смена пары МЕС-узла
(а) МЕС-узел свободен				
(а)	$s = 0$	МЕС не обслуживает сервис; поступление любого k инициирует размещение сервиса на МЕС	Нет	$(0,0) \longrightarrow (1, k)$
(б) МЕС обслуживает сервис $s \neq 0$, поступает запрос текущего сервиса $k = s$				
(б-1)	$k = s$, $n_s < N_s$, $m < M_s$	МЕС не полностью загружен; новый пользователь сервиса s направляется на МЕС.	Нет	$(m, s) \longrightarrow (m + 1, s)$
(б-2)	$k = s$, $n_s < N_s$, $m = M_s$	МЕС полностью загружен; новый пользователь сервиса s обслуживается в облачном сервере сервиса s (состояние МЕС не меняется).	Нет	$(M_s, s) \rightarrow (M_s, s)$
(б) Поступает запрос текущего сервиса $k = s$, но допускается смена сервиса на МЕС на $l \neq s$				
(б-3)	$k = s$, $n_s < N_s$, $0 < n_l \leq M_l$, $l \in \mathcal{K} \setminus \{s\}$	После поступления запроса сервиса s выполняется смена сервиса на МЕС на l; на МЕС размещается полный текущий объем пользователей сервиса l.	Да (на сервис l, полная по l).	$(m, s) \rightarrow (n_l, l)$
(б-4)	$k = s$, $n_s < N_s$, $n_l > M_l$, $l \in \mathcal{K} \setminus \{s\}$	После поступления запроса сервиса sвыполняется смена сервиса на МЕС на l; из-за ограничения МЕС на МЕС размещается только допустимая часть пользователей сервиса l, то есть M_l.	Да (на сервис l, частич. по l).	$(m, s) \rightarrow (M_l, l)$.
(в) МЕС обслуживает сервис $s \in \mathcal{K}$, поступает запрос другого сервиса $k \neq s$				
(в-1)	$k \neq s$, $n_k < N_k$	Запрос сервиса k обслуживается в облачном сервере сервиса k, состояние МЕС не изменяется.	Нет	$(m, s) \rightarrow (m, s)$
(в-2)	$k \neq s$, $n_k < M_k$	Допускается смена сервиса на МЕС на k; после поступления новой заявки на МЕС размещается весь текущий объем пользователей сервиса k и поступивший пользователь.	Да (на сервис k, полная по k).	$(m, s) \rightarrow (n_k + 1, k)$.
(в-3)	$k \neq s$, $M_k \leq n_k < N_k$	Допускается смена сервиса на МЕС на k; из-за ограничения МЕС на МЕС размещается M_k пользователей сервиса k.	Да (на сервис k, частич. по k).	$(m, s) \rightarrow (M_k, k)$

Если МЕС-узел свободен ($s = 0$), на МЕС-узле не размещен ни один сервис. Поступление первой заявки сервиса k приводит к единственному допустимому действию: размещению сервиса k на МЕС-узле и началу обслуживания одной заявки на МЕС, что соответствует переходу $(0,0) \rightarrow (1, k)$ (строка (а) таблицы 3.2). Поэтому в этом случае множество допустимых действий имеет вид $\mathcal{A}_{1k}(\boldsymbol{x}) = \{(1, k)\}$.

Если МЕС-узел обслуживает сервис $s \neq 0$ и поступает заявка текущего сервиса ($k = s$), число заявок сервиса s после события становится $n_s + 1$. Таблица 3.2 показывает, что при $n_s < N_s$ и $m < M_s$ новая заявка направляется на МЕС-узел, и происходит переход $(m, s) \rightarrow (m + 1, s)$ (случай (б-1)). При $n_s < N_s$ и $m = M_s$ МЕС-узел полностью загружен, и новая заявка обслуживается в облачном сервере при неизменной конфигурации МЕС

$(M_s, s) \to (M_s, s)$ (случай (б-2)). Кроме того, допускается смена сервиса на МЕС-узле на сервис $l \in \mathcal{K} \setminus \{s\}$. Так при $0 < n_l \le M_l$ на МЕС-узле размещаются все n_l заявок сервиса l, переход $(m, s) \to (n_l, l)$ (случай (б-3)). И при $n_l > M_l$ на МЕС-узел выносится только M_l заявок сервиса l, переход $(m, s) \to (M_l, l)$ (случай (б-4)).

Если МЕС-узел обслуживает сервис s и поступает заявка другого сервиса ($k \ne s$), то по таблице 3.2 новая заявка сервиса k может быть обслужена в соответствующем облачном сервере при сохранении текущей конфигурации МЕС, что задает переход $(m, s) \to (m, s)$ (случай (в-1)). Возможна и смена МЕС-сервиса на k. Если $n_k < M_k$, то после поступления все $n_k + 1$ заявок сервиса k размещаются на МЕС-узле, переход $(m, s) \to (n_k + 1, k)$ (случай (в-2)). И если $M_k \le n_k < N_k$, на МЕС-узле может обслуживаться только M_k заявок сервиса k, переход $(m, s) \to (M_k, k)$ (случай (в-3)).

Таблица 3.2 – Множество действий при поступлении запроса от пользователя

	$k = s$	$k \ne s$
все k-задачи могут поместиться на МЕС	Условие: $s \in \mathcal{K}, k = s, n_s < N_s, m < M_s$ Алгоритм: $\mathcal{A}_{1k}(\boldsymbol{x}) = \mathcal{K}$. Переход: $\boldsymbol{x}' = (\boldsymbol{n} + e_s, m + 1, s)$.» Интерпретации результата выбора: – если выбирается $s' = s \to$ нет миграции; – если выбирается $s' \ne s \to$ миграция; – если выбирается $l \ne s$, то: при $n_l \le M_l$ – «все заявки сервиса l» на МЕС; при $n_l > M_l$ – «часть остается в облачном сервере» (на МЕС только M_l).	Условие: $s \in \mathcal{K}, k \ne s, n_k < N_k$. Алгоритм: $\mathcal{A}_{1k}(\boldsymbol{x}) = \{s, k\}$. Переходы: Ветка «выбран s» (то есть МЕС остается на текущем сервисе): выбран $s \Rightarrow$ нет миграции; поступившая k-заявка не переводится на МЕС, поэтому $\boldsymbol{x}' = (\boldsymbol{n} + e_k, m, s)$. Ветка «выбран k» (то есть МЕС переключается на новый сервис): выбран $k \Rightarrow s' = k \Rightarrow$ миграция. Далее применяется правило «все/часть в облачном сервере» для сервиса k. Если $n_k < M_k$, то $n_k + 1 \le M_k$ – «все заявки сервиса k на МЕС», и состояние после поступления $\boldsymbol{x}' = (\boldsymbol{n} + e_k, n_k + 1, k)$. Интерпретации результата выбора: – $s' = s$ – нет миграции; – $s' = k$ – миграция; – при $s' = k$: если $n_k + 1 \le M_k$ – все заявки сервиса k на МЕС, если позже $n_k > M_k$ – часть заявок сервиса k остается в облачном сервере
не все k-задачи могут поместиться на МЕС	Условие: $s \in \mathcal{K}, k = s, n_s < N_s, m = M_s$. Алгоритм: $\mathcal{A}_{1k}(\boldsymbol{x}) = \mathcal{K}$. Переход: $\boldsymbol{x}' = (\boldsymbol{n} + \boldsymbol{e_s}, m, s)$. Интерпретации результата выбора: – если выбирается $s' = s$ – нет миграции; – если выбирается $s' \ne s$ – миграция; – если выбирается $l \ne s$, то: - при $n_l \le M_l$ – «все заявки сервиса l» на МЕС; - при $n_l > M_l$ – «часть остается в облачном сервере» (на МЕС только M_l). Поскольку МЕС по текущему сервису s уже заполнен ($m = M_s$), поступившая s-заявка при сохранении сервиса на МЕС не увеличивает m: сервис s сохраняется, и $\boldsymbol{x}' = (\boldsymbol{n} + e_s, m, s)$, т.е. новая заявка уходит в облачный сервер, а МЕС остается с тем же числом пользователей сервиса s.	Условие: $s \in \mathcal{K}, k \ne s, M_k \le n_k < N_k$. Алгоритм: $\mathcal{A}_{1k}(\boldsymbol{x}) = \{s, k\}$. Переходы: Ветка «выбран s» (МЕС остается на сервисе s): выбран $s \Rightarrow$ нет миграции, $\boldsymbol{x}' = (\boldsymbol{n} + e_k, m, s)$. Ветка «выбран k» (МЕС переключается на сервис k): выбран $k \Rightarrow$ миграция. Далее применяется правило «часть остается в облачном сервере» по сервису k. Поскольку $M_k \le n_k < N_k$, после поступления $n_k + 1 > M_k$, поэтому на МЕС может быть размещено только M_k заявок сервиса k. Следовательно, МЕС-часть фиксируется как $\boldsymbol{x}' = (\boldsymbol{n} + e_k, M_k, k)$. Интерпретации результата выбора: – $s' = s$ – нет миграции; – $s'\ k$ – миграция; – при $s' = k$: на МЕС обслуживаются M_k заявок сервиса k, остальные $(n_k + 1 - M_k)$ заявки сервиса k остаются в облачном сервере.

Собирая эти варианты воедино, получаем множества допустимых действий $\mathcal{A}_{1k}(\boldsymbol{x})$, которые резюмированы в таблице 3.3. При $s = 0, \mathcal{A}_{1k}(\boldsymbol{x}) = \{(1, k)\}$. Далее при $s \neq 0,\ k = s$ множество $\mathcal{A}_{1s}(\boldsymbol{x})$ содержит действие, соответствующее сохранению сервиса s на МЕС-узле с числом заявок $min\ (n_s + 1, M_s)$, и действия, соответствующие переключению МЕС-узла на сервисы $j \in \mathcal{K} \setminus \{s\}$ с числом заявок $min\ (n_j, M_j)$. При $s \neq 0, k \neq s$ множество $\mathcal{A}_{1k}(\boldsymbol{x})$ состоит из действия сохранения, текущего МЕС-сервиса s и действия переключения МЕС-узла на сервис k с числом заявок $min\ (n_k + 1, M_k)$. Таким образом, для всех событий поступления A_{1k} множество допустимых действий $\mathcal{A}_{1k}(\boldsymbol{x})$, полученное из физических ограничений модели и отраженное в таблице 3.2, полностью совпадает с формальным описанием, приведенным в таблице 3.3.

б) Множество действий при завершении обслуживания пользователя

Пусть реализуется событие A_{2k}, соответствующее завершению обслуживания пользователя сервиса k в состоянии $\mathbf{x} = (\mathbf{n}, \mathrm{m}, \mathrm{s})$. Все возможные переходы при этом перечислены в таблице 3.3.

Таблица 3.3 – Завершение обслуживания пользователя

№	Условие	Интерпретация	Миграция	Смена пары МЕС-узла
(г) завершение обслуживания пользователя на МЕС-узле (текущий сервис s)				
(г-1)	$s \in \mathcal{K}$, $m > 0$, $n_s = m$	Завершается обслуживание пользователя сервиса s на МЕС, при этом все пользователи сервиса s находились на МЕС (облачной части для s нет).	Нет	$(m, s) \rightarrow (\ m - 1, s)$
(г-2)	$s \in \mathcal{K}$, $m = M_s$, $n_s > M_s$	Завершается обслуживание на МЕС, но для сервиса s в облачном сервере еще есть пользователи ($n_s - m > 0$), поэтому освободившееся место на МЕС немедленно занимает пользователь того же сервиса s из облачного сервера (поддерживается $m = M_s$).	Нет	$(M_s, s) \rightarrow (\ M_s, s)$
(г-3)	$\mathrm{s} \in \mathcal{K}$, $\mathrm{m} > 0$, $0 < \mathrm{n_l} \leq \mathrm{M_l}$, $\mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}$	После завершения обслуживания на МЕС (для сервиса s) допускается смена сервиса на МЕС на l, на МЕС размещается весь текущий объем пользователей сервиса l.	Да (на сервис l, полная по l).	$(m, s) \rightarrow (\ n_l, l)$
(г-4)	$\mathrm{s} \in \mathcal{K}$, $\mathrm{m} > 0$, $\mathrm{n_l} > \mathrm{M_l}$, $\mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}$	После завершения обслуживания на МЕС допускается смена сервиса на МЕС на l, но из-за ограничения МЕС на МЕС размещается только допустимая часть пользователей сервиса l, то есть M_l.	Да (на сервис l, частич. по l).	$(m, s) \rightarrow (\ M_l, l)$
(д) завершение обслуживания пользователя в k-облачном сервере				
(д-1)	$\mathrm{k} \in \mathcal{K} \setminus \{\mathrm{s}\}, \mathrm{n_k} > 0$	Завершается обслуживание пользователя сервиса k в облачном сервере сервиса k, состояние МЕС не меняется.	Нет	$(m, s) \rightarrow (\ m, s)$
(д-2)	$\mathrm{s} \in \mathcal{K}$, $\mathrm{n_s} - \mathrm{m} > 0$	Завершается обслуживание пользователя сервиса s в облачном сервере сервиса s, МЕС-узел продолжает обслуживать сервис s и величина m не изменяется.	Нет	$(m, s) \rightarrow (\ m, s)$

Сначала рассмотрим ситуацию, когда завершается обслуживание пользователя на МЕС-узле, то есть $k = s$ (текущий МЕС-сервис). В таблице 3.4 этот случай описывается

в блоке (г). Если $s \in \mathcal{K}$, $m > 0$ и $n_s = m$, то завершается обслуживание пользователя сервиса s на МЕС-узле, при этом в облачном сервере для сервиса s больше нет пользователей. Происходит переход $(m, s) \to (\, m-1, s)$. Если же МЕС-узел остается на сервисе s, миграции нет, а тогда число заявок на МЕС-узле уменьшается на единицу (строка (г-1)). Далее если $s \in \mathcal{K}$, $m > 0$ и $n_s > m$, то завершается обслуживание пользователя на МЕС-узле, но в облачном сервере для сервиса s остаются заявки, поэтому освобожденное место на МЕС немедленно занимает пользователь того же сервиса s из облачного сервера, и число заявок на МЕС-узле не меняется (переход $(M_s, s) \to (\, M_s, s)$, строка (г-2)). В обоих подслучаях конфигурация МЕС-узла (сервис s и допустимое число заявок $min\,(n_s - 1, M_s)$ или M_s) остается в допустимых пределах.

Таблица 3.4 также предусматривает возможность смены сервиса на МЕС-узле после завершения обслуживания пользователя сервиса s. При $s \in \mathcal{K}$, $m > 0$ и наличии пользователей других сервисов $l \in \mathcal{K} \setminus \{s\}$ после события A_{2s} МЕС-узел может быть переключен на сервис l. Если $0 < n_l \leq M_l$, то на МЕС-узел переносится весь текущий объем заявок сервиса l, и переход имеет вид $(m, s) \to (\, n_l, l)$ (строка (г-3)). Далее если $n_l > M_l$, то на МЕС-узле размещается только M_l заявок сервиса l, то есть $(m, s) \to (\, M_l, l)$ (строка (г-4)). Эти переходы соответствуют режимам «все заявки сервиса l на МЕС» и «часть заявок сервиса l остается в облачном сервере» соответственно.

Теперь рассмотрим ситуацию, когда завершается обслуживание пользователя в облачном сервере, то есть $k \neq s$. В нижнем блоке таблицы 3.4 (случай (д)) показано, что при $k \in \mathcal{K} \setminus \{s\}$, $n_k > 0$ завершение обслуживания заявки сервиса k в облачном сервере не влияет на конфигурацию МЕС-узла, то происходит переход $(m, s) \to (\, m, s)$, МЕС-узел продолжает обслуживать сервис s с тем же числом заявок m, следовательно миграции нет (строка (д-1)). Аналогично, когда завершается обслуживание заявки сервиса s в облачном сервере (при $n_s - m > 0$), МЕС-узел остается на сервисе s, и конфигурация МЕС не меняется, что отражено в строке (д-2) таблицы 3.4.

Собирая все рассмотренные случаи, приходим к следующему описанию множества допустимых действий при событии A_{2k}. Если завершается обслуживание пользователя текущего МЕС-сервиса $(k = s,\ \ s \neq 0,\ \ m > 0)$, то $\mathcal{A}_{2s}(\boldsymbol{x})$ содержит действие, соответствующее уменьшению числа заявок сервиса s на МЕС-узле до $min\,(n_s - 1, M_s)$ при сохранении текущего сервиса, а также действия, соответствующие переключению МЕС-узла на любой сервис $j \in \mathcal{K} \setminus \{s\}$ с числом заявок $min\,(n_j, M_j)$, как это

зафиксировано переходами (г-3) и (г-4). Если же завершается обслуживание пользователя сервиса $k \neq s$ в облачном сервере, то конфигурация МЕС-узла не изменяется, и $\mathcal{A}_{2k}(\boldsymbol{x}) = \{(m, s)\}$. Эти множества в компактной форме представлены в формуле леммы для $\mathcal{A}_{2k}(\boldsymbol{x})$ и резюмированы в таблице 3.4, которая, подобно таблице 3.2, обобщает все подслучаи завершения обслуживания, подробно перечисленные в таблице 3.3.

Таблица 3.4 - Множество действий при завершении обслуживания пользователя

	$k = s$	$k \neq s$
все l-задачи могут поместиться на МЕС	Условие: $s \in \mathcal{K}, m > 0, 0 < n_l \leq M_l$. Алгоритм: $\mathcal{A}_{2k}(\boldsymbol{x}) = \mathcal{K}$. Переходы: - Ветка «выбран s»: Переход $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_s}, m - 1, s)$. Действие $(m', s') = (m - 1, s)$ и реализует $(min\ (n_s - 1, M_s), s)$; - Ветка «выбран l», $l \in \mathcal{K} \setminus \{s\}$: выбран $l \Rightarrow s' = l \Rightarrow$ миграция; «все задачи»: 0 < n_l ≤ M_l ⇒ «все задачи» на МЕС и переход $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_s}, n_l, l)$и действие $(m', s') = (n_l, l) = (min\ (n_l, M_l), l)$ Интерпретации результата выбора: - если выбирается $s' = s$ – нет миграции; - если выбирается $s' \neq s$ – миграция; - если выбирается $l \neq s$: при $n_l \leq M_l$ – «все пользователи класса l на МЕС», при $n_l > M_l$ – «часть остается в облачном сервере» (на МЕС только M_l)	Условие: $k \in \mathcal{K} \setminus \{s\}$, $n_k > 0$. Алгоритм: нет, т.к. состояние МЕС не изменяется Переход: $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_k}, m, s)$. Миграция: нет.
Не все l-задачи могут поместиться на МЕС	Условие: $s \in K,\ m > 0, n_l > M_l$ Алгоритм: $A_{2k}(\mathbf{x}) = \mathcal{K}$. Переходы: →Ветка «выбран s» (МЕС остается на s): выбран s ⇒нет миграции обычный уход: $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_s}, m - 1, s)$, уход при полной загрузке (как у тебя отдельно выделено): $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_s}, M_s, s)$, действие $(m', s') = (min\ (n_s - 1, M_s), s)$ →Ветка «выбран l», где $l \in \mathcal{K} \setminus \{s\}$:выбран $l \Rightarrow s' \neq s \Rightarrow$ миграция Теперь режим «не все задачи помещаются на МЕС» означает: $n_l > M_l \Rightarrow$ «часть остается в облачном сервере», на МЕС M_l. Следовательно МЕС-часть фиксируется как: $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_s}, M_l, l)$ и действие $(m', s') = (M_l, l) = (min\ (n_l, M_l), l)$ Интерпретации результата выбора: →если выбирается s - нет миграции; →если выбирается $s' \neq s$ - миграция; →если выбирается $l \neq s$: при $n_l \leq M_l$- все на МЕС; при $n_l > M_l$- часть в облачном сервере, на МЕС M_l.	Условие: $k \in \mathcal{K} \setminus \{s\}, n_k > 0$. Алгоритм: нет, т.к. состояние МЕС не изменяется Переход: $\boldsymbol{x}' = (\boldsymbol{n} - \boldsymbol{e_k}, m, s)$. Миграция: нет.

Лемма доказана. □

Политика управления представляет собой правило, задающее выбор действия при каждом возможном событии в каждом состоянии системы. Формально под политикой будем понимать отображение, которое для любой пары «состояние системы $\boldsymbol{x} = (\boldsymbol{n}, m, s)$ событие A_{ik}» однозначно выбирает одно допустимое действие (m', s') из множества $\mathcal{A}_{ik}(\boldsymbol{x})$. Иначе говоря, политика – это полный набор правил выбора действий (механизмов приоритета, перераспределения ресурса, определения сервиса на МЕС-узле) во всех состояниях системы.

