Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ”

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

АНАЛИЗ НЕСТАЦИОНАРНЫХ ПРОЦЕССОВ В ОБЛАЧНЫХ СИСТЕМАХ

Автор Мартынчук Илья Геннадьевич _____
(Фамилия, Имя, Отчество) (Подпись)

Направление подготовки (специальность) 09.04.01
(код, наименование)
Информатика и вычислительная техника

Квалификация Магистр
(бакалавр, магистр)

Руководитель ВКР Алиев Т. И., профессор, д.т.н. _____
(Фамилия, И., О., ученое звание, степень) (Подпись)

К защите допустить

Руководитель ОП Алиев Т. И., профессор, д.т.н. _____
(Фамилия, И.О., ученое звание, степень) (Подпись)

“_____” _____ 20 ____г.

Санкт-Петербург, 2019 г.

Содержание

Введение

Актуальность темы исследования. В настоящее время широко распространены вычисления в облачных системах, которые предоставляют широкий спектр решаемых задач при малом времени обработки пользовательских запросов. Примерами таких систем могут служить различные интернет-сервисы преобразования форматов изображений и медиа файлов, сервисы для выполнения математических расчетов и сервисы для коллективной работы с офисными документами. Ввиду того, что облачные системы, как правило, имеют веб-интерфейс, для работы с ними не требуется иметь специализированное программное обеспечение, поэтому постоянно увеличивается число их пользователей и возрастает суммарная нагрузка на них.

Процессы, протекающие в реальных системах, обладают определенной степенью случайности [1]. Это означает, что характеристики таких процессов непостоянны и меняются в зависимости от некоторого набора факторов. На практике случайность может проявляться практически во всех процессах: процессы поступления запросов в некоторую систему (или разного рода нагрузочные процессы), процесс обслуживания пользовательских запросов в этой системе и т.д.

Процесс поступления пользовательских запросов к крупным облачным приложениям представляет из себя совокупность некоторого числа потоков. Такие потоки могут отличаться не только интенсивностью, но и периодичностью из-за различного географического положения пользователей [2]. Это затрудняет решение задачи проектирования облачной системы. Поэтому возникает необходимость разработки способа описания и моделирования нагрузочных процессов, не обладающих свойством стационарности и представляющими из себя совокупность нескольких потоков с различными параметрами. Таким образом, наличие адекватной мо-

дели таких систем с учетом нестационарности нагрузки, обеспечит возможность проектирования и управления облачными системами, а также позволит оптимизировать распределение нагрузки и количество вычислительных узлов.

Степень теоретической проработанности темы. В открытом доступе существует ряд научных трудов, описывающих нестационарные процессы [3–6]. Однако данные материалы дают лишь поверхностное описание нестационарных процессов. Также в открытом доступе не были найдены материалы, посвященные исследованию влияния нестационарных процессов на характеристики функционирования систем массового обслуживания и облачных систем в частности. Из сказанного выше можно сделать вывод, что рассматриваемая тема имеет низкую степень теоретической проработки.

Цель и задачи исследования. Целью исследования является обеспечение возможности прогнозирования влияния нестационарных процессов на характеристики функционирования облачных систем. Для достижения поставленной цели требуется решить ряд задач.

1. Предложить классификацию нестационарных процессов.
2. Определить характер нестационарности входного потока пользовательских запросов в реальных системах.
3. Формализовать способ аналитического описания нестационарных процессов.
4. Предложить способ композиции нестационарных распределений.
5. Разработать модель облачной системы и проанализировать её свойства в условиях нестационарности нагрузочных процессов.
6. На основе результатов моделирования сформулировать выводы о

влиянии нестационарности входного потока пользовательских запросов на характеристики функционирования облачных систем.

Область исследования. Проведенный анализ нестационарных процессов в облачных системах полностью соответствует специальности «Информатика и вычислительная техника», а содержание диссертации – техническому заданию.

Объектом исследования выбраны системы с очередями, используемые в качестве моделей облачных систем.

Предметом исследования являются нестационарные процессы и характеристики функционирования облачных систем.

Теоретическую основу исследования составляют научные труды отечественных и зарубежных авторов в области облачных технологий, теории вероятностей и теории очередей.

Методологическую основу исследования составляют методы теории вероятностей, методы теории очередей, методы математической статистики и метод имитационного моделирования.

Информационная база исследования. Основными научными источниками диссертации являются книга В. Столлингса «Современные компьютерные сети» [1], книга Т. И. Алиева «Основы моделирования дискретных систем» [7] и книга Д. Риза «Облачные вычисления» [8].

Научная новизна работы заключается в следующем.

1. Предложена не представленная ранее классификация нестационарных процессов по задающим распределениям и природе самого процесса.

2. Предложено аналитическое описание нестационарных процессов, позволяющее в дальнейшем строить аналитические модели процессов, заданных различными распределениями с изменяющимися во времени параметрами.

3. Формализован способ композиции нестационарных распределений. Данные результаты позволяют прогнозировать нагрузку на облачные системы, нагрузочные процессы на которые состоят из нескольких потоков с различной интенсивностью и периодичностью.

4. Проанализировано влияние нестационарных процессов и их композиции на характеристики функционирования систем с очередями, используемых в качестве моделей облачных систем. Полученные результаты позволяют на основе параметров нестационарных нагрузочных процессов определять эффективность функционирования облачной системы, что может положительно сказаться на процессах проектирования и управления облачными системами.

Практическая значимость исследования. Полученные аналитические зависимости, разработанная имитационная модель облачной системы, выявленные свойства могут быть использованы при проектировании и управлении облачными системами. И в частности, в алгоритмах автоматического масштабирования облачных систем и систем контейнерной кластеризации.

Апробация результатов исследования. Основные результаты исследования обсуждались на IX Научно-практической конференции молодых ученых «Вычислительные системы и сети (Майоровские чтения)», XLVII научной и учебно-методической конференции Университета ИТМО, VI Научно-практической конференции «Наука настоящего и будущего», VII Всероссийском конгрессе молодых ученых, X международной научно-практической конференции «Программная инженерия и компьютерная техника», XLVIII научной и учебно-методической конференции Университета ИТМО и на VIII Всероссийском конгрессе молодых ученых Университет ИТМО. Материалы, отражающие основное содержание работы, изложены в [2, 13, 36, 39, 42, 47].

Объем и структура работы. Выпускная квалификационная работа

содержит 65 страниц машинописного текста, 11 рисунков, 12 таблиц и список литературы, включающий 49 источников. Структурно работа состоит из введения, трех частей и заключения. Во введении обоснована актуальность и новизна исследования, определены цели и задачи, объект и предмет исследования. Первая часть посвящена теоретическим основам исследования и формулировке проблемы. Во второй части представлен анализ нестационарных процессов и их композиции. Третья часть исследования посвящена анализу влияния нестационарных процессов на характеристики функционирования облачных систем. В заключении приведены основные результаты работы.

1 Обзор предметной области

1.1 Облачные системы

Облачные вычисления являются вычислительной парадигмой, которая описывает большой набор систем, соединённых с помощью частных или публичных сетей. Такие системы предоставляют динамически масштабируемую инфраструктуру для приложений, данных и файловых хранилищ, обеспечивая тем самым возможность адаптации к изменениям в нагрузочных процессах. Данный механизм называется автоматическим масштабированием облачного приложения. Введение представленной технологии позволило существенно снизить затраты на размещение приложений, вычисления, а также хранение и доставку содержимого [9].

В типовом случае, в контексте облачных систем могут быть выделены три заинтересованные стороны: пользователь облачной системы, владелец облачного приложения, владелец облачной системы. За счёт процесса автоматического масштабирования владелец облачного приложения может снизить затраты на аренду ресурсов облачной системы при неизменном качестве обслуживания пользовательских запросов. В то же время, путём выключения и перераспределения аппаратных мощностей, владелец облачной системы может снизить затраты на потребляемые энергетические ресурсы [10].

В открытом доступе не было найдено документации должно уровня, которая бы предоставляла возможность анализа и реализации программных модулей для автоматического управления вычислительными ресурсами облачных приложений. В первую очередь, выполнен анализ восьми подходов к построению архитектуры, используемых экспертами в более чем 95 процентов случаев: ARIS, TOGAF, Zachman Framework, IDEF, SWEBOK, FEAF, 4 plus 1 view, RM-ODP [11]. Среди рассмотреннх способов первые три предлагают более широкий взгляд на корпоративную

архитектуру. Поскольку фреймворк ARIS является закрытым корпоративным, его использование сильно ограничено. Модель Захмана использовалась в конце 20-го века и в настоящее время не учитывает всех аспектов корпоративной архитектуры. Подход TOGAF, непрерывно развиваемый и в настоящее время, вобрал в себя достоинства двух предыдущих, поэтому был выбран для данной работы. Ключевые элементы архитектуры и их взаимосвязи представлены на рис. 1. Достижение целевого состояния обуславливает используемую ИТ-стратегию, в рамках которой системе необходима возможность автоматизированного управления ресурсами. Данная возможность обеспечивается модулем автоматического масштабирования, входящего в состав облачного приложения. Облачное приложение, в общем случае, состоит из программного кода, платформы или среды выполнения программного кода и облачной инфраструктуры. Вместе представленные компоненты предоставляют сервис приложения, за счет которого исполняется основной бизнес-процесс и пользователям предоставляется внешний бизнес-сервис. Модуль автоматического масштабирования использует облачную инфраструктуру на базе арендованных серверов из множества серверов центра обработки данных. Арендованные серверы размещают облачную инфраструктуру с помощью технологий виртуализации, отвечающих не только за непосредственно экземпляры узлов, на которых размещается платформа и программный код, но и за сетевые ресурсы и объекты систем хранения данных.

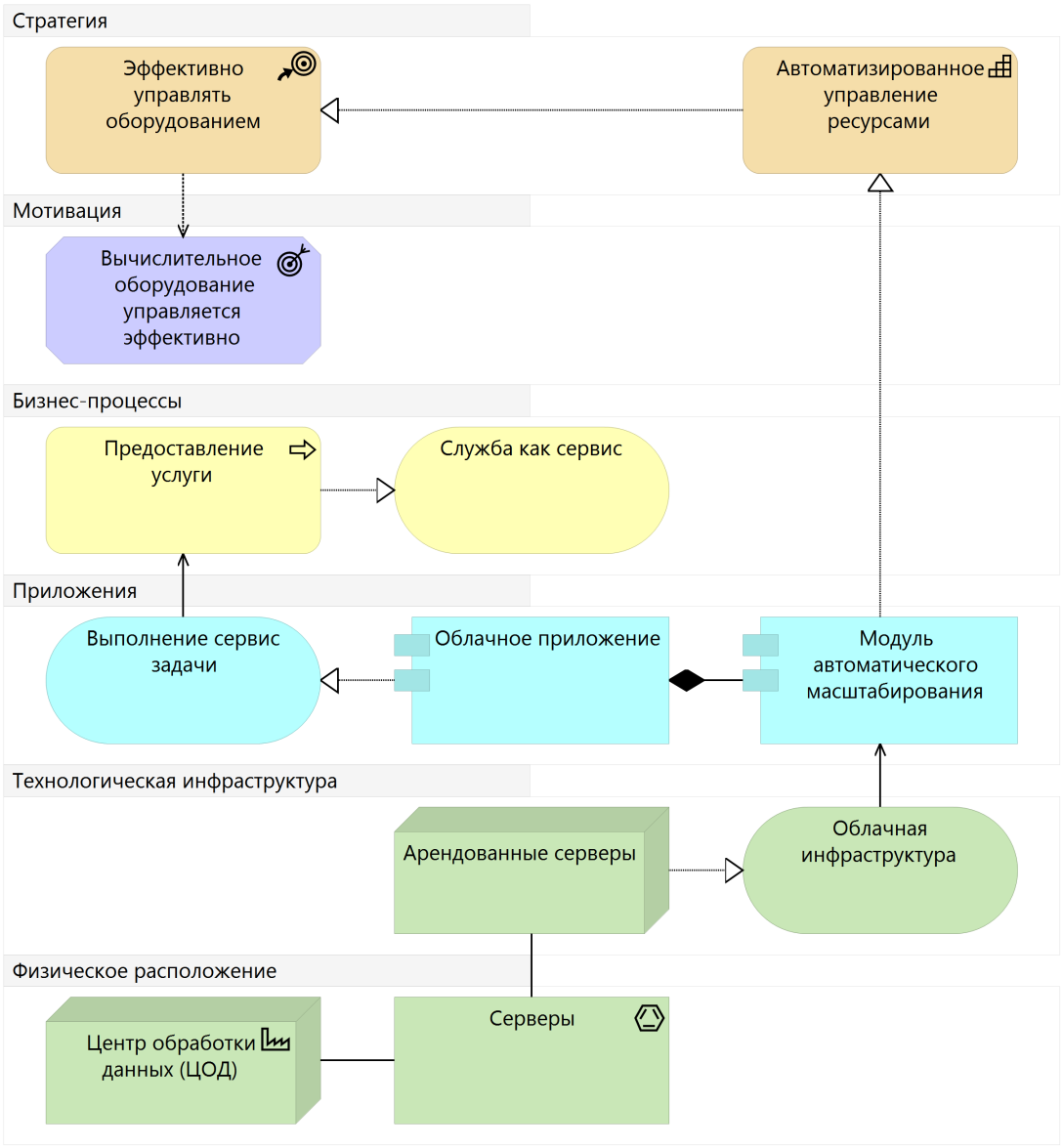


Рис. 1. Ракурс уровней

Сервисы, предоставляемые владельцами облачных приложений и систем, можно разделить на три основные категории.