Для поиска оптимальной политики используется подход минимизации межконцевой суммарной задержки пользователей.

$$(m^*(\boldsymbol{n},m,s,i,k),s^*(\boldsymbol{n},m,s,i,k)) \in arg \max_{(m',s')\in\mathcal{A}_{ik}(\boldsymbol{x})} m'(d_{s'}-d_0), \quad (3.8)$$

Минимизация суммарной задержки в следующем состоянии по всем допустимым действиям $(m',s')\in\mathcal{A}_{ik}(\boldsymbol{x})$ эквивалентна максимизации величины $m'(d_{s'}-d_0)$. Для модели с миграцией сервисов в гибридной граничной облачной инфраструктуре с пуассоновским входящим потоком, в которой суммарная межконцевая задержка в состоянии $(\boldsymbol{n},m,s)$ справедлива следующая теорема.

Теорема 3.1. Для модели с перемещением заявок между общей и индивидуальными группами приборов при наступлении события A_{ik} в состоянии $\boldsymbol{x}=(\boldsymbol{n},m,s)$ политика минимизации межконцевой суммарной задержки выглядит следующим образом

$$\big(m^*(\boldsymbol{n},m,s,i,k),s^*(\boldsymbol{n},m,s,i,k)\big)=$$

$$=\begin{cases}
(1,k), & i=1,s=0,\\
(m+1,k), & i=1,s\neq 0,k=s,n_k<N_k,m<M_k,m(d_k-d_0)\geq \max\limits_{l\in\mathcal{K}\setminus\{s\}}\min(n_l,M_l)(d_l-d_0),\\
(M_k,k), & i=1,s\neq 0,k=s,n_k<N_k,m=M_k,M_k(d_k-d_0)\geq \max\limits_{l\in\mathcal{K}\setminus\{s\}}\min(n_l,M_l)(d_l-d_0),\\
(n_l,l), & i=1,s\neq 0,k=s,n_k<N_k,0<n_l\leq M_l,min(n_l,M_l)\,(d_l-d_0)>m(d_k-d_0),\\
(M_l,l), & i=1,s\neq 0,k=s,n_k<N_k,n_l>M_l,min(n_l,M_l)\,(d_l-d_0)>m(d_k-d_0),\\
(m,s), & i=1,s\neq 0,k\neq s,n_k<N_k,m(d_s-d_0)\geq min(n_k+1,M_k)\,(d_k-d_0),\\
(n_k+1,k), & i=1,s\neq 0,k\neq s,n_k<M_k,m(d_s-d_0)<(n_k+1)(d_k-d_0),\\
(M_k,k), & i=1,s\neq 0,k\neq s,M_k\leq n_k<N_k,m(d_s-d_0)<M_k(d_k-d_0),\\
(m-1,s), & i=2,s\neq 0,k=s,m>0,n_s=m,(m-1)(d_s-d_0)\geq \max\limits_{l\in\mathcal{K}\setminus\{s\}}\min(n_l,M_l)(d_l-d_0),\\
(M_s,s), & i=2,s\neq 0,k=s,m=M_s,n_s>M_s,M_s(d_s-d_0)\geq \max\limits_{l\in\mathcal{K}\setminus\{s\}}\min(n_l,M_l)(d_l-d_0),\\
(n_l,l), & i=2,s\neq 0,k=s,m>0,0<n_l\leq M_l,min(n_l,M_l)\,(d_l-d_0)>(m-1)(d_s-d_0),\\
(M_l,l), & i=2,s\neq 0,k=s,m>0,n_l>M_l,min(n_l,M_l)\,(d_l-d_0)>(m-1)(d_s-d_0),\\
(m,s), & i=2,s\neq 0,k\neq s,n_k>0.
\end{cases} \quad (3.9)$$

Доказательство.

Для сравнения альтернативных состояний используем эквивалентность по суммарной задержке пользователей. В частности, для двух $\boldsymbol{x}=(\boldsymbol{n},m,s)$ и $\boldsymbol{x}'=(\boldsymbol{n},m',s')$ справедливо соотношение

$$d(\boldsymbol{n},m,s)\leq d(\boldsymbol{n},m',s') \;\Leftrightarrow\; m(d_s-d_0)\geq m'(d_{s'}-d_0). \quad (3.10)$$

Тем самым задача сравнения суммарных задержек сводится к сравнению величин, зависящих только от параметров сервисов и числа пользователей, обслуживаемых на МЕС-узле. Дополнительно используем эквивалентное соотношение, позволяющее

заменить задачу минимизации суммарной задержки задачей максимизации соответствующих выражений: если из состояния $(\boldsymbol{n}, m, s)$ возможен набор допустимых переходов в состояния $(\boldsymbol{n}, m_1, s_1), \dots, (\boldsymbol{n}, m_I, s_I)$, то

$$min\big(d(\boldsymbol{n}, m_1, s_1), d(\boldsymbol{n}, m_2, s_2), \dots, d(\boldsymbol{n}, m_I, s_I)\big) = \\ = max\left(m_1\big(d_{s_1} - d_0\big), m_2\big(d_{s_2} - d_0\big), \dots, m_I\big(d_{s_I} - d_0\big)\right). \quad (3.11)$$

Таким образом, выбор предпочтительного состояния из множества допустимых альтернатив при фиксированном событии может быть выполнен путем сравнения указанных величин.

В дальнейшем, для случаев, в которых из состояния $\boldsymbol{x} = (\boldsymbol{n}, m, s)$ возможно несколько допустимых переходов, вводится правило выбора состояния $\boldsymbol{x}'$ по минимальному значению функции суммарной задержки $d(\cdot)$. Это правило однозначно определяет функции $m^*(\boldsymbol{n}, m, s, i, k)$ и $s^*(\boldsymbol{n}, m, s, i, k)$, заданные в формулировке теоремы. Покажем, что для каждого сочетания (i, k) эти функции действительно совпадают с формулами теоремы.

Рассмотрим произвольное состояние $\boldsymbol{x} = (\boldsymbol{n}, m, s) \in \mathcal{X}$ и событие A_{ik}, $i = 1,2$, $k \in \mathcal{K}$. В соответствии с определением про события A_{ik} и таблицами 3.1–3.4, множество допустимых действий $\mathcal{A}_{ik}(\boldsymbol{x})$ состоит из пар (m, s), описывающих конфигурацию МЕС-узла после события A_{ik}. Пусть $\boldsymbol{x}' = (\boldsymbol{n}, m', s')$ есть состояние после события A_{ik}, где $\boldsymbol{n}'$ отличается от $\boldsymbol{n}$ только по компоненте класса k, в т.ч. по типу события i. Тогда по определению

$$(m^*(\boldsymbol{n}, m, s, i, k), s^*(\boldsymbol{n}, m, s, i, k)) = arg \min_{(m', s') \in \mathcal{A}_{ik}(\boldsymbol{x})} d(\boldsymbol{n}', m', s'), \quad (3.12)$$

и, в силу приведенной выше эквивалентности (3.12), имеем

$$\big(m^*(\boldsymbol{n}, m, s, i, k), s^*(\boldsymbol{n}, m, s, i, k)\big) = arg \max_{(m', s') \in \mathcal{A}_{ik}(\boldsymbol{x})} m'(d_{s'} - d_0). \quad (3.13)$$

Начнем со случая поступления заявки при пустом МЕС-узле, т.е. $i = 1$ и $s = 0$. Если $s = 0$, то по определению множества $\mathcal{A}_{1k}(\boldsymbol{x})$ и таблице 3.1 единственное допустимое действие поступление заявки на МЕС-узел и назначение его на сервис k, тогда $\mathcal{A}_{1k}(\boldsymbol{x}) = \{(1, k)\}$. Следовательно,

$$\big(m^*(\boldsymbol{n}, 0,0,1, k), s^*(\boldsymbol{n}, 0,0,1, k)\big) = (1, k). \quad (3.14)$$

Случай поступления заявки сервиса s при активном МЕС-узле, т.е. $i = 1$ и $k = s$. Пусть $s \in \mathcal{K}$ и поступает новая заявка сервиса s. При $n_s < N_s$ и $m < M_s$ по таблице 3.2 множество $\mathcal{A}_{1s}(\mathbf{x})$ включает два действия: действие без миграции: $(m', s') = (m + 1, s)$ и действия с миграцией на сервис $l \in \mathcal{K} \setminus \{s\}$, такое что $(m', s') = (min\ (n_l, M_l), l)$.

Введем вспомогательные функции

$$\tilde{d}_k(\boldsymbol{n}, m, s) = \begin{cases} d(\boldsymbol{n} + \boldsymbol{e}_s, m+1, s), & k = s, \\ d(\boldsymbol{n} + \boldsymbol{e}_s, n_k, k), & k \neq s,\ 0 < n_k \leq M_k, \\ d(\boldsymbol{n} + \boldsymbol{e}_s, M_k, k), & k \neq s,\ n_k > M_k. \end{cases} \tag{3.15}$$

и задачу

$$\boldsymbol{x}' = arg \min_{k \in \mathcal{K}} \tilde{d}_k(\boldsymbol{x}). \tag{3.16}$$

Используя эквивалентность между $d(\cdot)$ и $m(d_s - d_0)$, перепишем ее в виде

$$\boldsymbol{x}' = arg\ max\ ((m+1)(d_s - d_0), \max_{k \neq s}\ min\ (n_k, M_k)(d_k - d_0)) = \boldsymbol{y}_1(\boldsymbol{n}, m, s), \tag{3.17}$$

где $\mathbf{y}_1(\mathbf{n}, \mathrm{m}, \mathrm{s})$ определена ранее. Отсюда

$$\boldsymbol{y}_1(\boldsymbol{n}, m, s) = \\ = \begin{cases} (\boldsymbol{n} + \boldsymbol{e}_s, m+1, s),\ если\ (m+1)(d_s - d_0) \geq \max\limits_{l \in \mathcal{K} \setminus \{s\}} \min\ (n_l, M_l)(d_l - d_0), \\ (\boldsymbol{n} + \boldsymbol{e}_s, min(n_l, M_l), l),\ если\ (м+1)(d_s - d_0) < \max\limits_{l \in \mathcal{K} \setminus \{s\}} \min\ (n_l, M_l)(d_l - d_0). \end{cases} \tag{3.18}$$

В результате правило выбора перехода принимает вид:

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \xrightarrow{\lambda_s} (\boldsymbol{n} + \boldsymbol{e}_s, m+1, s) = \boldsymbol{x}', \qquad \boldsymbol{y}_1(\boldsymbol{n}, m, s) = (\boldsymbol{n} + \boldsymbol{e}_s, m+1, s),$$

при $\mathrm{n_s} < \mathrm{N_s}$, $\mathrm{m} < \mathrm{M_s}$, $\mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \xrightarrow{\lambda_s} (\boldsymbol{n} + \boldsymbol{e}_s, n_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_1(\boldsymbol{n}, m, s) = (\boldsymbol{n} + \boldsymbol{e}_s, n_l, l),$$

при $\mathrm{n_s} < \mathrm{N_s}$, $0 < \mathrm{n_l} \leq \mathrm{M_l}$, $\mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}, \mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \xrightarrow{\lambda_s} (\boldsymbol{n} + \boldsymbol{e}_s, M_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_1(\boldsymbol{n}, m, s) = (\boldsymbol{n} + \boldsymbol{e}_s, M_l, l),$$

при $n_s < N_s$, $n_l > M_l$, $l \in \mathcal{K} \setminus \{s\}$, $s \in \mathcal{K}$.

Следовательно,

$$\left(m^*(\boldsymbol{n}, m, s, 1, s), s^*(\boldsymbol{n}, m, s, 1, s)\right) = \\ = \begin{cases} (m+1, s), & (m+1)(d_s - d_0) \geq \max\limits_{l \in \mathcal{K} \setminus \{s\}} \min\ (n_l, M_l)(d_l - d_0), \\ (min\ (n_l, M_l), l)), & (m+1)(d_s - d_0) < \max\limits_{l \in \mathcal{K} \setminus \{s\}} \min\ (n_l, M_l)(d_l - d_0), \end{cases} \tag{3.19}$$

что совпадает с соответствующей частью формул теоремы для $i = 1, k = s, m < M_s$.

Если $n_s < N_s$, $m = M_s$, то поступающая заявка сервиса s направляется в облачный сервер, а множеству $\mathcal{A}_{1s}(\boldsymbol{x})$ соответствуют два действия: действие без миграции: $(m', s') = (M_s, s)$ и действия с миграцией на сервис $l \neq s$: $(m', s') = (\min\ (n_l, M_l), l)$. Определим

$$\tilde{d}_k(\boldsymbol{n}, m, s) = \begin{cases} d(\boldsymbol{n} + \boldsymbol{e}_s, M_s, s), & k = s, \\ d(\boldsymbol{n} + \boldsymbol{e}_s, n_k, k), & k \neq s,\ 0 < n_k \leq M_k, \\ d(\boldsymbol{n} + \boldsymbol{e}_s, M_k, k), & k \neq s,\ n_k > M_k. \end{cases} \tag{3.20}$$

Тогда

$$\boldsymbol{x}' = arg\ max\ (M_s(d_s - d_0), \max_{k \neq s}\ min\ (n_k, M_k)(d_k - d_0)) = \boldsymbol{y}_2(\boldsymbol{n}, M_s, s). \tag{3.21}$$

По определению $\mathbf{y}_2(\mathbf{n}, M_s, s)$ имеем

$$\boldsymbol{y}_2(\boldsymbol{n}, M_s, s) = \\ = \begin{cases} (\boldsymbol{n} + e_s, M_s, s), & если\ M_s(d_s - d_0) \geq \max\limits_{l \in \mathcal{K} \setminus \{s\}} min\ (n_l, M_l)(d_l - d_0), \\ (\boldsymbol{n} + e_s, min\ (n_{l^*}, M_{l^*}), l^*), & если\ M_s(d_s - d_0) < \max\limits_{l \in \mathcal{K} \setminus \{s\}} min\ (n_l, M_l)(d_l - d_0), \end{cases} \tag{3.22}$$

Следовательно, правило выбора перехода в рассматриваемом случае имеет вид:

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{\lambda_s}{\to} (\boldsymbol{n} + \boldsymbol{e}_s, M_s, s) = \boldsymbol{x}', \qquad \boldsymbol{y}_2(\boldsymbol{n}, m, s) = (\boldsymbol{n} + \boldsymbol{e}_s, M_s, s),$$

при $\mathrm{n_s} < \mathrm{N_s}$, $\mathrm{m} = \mathrm{M_s}$, $\mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{\lambda_s}{\to} (\boldsymbol{n} + \boldsymbol{e}_s, n_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_2(\boldsymbol{n}, m, s) = (\boldsymbol{n} + \boldsymbol{e}_s, n_l, l),$$

при $n_s < N_s$, $0 < n_l \le M_l$, $l \in \mathcal{K} \setminus \{s\}$, $s \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{\lambda_s}{\to} (\boldsymbol{n} + \boldsymbol{e}_s, M_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_2(\boldsymbol{n}, m, s) = (\boldsymbol{n} + \boldsymbol{e}_s, M_l, l),$$

при $\mathrm{n_s} < \mathrm{N_s}$, $\mathrm{n_l} > \mathrm{M_l}$, $\mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}$, $\mathrm{s} \in \mathcal{K}$.

Тогда

$$\begin{gathered}\big(m^*(\boldsymbol{n}, M_s, s, 1, s), s^*(\boldsymbol{n}, M_s, s, 1, s)\big) = \\ = \begin{cases} (M_s, s), & M_s(d_s - d_0) \ge \max\limits_{l \in \mathcal{K}\setminus\{s\}} \min\,(n_l, M_l)(d_l - d_0), \\ (min(n_{l^*}, M_{l^*}), l^*), & M_s(d_s - d_0) < \max\limits_{l \in \mathcal{K}\setminus\{s\}} \min\,(n_l, M_l)(d_l - d_0), \end{cases}\end{gathered} \tag{3.23}$$

что совпадает с частью формул теоремы для $i = 1, k = s, m = M_s$.

Рассмотрим случай поступления заявки сервиса $k \neq s$ ($i = 1$). По таблицам 3.2 имеются две альтернативы: заявка поступает в соответствующий облачный сервер, MEC-узел остается на сервисе s: $(m', s') = (m, s)$ и MEC-узел переключается на сервис k:

$$(m', s') = \begin{cases} (n_k + 1, k), & n_k < M_k, \\ (M_k, k), & M_k \le n_k < N_k. \end{cases} \tag{3.24}$$

Сравнивая задержки по приведенной выше эквивалентности, получаем

$$\begin{gathered} d(\boldsymbol{n} + e_k, m, s) \le d(\boldsymbol{n} + e_k, n_k + 1, k) \iff m(d_s - d_0) \ge (n_k + 1)(d_k - d_0), \\ d(\boldsymbol{n} + e_k, m, s) \le d(\boldsymbol{n} + e_k, M_k, k) \iff m(d_s - d_0) \ge M_k(d_k - d_0). \end{gathered} \tag{3.25}$$

Отсюда следует, что

(а) если $n_k < M_k$ и $m(d_s - d_0) \ge (n_k + 1)(d_k - d_0)$, оптимально $(m^*, s^*) = (m, s)$, иначе $(m^*, s^*) = (n_k + 1, k)$,

(б) если $M_k \le n_k < N_k$ и $m(d_s - d_0) \ge M_k(d_k - d_0)$, оптимально $(m^*, s^*) = (m, s)$, иначе $(m^*, s^*) = (M_k, k)$.

Эти выражения полностью совпадают с формулами теоремы для $i = 1, k \neq s$.

Случай завершения обслуживания на MEC-узле, $k = s$ ($i = 2$). Пусть $n_s = m > 0$. Тогда по таблицам 3.3 возможны тоже два действия: действие без миграции: $(m', s') = (m - 1, s)$ и действия с миграцией на сервис $l \in \mathcal{K} \setminus \{s\}$: $(m', s') = (\min\,(n_l, M_l), l)$.

Введем

$$\tilde{d}_k(\boldsymbol{n}, m, s) = \begin{cases} d(\boldsymbol{n} - \boldsymbol{e}_s, m - 1, s), & k = s, \\ d(\boldsymbol{n} - \boldsymbol{e}_s, n_k, k), & k \neq s,\ 0 < n_k \le M_k, \\ d(\boldsymbol{n} - \boldsymbol{e}_s, M_k, k), & k \neq s,\ n_k > M_k. \end{cases} \tag{3.26}$$

и рассмотрим

$$\boldsymbol{x}' = arg\ max\ ((m - 1)(d_s - d_0), \max_{k \in \mathcal{K}\setminus\{s\}} min\ (n_k, M_k)(d_k - d_0)) = \boldsymbol{y}_3(\boldsymbol{n}, m, s). \tag{3.27}$$

По определению $\boldsymbol{y}_3(\boldsymbol{n}, m, s)$

$$\boldsymbol{y}_3(\boldsymbol{n}, m, s) =$$
$$= \begin{cases} (\boldsymbol{n} - e_s, m-1, s), \text{ если } (m-1)(d_s - d_0) \geq \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \\ (\boldsymbol{n} - e_s, \min(n_l, M_l), l), \text{ если } (m-1)(d_s - d_0) < \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \end{cases} \quad (3.28)$$

при $\mathrm{n_s} = \mathrm{m}, \mathrm{m} > 0, \mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{m\mu_s}{\rightarrow} (\boldsymbol{n} - \boldsymbol{e}_s, n_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_3(\boldsymbol{n}, m, s) = (\boldsymbol{n} - \boldsymbol{e}_s, n_l, l),$$

при $\mathrm{n_s} = \mathrm{m}, \mathrm{m} > 0, 0 < \mathrm{n_l} \leq \mathrm{M_l}, \mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}, \mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{m\mu_s}{\rightarrow} (\boldsymbol{n} - \boldsymbol{e}_s, M_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_3(\boldsymbol{n}, m, s) = (\boldsymbol{n} - \boldsymbol{e}_s, M_l, l),$$

при $\mathrm{n_s} = \mathrm{m}, \mathrm{m} > 0, \mathrm{n_l} > \mathrm{M_l}, \mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}, \mathrm{s} \in \mathcal{K}$.