1. Программное обеспечение как услуга (SaaS). В данной модели конечному пользователю в качестве сервиса по требованию предоставляется законченный экземпляр некоторого приложения. Со стороны пользователей не требуется никаких вложений в серверное оборудование или лицензии на программное обеспечение [12]. В настоящее время программное обеспечение как услугу предостав-

ляют такие компании как Google, Microsoft, Zoho и др.

2. Платформа как услуга (PaaS). Здесь в качестве сервиса уже предоставляется программное обеспечение или среда разработки, на основе которых размещаются сервисы более высокого уровня. Пользователям предоставлена возможность запускать собственные приложения, размещая их на инфраструктурных ресурсах провайдера. Для успешного развертывания и управления пользовательскими приложениями платформа должна содержать определённый набор программного обеспечения. В типовом случае этот набор может включать в себя операционную систему и сервер приложений. Примером такого набора может служить платформа LAMP (Linux, Apache, MySQL и PHP) или ограниченная корпоративная версия платформы Java. В качеств примеров готовых PaaS решений можно привести Goole App Engine, Heroku Runtime, Amazon AWS Elastic Beanstalk и др.

3. Инфраструктура как сервис (IaaS). В данной модели в качестве сервиса пользователю предоставляются инфраструктурные ресурсы, включающие в себя сервера, системы хранения данных, сетевое оборудование, место в центре обработки данных. Отличительной особенностью облачных платформ в данном случае является то, что все выше перечисленные ресурсы предоставляются с использованием технологий виртуализации. Таким образом, пользователю предоставлена возможность построения собственной виртуальной инфраструктуры поверх физической инфраструктуры центра обработки данных. Примерами IaaS решений могут служить такие платформы как Amazon AWS, Microsoft Azure и Google Cloud Platform.

1.2 Облачные сервисы на инфраструктурном уровне

Нестационарные процессы представляют наибольший интерес на инфраструктурном уровне, так как процесс горизонтального масштабирования того или иного приложения напрямую связан с выделением и перераспределением виртуальных инфраструктурных ресурсов владельца облачной системы [13]. Как правило, ресурсы облачных платформ формируются путём объединения физических ресурсов одного или нескольких, географически распределенных центров обработки данных. Возможность размещения экземпляров облачного приложения на географически распределенных вычислительных ресурсах составляет основу механизма обеспечения высокой доступности сервисов и соблюдения соглашений об уровне предоставления услуг [14]. В настоящее время можно выделить две основные технологии, позволяющие реализовывать виртуальную инфраструктуру: виртуальные машины и контейнеры.

1.2.1 Виртуальные машины

Виртуальная машина на программном уровне эмулирует поведение реального физического узла. Такой механизм виртуализации реализуется с помощью аппаратных или программных гипервизоров [15]. Примером аппаратного гипервизора может служить VMware ESXi, программного – VMware Workstation, VirtualBox. Гипервизор обеспечивает для виртуальных машин доступ к аппаратным ресурсам физического узла. В типовом случае такие ресурсы включают в себя потоки центрального процессора, адресное пространство оперативной памяти, блоки системы хранения данных и сетевые интерфейсы. Количество выделенных ресурсов для каждой виртуальной машины является конфигурируемым параметром, что, в свою очередь, повышает гибкость процесса перераспределения ресурсов в зависимости от нагрузки на систему. Большинство облачных платформ, как правило, имеют веб-интерфейс, а также ряд ути-

лит командной строки, позволяющих напрямую управлять виртуальной инфраструктурой системы. Всё это предоставляет владельцу облачного приложения широкий спектр возможностей по управлению инфраструктурными ресурсами [16]. Например, оперативно добавить виртуальную машину в вычислительный процесс при возрастании нагрузки или, наоборот, исключить какой-либо узел при избыточной для решения конкретной задачи вычислительной мощности.

1.2.2 Контейнеры

Приложение, в типовом случае, состоит из среды выполнения, библиотек и исходного кода. Часто приложения имеют зависимости от таких внешних библиотек как libc и libssl. Внешние библиотеки в основном представляют собой разделяемый компонент операционной системы, установленной, в случае облачных систем, на виртуальной машине. Разработка и развертывание программного обеспечения крупных проектов требует взаимодействия различных отделов и групп специалистов, что, в свою очередь, может являться причиной возникновения ошибок и повышения рисков. Часто проблемы возникают, когда приложение, разработанное на персональном компьютере программиста, зависит от разделяемых библиотек, которые недоступны в момент развертывания приложения в рабочей среде [17]. Традиционный метод запуска нескольких приложений на одном узле требует, чтобы все приложения использовали одинаковые версии разделяемых библиотек в системе [18]. Если приложения разработаны разными командами, то такие зависимости усложняют процессы разработки и развертывания программного обеспечения, а также добавляют ненужные связи между командами разработки. Перечисленные выше проблемы можно решить с использованием механизмов контейнеризации. Контейнеры представляют собой легковесную виртуализацию на уровне операционной системы, позволяющую запускать при-

ложения с необходимыми версиями зависимостей и разделяемых библиотек в изолированном пространстве. Все необходимые для запуска приложения компоненты упаковываются в отдельный образ и могут быть повторно использованы [19].

Таким образом, можно выделить следующие основные отличия между виртуальными машинами и контейнерами. Виртуальные машины в большинстве случаев содержат операционную систему и само приложение, что в итоге способствует росту образа виртуальной машины до нескольких гигабайт. Кроме того, для запуска и управления виртуальной машиной необходим гипервизор. Запуск виртуальной машины обычно занимает несколько минут. В свою очередь, размеры контейнеров в основном исчисляются в мегабайтах, а их запуск происходит практически мгновенно [20].

1.2.3 Управление контейнерами

Как и в случае с виртуальными машинами, рост количества контейнеров требует введения механизмов управления и обслуживания. Для виртуальных машин такие механизмы реализованы с помощью облачных систем, для контейнеров – с помощью систем оркестровки. Системы оркестровки контейнеров обеспечивают возможность развёртывания, масштабирования и управления контейнеризованными приложениями. Примерами таких систем могут служить Kubernetes и Docker Swarm.

1.2.4 Автоматическое масштабирование

С развитием облачного приложения растет количество виртуальных машин и контейнеров, выполняющий экземпляры этого приложения. В больших проектах счет виртуальных узлов идет на тысячи. Соответственно, необходимо каким-то автоматизировать процесс управления вычислительными ресурсами в зависимости от текущей нагрузи на облачное

приложение. В облачных системах и системах управления контейнерами существует такой компонент как сервис автоматического масштабирования. Данный компонент изменяет инфраструктуру, на которой развернуто облачное приложение, в соответствии с текущей загрузкой системы. В процессе изучения предметной области были проанализированы наиболее распространенные облачные платформы с открытым исходным кодом: Apache CloudStack, OpenStack, OpenNebula и HPE Eucalyptus [21]. С использованием косвенных признаков: интерфейса конфигурации и пользовательских параметров, кроме того, выполнен анализ облачных систем с закрытым исходных кодом: Amazon AWS и VMware vSphere. Также проанализированы сервисы автоматического масштабирования в системах управления контейнерами с открытым исходным кодом: Kubernetes, Docker Swarm. В процессе анализа сделан вывод, что сервисы автоматического масштабирования в облачных и контейнерных системах функционируют по одному алгоритму. Таким образом, дальнейшее рассмотрение и использование компонентов автоматического масштабирования одновременно и облачных систем, и систем управления контейнерами избыточно. В силу того, что облачные системы в данный момент имеют большее распространение по сравнению с системами управления контейнерами [22], было принято решение в дальнейшей работе опираться на сервисы автоматического масштабирования облачных систем.

Сервис автоматического масштабирования в облачных системах управляет жизненным циклом виртуальных машин. Можно выделить следующие основные функции и возможности данного сервиса:

- создание виртуальных машин из шаблона;
- запуск и останов виртуальных машин;
- установка пороговых значение загрузки виртуальных машин;
- получение значений характеристик функционирования виртуальных

машин.

Под шаблоном понимается конфигурационный файл, описывающий основные параметры виртуальной машины, который используется гипервизором для ее инициализации и выделения физических ресурсов [23]. Существуют следующие ключевые параметры, использующиеся при создании шаблонов виртуальных машин:

- количество виртуальных центральных процессоров (потоков физического центрального процессора);
- объем оперативной памяти;
- образ операционной системы;
- конфигурация дисковой подсистемы;
- конфигурация сетевых интерфейсов.

Во всех рассмотренных системах алгоритм автоматического масштабирования использует мгновенную загрузку аппаратных ресурсов в качестве условия для изменения числа запущенных копий облачного приложения. При достижении заданного верхнего порогового значения мгновенной загрузки происходит запуск очередного вычислительного узла, в результате чего величина загрузки уменьшается [24]. Аналогичный процесс происходит при достижении заданной нижней границы загрузки: число вычислительных узлов уменьшается в соответствии с конфигурацией модуля автоматического масштабирования. Диаграмма активности алгоритма в нотации UML представлена на рис. 2. То есть, используемые алгоритмы не учитывают исторических данных, а также не могут выполнять масштабирование на основании прогнозируемой нагрузки[13]. Это позволяет сделать вывод о низкой эффективности используемых алгоритмов в условиях реальных нагрузок.

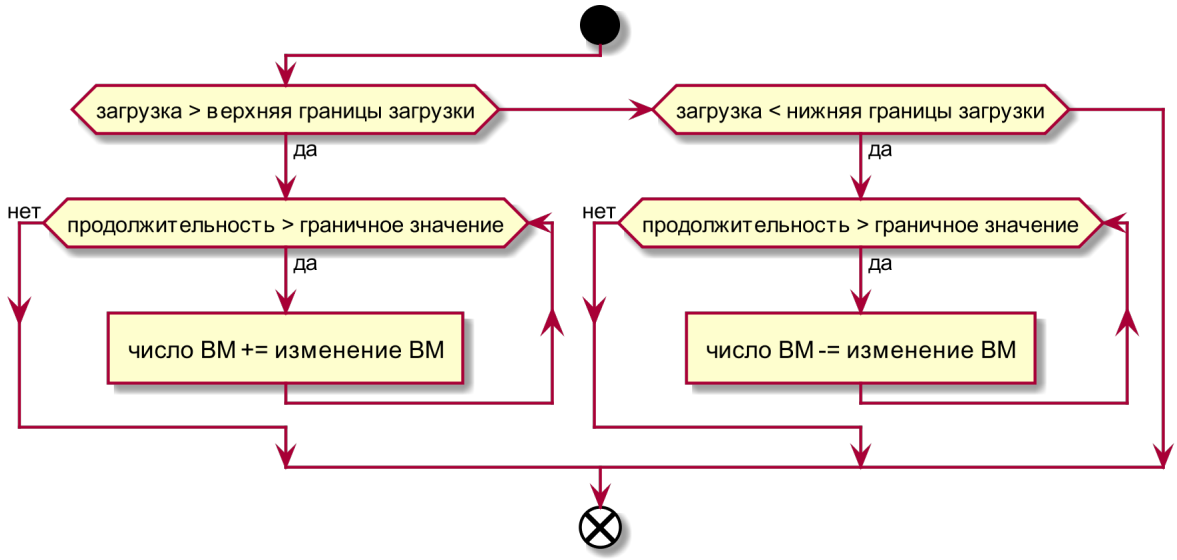


Рис. 2. Диаграмма активности алгоритма автоматического масштабирования

1.3 Разработанный ранее программный комплекс

В ранних трудах автора был разработан программный комплекс для автоматического управления ресурсами облачной вычислительной системы [24]. В качестве облачной платформы использовалось программное обеспечение OpenNebula. Полученное решение добавляет в алгоритм автоматического масштабирования возможность сохранения информации о мгновенном состоянии системы в конкретный момент времени, что позволило ввести учет исторических данных и предсказывать возможные состояния системы в будущем. Разработанный программный комплекс включает в себя следующие модули (рис. 3):

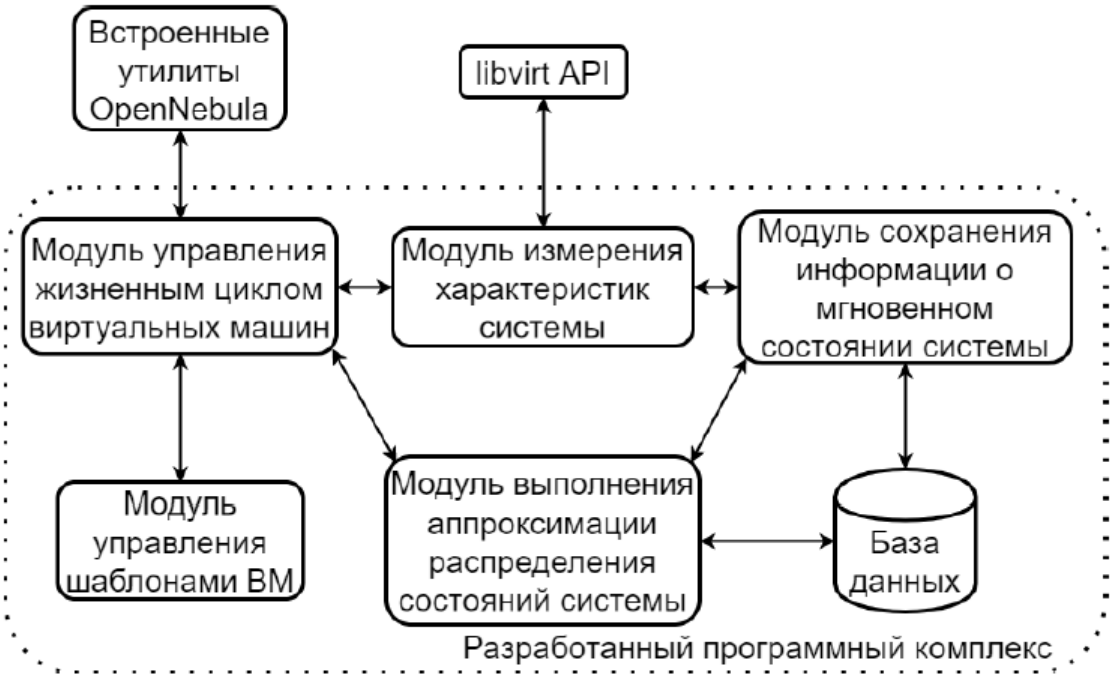


Рис. 3. Разработанный ранее программный комплекс

1. Модуль измерения характеристик. Отвечает за сбор данных о характеристиках функционирования виртуальных машин.

2. Модуль управления жизненным циклом виртуальных машин. Отвечает за запуск и останов виртуальных машин в зависимости от их общей загрузки.

3. Модуль управления шаблонами виртуальных машин. Осуществляет создание виртуальной машины из конкретного заранее созданного шаблона.

4. Модуль сохранения информации о мгновенном состоянии системы. Заносит в базу данных результаты измерений характеристик функционирования виртуальных машин.

5. Модуль выполнения аппроксимации распределения состояний системы. Выполняет линейную аппроксимацию выборки, полученной

с помощью модулей измерения характеристик и модуля сохранения информации о мгновенном состоянии системы [25].

Основываясь на результатах тестирования, можно сделать вывод, что разработанный программный комплекс позволяет добиться снижения среднего времени отклика системы в среднем на 17 %, а величину 95-го перцентиля при максимальных нагрузках на 37 % по с равнению со стандартным решением автоматического масштабирования, представленным в облачной платформе OpenNebula [26].

1.4 Формулировка проблемы

По результатам анализа алгоритмов работы сервисов автоматического масштабирования наиболее популярных облачных систем, а также разработанного автором в ходе ранних исследований программного комплекса, сделан вывод, что алгоритмы, используемые в таких системах имеют ряд недостатков, основным из которых является отсутствие возможности построения прогнозных моделей нагрузочных процессов. Интенсивность пользовательских запросов к системам, расположенным в сети Интернет, изменяется в зависимости от многих факторов. Следовательно, необходимо каким-то образом описывать процессы поступления пользовательских запросов для обеспечения возможности построение прогнозных моделей. Кроме того, процесс поступления пользовательских запросов к крупным облачным приложениям представляет из себя совокупность некоторого числа потоков. Такие потоки могут отличаться не только интенсивностью, но и периодичностью из-за различного географического положения пользователей. Это затрудняет решение задачи проектирования облачной системы. Поэтому возникает необходимость разработки способа описания и моделирования нагрузочных процессов, не обладающих постоянной интенсивностью и представляющими из себя совокупность нескольких потоков с различными параметрами. Таким образом, наличие

адекватной модели таких систем с учетом характера нагрузки, обеспечит возможность проектирования и управления облачными системами, а также позволит оптимизировать распределение нагрузки и количество вычислительных узлов.

2 Аналитическое представление нестационарных процессов

2.1 Нестационарные процессы в реальных системах

Процессы, протекающие в реальных системах, обладают определенной степенью случайности [27]. Это означает, что характеристики таких процессов непостоянны и меняются в зависимости от некоторого набора факторов. На практике случайность может проявляться практически во всех процессах: процессы поступления запросов в некоторую систему (или разного рода нагрузочные процессы), процесс обслуживания пользовательских запросов в этой системе и т.д.

Для определения характера нагрузочных процессов, протекающих в реальных системах, в работе была проанализирована нагрузка на сетевое оборудование крупных систем обмена данными [28]. Для анализа выбраны следующие крупные точки обмена данными, предоставляющие информацию об объемы передаваемых данных: M-9 (MSK-IX), OST Dataline, StoreData, LinxTelecom, TrustInfo и DataSpace. Данные сервисы представляют из себя наивысший уровень иерархии сети Интернет и предназначены для высокоскоростного обмена данными между географически распределенными регионами. На рис. 4 изображен график сетевой активности крупнейшей в России точки обмена данными MSK-IX.

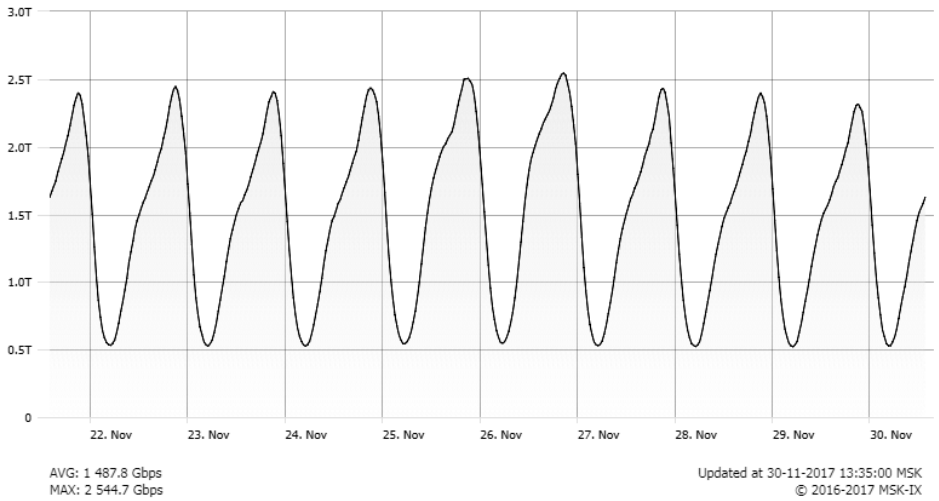


Рис. 4. Суммарный трафик, передаваемый через узлы MSK-IX

По представленному графику видно, что процесс передачи данных в сети Интернет имеет нестационарный характер, то есть его характеристики зависят от времени суток.

2.2 Нестационарность в моделях реальных систем

В основных подходах к моделированию реальных систем различного назначения случайность процессов, протекающих в них, описывается законами распределений случайных величин (или законами распределения вероятностей). Закон распределения описывается функцией распределения [29]. Функция распределения вероятностей – это математическая функция, которая описывает вероятность получения возможных значений, которые может принять случайная величина [30]. Примерами законов распределения вероятностей могут служить экспоненциальный закон, закон Пуассона, закон Гаусса и т.д.

Можно выделить два основных подхода к моделированию систем. Первый подход предполагает формирование математического представления системы путем использования аналитических зависимостей, позволяющих рассчитывать характеристики функционирования системы на основе набора входных параметров. Основным преимуществом такого

подхода является возможность проведения детального анализа процессов, протекающих в системе, при практически любых возможных значениях входных параметров [31]. Также, в случае приемлемой вычислительной сложности полученных аналитических зависимостей, можно добиться достаточно быстрого получения результатов моделирования с низкой степенью частности. Однако успешность и временные затраты, в случае математического моделирования, сильно зависят от степени сложности исследуемой системы, т.е. от количества математических манипуляций и формы используемых аналитических зависимостей . Например, в некоторых случаях, небольшие изменения в структуре уравнения могут потребовать огромных изменений в математических методах.

Второй подход предполагает создание цифрового прототипа исследуемой системы. Данный прототип подразумевает под собой совокупность некоторых объектов, имитирующих поведение объектов реальной системы. Такие объекты, в типовом случае, реализуются с помощью программных модулей, функционирующих на основе определенных алгоритмов, позволяющих в последствии рассчитывать характеристики функционирования системы с помощью численно-аналитических методов. Основным преимуществом такого подхода является высокая степень универсальности, позволяющая строить модели практически любой сложности [32]. Существующие программный комплексы, формирующие среды имитационного моделирования, предоставляют широкий спектр программных реализаций объектов реальных систем, что существенно упрощает процесс построения имитационных моделей. Однако процесс моделирования в случае сложных систем может оказаться достаточно требовательным к вычислительным ресурсам, что является критичным в ряде случаев. Кроме того, имитационное моделирование обладает высокой степенью частности результатов, определяющих значения и зависимости характеристик только в некоторых точках. Для снижения частности результатов

и статистической погрешности необходимо выполнить большое количество экспериментов, что может сильно повлиять на трудоемкость и продолжительность процесса моделирования.

На практике, при моделировании реальных систем используется ряд различных допущений [33]. Целью использования таких допущений является упрощение моделей исследуемых систем, позволяющее вычислять характеристики их функционирования при сохранении приемлемой степени адекватности. Основных из таких допущений является предположение о стационарности случайных процессов, протекающих в моделируемых системах [34]. Следовательно, такое допущение подразумевает под собой использование стационарных законов распределения, что, в свою очередь, позволяется использовать широко известные аналитические методы теории массового обслуживания. Однако, как было показано на рис. 4, процессы, протекающие в реальных системах, не обладают свойством стационарности. Это объясняется различными факторами [35], влияющими на характер потоков пользовательских запросов: время суток, время года, календарные события, личные предпочтения пользователей и т.д. Следовательно, возникает вопрос адекватности результатов моделирования, полученных путем использования стационарных распределений. Кроме того, как говорилось ранее, при проектировании облачных систем для размещения крупных приложений возникает проблема географической распределенности пользователей [36]. Потоки, создаваемые пользователями крупных облачных приложений, могу отличаться различным набором параметров: периодичностью, интенсивностью и т.д. Переход к стационарным процессам при моделировании не позволяет учитывать нестационарность процессов поступления пользовательских запросов к таким приложениям. Следовательно, отсутствует возможность выполнения операций с нестационарными распределениями, описывающими нагрузочные процессы на облачные системы. Такие операции, как сложение

и объединение нескольких потоков пользовательских запросов, заданных нестационарными распределениями, могут позволить более точно предсказывать нагрузку при проектировании и управлении облачными системами. Данная проблема, зачастую, решается с помощью имитационного моделирования. Однако во многих случаях данный подход может быть неприемлем в силу возможной трудоемкости процесса моделирования и частности полученных результатов.

2.3 Классификация нестационарных процессов

В работе предложены следующая классификация нестационарных процессов. В зависимости от характера задающего распределения можно выделить следующие типы нестационарных процессов:

- с изменяющимися параметрами закона распределения;
- с изменяющимся законом распределения, но постоянными параметрами;
- с изменяющимися как законом распределения, так и его параметрами.