Тогда

$$\left(m^*(\boldsymbol{n}, m, s, 2, s), s^*(\boldsymbol{n}, m, s, 2, s)\right) =$$
$$= \begin{cases} (m-1, s), & (m-1)(d_s - d_0) \geq \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \\ (\min(n_{l^*}, M_{l^*}), l^*), & (m-1)(d_s - d_0) < \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \end{cases} \quad (3.29)$$

Если $m = M_s$ и $n_s > M_s$, то завершение на МЕС не изменяет m, и по таблицам 3.3 возможны следующие действия: действие без миграции: $(m', s') = (M_s, s)$ и действия с миграцией на сервис $l \neq s$: $(m', s') = (\min(n_l, M_l), l)$.

Пусть

$$\tilde{d}_k(\boldsymbol{n}, M_s, s) = \begin{cases} d(\boldsymbol{n} - \boldsymbol{e}_s, M_s, s), & k = s, \\ d(\boldsymbol{n} - \boldsymbol{e}_s, n_k, k), & k \neq s,\ 0 < n_k \leq M_k, \\ d(\boldsymbol{n} - \boldsymbol{e}_s, M_k, k), & k \neq s,\ n_k > M_k. \end{cases} \quad (3.30)$$

тогда

$$\boldsymbol{x}' = arg\ \max\left(M_s(d_s - d_0), \max_{k \neq s} \min(n_k, M_k)(d_k - d_0)\right) = \boldsymbol{y}_4(\boldsymbol{n}, M_s, s). \quad (3.31)$$

По определению $\boldsymbol{y}_4$

$$\boldsymbol{y}_4(\boldsymbol{n}, M_s, s) =$$
$$= \begin{cases} (\boldsymbol{n} - e_s, M_s, s), & \textit{если } M_s(d_s - d_0) \geq \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \\ (\boldsymbol{n} - e_s, \min(n_{l^*}, M_{l^*}), l^*), & \textit{если } M_s(d_s - d_0) < \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \end{cases} \quad (3.32)$$

Следовательно, правило выбора перехода в рассматриваемом случае имеет вид:

$$\boldsymbol{x} = (\boldsymbol{n}, M_s, s) \overset{M_s\mu_s}{\rightarrow} (\boldsymbol{n} - \boldsymbol{e}_s, M_s, s) = \boldsymbol{x}', \qquad \boldsymbol{y}_4(\boldsymbol{n}, M_s, s) = (\boldsymbol{n} - \boldsymbol{e}_s, M_s, s),$$

при $\mathrm{m} = \mathrm{M_s}, \mathrm{n_s} > \mathrm{M_s}, \mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{m\mu_s}{\rightarrow} (\boldsymbol{n} - \boldsymbol{e}_s, n_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_4(\boldsymbol{n}, M_s, s) = (\boldsymbol{n} - \boldsymbol{e}_s, n_l, l),$$

при $\mathrm{m} = \mathrm{M_s}, \mathrm{n_s} > \mathrm{M_s}, 0 < \mathrm{n_l} \leq \mathrm{M_l}, \mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}, \mathrm{s} \in \mathcal{K}$;

$$\boldsymbol{x} = (\boldsymbol{n}, m, s) \overset{m\mu_s}{\rightarrow} (\boldsymbol{n} - \boldsymbol{e}_s, M_l, l) = \boldsymbol{x}', \qquad \boldsymbol{y}_4(\boldsymbol{n}, M_s, s) = (\boldsymbol{n} - \boldsymbol{e}_s, M_l, l),$$

при $\mathrm{m} = \mathrm{M_s}, \mathrm{n_s} > \mathrm{M_s}, \mathrm{n_l} > \mathrm{M_l}, \mathrm{l} \in \mathcal{K} \setminus \{\mathrm{s}\}, \mathrm{s} \in \mathcal{K}$,

и, значит,

$$\left(m^*(\boldsymbol{n}, M_s, s, 2, s), s^*(\boldsymbol{n}, M_s, s, 2, s)\right) = \\ = \begin{cases} (M_s, s), & M_s(d_s - d_0) \geq \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \\ (\min(n_{l^*}, M_{l^*}), l^*), & M_s(d_s - d_0) < \max\limits_{l \neq s} \min(n_l, M_l)(d_l - d_0), \end{cases} \quad (3.33)$$

что соответствует формулировке теоремы для $i = 2, k = s, m = M_s$.

Случай завершения в облачном сервере, $k \neq s$ $(i = 2)$. Для событий $A_{2k}, k \neq s$, по таблице 3.4 множество допустимых действий $\mathcal{A}_{2k}(\boldsymbol{x})$ содержит единственный элемент (m, s), то есть конфигурация МЕС-узла не изменяется. Поэтому

$$(m^*(\boldsymbol{n}, m, s, 2, k), s^*(\boldsymbol{n}, m, s, 2, k)) = (m, s), k \neq s, \quad (3.34)$$

что совпадает с последней строкой формул теоремы.

Таким образом, во всех возможных случаях (i, k) оптимальное действие $(m^*(\boldsymbol{n}, m, s, i, k), s^*(\boldsymbol{n}, m, s, i, k))$, минимизирующее суммарную задержку пользователей, задается формулами теоремы через функции $\boldsymbol{y}_1(\boldsymbol{n}, m, s)$, $\boldsymbol{y}_2(\boldsymbol{n}, m, s)$, $\boldsymbol{y}_3(\boldsymbol{n}, m, s)$, $\boldsymbol{y}_4(\boldsymbol{n}, m, s)$ и сводку условий миграции.

Таблица 3.5 - Сравнение фаз нагрузки и допустимых состояний

Тип решения МЕС-узла	Условие	Результат для (m', s')
Оставаться на сервисе s	$m'(d_s - d_0) \geq \max\limits_{l \in \mathcal{K} \setminus \{s\}} \min(n_l, M_l)(d_l - d_0)$	$(m', s') = (m, s)$
Перейти на сервис l (полный перенос)	$n_l \leq M_l, n_l(d_l - d_0) > m'(d_s - d_0)$	$(m', s') = (n_l, l)$
Перейти на сервис l (частичный перенос)	$n_l > M_l, M_l(d_l - d_0) > m'(d_s - d_0)$	$(m', s') = (M_l, l)$

Здесь m' число заявок на МЕС-узле после события и выбранного действия, s текущий сервис МЕС-узла до миграции, l кандидатный сервис для миграции.

Теорема доказана. □

Полученная теорема задает оптимальную стратегию управления конфигурацией МЕС-узла: для каждого состояния $(\boldsymbol{n}, m, s)$ и каждого события A_{ik} функции $m^*(\boldsymbol{n}, m, s, i, k)$, $s^*(\boldsymbol{n}, m, s, i, k)$ однозначно определяют, будет ли выполнена миграция и на какой сервис она произойдет. На основе этих правил можно записать матрицу интенсивностей $\boldsymbol{Q}$, соответствующую работе системы при оптимальной политике.

Следствие 3.1. Для модели с перемещением заявок между общей и индивидуальными группами приборов при оптимальной стратегии миграции ненулевые внедиагональные элементы матрицы $\boldsymbol{Q}$.

$$\begin{aligned} &\boldsymbol{Q}[(\boldsymbol{0}, 0, 0), (\boldsymbol{e}_k, 1, k)] = \lambda_k, k \in \mathcal{K}, \\ &\boldsymbol{Q}[(\boldsymbol{n}, m, s), (\boldsymbol{n} + \boldsymbol{e}_s, m + 1, s)] = \lambda_s, \\ &\qquad n_s < N_s, m < M_s, (m+1)(d_s - d_0) \geq \max_{l \in \mathcal{K} \setminus \{s\}} \min(n_l, M_l)(d_l - d_0), \\ &\boldsymbol{Q}[(\boldsymbol{n}, m, s), (\boldsymbol{n} + \boldsymbol{e}_s, M_s, s)] = \lambda_s, \\ &\qquad n_s < N_s, m = M_s, M_s(d_s - d_0) \geq \max_{l \in \mathcal{K} \setminus \{s\}} \min(n_l, M_l)(d_l - d_0), \\ &\boldsymbol{Q}[(\boldsymbol{n}, m, s), (\boldsymbol{n} + \boldsymbol{e}_s, n_l, l)] = \lambda_s, \\ &\qquad n_s < N_s, 0 < n_l \leq M_l, n_l(d_l - d_0) > m(d_s - d_0), l \in \mathcal{K} \setminus \{s\}, \\ &\boldsymbol{Q}[(\boldsymbol{n}, m, s), (\boldsymbol{n} + \boldsymbol{e}_s, M_l, l)] = \lambda_s, \\ &\qquad n_s < N_s, n_l > M_l, M_l(d_l - d_0) > m(d_s - d_0), l \in \mathcal{K} \setminus \{s\}, \end{aligned} \quad (3.34)$$

$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}+\boldsymbol{e}_k,m,s)]=\lambda_k,$$
$$n_k<N_k, m(d_s-d_0)\geq(n_k+1)(d_k-d_0), k\in\mathcal{K}\setminus\{s\},$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}+\boldsymbol{e}_k,n_k+1,k)]=\lambda_k,$$
$$n_k<M_k, (n_k+1)(d_k-d_0)>m(d_s-d_0), k\in\mathcal{K}\setminus\{s\},$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}+\boldsymbol{e}_k,M_k,k)]=\lambda_k,$$
$$M_k\leq n_k<N_k, M_k(d_k-d_0)>m(d_s-d_0), k\in\mathcal{K}\setminus\{s\}.$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}-\boldsymbol{e}_s,m-1,s)]=m\mu_s,$$
$$n_s=m, m>0, (m-1)(d_s-d_0)\geq\max_{l\in\mathcal{K}\setminus\{s\}}\min(n_l,M_l)(d_l-d_0),$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}-\boldsymbol{e}_s,M_s,s)]=M_s\mu_s,$$
$$n_s>M_s, M_s(d_s-d_0)\geq\max_{l\in\mathcal{K}\setminus\{s\}}\min(n_l,M_l)(d_l-d_0),$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}-\boldsymbol{e}_s,n_l,l)]=m\mu_s,$$
$$m>0, 0<n_l\leq M_l,\ n_l(d_l-d_0)>(m-1)(d_s-d_0),$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}-\boldsymbol{e}_s,M_l,l)]=m\mu_s,$$
$$m>0, n_l>M_l, M_l(d_l-d_0)>(m-1)(d_s-d_0),$$
$$\boldsymbol{Q}[(\boldsymbol{n},m,s),(\boldsymbol{n}-\boldsymbol{e}_k,m,s)]=n_k\mu_k, n_k>0, k\in\mathcal{K}.$$

3.3 СТАЦИОНАРНОЕ РАСПРЕДЕЛЕНИЕ В МУЛЬТИПЛИКАТИВНОМ ВИДЕ И ЧИСЛЕННЫЙ АНАЛИЗ

В рамках модели, введенной в разделе 3.1, динамика системы описывается марковским процессом $\boldsymbol{X}(t), t\geq 0$, где состояние системы в момент времени t задается расширенным вектором $\boldsymbol{X}(t)=\big(\boldsymbol{N}(t),M(t),S(t)\big)$. При фиксированном правиле выбора (m,s) случайный процесс $\mathbf{N}(\mathrm{t})=\big(N_1(t),\dots,N_K(t)\big), t\geq 0$, является также марковским, причем как для мультисервисной модели Эрланга с явными потерями. В этом случае параметры (m,s) однозначно определяются текущим вектором $\boldsymbol{N}(t)$ и не рассматриваются как независимые компоненты состояния. Таким образом, расширенное описание состояния системы может быть сужено до описания, основанного только на векторе $\boldsymbol{N}(t)$, при сохранении структуры переходов и их интенсивностей. Тогда пространство состояний суженного процесса имеет вид

$$\mathcal{N}=\{\boldsymbol{n}=(n_1,\dots,n_K):0\leq n_k\leq N_k, k\in\mathcal{K}\}. \qquad (3.35)$$

При этом

$$\mathcal{N}=\bigcup_{k\in\mathcal{K}}\mathcal{N}_k\cup\{\mathbf{0}\},\qquad \mathcal{N}_k=\{\boldsymbol{n}\in\mathcal{N}:s(\boldsymbol{n})=k\}, k\in\mathcal{K}.$$

Далее можно сформулировать следующее утверждение.

Утверждение 3.1. Для модели с перемещением заявок между общей и индивидуальными группами приборов при фиксированной политике стационарное распределение марковского процесса $\boldsymbol{N}(t)$ имеет мультипликативный вид

$$p(\boldsymbol{n}) = \frac{1}{G}\prod_{k=1}^{K}\left(\frac{\lambda_k}{\mu_k}\right)^{n_k}\frac{1}{n_k!}, \tag{3.36}$$

$$= \frac{1}{p(\mathbf{0})} = \left(\sum_{\boldsymbol{n}\in\mathcal{N}}\prod_{k=1}^{K}\frac{\rho_k^{n_k}}{n_k!}\right) \tag{3.37}$$

Зная стационарное распределение можем вывести следующие характеристики распределения нагрузки между MEC-узлом и облачными серверами. Вероятность того, что активным является сервис s

$$p_k = \sum_{\boldsymbol{n}\in\mathcal{N}_k} p(\boldsymbol{n}), k \in \mathcal{K} \tag{3.38}$$

Среднее число заявок сервиса s на MEC (при условии, что s активен)

$$\bar{m}_k = \frac{1}{p_k}\sum_{\boldsymbol{n}\in\mathcal{N}_k} min(n_k, M_k)\, p(\boldsymbol{n}), k \in \mathcal{K} \tag{3.39}$$

Средняя суммарная задержка

$$\sum_{\boldsymbol{n}\in\mathcal{N}_k} p(\boldsymbol{n})\left(\sum_{k\in\mathcal{K}} n_k\, d_k \;+\; m\,(d_0 - d_s)\right). \tag{3.40}$$

Средняя суммарная задержка для сервиса k и для любого сервиса

$$\begin{gathered}\tilde{d}_k = \bar{m}_k(d_k - d_0),\ k \in \mathcal{K} \\ \tilde{d} \;\; = \sum_{k\in\mathcal{K}} \tilde{d}_k\, p_k\end{gathered} \tag{3.41}$$

Вероятность того, что MEC узел полностью занят

$$p_{MEC} = \sum_{\boldsymbol{n}\in\mathcal{N}_{MEC}} p(\boldsymbol{n}),\ \ \mathcal{N}_{MEC} = \{\boldsymbol{n} \in \mathcal{N} : m(\boldsymbol{n}) = M_s(\boldsymbol{n})\} \tag{3.42}$$

Далее проводится численный анализ предложенной модели системы для трех 6G-сервисов: дополненная и виртуальная реальность (Augmented reality, AR, Virtual reality, VR, AR/VR), облачный гейминг и кооперативные автомобильные сервисы (Vehicle-to-Everything, V2X). AR/VR-трафик моделирует иммерсивные приложения расширенной реальности, требующие очень низкой задержки и высокой пропускной способности канала. Облачный гейминг представляет численно доминирующий интерактивный мультимедийный трафик со средними требованиями к задержке, но высоким битрейтом на пользователя. V2X-сервисы описывают обмен сенсорными данными и кооперативное восприятие между транспортными средствами и

инфраструктурой, где критичны надежность и устойчивость задержки, а битрейт умеренный. [144]. Параметры сценария предсталены в Таблице 3.6.

Таблица 3.6 - Сравнение фаз нагрузки и допустимых состояний

Параметр	k=1 AR/VR	k=2 Облачный гейминг	k=3 V2X
Интенсивность поступления λ_k, заявок/с	50	80	30
Скорость обслуживания на МЕС μ_k, 1/с	$1/0{,}008 \approx 125$	$1/0{,}012 \approx 83{,}3$	$1/0{,}006 \approx 166{,}7$
Среднее время обслуживания на МЕС, с	0.008	0.012	0.006
Задержка в облачном сервере d_k(ближний ЦОД), с	0.030	0.040	0.035
Максимальное число пользователей на МЕС M_k	5	10	15
Максимальное число пользователей в облачном сервере N_k	15	20	25

Для моделирования качества облачной инфраструктуры рассматриваются три сценария, различающиеся по задержкам обработки. В базовом сценарии «ближний облачный дата-центр» используются значения задержек d_k, приведенные в таблице, и предполагается, что облачные серверы географически близки к МЕС-узлу. В сценарии «региональный облачный дата-центр» задержка обработки в облачном сервере для каждого сервиса увеличивается примерно на 30% по сравнению с базовым вариантом, то есть для сервиса k используется величина $d_k^{\text{per}} = 1{,}3\, d_k$. В сценарии «магистральный облачный дата-центр» задержки возрастают примерно на 70% относительно базового сценария, $d_k^{\text{маг}} = 1{,}7\, d_k$, что соответствует размещению облачного сервера на существенно большем расстоянии и учету дополнительной транспортной задержки [145]. Результаты численного анализа представлены на рисунках 3.2-3.4.

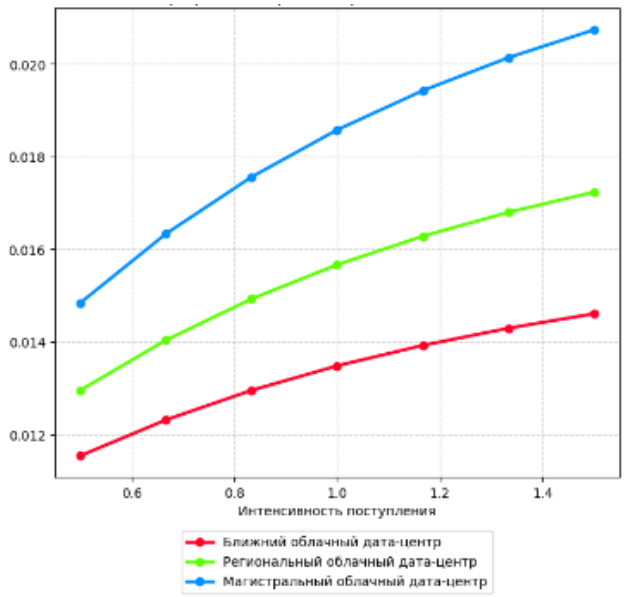

Рисунок 3.2. - Средняя задержка от интенсивности при разных вариантах размещения облачных серверов

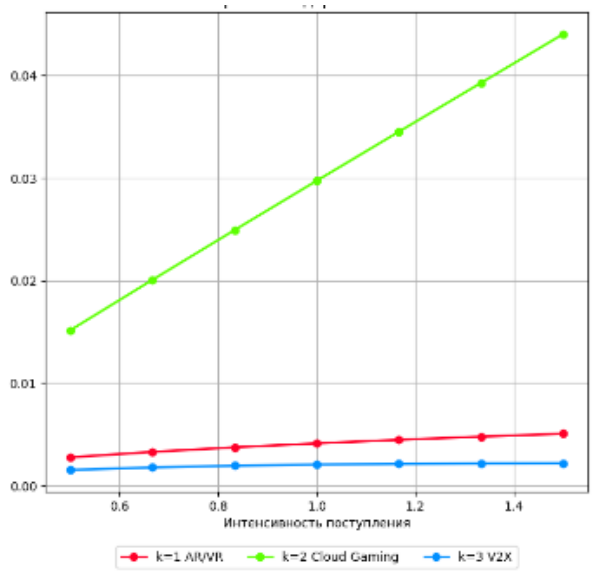

Рисунок 3.3. - Средняя суммарная задержка (ближний дата-центр)

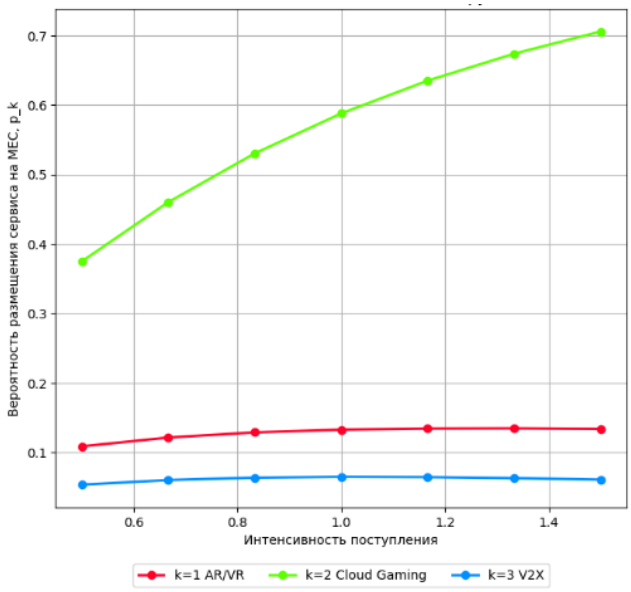

Рисунок 3.4. - Вероятность размещения сервисов на MEC

В качестве базового для численного анализа выбран сценарий «ближнего облачного дата-центра», поскольку он наилучшим образом отражает современную практику развертывания региональных облачных площадок в непосредственной близости к крупным узлам трафика. Такой подход задает минимальный реалистичный уровень задержки.