В свою очередь, по природе изменений нестационарные процессы делаться на:

- периодические;
- апериодические;
- хаотические.

В результате, в соответствии с предложенной классификацией, а также основными положениями пункта 2.1, сделан вывод, что процессы, определяющие нагрузку на облачные системы, относятся к классу периодических нестационарных процессов с изменяющимися параметрами закона

распределения. Таким образом, дальнейшее исследование целесообразно проводить именно с данным классом процессов.

2.4 Представление и композиция нестационарных распределений

В общем случае, нестационарный процесс может быть задан следующей системой уравнений:

$$\begin{cases} F(x,t) = f1(x, \lambda(t)) \\ \lambda(t) = f2(t) \end{cases}, \tag{1}$$

где $F(x,t)$ – закон распределения вероятностей, $f1(x, \lambda(t))$ – функция, задающая закон распределения вероятностей, $\lambda(t)$ – параметр распределения, $f2(t)$ – функция изменения параметра распределения.

Пусть два нестационарных процесса заданы нестационарными распределениями (2) и (4).

$$\begin{cases} f(x,t) = a(\lambda_f(t), x) \\ \lambda_f(t) = c(t) \end{cases} \tag{2}$$

$$\begin{cases} sum(x,t) = a(\lambda_f(t), x) \otimes \\ \quad \otimes b(\lambda_g(t), x) \\ \lambda_h(t) = c(t) + d(t) \end{cases} \tag{3}$$

$$\begin{cases} g(x,t) = b(\lambda_g(t), x) \\ \lambda_g(t) = d(t) \end{cases} \tag{4}$$

$$\begin{cases} comp(x,t) = (a(\lambda_f(t), x) + \\ \quad + b(\lambda_g(t), x))/2 \\ \lambda_h = c(t) + d(t) \end{cases} \tag{5}$$

Для вычисления суммы этих процессов сделано предположение, что функция плотности распределения может быть представлена как свёртка исходных плотностей распределения (3), а для вычисления их композиции, плотность распределения можно рассчитать как взвешенную смесь с равными весами (5).

Под суммой понимается последовательное сложение двух случайных величин [37]. В общем виды свертка двух плотностей распределения может быть представлена следующей формулой:

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-u) \cdot g(u)du \tag{6}$$

.

На практике последовательное сложение потоков входящих заявок встречается редко, так как в большинстве случаев системы проектируются для возможности обслуживания параллельных и не зависящих друг от друга потоков пользовательских запросов [38]. Кроме того, в случае нестационарности задающих распределений, для расчета свертки может потребоваться большое количество математических манипуляций и использование сложных форм аналитических зависимостей. Таким образом, в силу параллельности, а не последовательности процессов поступления пользовательских запросов, выполнение сложных математических расчетов в случае свертки себя не оправдывает.

Композиция, в свою очередь, подразумевает под собой параллельное сложение двух нестационарных распределений, что гораздо точнее характеризует процессы, протекающие в реальных системах [39].

В обоих случаях, результирующая функция изменения параметра распределения может быть определена как сумма исходных. При этом, средняя и максимальная интенсивность потока заявок может быть найдена как сумма соответствующих значений $\overline{\lambda(t)} = \overline{c(t)} + \overline{d(t)}$ и $\lambda_{max}(t) = c_{max}(t) + d_{max}(t)$

2.5 Проверка адекватности полученных зависимостей

Задачей моделирования на данном этапе работы является проверка адекватности предложенных ранее аналитических зависимостей. Как го-

ворилось ранее, вычисление свертки в реальных системах может привести к неоправданным временным и мощностным затратам [40]. Поэтому решено основное внимание уделить именно параллельному сложению нестационарных процессов, т.е. композиции. Облачные системы относятся к классу систем, функционирующих на основе сетевого взаимодействия [41]. В процессе изучения предметной области выявлено, что при моделировании сетей передачи данных, наиболее часто используемыми распределениями являются экспоненциальное и мультиэкспоненциальное [42]. Таким образом, для проверки полученных зависимостей были выбраны именно эти распределения. Для проведения имитационных экспериментов в среде имитационного моделирования Anylogic Professional 7.0.2 была построена модель, структура которой представлена на рис. 5.

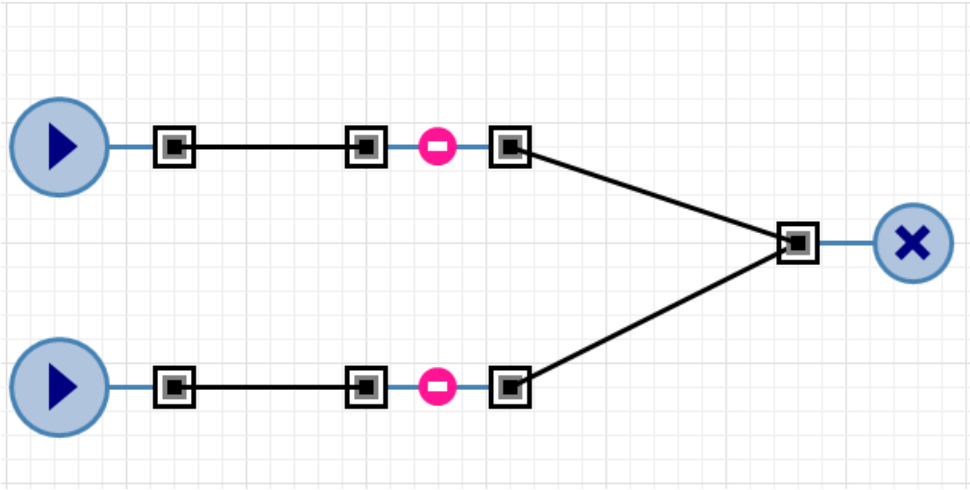


Рис. 5. Имитационная модель композиции распределений

Время между последовательными поступлениями заявок генерируются в соответствии с заданным законом распределения. Блоки задержки необходимы для подсчета времени между прибытиями заявок и расчета статистики. Подсчет суммарной интенсивности и времени между прибытиями заявок происходит в заключительном элементе, объединяющем два исходных потока. Таким образом, к задаваемым параметрам модели относятся следующие величины:

- закон распределения, задающий время между прибытиями заявок в первом потоке F_1;
- закон распределения, задающий время между прибытиями заявок во втором потоке F_2;
- константа или функция, задающая среднюю интенсивность входящего потока заявок в первом потоке λ_1;
- константа или функция, задающая среднюю интенсивность входящего потока заявок во втором потоке λ_2;

В свою очередь, к исследуемым характеристикам относятся:

- среднюю интенсивность суммарного потока заявок λ_c;
- среднее время между последовательными поступлениями заявок в суммарном потоке a_c;

Для каждого набора задаваемых параметров проведено 10 экспериментов и получено 10 усредненных значений, с помощью которых для представленных в данных экспериментах результатов были рассчитаны доверительные интервалы с уровнем доверия 95%: $\Delta\lambda_c = 0,01$, $\Delta a_c = 0,012$.

Таблица 1. Результаты моделирования

F_1	F_2	λ_1	λ_2, мин$^{-1}$	λ_c, мин$^{-1}$	a_c, мин
$\lambda(t)e^{-\lambda(t)x}$	$\lambda(t)e^{-\lambda(t)x}$	0.3333	0.5	0.8333	1.2
$\lambda(t)e^{-\lambda(t)x}$	$\lambda(t)e^{-\lambda(t)x}$	$Asin(t)+B$	$Asin(t)+B$	0.71	0.01425
$\lambda(t)e^{-\lambda(t)x}$	$\lambda(t)e^{-\lambda(t)x}$	$saw(t)$	$saw(t)$	0,708	0.01427

Эксперименты проводились с различными стационарными и нестационарными распределениями. В качестве примера нестационарных распределений взяты экспоненциальные распределения с синусоидальной и пилообразной функцией интенсивностей. Синусоидальная интенсивность,

представленная в таблице 1, задана формулой $\lambda(t) = Asin(t) + B$, где A = 25 мин$^{-1}$ и B = 35 мин$^{-1}$. Пилообразная интенсивность в экспериментах имела такое же среднее значение, как и синусоидальная – 35 мин$^{-1}$. Таким образом, в обоих случаях средняя интенсивность каждого потока равна 35 мин$^{-1}$. По представленным в таблице результатам видно, что с учетом доверительных интервалов можно говорить о корректности суммирования функций и их средних значений для определения параметра результирующего распределения.

2.6 Выводы по главе

1. В результате анализа трафика, передаваемого через крупнейший в России транспортный узел MSK-IX, сделан вывод, что процесс передачи данных в сети Интернет имеет нестационарный характер, то есть его параметры зависят от времени суток.

2. На практике, при моделировании реальных систем, используется ряд различных допущений. Основным из таких допущений является предположение о стационарности параметров входящего потока пользовательских запросов. Данное допущение не позволяет строить адекватные модели реальных систем, так как нагрузочные процессы на них не обладают свойством нестационарности.

3. В силу низкой степени теоретической проработанности с точки зрения теории массового обслуживания исследование нестационарных процессов в достаточной степени затруднено. Поэтому, в первую очередь, необходимо выполнить классификацию нестационарных процессов для обеспечения возможности исследования нестационарных процессов, протекающих в реальных системах.

4. Предложена классификация нестационарных процессов по задающим распределениям и по природе изменения процесса. В соответ-

ствии с предложенной классификацией был сделан вывод, что процессы, определяющие нагрузку на облачные системы, относятся к классу периодических нестационарных процессов с изменяющимися параметрами закона распределения.

5. Предложено аналитическое представление нестационарных процессов и их композиции.

6. В результате имитационных экспериментов сделан вывод о корректности суммирования функций и их средних значений для определения параметра результирующего распределения.

3 Моделирование нестационарных процессов

3.1 Постановка экспериментов и описание модели

Задачей моделирования на данном этапе работы является выявление влияния нестационарных потоков входящих заявок на характеристики систем с очередями, используемых в качестве моделей облачных систем. Таким образом, для начала необходимо определить набор варьируемых параметров и измеряемых величин. Характер изменения интенсивности нагрузочных процессов, представленный на рис. 4, позволяет сделать вывод, что наибольший интерес при моделировании представляют процессы, имеющие синусоидальный характер изменения интенсивности входящего потока пользовательских запросов. Кроме того, в силу представленных ранее особенностей функционирования крупных облачных приложений важной задачей является определение влияния композиции нескольких таких потоков на характеристики функционирования облачных систем. В качестве закона распределения времени между приходом заявок в систему был выбран экспоненциальный закон распределения, так как, как было сказано ранее, при моделировании сетей передачи данных, наиболее часто используемыми распределениями являются экспоненциальное и мультиэкспоненциальное. Таким образом, в общем виде, нестационарный процесс поступления заявок в систему в соответствии с (1) можно представить следующей системой уравнений:

$$\begin{cases} F(x,t) = \lambda(t)e^{-\lambda(t)x} \\ \lambda(t) = Asin(t+\phi_0) + B \end{cases}. \tag{7}$$

Для исследования влияния нестационарных процессов на характеристики функционирования облачных систем в среде имитационного моделирования AnyLogic Professional 7.0.2 была разработана модель облачной системы. Графическое описание разработанной модели представлено

на рис. 6. Модель состоит из двух приложений с раздельными источниками заявок и накопителями ограниченной емкости. За обслуживания поступающих в систему заявок обоих приложений отвечает некоторый набор вычислительных узлов (приборов). В каждый момент времени на вычислительном узле могут обрабатываться заявки только одного приложения. Вычислительная инфраструктура в данной модели изменяется в зависимости от загрузки приборов по приложениям. Управление количеством вычислительных узлов осуществляется с помощью реализованного в модели сервиса автоматического масштабирования. Алгоритм автоматического масштабирования вычислительной инфраструктуры реализован в соответствии с исследованными алгоритмами, используемыми в реальных системах [43]. Масштабирование происходит с заданным периодом времени T_{as} в соответствии с определенными пороговыми значениями суммарной загрузки приборов по приложениям [44]. Таким образом, минимальное время резервирования прибора за приложением равна времени между двумя последовательными выполнениями алгоритма автоматического масштабирования. Если загрузка превышает верхнее пороговое значение P_u, то происходит добавление узлов в вычислительную инфраструктуру. В случае достижения нижнего порогового значения P_l происходит исключение узлов из вычислительного процесса. Количество добавляемых или исключаемых узлов за один проход процесса автоматического масштабирования принято равным 1, как и в типовой настройке сервиса автоматического масштабирования в реальных системах [45].

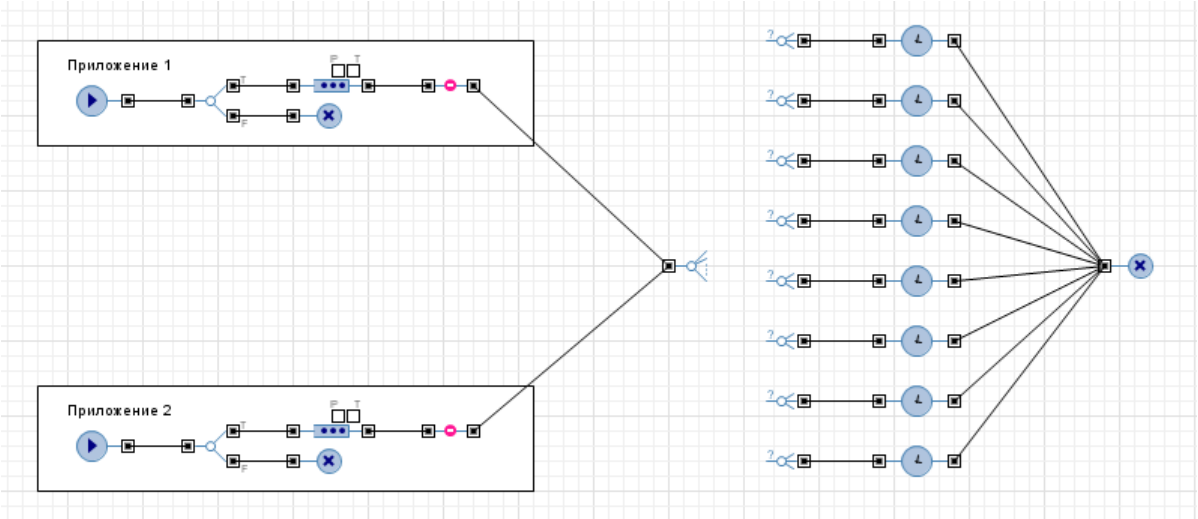


Рис. 6. СМО, используемая в качестве модели облачной системы

К задаваемым параметрам модели, в первую очередь, можно отнести следующие величины:

- закон распределения, задающий время между прибытиями заявок к первому приложению F_1;
- закон распределения, задающий время между прибытиями заявок ко второму приложению F_2;
- функция, задающая интенсивность входящего потока заявок к первому приложению λ_1;
- функция, задающая интенсивность входящего потока заявок ко второму приложению λ_2;
- ёмкость накопителя первого приложения H_1;
- ёмкость накопителя второго приложения H_2;
- закон распределения, задающий время обслуживания заявок в конкретном приборе G;

- среднее время обслуживания заявок в конкретном приборе b;
- количество приборов в вычислительной инфраструктуре N;
- время между двумя последовательными выполнениями алгоритма автоматического масштабирования T_{as}
- верхний порог загрузки по приложению P_u;
- нижний порог загрузки по приложению P_l.

Среди исследуемых характеристик наибольший интерес представляют:

- средняя загрузка приборов по первому приложению ρ_1;
- средняя загрузка приборов по второму приложению ρ_2;
- средняя длина очереди первого приложения l_1;
- средняя длина очереди второго приложения l_2;
- среднее время ожидания заявок первого приложения w_1;
- среднее время ожидания заявок второго приложения w_2;
- вероятность потери заявок первого приложения π_1
- вероятность потери заявок второго приложения π_2
- среднее количество свободных приборов $\widetilde{N}$.

Целью исследования и моделирования на данном этапе работы является проверка адекватности перехода к стационарным распределениям при моделировании систем с очередями, используемых в качестве моделей облачных систем.

Значения исходных параметров системы представлены в таблице 2. Эксперимент был разбит на два этапа. На первом этапе на вход модели подавалась нестационарная нагрузка. Во второй части эксперимента на вход модели подавалась стационарная нагрузка со средней интенсивность, равной B, т.е. средней интенсивностью нестационарной нагрузки. Признаком завершения одного конкретного эксперимента было выбрано суммарное количество пройденных через систему заявок [32]. Экспериментальным путём получено, что при прохождении через систему 50 миллионов заявок удается добиться изменения характеристик в пределах 0,01 %, таким образом минимизировав влияние переходных и стохастических процессов на систему. Эксперименты проводились с разными соотношениями фаз интенсивностей потоков заявок к каждому из приложений: противофазные, синфазные, со смещением.

Таблица 2. Исходные параметры модели

Параметр	**Значение**
F_1	$F(x,t) = \lambda(t)e^{-\lambda(t)x}$
F_2	$F(x,t) = \lambda(t)e^{-\lambda(t)x}$
H_1	1000
H_2	1000
G	$F(x,t) = \lambda e^{-\lambda x}$
b	0.08 мин
N	8
T_{as}	5 мин
P_u	0.92
P_l	0.45

3.2 Результаты имитационных экспериментов с синусоидальной интенсивностью

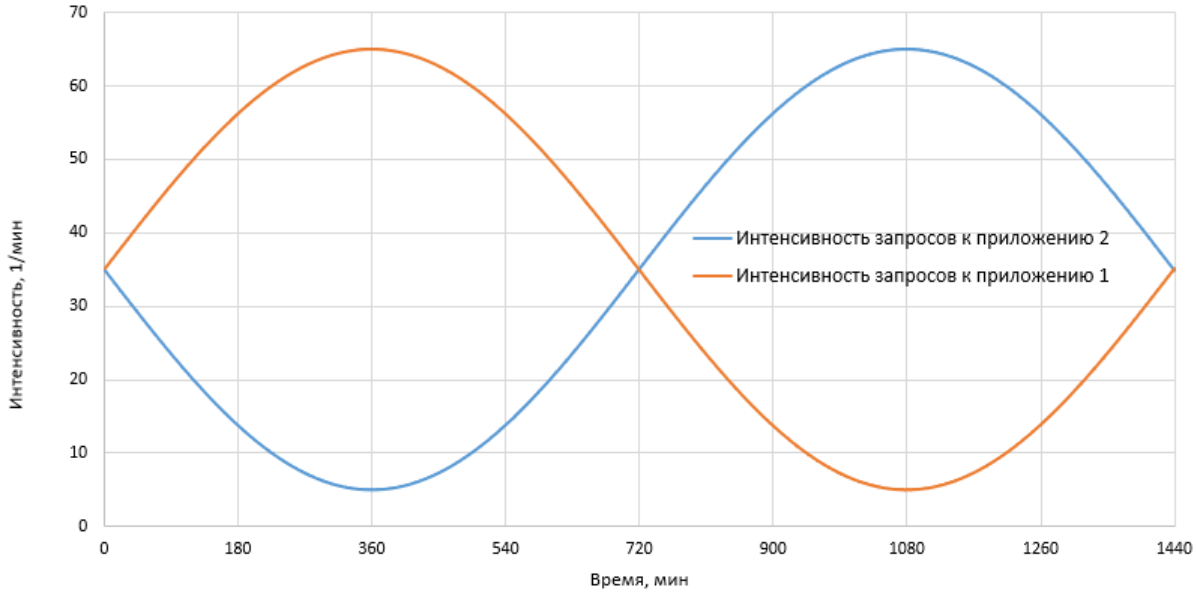


Рис. 7. Противофазные интенсивности

В данном эксперименте на вход модели подавались потоки заявок с противофазными интенсивностями, заданными синусоидальными функциями. Таким образом, интенсивность входящих пользовательских запросов в общем случает представляет собой уравнение гармонического колебания $\lambda(t) = Asin(t + \phi_0) + B$. Период нестационарной нагрузки – 1440 мин (24 часа). Амплитуда нестационарной нагрузки – 25 запросов в минуту. В эксперименте осуществлялось варьирование средней интенсивности B от 35 до 50 запросов в минуту. На рис. 7 представлен пример подаваемых на вход модели нестационарных интенсивностей со средней интенсивностью 35 запросов в секунду.

Для каждого набора задаваемых параметров проведено 10 экспериментов и получено 10 усредненных значений, с помощью которых для представленных в данных экспериментах результатов были рассчитаны доверительные интервалы с уровнем доверия 95%.

Таблица 3. Доверительные интервалы (нестационарная интенсивность)

B, мин$^{-1}$	ρ	l	w, мин	π	$\widetilde{N}$
30	$\pm 0,002$	$\pm 0,03$	$\pm 0,002$	$\pm 0,0$	$\pm 0,001$
35	$\pm 0,002$	$\pm 5,5$	$\pm 0,1$	$\pm 0,0$	$\pm 0,002$
40	$\pm 0,002$	$\pm 8,87$	$\pm 0,12$	$\pm 0,001$	$\pm 0,01$
45	$\pm 0,003$	$\pm 10,22$	$\pm 0,32$	$\pm 0,002$	$\pm 0,012$
50	$\pm 0,004$	$\pm 11,1$	$\pm 0,41$	$\pm 0,003$	$\pm 0,014$

Таблица 4. Доверительные интервалы (стационарная интенсивность)

B, мин$^{-1}$	ρ	l	w, мин	π	$\widetilde{N}$
30	$\pm 0,001$	$\pm 0,005$	$\pm 0,001$	$\pm 0,0$	$\pm 0,001$
35	$\pm 0,001$	$\pm 0,007$	$\pm 0,001$	$\pm 0,0$	$\pm 0,002$
40	$\pm 0,001$	$\pm 0,008$	$\pm 0,003$	$\pm 0,0$	$\pm 0,002$
45	$\pm 0,001$	$\pm 0,01$	$\pm 0,004$	$\pm 0,0$	$\pm 0,003$
50	$\pm 0,001$	$\pm 5,28$	$\pm 0,02$	$\pm 0,0$	$\pm 0,005$

Таблица 5. Результаты моделирования (противофазная синусоидальная нестационарная интенсивность)

B, мин$^{-1}$	ρ_1	l_1	w_1, мин	π_1	ρ_2	l_2	w_2, мин	π_2	$\widetilde{N}$
30	0,649	1,342	0,045	0	0,652	1,512	0,05	0	3,201
35	0,744	77,311	2,217	0,004	0,746	86,184	2,476	0,005	2,427
40	0,794	217,53	5,614	0,032	0,794	221,698	5,735	0,033	1,808
45	0,846	357,366	8,64	0,082	0,846	358,256	8,688	0,084	1,389
50	0,873	495,358	11,309	0,125	0,874	499,07	11,451	0,128	1,009

Таблица 6. Результаты моделирования (стационарная интенсивность)

B, мин$^{-1}$	ρ_1	l_1	w_1, мин	π_1	ρ_2	l_2	w_2, мин	π_2	$\widetilde{N}$
30	0,622	0,658	0,022	0	0,622	0,668	0,022	0	3,201
35	0,7	1,006	0,029	0	0,7	1,005	0,029	0	2,4
40	0,8	2,391	0,06	0	0,8	2,399	0,06	0	1,6
45	0,9	7,123	0,158	0	0,9	7,144	0,159	0	0,799
50	0,999	512,602	10,266	0,001	0,999	456,126	9,132	0,001	0,009

По представленным результатам видно, что при переходе от нестационарных распределений к стационарным имеет место различие во все исследуемых характеристиках. Особенно значимые результаты получены в условиях высоких нагрузок, близких к 1. При интенсивности запросов, близкой 50 мин$^{-1}$ при стационарной интенсивности наблюдается резкий рост таких характеристик, как w и l. Это связано с тем, что система приближается к переходу в перегруженное состояние и не справляется с потоком пользовательских запросов, от чего и возникают потери [46]. Однако при нестационарной интенсивности с таким же средним значением средняя загрузка системы меньше единицы. Это означает, что система на длительном промежутке времени не переходит в перегруженное состояние. Но стоит отметить, что величины w и l имеют гораздо большее значение. Данный результат объясняется тем, что алгоритм автоматического масштабирования имеет некоторый период. Ввиду того, что нагрузка на приложения имеет синусоидальных характер, то имеют место возрастания и убывания интенсивностей запросов [47]. Для того, чтобы зарезервировать под приложение какой-то конкретный узел, сервис автоматического масштабирования должен выбрать его из списка свободных. В случае отсутствия свободных узлов данный сервис отказывает приложению в выдаче ресурсов, так как в большинстве реальных систем абсолютный приоритет у приложений в дисциплине обслуживания отсутствует, т.е. отсутствует возможность конфигурации таких приоритетов с помощью средств сервиса автоматического масштабирования [48]. Таким образом, в условиях меняющихся нагрузок, может возникнуть ситуация, когда в момент пика нагрузки на какое-то приложение, сервис автоматического масштабирования выдал испытывающему высокую нагрузку приложению большое количество приборов. В силу противофазности синусоидальных интенсивностей, в тоже время, начала возрастать нагрузка на другое приложение, но сервис автоматического масштабирования не смо-

жет зарезервировать ему вычислительный узел, что негативно скажется на значениях w, l и π. Кроме того, из-за наличия периода выполнения алгоритма автоматического масштабирования можно сделать предположение, что слишком большой период приводит к недостаточно реактивной реакции системы на изменяющиеся нагрузочные процессы [49]. Следовательно, необходимо выявить зависимость характеристик функционирования системы от периода выполнения автоматического масштабирования.