3.4 Система массового обслуживания с перемещением заявок и коррелированным потоком

Рассматриваемая в данном разделе модель представляет собой развитие общей постановки, введенной в начале главы 3, и предназначена для анализа влияния нестационарной нагрузки и фазо-зависимого управления задачами на поведение гибридной системы граничных облачных вычислений. В отличие от базовой модели, в данной постановке основное внимание уделяется структурной динамике распределения задач между уровнями вычислений при изменяющейся интенсивности входного потока. Связь с моделью, представленной в начале главы 3, заключается в сохранении архитектуры системы и принципов обслуживания, однако задержки в явном виде не вводятся. Предполагается единый тип сервиса с одинаковой интенсивностью обслуживания как на узле MEC, так и на облачном сервере. Миграция также инициируется при поступлении запроса пользователей на предоставление сервиса и завершении обслуживания пользователя.

Для описания входного трафика используется MMPP-поток с двумя фазами нагрузки. Фазовое состояние представлено как непрерывная марковская цепь с интенсивностями переходов α и β, а интенсивность поступления задач в фазе l равна λ_l, причем $\lambda_1 > \lambda_0$. Использование MMPP позволяет учитывать коррелированную нестационарность входной нагрузки и связать изменение интенсивности поступления задач с фазовым состоянием системы. При этом рассматриваемый MMPP-поток является частным случаем MAP-процесса, параметры которого построенными в разделе 1.4. Архитектура системы включает узел MEC с ограниченной вычислительной емкостью M и удаленный облачный вычислительный центр (рис.3.5). Общее число задач в системе ограничено величиной N. Все задачи относятся к одному классу сервиса и обслуживаются с одинаковой интенсивностью μ, независимо от места выполнения.

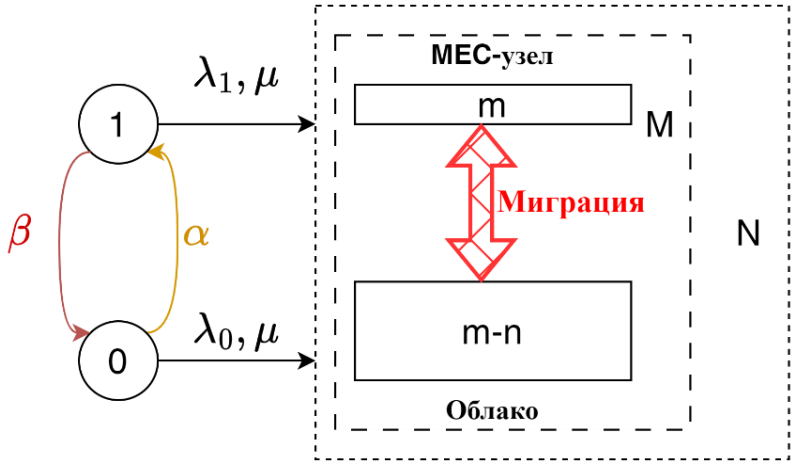

Рисунок 3.5 – Система модели с перемещением заявок и коррелированным потоком

Состояние системы в момент времени t задается вектором $\boldsymbol{X}(t) = \big(n(t), m(t), l(t)\big), t \geq 0$, где $n(t)$ общее число задач в системе, $m(t)$ число задач, размещенных на узле MEC, а $l(t)$ фазовое состояние нагрузки. Число задач, размещенных в облачном сервере, равно $n(t) - m(t)$ и однозначно определяется первыми двумя компонентами состояния. В таком случае состояние гибридной граничной облачной системы в произвольный момент времени описывается тройкой $\boldsymbol{x} = (n, m, l)$.

Ключевой особенностью модели является фазо-зависимая структура пространства состояний. В фазе низкой нагрузки $(l = 0)$ при фиксированном n допустимы все состояния, удовлетворяющие условию $0 \leq m \leq min(n, M)$. В фазе высокой нагрузки $(l = 1)$ вследствие принятой политики использования узла MEC допустимы только состояния

вида $m = min(n, M)$. Таким образом, полное пространство состояний имеет составную структуру и представляется в виде объединения

$$\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1, \tag{3.43}$$

где

$$\begin{gathered}\mathcal{X}_0 = \{(n, m, 0): 0 \le n \le N,\ 0 \le m \le min(n, M)\}, \\ \mathcal{X}_1 = \{(n, m, 1): 0 \le n \le N,\ m = min(n, M)\}.\end{gathered} \tag{3.44}$$

При поступлении новая задача направляется на узел МЕС, если $m < M$, и в облачный сервер в противном случае. Указанное правило не зависит от фазового состояния нагрузки.

Политика миграции является и фазо-зависимой. При переходе системы из фазы низкой нагрузки в фазу высокой нагрузки осуществляется миграция задач из облачного сервера на узел МЕС, в результате чего число задач на МЕС мгновенно изменяется в соответствии с правилом $m \to min(n, M)$. Обратный переход $l = 1 \to l = 0$ миграцией не сопровождается, и конфигурация системы сохраняется.

Правила завершения обслуживания зависят от фазового состояния нагрузки. В фазе низкой нагрузки завершение обслуживания на узле МЕС приводит к переходу $(n, m, 0) \to (n - 1, m - 1,0)$, а завершение обслуживания в облачном сервере к переходу $(n, m, 0) \to (n - 1, m, 0)$.

В фазе высокой нагрузки при завершении обслуживания на узле МЕС возможны два случая. Если в облачном сервере имеются ожидающие задачи $(n > m)$, осуществляется немедленная подстановка задачи из облачного сервера, и переход имеет вид $(n, m, 1) \to (n - 1, m, 1)$. Если же все задачи находятся на узле МЕС $(n = m)$, завершение обслуживания приводит к переходу $(n, m, 1) \to (n - 1, m - 1,1)$. Завершение обслуживания в облачном сервере в фазе высокой нагрузки описывается переходом $(n, m, 1) \to (n - 1, m, 1)$.

Таблица 3.7 - Сравнение фаз нагрузки и допустимых состояний

Характеристика	Фаза $l = 0$	Фаза $l = 1$
Интенсивность поступления	λ_0	$\lambda_1, \lambda_1 > \lambda_0$
Допустимые m при фикс. n	$0 \leqslant m \leqslant \min(n, M)$	$m = \min(n, M)$
Пространство состояний	$\mathcal{X}_0$	$\mathcal{X}_1$
Фазо-зависимая миграция	отсутствует	при $l: 0 \to 1: m \to \min(n, M)$
Завершение обслуживания на МЕС	$(n, m, 0) \to (n - 1, m - 1,0)$	если $n > m$: $(n, m, 1) \to (n - 1, m, 1)$; если $n = m$: $(n, m, 1) \to (n - 1, m - 1,1)$
Завершение обслуживания в облачном сервере	$(n, m, 0) \to (n - 1, m, 0)$	$(n, m, 1) \to (n - 1, m, 1)$

Для наглядного представления фазо-зависимой динамики системы и структуры допустимых переходов между состояниями на рис. 3.6 приведен граф переходов

марковского процесса. Граф отражает все типы событий, определяющих поведение системы: поступления задач, завершения обслуживания облачном сервере и на МЕС-узле, а также фазовые переходы $l = 0 \to l = 1$ и $l = 1 \to l = 0$ с соответствующими правилами миграции задач.

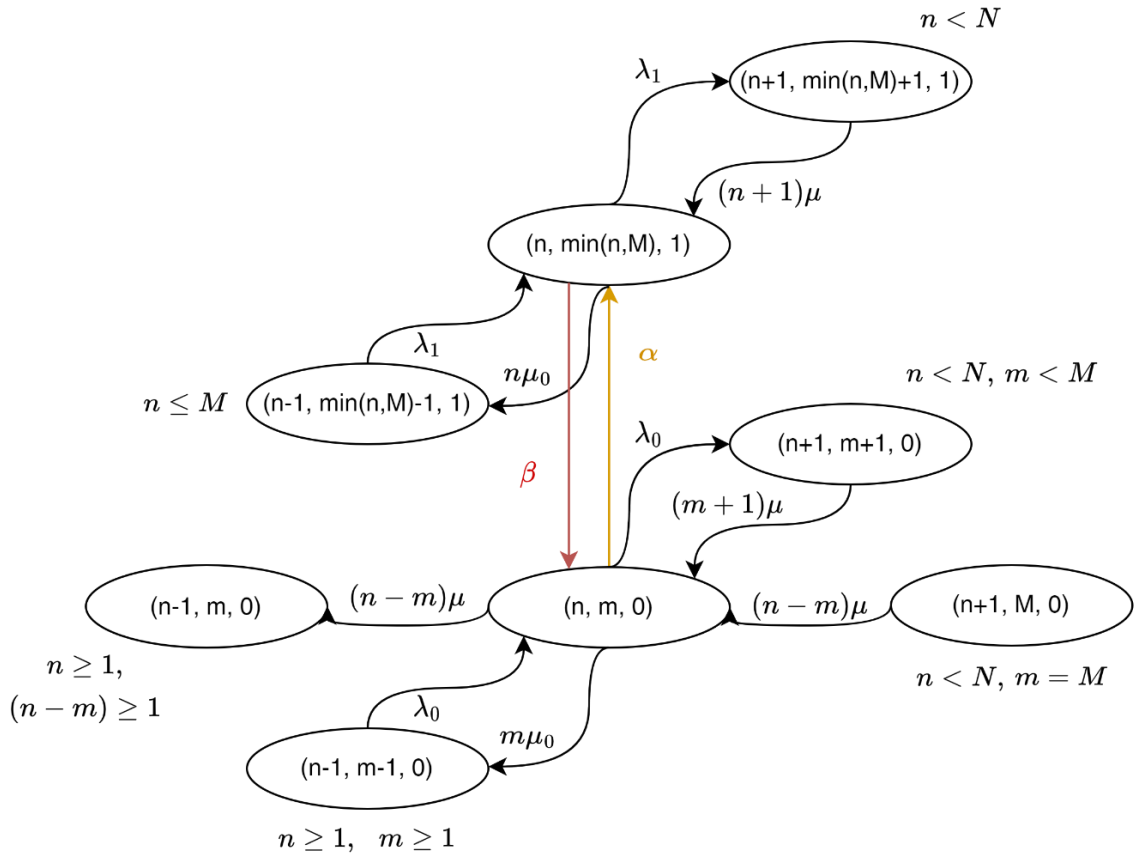

Рисунок 3.6 – Диаграмма интенсивностей переходов для центрального состояния

В качестве иллюстрации структуры пространства состояний и фазо-зависимых правил управления рассмотрим малый численный пример при параметрах $M = 2$ и $N = 3$. В этом случае в фазе низкой нагрузки ($l = 0$) допустимые состояния системы имеют вид $(n, m, 0)$, где $n = 0,1,2,3$ и $0 \le m \le min(n, 2)$. В фазе высокой нагрузки ($l = 1$) пространство состояний вырождается в набор состояний вида $(n, min(n, 2), 1)$, где $n = 0,1,2,3$. Таким образом, при $M = 2$ и $N = 5$, т.е. максимальное количество задач в облачном сервере равно 3. Пространство состояний включает в фазе $l = 0$ девять состояний и в фазе $l = 1$ четыре состояния и может быть представлено в виде $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1$, где

$$\mathcal{X}_0 = \{(0,0,0), (1,0,0), (1,1,0), (2,0,0), (2,1,0), (2,2,0), (3,0,0), (3,1,0), (3,2,0)\},$$
$$\mathcal{X}_1 = \{(0,0,1), (1,1,1), (2,2,1), (3,2,1)\}. \quad (3.45)$$

Граф интенсивностей переходов для данного примера приведен на рис. 3.7.

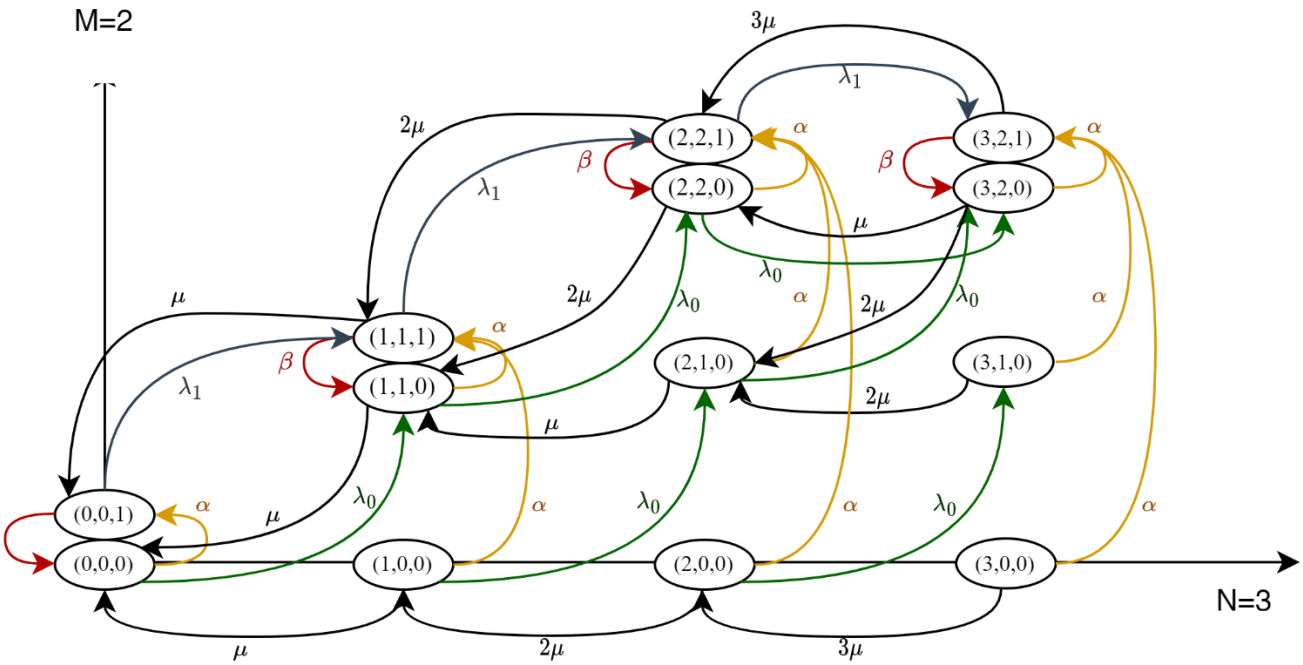

Рисунок 3.7 – Пример диаграммы интенсивностей переходов

3.5 Матричный рекуррентный алгоритм расчета стационарного распределения

Для анализа стационарного режима функционирования системы рассмотрим матрицу интенсивностей переходов $\boldsymbol{Q}$ марковского процесса $\boldsymbol{X}(t)$, описывающего динамику состояний на пространстве $\mathcal{X}$. Введем лексикографический порядок состояний, позволяющий представить матрицу $\boldsymbol{Q}$ в блочно-трехдиагональном виде. Введем обозначение $r_n = min(n, M)$. Тогда для фиксированного уровня n лексикографически упорядоченный набор состояний имеет вид

$$(n, 0,0), (n, 1,0), \dots, (n, r_n, 0), (n, r_n, 1).$$

Таким образом, на уровне n имеется $r_n + 1$ состояний с фазой $l = 0$ и одно состояние с фазой $l = 1$. Общее число состояний уровня n равно $d_n = r_n + 2$.

Вектор стационарных вероятностей марковского процесса представляется в виде $\boldsymbol{\pi} = (\pi_0, \pi_1, \dots, \pi_N)$, где π_n вектор-строка размерности d_n, элементы которого соответствуют стационарным вероятностям всех состояний уровня n, упорядоченных в соответствии с введенным лексикографическим порядком. Введенное упорядочивание позволяет сформулировать следующую лемму.

Лемма 3.2. Для модели с перемещением заявок и коррелированным потоком, если на пространстве $\mathcal{X}$ в пространстве состояний СП $\boldsymbol{X}(t)$ введен лексикографический порядок

$$\begin{gathered}\boldsymbol{x} = (n, m, l) < (n', m', l') = \boldsymbol{x}' \Leftrightarrow \\ (n < n') \vee \left(n = n', (l < l') \vee (l = l', m < m')\right)\end{gathered} \tag{3.46}$$

тогда матрица интенсивностей переходов СП $\boldsymbol{X}(t)$ представима в блочно-трехдиагональном виде

$$\begin{gathered}\boldsymbol{Q} = diag\,(\boldsymbol{B}_o, \dots, \boldsymbol{B}_{M-1}, \boldsymbol{B}_M, \dots, \boldsymbol{B}_M, \boldsymbol{B}_N) + \\ + diag^{-}(\boldsymbol{C}_1, \dots \boldsymbol{C}_{M+1}, \dots, \boldsymbol{C}_{M+1}) + \\ + diag^{+}(\boldsymbol{A}_0, \dots, \boldsymbol{A}_{M-1}, \boldsymbol{A}_M, \dots, \boldsymbol{A}_M)\end{gathered} \tag{3.47}$$

где ненулевые блоки определяются следующим образом

$$\boldsymbol{A}_n = (\boldsymbol{A}_{n0}, \boldsymbol{a}_{n1}^T), \qquad \boldsymbol{A}_{n0} = diag^{+}(\lambda_0, \dots, \lambda_0), \boldsymbol{a}_{n1}^T = \lambda_1 \cdot \boldsymbol{e}_{n+2}, n = 0, \dots, M-1, \tag{3.48}$$

$$\boldsymbol{A}_M = \begin{pmatrix} \boldsymbol{A}_{M2} & \boldsymbol{0^T} \\ \boldsymbol{0} & \lambda_1 \end{pmatrix}, \qquad \boldsymbol{A}_{M2} = diag^{+}(\lambda_0, \dots, \lambda_0) + diag\,(0, \dots, 0, \lambda_1), n = M, \dots, N \tag{3.49}$$

$$\boldsymbol{C}_n = \begin{pmatrix} \boldsymbol{C}_{n0} \\ \boldsymbol{c}_{n1} \end{pmatrix}, \qquad \boldsymbol{C}_{n0} = diag\,(n\mu, (n-1)\mu, \dots, \mu, 0) + diag^{-}(\mu, 2\mu, \dots, n\mu), \tag{3.50}$$

$$\boldsymbol{c}_{n1} = n\mu\, \boldsymbol{e}_{n+1}, n = 0, \dots, M,$$

$$\boldsymbol{C}_{M+1} = \begin{pmatrix} \boldsymbol{C}_{M2} & \boldsymbol{0^T} \\ \boldsymbol{0} & n\mu \end{pmatrix}, \tag{3.51}$$

$$\boldsymbol{C}_{M2} = diag\,(n\mu, (n-1)\mu, \dots, \mu, 0) + diag^{-}(\mu, 2\mu, \dots, n\mu),$$

$$\boldsymbol{B}_0 = \begin{pmatrix} -(\lambda\_0 + \alpha) & \alpha \\ \beta & -(\lambda_1 + \beta) \end{pmatrix}, \tag{3.52}$$

$$\boldsymbol{B}_n = \begin{pmatrix} \boldsymbol{B}_{n00} & \boldsymbol{b}_{n01}^{\boldsymbol{T}} \\ \boldsymbol{b}_{n10} & -(\lambda_1 + n\mu + \beta) \end{pmatrix}, \tag{3.53}$$

$$\boldsymbol{B}_{n00} = -[-(\lambda_0 + n\mu + \alpha)] \cdot \boldsymbol{I}_{M+1}, \boldsymbol{b}_{n01} = (\alpha, \dots . \alpha), \qquad \boldsymbol{b}_{n10} = (0, \dots . 0, \beta), \tag{3.54}$$

$$n = 1, \dots, N-1,$$

$$\boldsymbol{B}_N = \begin{pmatrix} \boldsymbol{B}_{N00} & \boldsymbol{b}_{N01}^{\boldsymbol{T}} \\ \boldsymbol{b}_{N10} & -(N\mu + \beta) \end{pmatrix}, \tag{3.55}$$

$$\boldsymbol{B}_{N00} = -[N\mu + \alpha] \cdot \boldsymbol{I}_N, \boldsymbol{b}_{N01}^{\boldsymbol{T}} = \boldsymbol{b}_{n01}, \boldsymbol{b}_{N10} = \boldsymbol{b}_{n10}.$$

Доказательство.