3.3 Результаты имитационных экспериментов с синусоидальной интенсивностью и варьированием периода автоматического масштабирования

Таблица 7. Результаты моделирования (противофазная синусоидальная нестационарная интенсивность, первое приложение)

B, мин$^{-1}$	T_{as}, мин	ρ_1	l_1	w_1, мин	π_1
30	5	0,649	1,342	0,045	0
35	5	0,744	77,311	2,217	0,004
40	5	0,794	217,53	5,614	0,032
45	5	0,846	357,366	8,64	0,082
50	5	0,873	495,358	11,309	0,125
30	3	0,644	1,202	0,04	0
35	3	0,756	72,46	2,073	0,001
40	3	0,812	251,336	6,528	0,038
45	3	0,861	320,52	7,59	0,062
50	3	0,884	488,382	11,071	0,118
30	1	0,642	1,014	0,034	0
35	1	0,745	3,992	0,114	0
40	1	0,847	112,627	2,82	0,002
45	1	0,904	314,362	7,252	0,037
50	1	0,949	491,159	10,462	0,061

Таблица 8. Результаты моделирования (противофазная синусоидальная нестационарная интенсивность, второе приложение)

B, мин$^{-1}$	T_{as}, мин	ρ_2	l_2	w_2, мин	π_2
30	5	0,652	1,512	0,05	0
35	5	0,746	86,184	2,476	0,005
40	5	0,794	221,698	5,735	0,033
45	5	0,846	358,256	8,688	0,084
50	5	0,874	499,07	11,451	0,128
30	3	0,646	1,234	0,041	0
35	3	0,755	74,075	2,119	0,001
40	3	0,813	255,926	6,668	0,04
45	3	0,861	323,132	7,681	0,065
50	3	0,887	501,399	11,452	0,124
30	1	0,642	1,012	0,034	0
35	1	0,745	4,216	0,12	0
40	1	0,848	116,98	2,932	0,003
45	1	0,904	315,467	7,289	0,038
50	1	0,95	494,542	10,556	0,063

Таблица 9. Результаты моделирования (противофазная синусоидальная нестационарная интенсивность, среднее число свободных приборов)

B, мин$^{-1}$	30	35	40	45	50	30	35	40	45	50	30	35	40	45	50
T_{as}, мин	1	1	1	1	1	3	3	3	3	3	5	5	5	5	5
$\widetilde{N}$	3,201	2,427	1,808	1,389	1,009	3,198	2,407	1,849	1,257	0,969	3,199	2,4	1,613	1,07	0,497

Таблица 10. Результаты моделирования (стационарная интенсивность, первое приложение)

B, мин$^{-1}$	T_{as}, мин	ρ_1	l_1	w_1, мин	π_1
30	5	0,622	0,658	0,022	0
35	5	0,7	1,006	0,029	0
40	5	0,8	2,391	0,06	0
45	5	0,9	7,123	0,158	0
50	5	0,999	512,602	10,266	0,001
30	3	0,644	0,873	0,029	0
35	3	0,701	1,027	0,029	0
40	3	0,8	3,401	0,085	0
45	3	0,9	7,123	0,158	0
50	3	0,999	483,555	9,68	0,001
30	1	0,678	1,186	0,04	0
35	1	0,745	1,544	0,044	0
40	1	0,721	5,08	0,127	0
45	1	0,903	37,77	0,839	0
50	1	0,999	523,183	10,479	0,002

Таблица 11. Результаты моделирования (стационарная интенсивность, второе приложение)

B, мин$^{-1}$	T_{as}, мин	ρ_2	l_2	w_2, мин	π_2	$\widetilde{N}$
30	5	0,622	0,668	0,022	0	3,201
35	5	0,7	1,005	0,029	0	2,4
40	5	0,8	2,399	0,06	0	1,6
45	5	0,9	7,144	0,159	0	0,799
50	5	0,999	456,126	9,132	0,001	0,009
30	3	0,645	0,88	0,029	0	3,201
35	3	0,701	1,034	0,03	0	2,4
40	3	0,8	3,234	0,081	0	1,601
45	3	0,9	7,08	0,157	0	0,8
50	3	0,999	461,568	9,241	0,001	0,01
30	1	0,678	1,184	0,039	0	3,2
35	1	0,721	1,536	0,044	0	2,4
40	1	0,806	5,143	0,129	0	1,6
45	1	0,901	31,604	0,703	0	0,804
50	1	0,999	488,602	9,781	0,001	0,009

Таблица 12. Результаты моделирования (стационарная интенсивность, среднее число свободных приборов)

B, мин$^{-1}$	30	35	40	45	50	30	35	40	45	50	30	35	40	45	50
T_{as}, мин	1	1	1	1	1	3	3	3	3	3	5	5	5	5	5
$\widetilde{N}$	3,201	2,4	1,6	0,799	0,009	3,201	2,4	1,601	0,8	0,01	3,2	2,4	1,6	0,804	0,009

По представленным результатам можно сделать вывод, что уменьшение периода выполнения алгоритма автоматического масштабирования в случае нестационарности входного потока заявок позволяет добиться улучшения таких характеристика как l и w. Это объясняется улучшением реакции системы на возникающие перегрузки. Более своевременное добавление узлов в вычислительную инфраструктуру позволяет обработать большее количество заявок, находящихся в накопителях. Кроме того, в случае нестационарности можно наблюдать увеличение загрузки прибо-

ров и рост количества свободных вычислительных узлов. Данный результаты объясняются всё тем же улучшением реакции системы на изменения интенсивности входящих запросов. Поскольку в данном эксперименте синусоидальные интенсивности находились в противофазе, то важным свойством системы является как можно более своевременное освобождение узлов от выполнения одного приложения, интенсивность запросов к которому начинает снижаться, и передача их другому приложению, обрабатывающему поток заявок с уже растущей интенсивностью. Повышение количества свободных вычислительных узлов при уменьшении l и w свидетельствует о более эффективном управлении вычислительной инфраструктурой, что в данном эксперименте достигается путём снижения периода выполнения алгоритма автоматического масштабирования.

В случае стационарности входного потока наблюдается ухудшение показателей l и w. Данные результаты, на первый взгляд, не совсем очевидны. Однако стоит учитывать, что принятие решения о включении и исключении узлов из вычислительного процесса происходит на основе верхнего и нижнего порогов загрузки по приложениям. В представленных экспериментах значения ρ_u и ρ_l были равны 0,45 и 0,92 соответственно. Во время срабатывания алгоритма автоматического масштабирования происходит подсчет загрузки, начиная с предыдущего выполнения алгоритма и заканчивая текущим, т.е. за период T_{as}. На рис. 8 и 9 изображен характер изменения измеряемой во время выполнения алгоритма автоматического масштабирования загрузки.

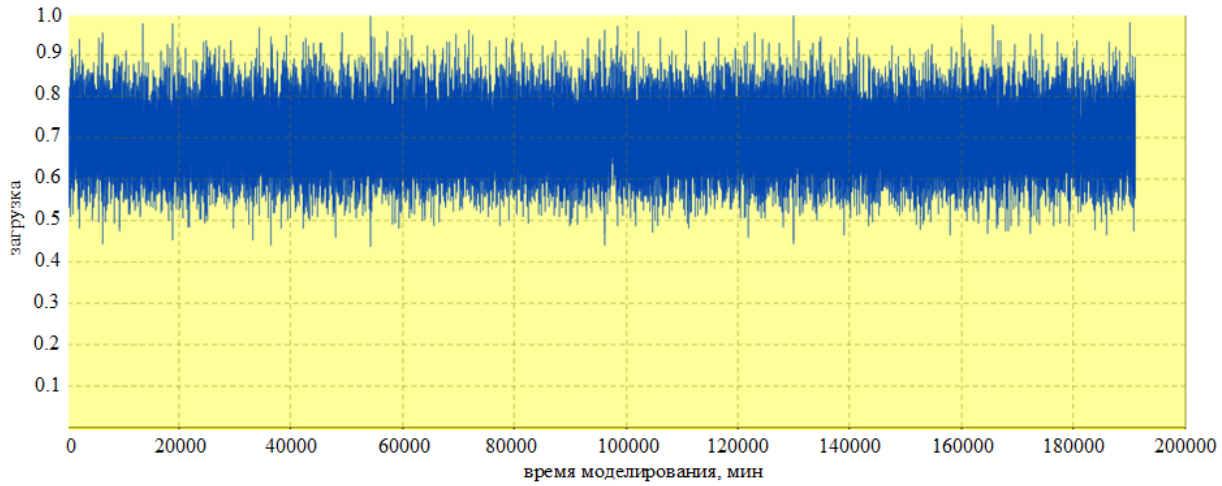


Рис. 8. Изменение измеряемой загрузки при T_{as} = 5 мин

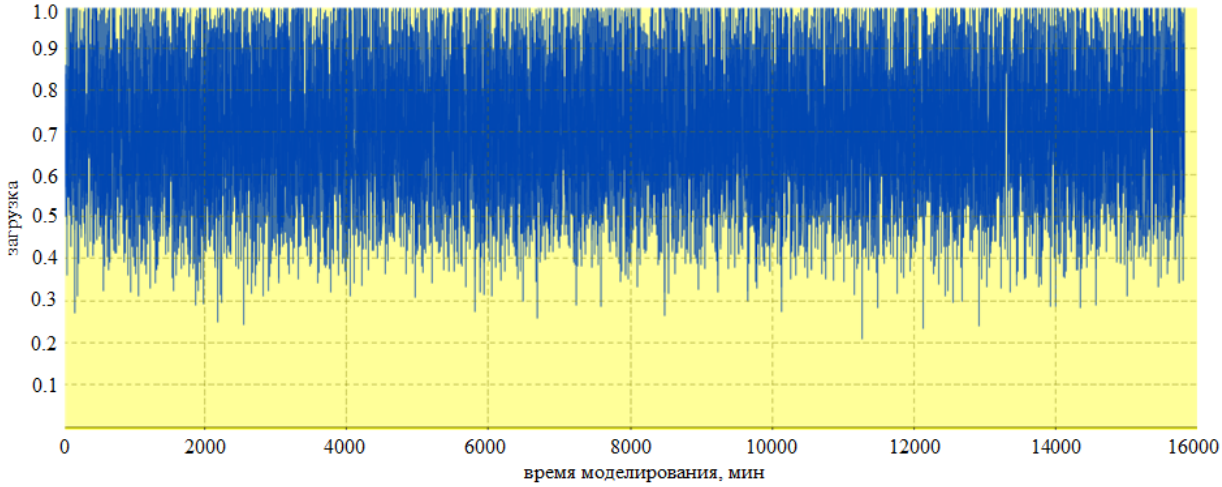


Рис. 9. Изменение измеряемой загрузки при T_{as} = 1 мин

По представленным график видно, что при уменьшении периода автоматического масштабирования увеличивается отклонение загрузки от средней величины. Таким образом, величина загрузки часто на короткий промежуток времени становится > ρ_u, либо < ρ_l. Такие показатели вызывают срабатывание алгоритма автоматического масштабирования. Однако в силу кратковременности превышения пороговых значений происходит обратный процесс, и сервис автоматического масштабирования приводит систему в состояние, в котором она находилась ранее.