(а) Введение лексикографического порядка

Рассмотрим допустимые переходы марковского процесса в пространстве состояний $\mathcal{X}$. Переходы, изменяющие координату n, имеют вид $(n, m, l) \to (n', m', l')$ и соответствуют поступлению заявок и завершению их обслуживания. При фиксированном значении n допускаются переходы $(n, m, l) \to (n, m, l')$, связанные с изменением фазового состояния. Изменение координаты m возможно лишь при фиксированных n и l. В соответствии с данной структурой переходов на пространстве состояний $\mathcal{X}$ вводится лексикографический порядок. При таком порядке множество $\mathcal{X}$ имеет вид:

$\mathcal{X} = \{(0,0,0), (0,0,1), (1,0,0), \dots, (3,0,0), (3,1,0), \dots, (n, m, 0), \dots, (n, m, 1), (n, n, 0), (n, n, 1)\}$

б) Описание блоков матрицы

С учетом введенного лексикографического порядка и ненулевых элементов матрицы $\boldsymbol{Q}$ интенсивностей переходов, матрица $\boldsymbol{Q}$ имеет блочный вид

$\boldsymbol{Q}$	0	1	2	⋯	$M-1$	M	$M+1$	⋯	$N-1$	N
0	$\boldsymbol{B}_0$	$\boldsymbol{A}_0$	0	0	0	0	0	0	0	0
1	$\boldsymbol{C}_1$	$\boldsymbol{B}_1$	$\boldsymbol{A}_1$	0	0	0	0	0	0	0
2	0	$\boldsymbol{C}_2$	$\boldsymbol{B}_2$	⋱	0	0	0	0	0	0
⋮	0	0	⋱	⋱	⋱	0	0	0	0	0
$M-1$	0	0	0	⋱	$\boldsymbol{B}_{M-1}$	$\boldsymbol{A}_{M-1}$	0	0	0	0
M	0	0	0	0	$\boldsymbol{C}_M$	$\boldsymbol{B}_M$	$\boldsymbol{A}_M$	0	0	0
$M+1$	0	0	0	0	0	$\boldsymbol{C}_{M+1}$	$\boldsymbol{B}_M$	$\boldsymbol{A}_M$	0	0
⋮	0	0	0	0	0	0	$\boldsymbol{C}_{M+1}$	⋱	⋱	0
$N-1$	0	0	0	0	0	0	0	⋱	$\boldsymbol{B}_M$	$\boldsymbol{A}_M$
N	0	0	0	0	0	0	0	0	$\boldsymbol{C}_{M+1}$	$\boldsymbol{B}_N$

Рассмотрим каждый из блоков по отдельности.

1) Рассмотрим матрицу $\boldsymbol{A}_n$

а) $\mathrm{n} = 0, \dots, M-1, dim\, \boldsymbol{A}_n = (n+2) \times (n+3)$, прямоугольная:

$\boldsymbol{A}_n$	$(n+1,0,0)$	$(n+1,1,0)$	⋯	⋯	$(n+1,m+1,0)$	⋯	$(n+1,n+1,0$	$(n+1,n+1,1$
$(n,0,0)$	0	λ_0	0	0	0	0	0	0
$(n,1,0)$	0	0	λ_0	0	0	0	0	0
⋮	0	0	0	⋱	0	0	0	0
$(n,m,0)$	0	0	0	0	λ_0	0	0	0
⋮	0	0	0	0	0	⋱	0	0
$(n,n,0)$	0	0	0	0	0	0	λ_0	0
$(n,n,1)$	0	0	0	0	0	0	0	λ_1

б) $\mathrm{n} = M, \dots, N, dim\, \boldsymbol{A}_n = (M+2) \times (M+2)$, квадратная:

$\boldsymbol{A}_n$	$(n+1,0,0)$	$(n+1,1,0)$	⋯	⋯	$(n+1,m+1,0)$	⋯	$(n+1,n+1,0$	$(n+1,n+1,1$
$(n,0,0)$	0	λ_0	0	0	0	0	0	0
$(n,1,0)$	0	0	λ_0	0	0	0	0	0
⋮	0	0	0	⋱	0	0	0	0
$(n,m,0)$	0	0	0	0	λ_0	0	0	0
⋮	0	0	0	0	0	⋱	0	0
$(n,M-1,0$	0	0	0	0	0	0	λ_0	0
$(n,M,0)$	0	0	0	0	0	0	λ_0	0
$(n,n,1)$	0	0	0	0	0	0	0	λ_1

2) Рассмотрим матрицу $\boldsymbol{C}_n$

а) $n = 0, \dots, M, dim\, \boldsymbol{C}_n = (n+2) \times (n+1)$, прямоугольная:

$\boldsymbol{C}_n$	$(n-1,0,0)$	$(n-1,1,0)$	⋯	⋯	$(n-1,n-1,0$	$(n-1,n-1,1$
$(n,0,0)$	$n\mu$	0	0	0	0	0
$(n,1,0)$	μ	$(n-1)\mu$	0	0	0	0
⋮	0	2μ	⋱	0	0	0
⋮	0	0	⋱	2μ	0	0

⋮	0	0	0	$(n-1)\mu$	μ	0
$(n,m,0)$	0	0	0	0	$n\mu$	0
$(n,n,1)$	0	0	0	0	0	$n\mu$

б) $n = M+1, \dots, N$, $dim\, \boldsymbol{C}_n = (M+2) \times (M+2)$, квадратная:

$\boldsymbol{C}_n$	$(n-1,0,0)$	$(n-1,1,0)$	⋯	⋯	⋯	$(n-1,M,0)$	$(n-1,n-1,1$
$(n,0,0)$	$n\mu$			0	0	0	0
$(n,1,0)$	μ	$(n-1)\mu$		0	0	0	0
⋮	0	2μ	⋱	0	0	0	0
⋮	0	0	⋱	⋱	0	0	0
⋮	0	0	0	$(M-1)\mu$	$(n-M-1)\mu$	0	0
$(n,M,0)$	0	0	0	0	$M\mu$	$(n-M)\mu$	0
$(n,n,1)$	0	0	0	0	0	0	$n\mu$

3) Рассмотрим матрицу $\boldsymbol{B}_n$

а) $n = 0$, $dim\, \boldsymbol{B}_0 = 2 \times 2$, квадратная:

$\boldsymbol{B}_0$	$(0,0,0)$	$(0,0,1)$
$(0,0,0)$	$-(\lambda_0 + \alpha)$	α
$(0,0,1)$	β	$-(\lambda_1 + \beta)$

б) $n = 1, \dots, M-1$, $dim\, \boldsymbol{B}_n = (n+2) \times (n+2)$, квадратная:

$\boldsymbol{B}_n$	$(n,0,0)$	⋯	$(n,n,0)$	$(n,n,1)$
$(n,0,0)$	$-(\lambda_0 + n\mu + \alpha)$	0	0	α
⋮	0	⋱	0	α
$(n,n,0)$	0	0	$-(\lambda_0 + n\mu + \alpha)$	α
$(n,n,1)$	0	0	β	$-(\lambda_1 + n\mu + \beta)$

в) $n = M, \dots, N-1$, $dim\, \boldsymbol{B}_n = (M+2) \times (M+2)$, квадратная:

$\boldsymbol{B}_n$	$(n,0,0)$	⋯	$(n,N-1,0)$	$(n,N-1,1)$
$(n,0,0)$	$-(\lambda_0 + n\mu + \alpha)$	0	0	α
⋮	0	⋱	0	α
$(n,N-1,0)$	0	0	$-(\lambda_0 + n\mu + \alpha)$	α
$(n,N-1,1)$	0	0	β	$-(\lambda_1 + n\mu + \beta)$

г) $n = N$, $dim\, \boldsymbol{B}_N = (M+2) \times (M+2)$, квадратная:

$\boldsymbol{B}_N$	$(n,0,0)$	⋯	$(n,M,0)$	$(n,M,1)$
$(n,0,0)$	$-(N\mu + \alpha)$	0	0	α
⋮	0	⋱	0	α
$(n,M,0)$	0	0	$-(N\mu + \alpha)$	α
$(n,M,1)$	0	0	β	$-(N\mu + \beta)$

Поскольку пространство состояний $\mathcal{X}$ конечно и состояния упорядочены по уровням $n = 0,1, \dots, N$, матрица интенсивностей $\boldsymbol{Q}$ является конечной квадратной матрицей размера $|\mathcal{X}| \times |\mathcal{X}|$. При этом $|\mathcal{X}|$

$$|\mathcal{X}| = \sum_{n=0}^{M-1}(n+2) + \sum_{n=M}^{N}(M+2) = \\ = \sum_{n=0}^{M-1} n + 2M = \frac{(M-1)M}{2} + 2M + (N-M+1)(M+2) = \\ = MN + 2N - \frac{M^2}{2} + \frac{M}{2} + 2. \quad (3.56)$$

■

Для рассмотренного ранее примера матрица интенсивностей переходов $\boldsymbol{Q}$ имеет блочно-трехдиагональную структуру и выглядит следующим образом.

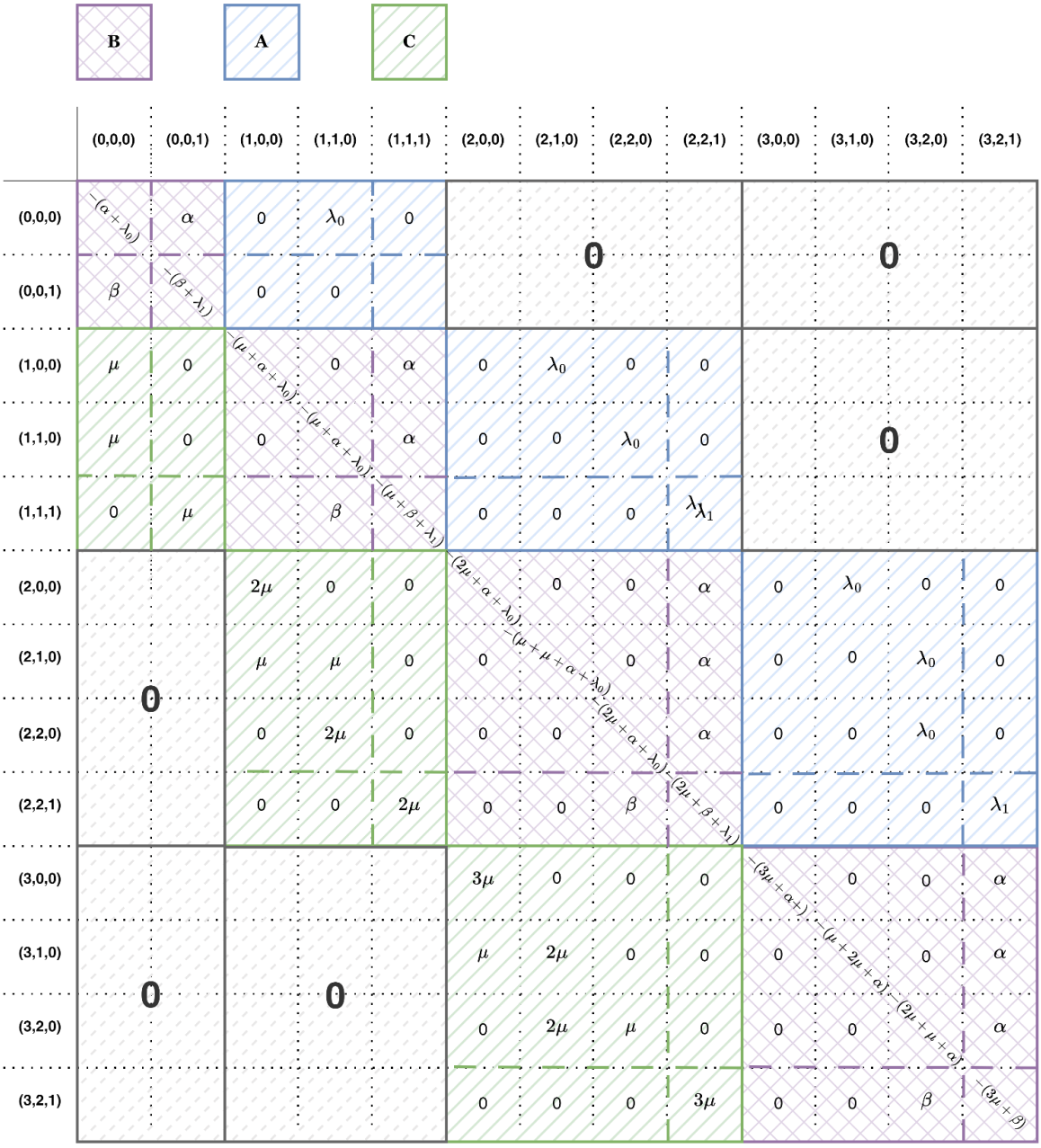

Рисунок 3.8 – Пример матрицы интенсивностей переходов

Далее сформулирована и доказана теорема, позволяющая найти стационарное распределение вероятностей.

Теорема 3.2. Для модели с перемещением заявок и коррелированным потоком стационарное распределение СП $\boldsymbol{X}(t)$ рассчитывается в матричном виде по формуле

$$\boldsymbol{\pi}_n = \boldsymbol{\pi}_0 \prod_{i=0}^{n-1} \boldsymbol{R}_i, \qquad n = \overline{1, N}\,, \tag{3.57}$$

где $\boldsymbol{\pi}_0$ является единственным решением системы уравнений

$$\begin{cases} \boldsymbol{\pi}_0(\boldsymbol{B}_0 + \boldsymbol{R}_0\boldsymbol{C}_1) = 0, \\ \sum_{n=0}^{N} \boldsymbol{\pi}_0 \prod_{i=0}^{n-1} \boldsymbol{R}_i\, \boldsymbol{1}^{\boldsymbol{T}} = 1 \end{cases} \tag{3.58}$$

а матрицы $\boldsymbol{R}_i$ вычисляются по рекуррентным соотношениям

$$\boldsymbol{R}_n = -\boldsymbol{A}_n\,(\boldsymbol{B}_{n+1} + \boldsymbol{R}_{n+1}\boldsymbol{C}_{n+2}\,)^{-1},\ \ n = 0, \dots, M-1,$$
$$\boldsymbol{R}_n = -\boldsymbol{A}_M\,(\boldsymbol{B}_M + \boldsymbol{R}_{n+1}\boldsymbol{C}_{M+1}\,)^{-1},\ \ n = M, \dots, N-2,$$
$$\boldsymbol{R}_{N-1} = -\boldsymbol{A}_M\,\boldsymbol{B}_N^{-1}.$$

Доказательство.

Запишем СУР $\boldsymbol{\pi Q} = \boldsymbol{0}$ с учетом блочно-трехдиагонального вида матрицы $\boldsymbol{Q}$ интенсивностей переходов

а) СУР в матричном виде

$$\begin{cases} \boldsymbol{\pi}_0\boldsymbol{B}_0 + \boldsymbol{\pi}_1\boldsymbol{C}_1 = 0, \\ \boldsymbol{\pi}_0\boldsymbol{A}_0 + \boldsymbol{\pi}_1\boldsymbol{B}_1 + \boldsymbol{\pi}_2\boldsymbol{C}_2 = 0, \\ \boldsymbol{\pi}_{n-1}\boldsymbol{A}_{n-1} + \boldsymbol{\pi}_n\boldsymbol{B}_n + \boldsymbol{\pi}_{n+1}\boldsymbol{C}_{n+1} = 0, n = 2 \dots M-2, \\ \boldsymbol{\pi}_{M-2}\boldsymbol{A}_{M-2} + \boldsymbol{\pi}_{M-1}\boldsymbol{B}_{M-1} + \boldsymbol{\pi}_M\boldsymbol{C}_M = 0, \\ \boldsymbol{\pi}_{M-1}\boldsymbol{A}_{M-1} + \boldsymbol{\pi}_M\boldsymbol{B}_M + \boldsymbol{\pi}_{M+1}\boldsymbol{C}_{M+1} = 0, \\ \boldsymbol{\pi}_{n-1}\boldsymbol{A}_M + \boldsymbol{\pi}_n\boldsymbol{B}_n + \boldsymbol{\pi}_{n+1}\boldsymbol{C}_{M+1} = 0, n = M+1 \dots N-2, \\ \boldsymbol{\pi}_{N-2}\boldsymbol{A}_M + \boldsymbol{\pi}_{N-1}\boldsymbol{B}_M + \boldsymbol{\pi}_N\boldsymbol{C}_{M+1} = 0, \\ \boldsymbol{\pi}_{N-1}\boldsymbol{A}_M + \boldsymbol{\pi}_N\boldsymbol{B}_N = 0. \end{cases} \tag{3.59}$$

Подставим вид матриц в полученный вид СУР

Уровень N

$$\boldsymbol{\pi}_{N-1}\boldsymbol{A}_M + \boldsymbol{\pi}_N\,\boldsymbol{B}_N = 0,$$
$$\boldsymbol{\pi}_N\boldsymbol{B}_N = -\boldsymbol{\pi}_{N-1}\boldsymbol{A}_M,$$
$$\boldsymbol{\pi}_N = \boldsymbol{\pi}_{N-1}\underbrace{(-\boldsymbol{A}_M\ \boldsymbol{B}_N^{-1})}_{\boldsymbol{R}_{N-1}} \Rightarrow\ \boldsymbol{\pi}_N = \boldsymbol{\pi}_{N-1}\boldsymbol{R}_{N-1}.$$

Уровень N-1

$$\boldsymbol{\pi}_{N-2}\boldsymbol{A}_M + \boldsymbol{\pi}_{N-1}\boldsymbol{B}_{N-1} + \boldsymbol{\pi}_N\boldsymbol{C}_{M+1} = 0,$$
$$\boldsymbol{\pi}_{N-1}(\boldsymbol{B}_{N-1} + \boldsymbol{R}_{N-1}\boldsymbol{C}_{M+1}\,) = -\boldsymbol{\pi}_{N-2}\boldsymbol{A}_M,$$
$$\boldsymbol{\pi}_{N-1} = \boldsymbol{\pi}_{N-2}\ \underbrace{-(\boldsymbol{A}_M\ (\boldsymbol{B}_{N-1} + \boldsymbol{R}_{N-1}\boldsymbol{C}_{M+1}\,)^{-1})}_{\boldsymbol{R}_{N-2}} \Rightarrow\ \boldsymbol{\pi}_{N-1} = \boldsymbol{\pi}_{N-2}\boldsymbol{R}_{N-2}.$$

Пусть $n = M+1, \dots, N-1$

$$\begin{cases} \boldsymbol{R}_n = -\boldsymbol{A}_M\,(\boldsymbol{B}_{n+1} + \boldsymbol{R}_{n+1}\boldsymbol{C}_{M+1}\,)^{-1} \\ \boldsymbol{\pi}_{n+1} = \boldsymbol{\pi}_n\boldsymbol{R}_n \end{cases},$$

тогда

$$\boldsymbol{\pi}_{n-1}\boldsymbol{A}_M + \boldsymbol{\pi}_n\boldsymbol{B}_n + \boldsymbol{\pi}_{n+1}\boldsymbol{C}_{M+1} = 0,$$

$$\boldsymbol{\pi}_{n-1}\boldsymbol{A}_M + \boldsymbol{\pi}_n\boldsymbol{B}_n + \boldsymbol{\pi}_n\boldsymbol{R}_n\boldsymbol{C}_{M+1} = 0,$$
$$\boldsymbol{\pi}_n(\boldsymbol{B}_n + \boldsymbol{R}_n\boldsymbol{C}_{M+1}) = -\boldsymbol{\pi}_{n-1}\boldsymbol{A}_M\,,$$
$$\boldsymbol{\pi}_n = \boldsymbol{\pi}_{n-1}\boldsymbol{R}_{n-1},$$
$$\boldsymbol{R}_{n-1} = -\boldsymbol{A}_M(\boldsymbol{B}_n + \boldsymbol{R}_n\boldsymbol{C}_{M+1}\,)^{-1}.$$

Следовательно,

$$\boldsymbol{\pi}_n = \boldsymbol{\pi}_{n-1}\boldsymbol{R}_{n-1}, \qquad n = M+1, \dots, N-1$$

Проверим уравнение уровня M

$$\boldsymbol{\pi}_{\mathrm{M}-1}\mathbf{A}_{\mathrm{M}-1} + \boldsymbol{\pi}_{\mathrm{M}}\mathbf{B}_{\mathrm{M}} + \boldsymbol{\pi}_{\mathrm{M}+1}\mathbf{C}_{\mathrm{M}+1} = 0$$
$$\boldsymbol{\pi}_M(\boldsymbol{B}_M + \boldsymbol{R}_M\boldsymbol{C}_{M+1}) = -\boldsymbol{\pi}_{M-1}\boldsymbol{A}_{M-1}\,,$$
$$\boldsymbol{\pi}_M = \boldsymbol{\pi}_{M-1}\underbrace{-(\boldsymbol{A}_{M-1}\,(\boldsymbol{B}_M + \boldsymbol{R}_M\boldsymbol{C}_{M+1}\,)^{-1})}_{\boldsymbol{R}_{M-1}} \Rightarrow \boldsymbol{\pi}_M = \boldsymbol{\pi}_{M-1}\boldsymbol{R}_{M-1}.$$