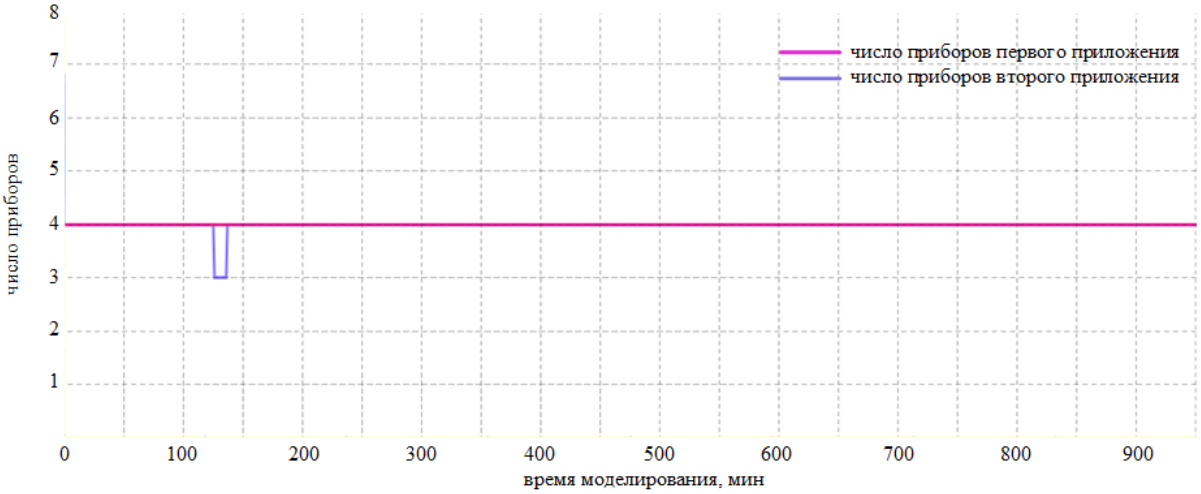


Рис. 10. Изменение количества приборов при T_{as} = 5 мин

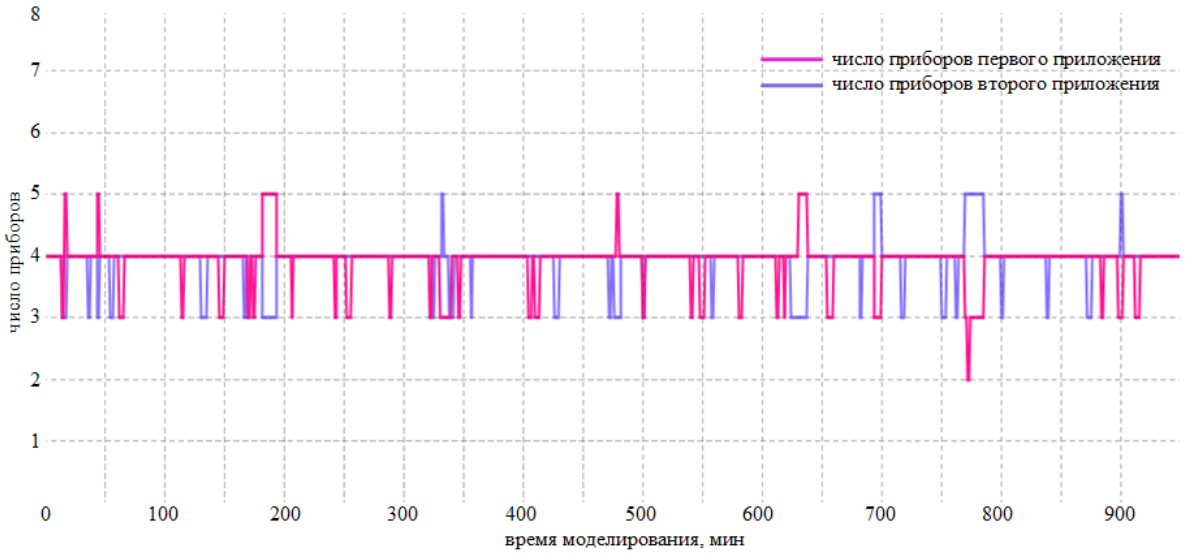


Рис. 11. Изменение количества приборов при T_{as} = 1 мин

На рис. 10 и 11 представлен характер изменения количества приборов, выделенных каждому приложению, от времени моделирования. Видно, что при T_{as} = 5 минут число приборов практически все время остается неизменным. А при T_{as} = 1 минуту происходит постоянное добавление или исключение вычислительных узлов по одному. В итоге, данное поведение способствует ухудшению таких характеристик как l и w, так как, в большинстве случаев, производительность системы становиться ниже при кратковременном исключении узлов.

3.4 Выводы по результатам моделирования

1. Замена нестационарных вероятностных распределений стационарными при условии равных средних интенсивностей входящих потоков пользовательских запросов привод к ощутимым отличиям в характеристиках функционирования разработанной модели облачной системы. Данные результаты, в первую очередь, связаны с тем, что нестационарность потоков пользовательских запросов напрямую коррелирует с их суточной активностью, которая характеризуется синусоидальной функцией, т.е., в общем виде, может быть представлена с помощью уравнения гармонического колебания. Интенсивность такого характера имеет ряд свойств, главным из которых является периодическая смена пиков и спадов. Именно это свойство оказывает наибольшее влияние на характеристики функционирования облачных систем и доказывается неадекватность использования стационарных распределений.

2. Наличие пиков и спадов позволяет системе не переходить в постоянно перегруженное состояние даже при такой средней интенсивности запросов, при которой, в случае использования стационарных распределений, наступает постоянная перегрузка. Однако в данном случае, при нестационарности, не обходится без увеличения количества потерянных заявок при приближении к максимально возможной нагрузке. Это связано с ростом синусоидальной нагрузки. Потери зависят от величины периода автоматического масштабирования: чем меньше период, тем быстрее система реагирует на изменение интенсивности запросов, и тем меньше становится количество потерянных заявок. С другой стороны, величина потерь отлична от 0 в ряде экспериментов из-за отсутствия свободных вычислительных узлов в момент роста интенсивности запросов к одному из прило-

жений. Отсутствие свободных приборов связано с тем, что, в случае противофазных синусоидальных интенсивностей, система не успевает обработать заявки одного из приложений, которое испытывало пиковую нагрузку, и освободить вычислительные узлы для второго приложения, интенсивность запросов к которому начала возрастать.

3. Характеристики функционирования облачных систем напрямую зависят от периода автоматического масштабирования. Меньшая величина периода позволяет ускорить реакцию системы на изменения нестационарной интенсивности, что положительным образом сказывается на среднюю длину очереди, среднее время ожидания и количество потерянных заявок. Однако в случае стационарной интенсивности слишком малая величина периода автоматического масштабирования может ухудшить ситуацию, так как измерение загрузки будет производится с недостаточной точностью, что может привести к некорректному добавлению или исключению вычислительных узлов.

Заключение

В работе были получены следующие результаты.

1. С помощью подхода TOGAF формализованы архитектурные уровни, лежащие в основе функционирования любых облачных сервисов и систем, начиная с физического расположения вычислительных ресурсов и заканчивая целевым состоянием системы. Данный результат предоставляет возможность анализа и реализации программных модулей для автоматического управления вычислительными ресурсами облачных приложений.

2. Предложена классификация нестационарных распределений в зависимости от характера задающего распределения и от природы изменений. Полученная классификация позволяет проводить исследования нестационарных процессов, протекающих в реальных системах.

3. Проведен анализ характера изменения нагрузочных процессов на сетевые узлы, являющиеся крупнейшими в России точками обмена данными. Определено, что характер изменения интенсивности поступления сетевых пакетов носит синусоидальный характер, что обусловлено суточной активностью пользователей различных систем. На основе полученной ранее классификации был сделан вывод, что процессы, определяющие нагрузку на такие системы, относятся к классу периодических нестационарных процессов.

4. Сформулирован способ аналитического описания нестационарных распределений, основанный на функции плотности распределения вероятности и функции изменения параметра распределения.

5. Предложен способ композиции нестационарных процессов. Выявлено, что для вычисления суммы таких процессов функция плот-

ности распределения может быть представлена как свёртка исходных плотностей распределения, а для вычисления их композиции, плотность распределения можно рассчитать как взвешенную смесь с равными весами. В обоих случаях результирующая функция изменения параметра распределения может быть определена как сумма исходных, ровно как и средняя и максимальная интенсивности. Полученные аналитические зависимости могут быть использованы при проектировании и управлении облачными системами, нагрузочные процессы на которые не обладают свойством стационарности и состоят из нескольких потоков с различными функциями изменения параметра распределения.

6. Правомерность полученных аналитических зависимостей подтверждена в ходе многочисленных имитационных экспериментов.

7. Разработана имитационная модель облачной системы, позволяющая проводить анализ влияния характера нагрузочных процессов на характеристики систем с динамическим перераспределением ресурсов.

8. В ходе проведения имитационных экспериментов с использованием разработанной модели облачной системы выявлено, что замена нестационарных вероятностных распределений стационарными при условии равных средних интенсивностей входящих потоков пользовательских запросов приводит к ощутимым отличиям в характеристиках функционирования разработанной модели облачной системы. Наличие пиков и спадов позволяет системе не переходить в постоянно перегруженное состояние даже при такой средней интенсивности запросов, при которой, в случае использования стационарных распределений, наступает постоянная перегрузка.

Полученные аналитические зависимости, а также результаты имита-

ционных экспериментов могут быть использованы при проектировании и управлении облачным системами различного назначения. Учет нестационарности нагрузочных процессов, а также их композиции, позволит не только облегчить процесс проектирования облачных систем, но и обеспечить снижение необходимого количества вычислительных ресурсов для решения конкретных задач при сохранении соглашения об уровне предоставления услуг.

Список литературы

1. Столлингс В. Современные компьютерные сети. – СПб.: Питер, 2003. – 820 с.

2. Zhmylev S.A., Martynchuk I.G., Kireev V.I. Analytical methods of nonstationary processes modeling // CEUR Workshop Proceedings. Т. 2344. 2019.

3. Data flow management and compliance in cloud computing / Jatinder Singh, Julia Elizabeth Powles, Thomas Pasquier [и др.]. 2015.

4. Павловский Ю. Н. Имитационное моделирование. – М.: издательский центр «Академия», 2008. – 237 с.

5. Improving resource utilisation in the cloud environment using multivariate probabilistic models / Sijin He, Li Guo, Moustafa Ghanem [и др.] // 2012 IEEE Fifth International Conference on Cloud Computing / IEEE. 2012. С. 574–581.

6. Khazaei Hamzeh, Misic Jelena, Misic Vojislav B. Performance analysis of cloud computing centers using m/g/m/m+ r queuing systems // IEEE Transactions on parallel and distributed systems. 2012. Т. 23, № 5. С. 936–943.

7. Алиев Т. И. Основы моделирования дискретных систем. – СПб.: СПбГУ ИТМО, 2009. – 363 с.

8. Риз Джордж. Облачные вычисления // СПб.: БХВ-Петербург. 2011. Т. 278.

9. Grossman R.L. The case for cloud computing // IT professional. 2009. Т. 11, № 2. С. 23–27.

10. Antonopoulos Nick, Gillam Lee. Cloud computing. Springer, 2010.

11. Cloud computing—The business perspective / Sean Marston, Zhi Li, Subhajyoti Bandyopadhyay [и др.] // Decision support systems. 2011. Т. 51, № 1. С. 176–189.

12. Cloud-trust—A security assessment model for infrastructure as a service (IaaS) clouds / Dan Gonzales, Jeremy M Kaplan, Evan Saltzman [и др.] // IEEE Transactions on Cloud Computing. 2017. Т. 5, № 3. С. 523–536.

13. Киреев В.Ю., Мартынчук И.Г., Жмылёв С.А. Аналитические методы исследования нестационарных процессов // Альманах научных работ молодых ученых Университета ИТМО. 2018. Т. 2. С. 258–261.

14. Scalable distributed computing hierarchy: Cloud, fog and dew computing / Karolj Skala, Davor Davidovic, Enis Afgan [и др.] // Open Journal of Cloud Computing (OJCC). 2015. Т. 2, № 1. С. 16–24.

15. Portnoy Matthew. Virtualization essentials. John Wiley & Sons, 2012. Т. 19.

16. Rafaels Ray J. Cloud Computing: From Beginning to End. CreateSpace Independent Publishing Platform, 2015.

17. Rodriguez Ismael, Llana Luis, Rabanal Pablo. A General Testability Theory: Classes, properties, complexity, and testing reductions // IEEE Transactions on software engineering. 2014. Т. 40, № 9. С. 862–894.

18. Jiang Qiqi, Qin Jianjun, Kang Lele. A literature review for open source software studies // International Conference on HCI in Business / Springer. 2015. С. 699–707.

19. Huizinga Dorota, Kolawa Adam. Automated defect prevention: best practices in software management. John Wiley & Sons, 2007.