Пусть $n = 1, \dots, M-1$

$$\begin{cases} \boldsymbol{R}_n = -\boldsymbol{A}_{M-1}\,(\boldsymbol{B}_M + \boldsymbol{R}_{n+1}\boldsymbol{C}_{M+1}\,)^{-1} \\ \boldsymbol{\pi}_{n+1} = \boldsymbol{\pi}_n\boldsymbol{R}_n \end{cases},$$

тогда

$$\boldsymbol{\pi}_{n-1}\boldsymbol{A}_{n-1} + \boldsymbol{\pi}_n\boldsymbol{B}_n + \boldsymbol{\pi}_{n+1}\boldsymbol{C}_{n+1} = 0,$$
$$\boldsymbol{\pi}_{n-1}\boldsymbol{A}_{n-1} + \boldsymbol{\pi}_n\boldsymbol{B}_n + \boldsymbol{\pi}_n\boldsymbol{R}_n\boldsymbol{C}_{n+1} = 0,$$
$$\boldsymbol{\pi}_n(\boldsymbol{B}_n + \boldsymbol{R}_n\boldsymbol{C}_{n+1}) = -\boldsymbol{\pi}_{n-1}\boldsymbol{A}_{n-1}\,,$$
$$\boldsymbol{\pi}_n = \boldsymbol{\pi}_{n-1}\boldsymbol{R}_{n-1}.$$

Следовательно

$$\boldsymbol{R}_n = -\boldsymbol{A}_n\,(\boldsymbol{B}_{n+1} + \boldsymbol{R}_{n+1}\boldsymbol{C}_{n+2}\,)^{-1}, \qquad n = 0, \dots, M-2$$
$$\boldsymbol{R}_{M-1} = -\boldsymbol{A}_{M-1}\,(\boldsymbol{B}_M + \boldsymbol{R}_M\boldsymbol{C}_{M+1}\,)^{-1},$$
$$\boldsymbol{R}_n = -\boldsymbol{A}_M\,(\boldsymbol{B}_M + \boldsymbol{R}_{n+1}\boldsymbol{C}_{M+1}\,)^{-1}, \qquad n = M, \dots, N-2.$$
$$\boldsymbol{R}_{N-1} = -\boldsymbol{A}_M\,\boldsymbol{B}_N^{-1}$$

б) Выражение π_n через π_0

$$\boldsymbol{\pi}_n = \boldsymbol{\pi}_{n-1}\boldsymbol{R}_{n-1} = \boldsymbol{\pi}_{n-2}\boldsymbol{R}_{n-2}\boldsymbol{R}_{n-1} = \cdots = \boldsymbol{\pi}_0\boldsymbol{R}_i \dots \boldsymbol{R}_{n-1} = \boldsymbol{\pi}_0\prod_{i=0}^{n-1}\boldsymbol{R}_i, n = \overline{1,N}, \tag{3.60}$$

в) Получение системы уравнений относительно π_0

$$\boldsymbol{\pi}_0\boldsymbol{B}_0 + \boldsymbol{\pi}_1\boldsymbol{C}_1 = 0,$$
$$\boldsymbol{\pi}_0\boldsymbol{B}_0 + \boldsymbol{\pi}_0\boldsymbol{R}_0\boldsymbol{C}_1 = 0,$$
$$\boldsymbol{\pi}_0(\boldsymbol{B}_0 + \boldsymbol{R}_0\boldsymbol{C}_1) = 0,$$

зная условие нормировки

$$\sum_{n=0}^{N}\boldsymbol{\pi}_n\,\mathbf{1}^{\boldsymbol{T}} = 1. \tag{3.61}$$

Тогда

$$\sum_{n=0}^{N}\boldsymbol{\pi}_0\prod_{i=0}^{n-1}\boldsymbol{R}_i\,\mathbf{1}^{\boldsymbol{T}} = 1 \tag{3.62}$$

Теорема доказана. □

3.6 Численный анализ модели с параметрами реального сетевого трафика

В данном разделе проводится численный анализ модели миграции сервиса в граничной облачной архитектуре с MMPP-потоком. Целью анализа является оценка того, как фазовая структура нагрузки и параметры MMPP, полученные на основе MAP-моделей трафика главы 1, влияют на распределение заявок между узлом MEC и облачным сервером, загрузку граничных серверов и среднее время отклика для различных классов сервисов. Все сервисы представлены в таблице 3.8.

Таблица 3.8 – Значения параметров для MMPP-потока по данным о трафике из раздела 1.3.

	Сервис	λ_0	λ_1	α	β	μ
1.	Потоковые приложения	1,8673051	1,8666943	1,0736854	0,0456169	1,5384870
2.	Другие приложения	4,9993535	4,9992132	0,4227921	4,0434638	6,2756205
3.	Веб-приложения	7,4643245	7,4640595	0,6736310	2,1898135	1,4038385
4.	Игры	4,0193734	4,0192981	1,2505095	7,0705579	31,9190739
5.	Электронная почта	2,0324334	2,0324509	2,4313324	0,5856852	14,0235315
6.	Передача файлов	2,6952114	2,6951995	0,1061536	0,0720156	0,4292618
7.	IP-телефония	1,4624261	1,4624227	1,2749329	0,0199447	0,8045442
8.	Приложения для обмена мгновенными сообщениями	3,0298573	3,0297607	0,7399089	0,2773495	2,5918172
9.	Безопасность	2,0526913	2,0526995	0,2628325	0,1850493	1,1353557
10.	Потоковая передача музыки	1,0131738	1,0131675	0,5189813	0,0295707	0,4706986
11.	Сетевые службы	2,7608742	2,7608729	1,3082660	0,0505938	1,5066014
12.	Мобильные терминалы	1,5337653	1,5337575	0,7768765	0,0820990	1,6823162
13.	Одноранговые приложения	5,1264921	5,1264192	5,3618994	2,5794773	8,5187078
14.	Транзакции баз данных	2,3569465	2,3569354	0,7435152	0,2840737	2,2236550
15.	Сетевые службы	1,6413285	1,6413252	0,6102959	0,0433039	0,8674576
16.	Файловые системы	2,4030431	2,4030271	0,9509579	0,2347422	1,8115727

Для каждой выделенной группы сервисов (№ 1 потоковое видео, № 3 веб-приложения, № 4 игровые приложения, № 7 IP-телефония, № 8 обмен мгновенными сообщениями) параметры MMPP: $\lambda_0, \lambda_1, \alpha, \beta$ и интенсивность обслуживания μ идентифицировались на основе матриц $\boldsymbol{D}_0, \boldsymbol{D}_1$, построенных в разделе 1.4. Далее представлены матрицы для выбранных типов сервисов.

$$\boldsymbol{D}_0^1 = \begin{bmatrix} -2{,}94099 & 1{,}07369 \\ 0{,}04562 & -1{,}91231 \end{bmatrix}, \boldsymbol{D}_1^1 = \begin{bmatrix} 1{,}86731 & 0{,}00 \\ 0{,}00 & 1{,}86669 \end{bmatrix}, \quad (3.63)$$

$$\boldsymbol{D}_0^3 = \begin{bmatrix} -8{,}13796 & 2{,}18981 \\ 0{,}67363 & -9{,}65387 \end{bmatrix}, \boldsymbol{D}_1^3 = \begin{bmatrix} 7{,}46432 & 0{,}00 \\ 0{,}00 & 7{,}46406 \end{bmatrix}, \quad (3.64)$$

$$\boldsymbol{D}_0^4 = \begin{bmatrix} -5{,}26988 & 1{,}25051 \\ 7{,}07056 & -11{,}08987 \end{bmatrix}, \boldsymbol{D}_1^4 = \begin{bmatrix} 4{,}01937 & 0{,}00 \\ 0{,}00 & 4{,}0193 \end{bmatrix}, \quad (3.65)$$

$$\boldsymbol{D}_0^7 = \begin{bmatrix} -1{,}59019 & 0{,}18168 \\ 2{,}05104 & -3{,}45949 \end{bmatrix}, \boldsymbol{D}_1^7 = \begin{bmatrix} 1{,}4085 & 0{,}00 \\ 0{,}00 & 1{,}40844 \end{bmatrix}, \quad (3.66)$$

$$\boldsymbol{D}_0^8 = \begin{bmatrix} -4{,}3612 & 0{,}07517 \\ 1{,}42264 & -5{,}70984 \end{bmatrix}, \boldsymbol{D}_1^8 = \begin{bmatrix} 4{,}28603 & 0{,}00 \\ 0{,}00 & 4{,}2872 \end{bmatrix}. \quad (3.67)$$

Графически распределение по выбранным типам сервисов представлено на риснке 3.9.

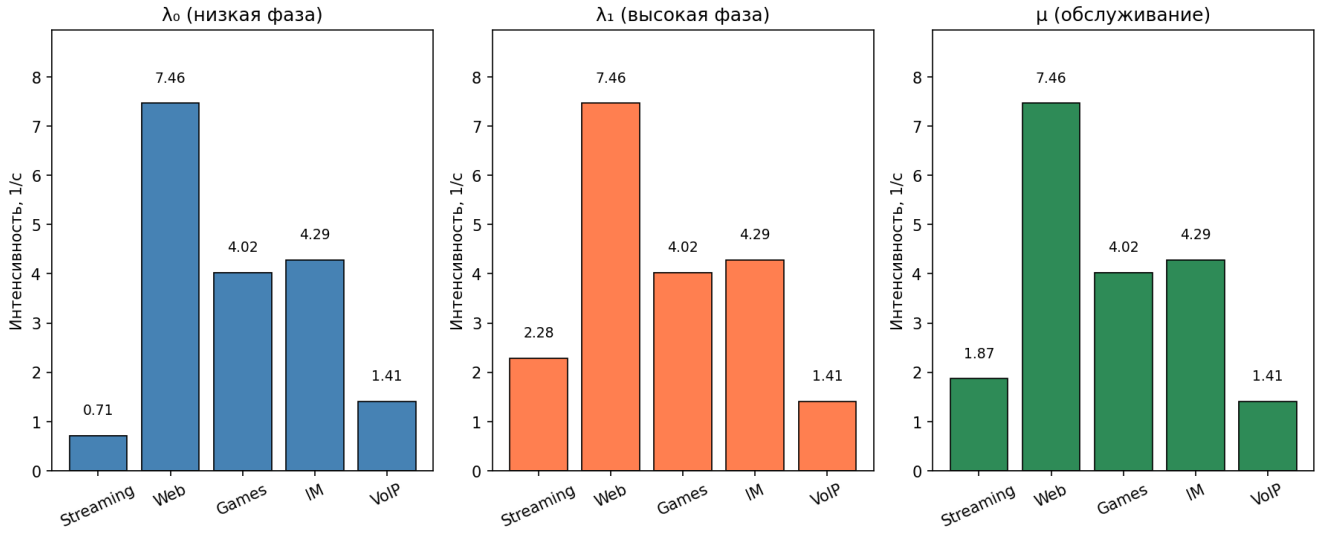

Рисунок 3.9 – Интенсивности входного потока и обслуживания в MMPP-модели по классам сервисов $(\lambda_0, \lambda_1, \mu)$

Зная стационарные вероятности можем вывести следующие характеристики распределения нагрузки между MEC-узлом и облачным сервером. Вероятность фаз

$$P_l = \sum_{\boldsymbol{x}=(n,m,l)\in\mathcal{X}} \boldsymbol{\pi}(\boldsymbol{x}), l = 0, 1, \quad (3.68)$$

среднее число заявок в системе

$$N = \sum_{\boldsymbol{x}=(n,m,l)\in\mathcal{X}} \boldsymbol{\pi}(\boldsymbol{x})\, n, \quad (3.69)$$

среднее число заявок в MEC

$$N_{MEC} = \sum_{\boldsymbol{x}=(n,m,l)\in\mathcal{X}} \boldsymbol{\pi}(\boldsymbol{x})\, m, \quad (3.70)$$

среднее число заявок в облачном сервере

$$N_c = \sum_{\boldsymbol{x}=(n,m,l)\in\mathcal{X}} \boldsymbol{\pi}(\boldsymbol{x})\, (n-m), \quad (3.71)$$

вероятность блокировки

$$B = \sum_{\boldsymbol{x}\in\mathcal{B}_v} \boldsymbol{\pi}(\boldsymbol{x}), \mathcal{B}_v = \{\boldsymbol{x} \in \mathcal{X}, n+1 > N\}, \quad (3.72)$$

средняя загрузка MEC

$$U_{MEC} = \frac{N_{MEC}}{M}. \quad (3.73)$$

В численном анализе также фиксируются размеры системы: максимальное число одновременных заявок сервиса принимается равным $N = 100$, а емкость узла MEC по

числу одновременно обслуживаемых заявок $M = 5$. Выбор этих параметров отражает сценарий, в котором MEC-узел имеет существенно более ограниченные вычислительные возможности по сравнению с облачным сервером. Это позволяет исследовать влияние коррелированной нагрузки и фазо-зависимой политики миграции на риск перегрузки MEC и перераспределение потока между граничным и облачным уровнями.

На рисунках 3.9 и 3.10 представлены значения интенсивностей поступления заявок в фазах низкой и высокой нагрузки (λ_0 и λ_1) и интенсивность обслуживания μ для каждого сервиса, что отражает существенные различия в требовательности сервиса к пропускной способности и скорости обработки. Например, для веб-приложений λ_0 и λ_1 заметно выше, чем для VoIP и потокового видео, тогда как для потокового видео характерны относительно малые интенсивности прихода при низкой интенсивности обслуживания, что соответствует большему среднему размеру сеанса.

В совокупности полученные результаты показывают, что учет коррелированного характера трафика через MMPP-модель существенно влияет на оценку эффективности миграции сервисов в граничной облачной архитектуре. Все показатели, полученные в ходе численного анализа представлены в таблице 3.9.

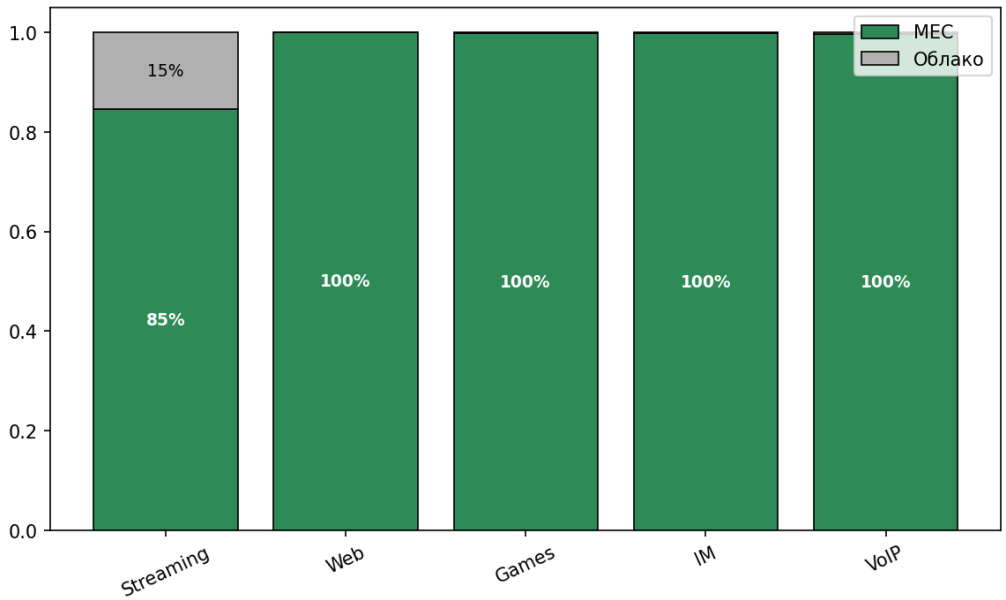

Рисунок 3.10 – Распределение заявок сервиса между MEC-узлом и облачным сервером

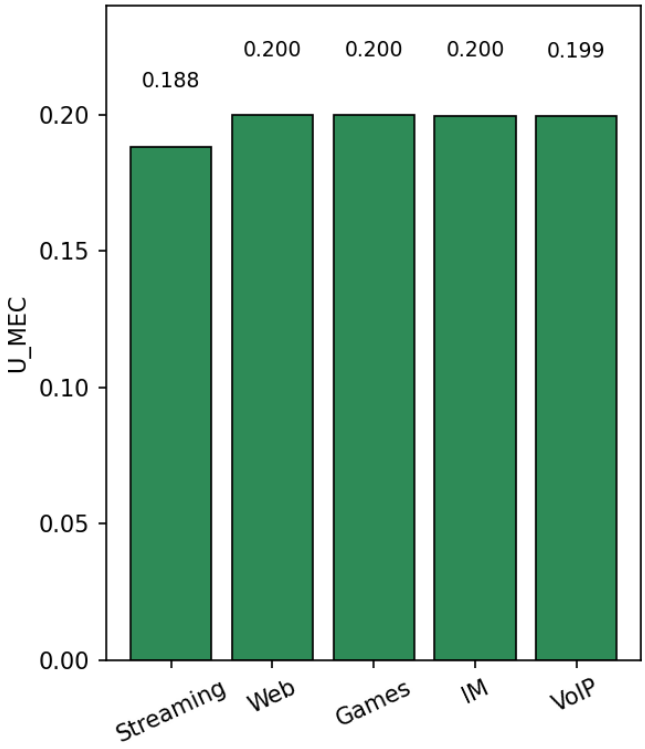

Рисунок 3.11 – Загрузка MEC-узла и доля заявок, обслуживаемых на MEC

Таблица 3.9 - Сравнение фаз нагрузки и допустимых состояний

Сервис	Загрузка MEC	Доля на MEC	Доля в облачном сервере	Вер-ть блокировки
Потоковые приложения	0,188373	0,845211	0,154789	7,25E-06
Веб-приложения	0,199854	0,999268	0,000732	1,67E-12
Игры	0,199791	0,998954	0,001046	3,89E-11
Приложения для обмена мгновенными сообщениями	0,199612	0,997689	0,002311	1,36E-08
IP-телефония	0,199434	0,997154	0,002846	1,92E-09

Заключение

Основные научные результаты диссертационной работы состоят в следующем.

1. Разработана модель миграции виртуальных машин с обслуживаемыми задачами между облачными серверами в виде системы массового обслуживания с перемещением всех заявок класса между группами приборов. Для предотвращения осцилляций решение о миграции принимается только в момент поступления новой заявки соответствующего класса. Формализованы два алгоритма миграции, направленные на минимизацию занятой пропускной способности серверов в облачной инфраструктуре и различающиеся моментом оценки занятости приборов: до и после принятия поступившей заявки. Получены формулы для расчета показателей эффективности миграции виртуальных машин, в том числе вероятности миграции и средней высвобождаемой пропускной способности сервера.
2. Разработана модель миграции пользователей сервисов между граничным и облачными серверами в виде системы массового обслуживания с перемещением заявок между общей группой приборов с эксклюзивным обслуживанием одного класса и индивидуальными приборами. Политика миграции формализована как задача минимизации суммарной межконцевой задержки по всем пользователям при каждом изменении состояния системы. Получено оптимальное распределение заявок между группами приборов в виде функции от числа заявок в системе. Показано, что стационарное распределение вероятностей состояний системы при оптимальной политике имеет мультипликативный вид.
3. Модель миграции с коррелированным потоком заявок пользователей в беспроводной гранично-облачной архитектуре построена в виде системы массового обслуживания с входным потоком, моделируемым марковски-модулированным пуассоновским процессом MMPP. Политика миграции зависит от текущей фазы MMPP-потока: в фазе высокой интенсивности решение о миграции пересматривается при каждом событии, в фазе низкой интенсивности – только при поступлении новой заявки или смене фазы. Матрица интенсивностей переходов представлена в блочно-трехдиагональном виде, на основе которого разработан матричный алгоритм расчета стационарного распределения. Численный анализ модели выполнен с параметрами MMPP-потока, оцененными по реальному сетевому трафику методом максимального правдоподобия на основе EM-алгоритма для различных типов сервисов.

СПИСОК ОСНОВНЫХ ОБОЗНАЧЕНИЙ

Параметры для первой модели, рассмотренной в главе 2.