20. O'Connor Patrick. Test engineering: a concise guide to cost-effective design, development and manufacture. Wiley, 2001.

21. Мартынчук И.Г., Жмылёв С.А. Сравнительный анализ систем для организации облачных вычислений // Сборник трудов VIII научно-практической конференции молодых ученых «Вычислительные системы и сети (Майоровские чтения)». 2017. С. 54–59.

22. Faynberg Igor, Lu Hui-Lan, Skuler Dor. Cloud computing: Business trends and technologies. John Wiley & Sons, 2016.

23. Truong Hong-Linh, Dustdar Schahram. Principles for engineering IoT cloud systems // IEEE Cloud Computing. 2015. Т. 2, № 2. С. 68–76.

24. Мартынчук И.Г., Жмылёв С.А. Архитектура и организация сервисов автомасштабирования в облачных системах // Альманах научных работ молодых ученых Университета ИТМО. 2017. Т. 5. С. 200–203.

25. Мартынчук И.Г., Жмылёв С.А. Разработка модуля для сервиса автомасштабирования облачной системы OpenNebula // Сборник тезисов докладов конгресса молодых ученых. – СПб., 2017.

26. Мартынчук И.Г. Разработка программного комплекса для автоматического управления ресурсами облачной вычислительной системы // Аннотированный сборник научно-исследовательских выпускных квалификационных работ бакалавров Университета ИТМО. 2017. С. 44–46.

27. Puterman Martin L. Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons, 2014.

28. Bogatyrev V.A., Bogatyrev A.V. Functional reliability of a real-time redundant computational process in cluster architecture systems // Automatic Control and Computer Sciences. 2015. Т. 49, № 1. С. 46–56.

29. Алиев Т. И. Аппроксимация вероятностных распределений в моделях массового обслуживания // Научно-технический вестник информационных технологий, механики и оптики. 2013. № 2. С. 88–93.

30. Гнеденко Б. В. Курс теории вероятности. – М.: Едиториал, 2005. – 448 с.

31. Cover Thomas M, Thomas Joy A. Elements of information theory. John Wiley & Sons, 2012.

32. Giambene Giovanni. Queuing theory and telecommunications. Springer, 2005.

33. Aliev T.I., Rebezova M.I., Russ A.A. Statistical methods for monitoring travel agencies in the settlement system // Automatic Control and Computer Sciences. 2015. Т. 49, № 6. С. 321–327.

34. Karlin Samuel. A first course in stochastic processes. Academic press, 2014.

35. Bogatyrev V.A. Protocols for dynamic distribution of requests through a bus with variable logic ring for reception authority transfer // Automatic Control and Computer Sciences. 1999. Т. 33, № 1. С. 57–63.

36. Мартынчук И.Г., Жмылёв С.А. Модели и методы композиции нестационарных распределений // Сборник тезисов докладов конгресса молодых ученых. – СПб., 2019.

37. Вентцель Е. С. Теория вероятностей. – 6 изд. М.: Высш. шк., 1969. – 576 с.

38. Santana Gustavo AA. Data center virtualization fundamentals: understanding techniques and designs for highly efficient data centers with Cisco Nexus, UCS, MDS, and beyond. Cisco Press, 2013.

39. Жмылёв С.А., Киреев В.Ю., Мартынчук И.Г. Исследование систем с нестационарными процессами // Сборник тезисов докладов конгресса молодых ученых. – СПб., 2018.

40. Гмурман В. Е. Теория вероятностей и математическая статистика. – 9 изд. М.: Высш. шк., 2003. – 479 с.

41. Bogatyrev V.A., Bogatyrev S.V., Golubev I.Y. Optimization and the process of task distribution between computer system clusters // Automatic Control and Computer Sciences. 2012. Т. 46, № 3. С. 103–111.

42. Жмылёв С.А., Мартынчук И.Г., Киреев В.Ю. Оценка длины периода нестационарных процессов в облачных системах // Известия высших учебных заведений. Приборостроение. 2018. Т. 61, № 8. С. 645–651.

43. Lorido-Botran Tania, Miguel-Alonso Jose, Lozano Jose A. A review of auto-scaling techniques for elastic applications in cloud environments // Journal of grid computing. 2014. Т. 12, № 4. С. 559–592.

44. Elastic application container: A lightweight approach for cloud resource provisioning / Sijin He, Li Guo, Yike Guo [и др.] // 2012 IEEE 26th International Conference on Advanced Information Networking and Applications / IEEE. 2012. С. 15–22.

45. Platform-as-a-service architecture for real-time quality of service management in clouds / Michael Boniface, Bassem Nasser, Juri Papay [и др.] // 2010 Fifth International Conference on Internet and Web Applications and Services / IEEE. 2010. С. 155–160.

46. Bogatyrev V., Vinokurova M. Control and safety of operation of duplicated computer systems // International Conference on Distributed Computer and Communication Networks / Springer. 2017. С. 331–342.

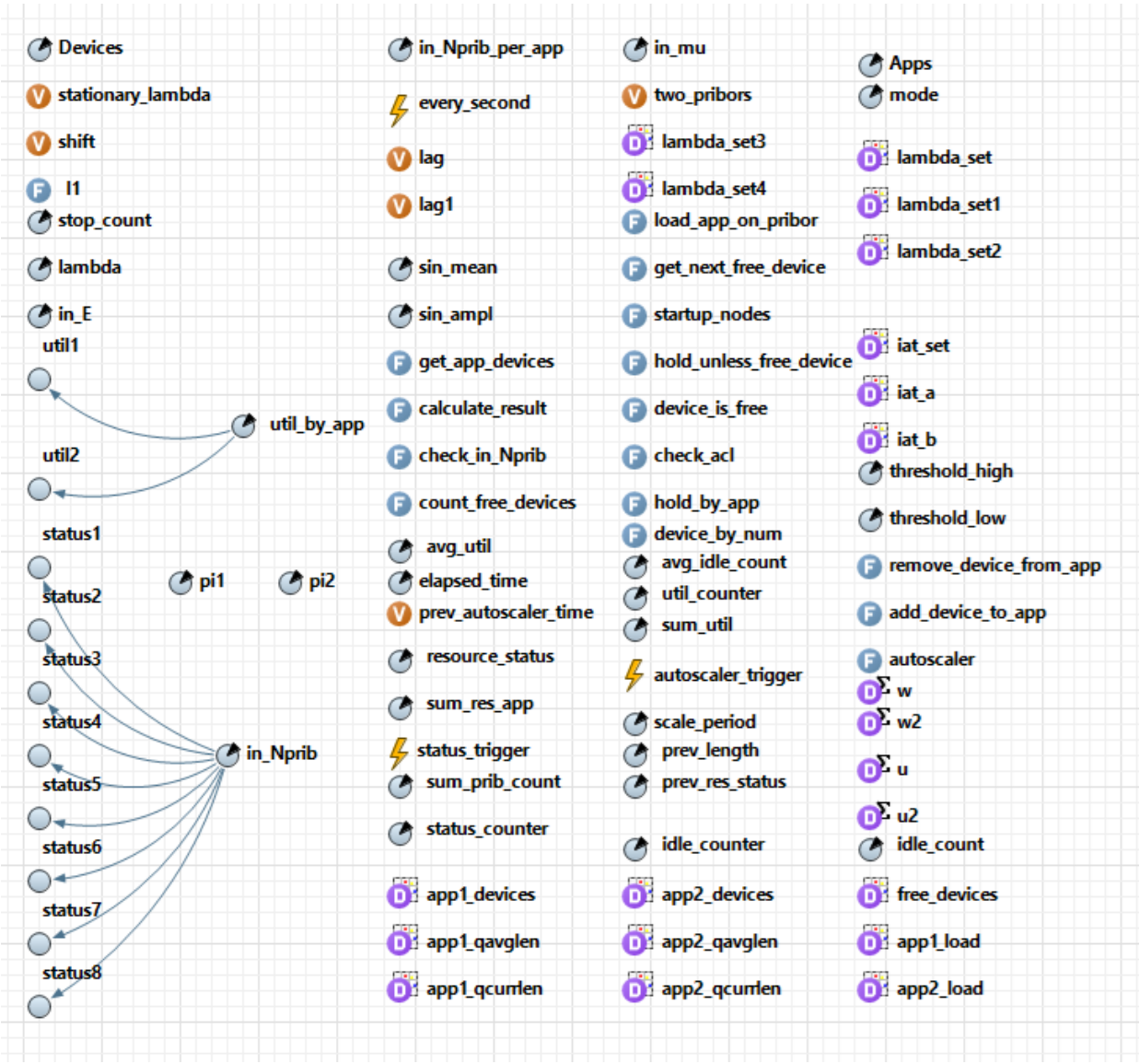
47. Жмылёв С.А., Мартынчук И.Г., Киреев В.Ю. Оценка длины периода нестационарных процессов в облачных системах // VI научно-практическая конференция с международным участием «Наука настоящего и будущего» для студентов, аспирантов и молодых ученых. 2018. С. 41–43.

48. Park K., Willinger W. Self-Similar Network Traffic and Performance Evaluation. – New-York: John Wiley and Sons, 2000. – 479 c.

49. The reservoir model and architecture for open federated cloud computing / Benny Rochwerger, David Breitgand, Eliezer Levy [и др.] // IBM Journal of Research and Development. 2009. Т. 53, № 4. С. 4–1.

Приложение 1. Ключевые участки реализации имитационной модели облачной системы на языке Java в среде AnyLogic

Ниже приведены необходимые для работы модели переменные, параметры, наборы данных и функции.



Java-код блока, реализующего сервис автоматического масштабирования, представлен в листинге 1.

Листинг 1. Функция выполнения атоматического масштабирования

```
switch (policy) {
   case 1:
   for (int app = Apps; app > 0; app--) {
      double util = 0, avg_util = 0;
      avg_util = (sum_res_app[app] * 1.0 / status_counter)/
      (sum_prib_count[app] * 1.0 / status_counter);
      if (avg_util > 1) {
         getEngine().pause();
      }
      sum_res_app[app] = 0;
      sum_prib_count[app] = 0;
      util_by_app[app] = avg_util;
      if (app == 1)
         app1_load.add(avg_util);
      else
         app2_load.add(avg_util);
      if (avg_util > threshold_high[app]) {
         add_device_to_app(app);
      }
      if (avg_util < threshold_low[app]) {
         remove_device_from_app(app);
      }
      sum_util[app] += avg_util;
   }
   break;
   default:
   break;
}
status_counter = 0;
prev_autoscaler_time = time();
util_counter++;
```

Получение характеристик функционирования системы, необходимых для работы сервиса автоматического масштабирования, представлено в листинге 2

Листинг 2. Полученние данных о загруженности узлов по приложению

```
int count1 = 0, count2 = 0;
for (int i = 1; i <= Devices; i++) {
   switch (resource_status[i]) {
      case 1: sum_res_app[1]++;
```

```
            count1++;
            break;
      case 2: sum_res_app[2]++;
            count2++;
            break;
      default:
            break;
   }
}
if (count1 > get_app_devices(1).length)
   sum_prib_count[1] += count1;
else
   sum_prib_count[1] += get_app_devices(1).length;

if (count2 > get_app_devices(2).length)
   sum_prib_count[2] += count2;
else
   sum_prib_count[2] += get_app_devices(2).length;

status_counter++;

for (int i = 1; i <= Devices; i++) {
   if (device_is_free(i) == 1) {
   idle_count++;
}
idle_counter++;
```

Получение узлов, выделенных под приложение, представлено в листинге 3.

Листинг 3. Функция получения списка выделенных под приложение узлов

```
int count_of_devices_per_app = 0;
for (int appi : in_Nprib) {
   count_of_devices_per_app += (appi == app) ? 1 : 0;
}
int[] rc = new int[count_of_devices_per_app];
int j = 0;
for (int i = 1; i < in_Nprib.length; i++) {
   if (in_Nprib[i] == app) {
      rc[j++] = i;
```

```
10      }
11  }
12  return rc;
```

Генерация функций изменения интенсивности поступающих в систему заявок представлена в листинге 4.

Листинг 4. Функции изменения интенсивности запросов

```
1  switch (mode) {
2     case 0: return stationary_lambda;
3     case 1: return (sin_ampl * sin(lag/24 + l_shift) + sin_mean);
4     case 2: if (l_shift == 0) { return (lag * lag)
5     / 100; } else { return (lag1 * lag1) / 100; }
6     case 3: return sin_mean;
7     default: return 0;
8  }
```