Параметр		**Описание**
$\mathcal{S} = \{1, \dots, S\}$	-	Множество серверов
$s \in S$	-	Индекс сервера
C_s	-	Пропускная способность сервера s
$\mathcal{V} = \{1, \dots, V\}$	-	Множество виртуальных машин
λ_v	-	Интенсивность пуассоновского потока задач, поступающих на виртуальную машину v
μ_v	-	Параметр экспоненциального распределения времени обслуживания задач виртуальной машиной v
b_v	-	Требуемая пропускная способность для одной задачи, обслуживаемой ВМ v
$\mathcal{X}$	-	Пространство состояний марковского процесса
$\boldsymbol{n} = (n_1, .., n_V)$	-	Число задач, выполняемых на ВМ
$\boldsymbol{s} = (s_1, .., s_V)$	-	Сервера, на которых работают ВМ
$\boldsymbol{x} = (\boldsymbol{n}, \boldsymbol{s})$	-	Произвольное состояние системы
$\mathcal{X}_v$	-	Подмножество критических состояний, инициирующих миграцию для ВМ v
$c_s(\boldsymbol{x})$	-	Суммарная нагрузка на сервер s в состоянии **x**
$\mathcal{S}_v(\boldsymbol{x})$	-	Множество серверов, допустимых для миграции виртуальной машины v из состояния $\boldsymbol{x}$
$s_v^*(\boldsymbol{x})$	-	Целевой сервер миграции виртуальной машины v в состоянии $\boldsymbol{x}$
$i = 1$	-	Политика минимизации прогнозируемой загрузки
$i = 2$	-	Политика минимизации текущей загрузки
M_s	-	Максимальное число задач на сервере s (для политики с порогом по числу задач)
$m_s(\boldsymbol{x})$	-	Число задач, размещенных на сервере s в состоянии $\boldsymbol{x}$ (для политики с порогом по числу задач)
$\boldsymbol{\pi}$	-	Стационарное распределение вероятностей

Параметры для второй модели, рассмотренной в главе 3, разделах 3.1. – 3.3.

Параметр		**Описание**
$\mathcal{K} = \{1, \dots, K\}$	-	Множество сервисов, множество облачных серверов
$k \in \mathcal{K}$	-	Число сервисов, число облачных серверов
λ_k	-	Интенсивность пуассоновского потока запросов сервиса k
μ_k	-	Параметр экспоненциального времени обслуживания запроса сервиса k
b_k	-	Требуемая пропускная способность для одного запроса сервис k
$\boldsymbol{n} = (n_1, \dots, n_K)$	-	Число пользователей сервисов
n_k	-	Число пользователей k-сервиса
(m, s)	-	Состояние МЕС-узла
m	-	Число пользователей на МЕС-узле
s	-	Номер сервиса на МЕС-узле
$\boldsymbol{x} = (\boldsymbol{n}, m, s)$	-	Произвольное состояние системы
$\widetilde{\mathcal{X}}$	-	Множество допустимых состояний $\boldsymbol{x}$, удовлетворяющих ограничениям по пропускной способности
$\mathcal{N}$		Множество допустимых состояний $\boldsymbol{n}$, при фиксированном правиле управления редуцированной модели
C_0	-	Пропускная способность канала между пользователями и МЕС-узлом
C_k	-	Пропускная способность канала между пользователями и облачной инфраструктурой для сервиса k
M_s	-	Максимальное число пользователей сервиса s, одновременно обслуживаемых на МЕС-узле
N_k	-	Максимальное число пользователей сервиса k, обслуживаемых в облачном сервере
$c_0(\boldsymbol{x})$	-	Занимаемая пропускная способность МЕС-узла в состоянии
$c_s(\boldsymbol{x})$	-	Занимаемая пропускная способность облачного сервера сервиса s в состоянии $\boldsymbol{x}$
$c_k(\boldsymbol{x})$	-	Занимаемая пропускная способность облачного сервера сервиса $k \neq s$ в состоянии $\boldsymbol{x}$

d_0	-	Межконцевая задержка одного пользователя при обслуживании на МЕС-узле
d_k	-	Межконцевая задержка одного пользователя сервиса k при обслуживании в облачном сервере
$d(\boldsymbol{x})$	-	Суммарная задержка, которую ощущают все пользователи в состоянии $\boldsymbol{x}$
$\mathcal{A}_{ik}(\boldsymbol{x})$	-	Множество допустимых действий
$\boldsymbol{\pi}$	-	Стационарное распределение вероятностей

Параметры для второй модели (результат № 3), рассмотренной в главе 3, разделах 3.4. – 3.6.

Параметр		**Описание**
$\mathcal{K} = \{1\}$	-	Множество сервисов, множество облачных серверов
α	-	Интенсивность смены фазы $0 \to 1$
β	-	Интенсивность смены фазы $1 \to 0$
λ_0	-	Интенсивность потока запросов пользователей в фазе 0
λ_1	-	Интенсивность потока запросов пользователей в фазе 0, $\lambda_1 > \lambda_0$
μ	-	Параметр экспоненциального времени обслуживания
n	-	Число пользователей в системе
m	-	Число пользователей на МЕС-узле
$n - m$	-	Число пользователей в облачном сервере
l	-	Фазовое состояние системы
$\boldsymbol{x} = (n, m, l)$	-	Произвольное состояние системы
M	-	Максимальное число пользователей, обслуживаемых МЕС-узле
N	-	Максимальное число пользователей, обслуживаемых в облачном сервере
$\mathcal{X}$	-	Пространство состояний марковского процесс
$\mathcal{X}_0$	-	Пространство состояний марковского процесс в фазе $0 \to 1$
$\mathcal{X}_1$	-	Пространство состояний марковского процесс в фазе $1 \to 0$

СПИСОК ЛИТЕРАТУРЫ

1. Что такое облачные вычисления? [Электронный ресурс] // SAP : сайт. – URL: https://www.sap.com/central-asia-caucasus/products/technology-platform/what-is-cloud-computing.html (дата обращения: 03.05.2025).
2. Журавлев Е. Е., Иванов С. В., Каменщиков А. А. и др. Интероперабельность в облачных вычислениях // Журнал радиоэлектроники. – 2013. – № 9. – С. 14.
3. Mell P., Grance T. The NIST Definition of Cloud Computing // NIST Special Publication 800-145. – 2011. – URL: https://doi.org/10.6028/NIST.SP.800-145 (дата обращения: 01.01.2025).
4. Google Search [Электронный ресурс] // Wikipedia. – URL: https://en.wikipedia.org/wiki/Google_Search (дата обращения: 26.02.2025).
5. Younis R. et al. A Comprehensive Analysis of Cloud Service Models: IaaS, PaaS, and SaaS in the Context of Emerging Technologies and Trends // 2024 International Conference on Electrical, Communication and Computer Engineering (ICECCE). – IEEE, 2024. – С. 1–6.
6. Buyya R., Garg S. K., Calheiros R. N. SLA-oriented resource provisioning for cloud computing: Challenges, architecture, and solutions // 2011 International Conference on Cloud and Service Computing. – IEEE, 2011. – С. 1–10.
7. Armbrust M., Fox A., Griffith R., Joseph A. D., Katz R., Konwinski A., Lee G., Patterson D., Rabkin A., Stoica I., Zaharia M. A view of cloud computing // Communications of the ACM. – 2010. – Vol. 53, No. 4. – P. 50–58.
8. Amazon Elastic Compute Cloud [Электронный ресурс] // Wikipedia. – URL: https://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud (дата обращения: 25.02.2025).
9. MacLean K. The Evolution Of Cloud Computing (The History Of Bare Metal Servers) // DataBank: Data Center Evolved. – 2024. – URL: https://www.databank.com/resources/blogs/the-evolution-of-cloud-computing-the-history-of-bare-metal-servers (дата обращения: 10.01.2024).
10. Gurung D., Rashid S. M. Z., Abdeen Z. u., Rath S. Cloud Revolution: Tracing the Origins and Rise of Cloud Computing // arXiv preprint arXiv:2512.06800. – 2025. – URL: https://arxiv.org/html/2512.06800v1 (дата обращения: 07.02.2026).

11. Bonomi F., Milito R., Zhu J., Addepalli S. Fog Computing and Its Role in the Internet of Things // MCC ’12: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. – New York: ACM, 2012. – С. 13–16.
12. OpenFog Reference Architecture for Fog Computing [Электронный ресурс] / OpenFog Consortium Architecture Working Group. – Fremont, CA: OpenFog Consortium, 2017. – С. 162 – URL: https://www.iiconsortium.org/pdf/OpenFog_Reference_Architecture_2_09_17.pdf (дата обращения: 10.01.2024).
13. OpenFog Consortium: An Ecosystem to Accelerate End-to-End IoT [Электронный ресурс] // Cisco Blogs. – 2015. – URL: https://blogs.cisco.com/innovation/openfog-consortium-announcement (дата обращения: 25.02.2026).
14. Fog computing [Электронный ресурс] // Wikipedia. – URL: https://en.wikipedia.org/wiki/Fog_computing (дата обращения: 25.02.2025).
15. 15. Samouylov K., Gudkova I., Markova E. Formalizing set of multiservice models for analyzing pre-emption mechanisms in wireless 3GPP networks // Communications in Computer and Information Science. – 2016. – Vol. 601. – С. 61–71.
16. Markova E., Moltchanov D., Gudkova I., Samouylov K., Koucheryavy Y. Performance assessment of QoS-aware LTE sessions offloading onto LAA WiFi systems // IEEE Access. – 2019. – Vol. 7. – С. 36300–36311.
17. Adhikari M., Hazra A. 6G-enabled ultra-reliable low-latency communication in edge networks // IEEE Communications Standards Magazine. – 2022. – Vol. 6, No. 1. – С. 67–74.
18. Телекоммуникационный стандарт 5G/IMT-2020 [Электронный ресурс]. – URL: https://www.itu.int/rec/R-REC-M.2150-2-202312-I/en (дата обращения: 07.11.2024).
19. Kushchazli A., Ageeva A., Kochetkova I., Kharin P., Chursin A., Shorgin S. Model of radio admission control for URLLC and adaptive bit rate eMBB in 5G network // CEUR Workshop Proceedings. – 2021. – Vol. 2946. – P. 74–84.
20. Kochetkova I., Makeeva E., Alkanhel R. Retrial queueing system for analyzing impact of priority ultra-reliable low-latency communication transmission on enhanced mobile broadband quality of service degradation in 5G networks // Mathematics. – 2023. – Vol. 11, No. 18. – Art. No. 3925.

21. MEC 002 V2.1.1. Multi-Access Edge Computing (MEC); Phase 2 – Use cases and requirements. – 2018.
22. Multi-access Edge Computing (MEC) // ETSI. – URL: https://www.etsi.org/technologies/multi-access-edge-computing (дата обращения: 11.01.2024).
23. Li C., Sun H., Tang H., Luo Y. Adaptive resource allocation based on the billing granularity in edge-cloud architecture // Computer Communications. – 2019. – DOI: 10.1016/j.comcom.2018.09.019.
24. Edge Computing Congress – Multi-access [Электронный ресурс]. – KNect365, 2017. – URL: https://intuitive-design.foleon.com/knect365/edge-computing-congress/multi-access/ (дата обращения: 25.02.2026).
25. Bringing Scale to the Edge with Multi-Access Edge Computing [Электронный ресурс] // NVIDIA Developer Blog. – 2022. – URL: https://developer.nvidia.com/blog/bringing-scale-to-the-edge-with-multi-access-edge-computing/ (дата обращения: 25.02.2026).
26. Sarah A., Nencioni G., Khan M. M. I. Resource Allocation in Multi-access Edge Computing for 5G-and-beyond networks. – University of Stavanger, Stavanger, Norway.
27. Ateya A. A., Alhussan A. A., Abdallah H. A., Khakimov A., Muthanna A. Edge Computing Platform with Efficient Migration Scheme for 5G/6G Networks // Computer Systems Science & Engineering. – 2023. – Т. 45, № 2.
28. Хакимов А. А. Разработка и исследование моделей адаптивного управления трафиком в сетях пятого поколения: дис. … канд. техн. наук: 05.13.17. – Санкт-Петербург, 2022. – 180 с.
29. Three Services from 5G: Latency and Reliability // Cisco Systems. – 2019. – URL: https://blogs.cisco.com/sp/three-services-from-5g-latency-and-reliability (дата обращения: 11.01.2024).
30. Zhao X., Liu J., Ji B., Wang L. Service Migration Policy Optimization considering User Mobility for E-Healthcare Applications // Journal of Healthcare Engineering. – 2021. – DOI: 10.1155/2021/6629999. – URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC8238570 (дата обращения: 11.01.2024).
31. Sharma R., Deswal S., Singh C. Ensuring Quality of Service in Beyond 5G Networks Through Evolving Architectures, Protocols, and Optimization Methods // 2025 2nd

International Conference on Computational Intelligence and Computing Applications (ICCICA). – IEEE, 2025. – С. 1–6.

32. Gieske J. Mobile Edge Computing (MEC) – How Does MEC Work? // Nybsys. – 2024. – URL: https://nybsys.com/mobile-edge-computing-mec (дата обращения: 11.03.2025).
33. Cloud Computing and Virtual Machine Migration [Электронный ресурс] // International Journal of Computer Science and Network. – 2015. – Vol. 4, No. 4. – URL: http://ijcsn.org/IJCSN-2015/4-4/A-Survey-Paper-Cloud-Computing-and-Virtual-Machine-Migration.pdf (дата обращения: 25.02.2026).
34. A survey on virtual machine migration and server consolidation for cloud data centers [Электронный ресурс]. – URL: https://shop.tarjomeplus.com/UploadFileEn/TPLUS_EN_1399.pdf (дата обращения: 20.01.2026).
35. Ghosh S. K. et al. Live virtual machine migration: A survey, research challenges, and future directions // Future Generation Computer Systems. – 2023. – Vol. 139. – P. 181–210.
36. Le T. A survey of live virtual machine migration techniques // Computer Science Review. – 2020. – DOI: 10.1016/j.cosrev.2020.100329.
37. Zhao J., Ding Y., Xu G., Hu L., Dong Y., Fu X. A Location Selection Policy of Live Virtual Machine Migration for Power Saving and Load Balancing // The Scientific World Journal. – 2013. – DOI: 10.1155/2013/983651.
38. Kushchazli A., Safargalieva A., Kochetkova I., Gorshenin A. Queuing model with customer class movement across server groups for analyzing virtual machine migration in cloud computing // Mathematics. – 2024. – Т. 12, № 3. – С. 468.
39. Sheetal A. P., Ravindranath K. High Efficient Virtual Machine Migration Using Glow Worm Swarm Optimization Method for Cloud Computing // Ingénierie des Systèmes d'Information. – 2021. – Т. 26, № 6.
40. Cao H., Hou Z. Krill Herd Algorithm for Live Virtual Machines Migration in Cloud Environments // International Journal of Advanced Computer Science and Applications. – 2023. – Т. 14, № 5.
41. Belgacem A., Mahmoudi S., Ferrag M. A. A machine learning model for improving virtual machine migration in cloud computing // The Journal of Supercomputing. – 2023. – Т. 79, № 9. – С. 9486–9508.

42. Ma Z., Ma D., Lv M., Liu Y. Virtual machine migration techniques for optimizing energy consumption in cloud data centers // IEEE Access. – 2023. – T. 11. – C. 86739–86753.
43. Kaur A., Kumar S., Gupta D., Hamid Y., Hamdi M., Ksibi A., Saini S. Algorithmic approach to virtual machine migration in cloud computing with updated SESA algorithm // Sensors. – 2023. – T. 23, № 13. – C. 6117.
44. Ahmad I., Shahnaz A., Asfand-e-Yar M., Khalil W., Bano Y. A Service Level Agreement Aware Online Algorithm for Virtual Machine Migration // Computers, Materials & Continua. – 2023. – T. 74, № 1.
45. Ma X., He W., Gao Y. Virtual machine migration strategy based on Markov decision and greedy algorithm in edge computing environment // Wireless Communications and Mobile Computing. – 2023. – T. 2023, № 1. – C. 6441791.
46. Li H., Liu J., Zhou Q. Research on energy-saving virtual machine migration algorithm for green data center // IET Control Theory & Applications. – 2023. – T. 17, № 13. – C. 1830–1839.
47. Tuli K., Malhotra M. Novel framework: meta-heuristic elastic scheduling approach in virtual machine selection & migration // International Journal of Engineering Trends and Technology. – 2023. – T. 71. – C. 436–452.
48. Wu Q., Ishikawa F., Zhu Q., Xia Y. Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters // IEEE Transactions on Services Computing. – 2016. – T. 12, № 4. – C. 550–563.
49. Yang C. T., Wan T. Y. Implementation of an energy saving cloud infrastructure with virtual machine power usage monitoring and live migration on OpenStack // Computing. – 2020. – T. 102, № 6. – C. 1547–1566.
50. Tang Z. et al. Study on Dynamic Service Migration Strategy with Energy Optimization in Mobile Edge Computing // Journal of Network and Computer Applications. – 2019. – Vol. 135. – P. 86–99.
51. Hua K., Su S., Wang Y., Yang L. Intelligent service migration for the Internet of Vehicles in edge computing: A mobility-aware deep reinforcement learning framework // Computer Networks. – 2025. – Vol. 257. – Art. No. 111021. – DOI: 10.1016/j.comnet.2024.111021.
52. 52. Liu X. et al. Digital twin-assisted and mobility-aware service migration in Mobile Edge Computing // Computer Networks. – 2024. – Vol. 237. – Art. No. 110036.

53. Zhang H. et al. Multitier Service Migration Framework Based on Mobility Prediction and Mobile Edge Computing [Электронный ресурс] // Wireless Communications and Mobile Computing. – 2021. – Art. ID 6638730. – URL: https://onlinelibrary.wiley.com/doi/10.1155/2021/6638730 (дата обращения: 22.01.2026).
54. Singh A. et al. LMTSA-MEC: A Lightweight Mobility Aware Task and Service Allocation Framework for MEC [Электронный ресурс] // Future Generation Computer Systems. – 2025. – Vol. 154. – P. 120–133.
55. Sun Y., Liu J., Zhang X., Wang H. Service migration for mobile edge computing based on partially observable Markov decision processes // Computers & Electrical Engineering. – 2023. – Vol. 108. – Art. No. 108739.
56. Zarro I. M. A machine learning approach to predict Virtual Machine Workload. – Instituto Superior Técnico, 2021. – URL: https://scholar.tecnico.ulisboa.pt/records/CyuiO9BQYLkFVN-7QbSo-P9cX7Qa9xSVjiID?lang=en (дата обращения: 01.02.2025).
57. Rathinaraja J., Anandkumar B., Kumara M. A. A., Nadra G., Anand P. Resource Management in Cloud and Cloud-influenced Technologies for Internet of Things Applications // ACM Computing Surveys. – 2023. – DOI: 10.1145/3571729. – URL: https://dl.acm.org/doi/10.1145/3571729 (дата обращения: 01.02.2025).
58. Kim W.-S. Progressive Traffic-Oriented Resource Management for Reducing Network Congestion in Edge Computing // Entropy. – 2021. – DOI: 10.3390/e23050539.
59. Wang Y., Chen J., Wu Z., Chen P., Li X., Hao J. Efficient task migration and resource allocation in cloud–edge collaboration: A DRL approach with learnable masking // Alexandria Engineering Journal. – 2025. – Vol. 111. – P. 107–122. – DOI: 10.1016/j.aej.2024.10.015.
60. Menascé D. A., Bardhan S. TDQN: Trace-driven analytic queuing network modeling of computer systems // Journal of Systems and Software. – 2019. – Т. 147. – С. 162–171.
61. Shaw R., Howley E., Barrett E. Applying reinforcement learning towards automating energy efficient virtual machine consolidation in cloud data centers // Information Systems. – 2022. – Т. 107. – С. 101722.

62. Shaw R. Applying machine learning towards automating resource management in cloud computing environments: doctoral thesis. – NUI Galway, 2021. – URL: http://hdl.handle.net/10379/16516 (дата обращения: 02.02.2025).
63. Парамонов А. И. Разработка и исследование комплекса моделей трафика для сетей связи общего пользования: дис. … канд. техн. наук: 05.12.13. – Санкт-Петербург, 2014. – 325 с.
64. Алгазир А. А. Исследование моделей трафика для сетей связи пятого поколения и разработка методов его обслуживания с использованием БПЛА: дис. … канд. техн. наук: 05.12.13. – Санкт-Петербург, 2023. – 147 с.
65. Vila S., Guirado F., Lérida J. L. Cloud computing virtual machine consolidation based on stock trading forecast techniques // Future Generation Computer Systems. – 2023. – Vol. 145. – P. 321–336.
66. Liu Y., Gong B., Xing C., Jian Y. A Virtual Machine Migration Strategy Based on Time Series Workload Prediction Using Cloud Model // Mathematical Problems in Engineering. – 2014. – Vol. 2014, Iss. 1. – P. 973069. – DOI: 10.1155/2014/973069.
67. Patel M., Chaudhary S., Garg S. Machine Learning Based Statistical Prediction Model for Improving Performance of Live Virtual Machine Migration // Journal of Engineering. – 2016. – Vol. 2016, Iss. 1. – P. 3061674. – DOI: 10.1155/2016/3061674.
68. Masdari M., Khezri H. Efficient VM migrations using forecasting techniques in cloud computing: a comprehensive review // Cluster Computing. – 2020. – Vol. 23, Iss. 4. – P. 2629–2658. – DOI: 10.1007/s10586-019-03032-x.
69. Yang Q., Zhang J. Modelling the queues of connected and autonomous vehicles at signal-free intersections considering the correlated vehicle arrivals // Journal of Computational Science. – 2024. – Vol. 82. – P. 102420.
70. Gortsev A. M., Nezhel'skaya L. A. Analytical Investigation of a Single-Server Queueing System with an Incoming MAP Event Flow // Avtomatika i telemehanika. – 2023. – No. 7. – P. 3–22. – DOI: 10.31857/S0005231023070012.
71. Kochetkova I., Kushchazli A., Burtseva S., Gorshenin A. Short-term mobile network traffic forecasting using seasonal ARIMA and Holt-Winters models // Future Internet. – 2023. – Vol. 15, No. 9. – P. 290. – DOI: 10.3390/fi15090290.

72. Gorshenin A., Kozlovskaya A., Gorbunov S., Kochetkova I. Mobile network traffic analysis based on probability-informed machine learning approach // Computer Networks. – 2024. – Vol. 247. – Art. No. 110433. – DOI: 10.1016/j.comnet.2024.110433.

73. Горшенин А. К., Горбунов С. А., Волканов Д. Ю. О кластеризации объектов сетевой вычислительной инфраструктуры на основе анализа статистических аномалий в трафике // Информационные технологии и вычислительные системы. – 2023. – Т. 17, вып. 3. – С. 76–87. – DOI: 10.14357/19922264230311.

74. Ермолаев А. М., Кущазли А. И., Хакимов А. А., Кочеткова И. А. Разработка модели трафика в задаче миграции сервисов между граничными MEC-узлами и облаком // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с международным участием ИТТММ-2025, Москва, 18–22 апреля 2025 г. – М.: РУДН, 2025. – С. 145–149.

75. Силкина М. А., Кущазли А. И., Кочеткова И. А., Горшенин А. К. К статистическому анализу профиля трафика мобильного оператора // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с международным участием ИТТММ-2022, Москва, 18–22 апреля 2022 г. – М.: РУДН, 2022. – С. 152–155.

76. Service Name and Transport Protocol Port Number Registry [Электронный ресурс] / Internet Assigned Numbers Authority (IANA). – URL: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt (дата обращения: 26.11.2024).

77. Cotton M., Eggert L., Touch J., Westerlund M., Cheshire S. Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry: RFC 6335. – IETF, 2011. – 32 p. – (Best Current Practice).

78. Arafa R. A. R., Abdel-haleem A. M., Ali H. A., Elazab M. M. E. A. A Comparative Study of Traffic Classification Techniques for Smart City Networks // Sensors. – 2021. – Vol. 21, No. 14. – Art. No. 4884.

79. Rezaei S., Liu X. Network traffic classification: Techniques, datasets, and challenges // ICT Express. – 2019. – Vol. 5, No. 1. – P. 2–14.

80. Network Traffic Classification Using Machine Learning [Электронный ресурс]. – arXiv preprint arXiv:2503.02141. – 2025. – URL: https://arxiv.org/abs/2503.02141 (дата обращения: 26.02.2025).
81. Reddy S. R. et al. Software-defined networking based network traffic classification using deep learning [Электронный ресурс] // IEEE Access. – 2024. – URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC11362285/ (дата обращения: 26.02.2025).
82. Azzouni A., Pujolle G. A new platform for machine-learning-based network traffic classification // Computer Networks. – 2023. – Vol. 237. – Art. No. 109438.
83. Ali A. A., Al-Jaleel M. H. F. Comparison of Machine Learning Algorithms' Performance in Network Traffic Classification // Tishreen University Journal for Research and Scientific Studies. Engineering Sciences Series. – 2024. – Vol. 46, No. 4. – P. 331–350.
84. Pereira R., de Jesus S., Monteiro P. NTCS: A real time flow-based network traffic classification system // Proc. 10th International Conference on Network and Service Management (CNSM). – IEEE, 2014. – P. 302–305.
85. Al-Mahdi M. S. et al. Comparative Study of Machine Learning Algorithms for Network Traffic Classification // International Journal of Intelligent Systems and Applications in Engineering. – 2023. – Vol. 11, No. 3. – P. 185–194.
86. Comparative Analysis of Classification Algorithms Using Bot_IoT Dataset // Proc. IEEE International Conference on Computer and Information Sciences. – 2023. – Doc. ID 10370699.
87. Li M. et al. Network Traffic Classification Using Ensemble Learning in Software-Defined Networks // Proc. 2021 International Conference on Artificial Intelligence and Computer Engineering (ICAICE). – IEEE, 2021. – P. 382–388.
88. Al-Dabbagh A. A. K., Mahmood S. A. Machine Learning-Based Traffic Classification in IP Networks // Journal of Information Systems Engineering and Management. – 2025. – Vol. 7, No. 4. – P. 1–15.
89. Benavoli A. et al. Evaluation metrics and statistical tests for machine learning // Scientific Reports. – 2024. – Vol. 14. – Art. No. 12345.
90. Hatfield C. M. Applying machine learning to categorize distinct categories of network traffic [Электронный ресурс]. – Honors Thesis, Eastern Michigan University, 2021. – URL: https://commons.emich.edu/cgi/viewcontent.cgi?article=1756&context=honors (дата обращения: 26.11.2024).

91. Traffic generation model [Электронный ресурс] // Wikipedia. – URL: https://en.wikipedia.org/wiki/Traffic_generation_model (дата обращения: 26.02.2025).
92. Лекция 23. Моделирование случайного события. Моделирование полной группы несовместных событий [Электронный ресурс]. – URL: https://stratum.ac.ru/education/textbooks/modelir/lection23.html (дата обращения: 27.11.2024).
93. Hammar A. K. P., Fiedler M. Poisson Packet Traffic Generation Based on Empirical Data [Электронный ресурс] // Proc. 10th World Multi-Conference on Systemics, Cybernetics and Informatics. – 2006. – P. 137–142. – URL: https://iiisci.org/journal/pdv/sci/pdfs/P186919.pdf (дата обращения: 26.02.2025).
94. Downey A. B. Think Stats: Exploratory Data Analysis / A. B. Downey; with contributions by Loukides M., Blanchette M., Demarest R. – 2nd ed. – Sebastopol, CA: O'Reilly Media, 2014. – 223 p.
95. Cryer J. D., Chan K. S. Time Series Analysis: With Applications in R. – 2nd ed. – Berlin; Heidelberg: Springer, 2008. – 491 p.
96. Gijón C., Toril M., Luna-Ramírez S., Marí-Altozano M. L., Ruiz-Avilés J. M. Long-term data traffic forecasting for network dimensioning in LTE with short time series // Electronics. – 2021. – Vol. 10, No. 10. – Art. 1151.
97. Fang L., Cheng X., Wang H., Yang L. Mobile demand forecasting via deep graph-sequence spatiotemporal modeling in cellular networks // IEEE Internet of Things Journal. – 2018. – Vol. 5. – P. 3091–3101.
98. Tran Q. T., Hao L., Trinh Q. K. Cellular network traffic prediction using exponential smoothing methods // Journal of Information and Communication Technology. – 2019. – Vol. 18, No. 1. – P. 1–18.
99. Vishnevsky V. M., Vytovtov K. A., Barabanova E. A., Semenova O. V. Transient behavior of the MAP/M/1/N queuing system // Mathematics. – 2021. – Vol. 9, No. 20. – Art. No. 2559. – DOI: 10.3390/math9202559. [
100. Neuts M. F. A versatile Markovian point process // Journal of Applied Probability. – 1979. – Т. 16, № 4. – С. 764–779.
101. Dudin A. N. et al. A Dual Tandem Queue as a Model of a Pick-Up Point with Batch Receipt and Issue of Parcels // Mathematics. – 2025. – Т. 13, № 3. – С. 488.

102. Дудин А. Н., Клименок В. И., Вишневский В. М. The theory of queueing systems with correlated flows. – Springer International Publishing, 2020. – DOI: 10.1007/978-3-030-32072-0.

103. Вишневский В. М., Клименок В. И., Семенова О. Г., Данг Мин Конг. Retrial tandem queueing system with correlated arrivals // AIMS Mathematics. – 2025. – Vol. 10, Iss. 5. – P. 10650–10674. – DOI: 10.3934/math.2025485.

104. Вишневский В. М., Дудин А. Н. Системы массового обслуживания с коррелированными входными потоками и их применение для моделирования телекоммуникационных сетей // Автоматика и телемеханика. – 2017. – № 8. – С. 3–59.

105. Наумов В. А., Самуйлов К. Е. Анализ сетей ресурсных систем массового обслуживания // Автоматика и телемеханика. – 2018. – № 5. – С. 59–68.

106. Лихтшнер Б. Я., Блатов И. А., Китаев Е. В. Средняя длина очереди для систем массового обслуживания с коррелированными входными потоками // T-Comm: Телекоммуникации и транспорт. – 2020. – Т. 14, № 8. – С. 13–20.

107. Лихтциндер Б. Я. Задержки в очередях СМО с коррелированными потоками заявок // Информационные технологии и нанотехнологии (ИТНТ-2020): труды VI Международной конференции и молодежной школы, Самара, 26–29 мая 2020 г. – Самара: Изд-во Самарского университета, 2020. – Т. 4. Науки о данных. – С. 568–573.

108. Любина Т. В., Назаров А. А. Исследование динамической и адаптивной RQ-систем с входящим MMPP-потоком заявок // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. – 2013. – № 3(24). – С. 104–112.

109. Задиранова Л. А., Моисеева С. П. Асимптотический анализ потока повторных обращений в системе MMPP|M|∞ с повторным обслуживанием // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. – 2015. – № 2(31). – С. 26–34.

110. Горцев А. М., Нежельская Л. А. Асинхронный дважды стохастический поток с инициированием лишних событий // Дискретная математика. – 2011. – Т. 23, № 2. – С. 60–65.

111. Морозов Е. В., Румянцев А. С. Модели многосерверных систем для анализа вычислительного кластера // Труды Карельского научного центра Российской академии наук. – 2011. – № 5. – С. 75–85.

112. Akar N., Bastopcu M., Ulukus S., Başar T. Modeling interfering sources in shared queues for timely computations in edge computing systems // arXiv preprint arXiv:2408.01327. – 2024.

113. Dudin A., Dudina O., Dudin S. Algorithmic analysis of queuing system with varying number of servers, phase-type service time distribution, and changeable arrival process depending on random environment // Mathematics. – 2025. – Vol. 13, Iss. 7. – P. 154. – DOI: 10.3390/math1307154.

114. Kim C., Dudin A., Dudin S., Dudina O. Mathematical Model of Operation of a Cell of a Mobile Communication Network With Adaptive Modulation Schemes and Handover of Mobile Users // IEEE Access. – 2021. – DOI: 10.1109/ACCESS.2021.3100561.

115. Ma A., Pan S., Zhou W. Service Migration Algorithm Based on Markov Decision Process with Multiple Service Types and Multiple System Factors // Chinese Journal of Electronics. – 2024. – Vol. 33, No. 6. – P. 1515–1525. – DOI: 10.23919/cje.2022.00.128.

116. Назаров А. А., Федорова Е. А. Исследование RQ-системы MMPP|GI|1 методом асимптотического анализа второго порядка в условии большой загрузки // Известия Томского политехнического университета. – 2014. – Т. 325, № 5. – С. 6–15.

117. Nazarov A. A., Paul S. V., Lizyura O. D. Heavy outgoing call asymptotics for MMPP|M|1 retrial queue with two way communication and multiple types of outgoing calls // Izvestiya of Saratov University. Mathematics. Mechanics. Informatics. – 2021. – Vol. 21, Iss. 1. – P. 111–124. – DOI: 10.18500/1816-9791-2021-21-1-111-124.

118. Shklennik M., Moiseev A. N., Morozova A. Repeated flow analysis in infinite-server queueing tandem with MMPP arrivals and feedback at the second stage // Global and Stochastic Analysis. – 2021. – Vol. 8, No. 3. – P. 111–122.

119. Чернышова Е. Н., Лисовская Е. Ю. Суммарный объем занятого ресурса в системе с параллельным обслуживанием и входящим MMPP-потоком // Известия Саратовского университета. Новая серия. Серия Математика. Механика. Информатика. – 2020. – Т. 20, № 3. – С. 400–410.

120. Okamura H., Dohi T. mapfit: An R-based tool for PH/MAP parameter estimation // Quantitative Evaluation of Systems: 12th International Conference, QEST 2015, Madrid,

Spain, September 1–3, 2015, Proceedings 12. – Springer International Publishing, 2015. – С. 105–112.

121. Nazarov A. A., Moiseev A. N., Paul S. V., Lapatin I. L., Fedorova E. A., Lizyura O. D., Salimzyanov R., Salimzyanova D. Mathematical model of cloud node using closed queueing system with service rate degradation // Proc. 21st International Asian School-Seminar on Optimization Problems of Complex Systems (OPCS 2025). – 2025. – DOI: 10.1109/OPCS67346.2025.11219374.
122. Sokolov A. M., Semenova O. V., Larionov A. A. Examining the performance of a distributed system through the application of queuing theory // Communications in Computer and Information Science. – 2024. – Vol. 2129. – P. 16–32. – DOI: 10.1007/978-3-031-61835-2_2.
123. Volkov A. O., Korobkina A. V., Stepanov S. N. Development of a model and algorithms for servicing real-time and data traffic in a cloud computing system // 2022 Systems of Signals Generating and Processing in the Field of on Board Communications (SOSG 2022): Conf. Proc. – 2022. – DOI: 10.1109/IEEECONF53456.2022.9744289.
124. Galiaskarov D. F., Stepanov S. N., Nikolaychuk N. N., Pshenichnikov A. P., Kanishcheva M. G. Dynamic load balancing in data center // 2024 Systems of Signals Generating and Processing in the Field of on Board Communications (SOSG 2024): Conf. Proc. – 2024. – DOI: 10.1109/IEEECONF60226.2024.10496723.
125. Полуэктов Д. С. Построение и анализ вероятностных моделей граничных многопользовательских систем и разделения ресурсов беспроводных сетей: дис. … канд. физ.-мат. наук: 1.2.3. – Москва, 2023. – 158 с.
126. Пауль С. В., Назаров А. А., Лапатин И. Л., Иванова А. С. Моделирование производительности облачного узла при коррелированном характере нагрузки // Информационные и математические технологии в науке и управлении. – 2025. – № 4(40). – С. 90–101. – DOI: 10.25729/ESI.2025.40.4.007.
127. Fedorova E. A., Lapatin I. L., Lizyura O. D., Moiseev A. N., Nazarov A. A., Paul S. V. Mathematical modeling of virtual machine life cycle using branching renewal process // Communications in Computer and Information Science. – 2023. – Vol. 1803. – P. 29–39. – DOI: 10.1007/978-3-031-32990-6_3.
128. Stepanov M. S., Stepanov S. N., Kroshin F. S. Effective algorithm of estimation the performance measures of group of servers with dependence of call repetition on the type

of call blocking // Lecture Notes in Computer Science. – 2022. – Vol. 13766. – P. 324–337. – DOI: 10.1007/978-3-031-23207-7_25.

129. Daraseliya A.V., Sopin E.S. Optimization of mobile device energy consumption in a fog-based mobile computing offloading mechanism // Discrete and Continuous Models and Applied Computational Science. 2021. Vol. 29, No. 1. P. 53–62.
130. Sopin E.S., Zolotous N., Ageev K.A., Shorgin S.Ya. Analysis of the response time characteristics of the fog computing enabled real-time mobile applications // LNCS. 2020. Vol. 12525. P. 99–109.
131. Dang Van Thang, Volkov A.A., Muthanna A.S.A. et al. Future of Telepresence Services in the Evolving Fog Computing Environment: A Survey on Research and Use Cases // Sensors. 2025. Vol. 25, No. 11.
132. Belogaev A.A., Elokhin A.A., Krasilov A., Khorov E.M. Cost Optimization for Computing Resource Management in Intelligent Transportation Systems // Journal of Communications Technology and Electronics. 2020. Vol. 65, No. 12. P. 1517–1524.
133. Belogaev A.A., Elokhin A.A., Krasilov A., Khorov E.M., Akyildiz I.F. Cost-effective V2X task offloading in MEC-assisted intelligent transportation systems // IEEE Access. 2020. Vol. 8. P. 169010–169023.
134. Korneev E., Liubogoshchev M., Khorov E.M. Studying Cloud-Based Virtual Reality Traffic // Journal of Communications Technology and Electronics. 2022. Vol. 67, No. 12. P. 1500–1505.
135. Кущазли А. И. Система массового обслуживания с перемещением классов заявок по группам приборов для анализа миграции сервисов в облачной инфраструктуре // Ломоносов-2025: материалы Международного молодежного научного форума, Москва, 11–25 апреля 2025 г. – М.: МАКС Пресс, 2025. – С. 216–217.
136. Кущазли А. И., Власкина А. С., Кочеткова И. А. Система массового обслуживания с перемещением классов заявок между группами приборов для анализа миграции виртуальных машин в облачных вычислениях // Информационные технологии и технические средства управления (ICCT-2024): материалы VIII Международной научной конференции, Владикавказ, 01–05 октября 2024 г. – М.: ИПУ РАН, 2024. – С. 404–405.
137. Кущазли А. И., Сафаргалиева А. И., Кочеткова И. А. Система массового обслуживания с перемещением классов заявок между группами приборов для

анализа миграции виртуальных машин // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с международным участием, Москва, 08–12 апреля 2024 г. – М.: РУДН, 2024. – С. 69–72.

138. Kushchazli A., Leonteva K., Kochetkova I., Khakimov A. Evaluating QoS in dynamic virtual machine migration: A multi-class queuing model for edge-cloud systems // Journal of Sensor and Actuator Networks. – 2025. – Vol. 14, No. 3. – Art. No. 47.
139. Aleksandrov D. S., Vishnevsky A. S., Kolesnikova A. A. Applying queue theory for modeling of cloud computing // Concurrency and Computation: Practice and Experience. – 2023. – Vol. 35, No. 18. – Art. e5186.
140. Muthanna A., Shamilova R., Ateya A. A., Paramonov A., Hammoudeh M. A mobile edge computing/software-defined networking-enabled architecture for vehicular networks // Internet Technology Letters. – 2020. – Vol. 3, No. 6. – Art. e109.
141. Ateya A. A., Abd El-Latif A. A., Muthanna A., Volkov A., Koucheryavy A. Chapter 6. Development of Research Methods for Assessing the Quality of Transmission of Telepresence and Metaverse Data // Enabling metaverse and telepresence services in 6G networks. – River Publishers, 2025. – DOI: 10.1201/9788770046749-8.
142. Kushchazli A., Leonteva K., Gaidamaka E., Kochetkova I. A delay-aware queuing model for performance analysis of service migration in MEC-Cloud environments // Lecture Notes in Computer Science. – 2026. – Vol. 16461. – P. 1–16. – В печати.
143. Башарин Г. П. Лекции по математической теории телетрафика. – М.: РУДН, 2009. – 342 с.
144. Li P., Fan J., Wu J. Exploring the key technologies and applications of 6G wireless communication network // iScience. – 2025. – Vol. 28, No. 5. – Art. 112281. – DOI: 10.1016/j.isci.2025.112281.
145. Ateya A. A., Abd El-Latif A. A., Muthanna A., Volkov A., Koucheryavy A. Chapter 8. 6G telepresence architecture for immersive AR/VR and holographic experiences: a layered approach to superior QoE // Enabling Metaverse and Telepresence Services in 6G Networks. – River Publishers, 2025. – DOI: 10.1201/9788770046749-8.