

Научная статья

УДК 004.27

DOI:10.31854/1813-324X-2023-9-2-81-93



Интегральное решение проблемы размещения контроллеров и балансировки нагрузки

Аммар Салех Али Мутханна, muthanna.asa@sut.ru

Санкт-Петербургский государственный университет телекоммуникаций им. М.А. Бонч-Бруевича,
Санкт-Петербург, 193232, Российская Федерация

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

Ключевые слова: сети связи пятого и последующих поколений, ядро сети, мультиконтроллер, балансировка нагрузки, алгоритм «роя частиц» (PSO), алгоритм хаотического «роя сальп» (CSSA), алгоритм «серого волка» (GWO), кластеризация

Источник финансирования: Работа выполнена в рамках прикладных научных исследований СПбГУТ, регистрационный номер 1022040500653-0 от 16.02.2023 в ЕГИСУ НИОКТР.

Ссылка для цитирования: Мутханна А.С.А. Интегральное решение проблемы размещения контроллеров и балансировки нагрузки // Труды учебных заведений связи. 2023. Т. 9. № 2. С. 81–93. DOI:10.31854/1813-324X-2023-9-2-81-93

Controller Location and Load Balancing Integrated Solution

Аммар Мутханна, muthanna.asa@sut.ru

The Bonch-Bruevich Saint-Petersburg State University of Telecommunications,
St. Petersburg, 193232, Russian Federation

Abstract: The usage of multi-controller SDNs is currently the most efficient approach for constructing the core of communication networks of the fifth and following generations. Due to the complexity of the problems being tackled, there are currently a number of load balancing algorithms and algorithms for arranging controllers in multi-controller networks that are based on meta-heuristic methods. These algorithms allow for the optimum possible utilisation of controller resources in such networks. However, a comprehensive solution to the load balancing and controller placement issues has yet to be discovered. The answer to such an issue is the focus of this article. The report suggests using network clustering in conjunction with the meta-heuristic chaotic salp swarm technique, which has shown to be effective in prior research on the challenges of creating multi-controller networks, to accomplish this goal. The salp swarm algorithm in the paper is adjusted to take into account the integral solution to the problem of

deploying controllers based on clustering of a multi-controller network and load balancing. By contrasting the simulation results with those from the well-known meta-heuristic particle swarm algorithms optimization and the grey wolf GWO, as well as the previous version of the chaotic salp swarm algorithm CSSA, the effectiveness of the proposed solution was evaluated.

Keywords: 5G networks, core network, multi-controller, load balancing, chaotic salp swarm algorithm, particle swarm optimization algorithms, grey wolf optimization algorithms, clustering

Funding: The research was supported by Applied Scientific Research under the SPbSUT state assignment No. 1022040500653-0, 2023.

For citation: Muthanna A. Controller Location and Load Balancing Integrated Solution. *Proc. of Telecom. Universities.* 2023;9(2):81–93. (in Russ.) DOI:10.31854/1813-324X-2023-9-2-81-93

1. Введение

Развитие сетей и систем связи в направлении создания сетей связи пятого и последующих поколений ставит все новые и новые задачи перед исследовательскими центрами во всем мире. Появление концепций Интернета Вещей и Тактильного Интернета [1] привело к созданию сверхплотных сетей и сетей связи с ультрамалыми задержками, принципиально изменивших представления о трафике, поступающем на сети, и о требованиях к качеству обслуживания в таких сетях. Это привело к необходимости пересмотра представлений о построении ядра сети. При этом наиболее подходящей новой технологией для построения сетей связи пятого и последующих поколений были признаны программно-конфигурируемые сети SDN (аббр. от англ. Software-Defined Network) [2, 3].

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

Основной вклад данной статьи заключается в:

- разработке алгоритма иерархической кластеризации мультиконтроллерной сети для решения проблемы интеграции размещения контроллеров в мультиконтроллерных сетях и балансировки нагрузки;
- разработке модифицированного алгоритма хаотического «роя сальп» (CSSA, аббр. от англ. Chaotic Salp Swarm Algorithm – хаотический алгоритм «роя сальп») для использования в иерархических кластерных сетях clus-CSSA.

Статья организована следующим образом: в разделе (2) приводится аналитическая информация о проблеме размещения контроллеров и балансировки нагрузки в мультиконтроллерных сетях SDN, в разделе (3) представлена разработанная кластерная архитектура для балансировки нагрузки в мультиконтроллерных сетях, в разделе (4) – сетевая модель исследуемой мультиконтроллерной

сети SDN; в разделе (5) представлена решаемая оптимизационная задача, в разделе (6) – оценка характеристик разработанных решений с использованием кластеризации и метаэвристического модифицированного хаотического «роя сальп» в сравнении с другими известными алгоритмами.

2. История вопроса и соответствующие работы

Размещение контроллеров в SDN является одной из критических проблем и привлекает большое внимание в литературе [4]. Проблема размещения контроллеров в мультиконтроллерных сетях впервые была исследована в [5], а затем и [6].

В [7] авторы разработали метод балансировки нагрузки для динамического добавления контроллеров в заданную сеть, при этом коммутаторы могут мигрировать между контроллерами в зависимости от нагрузки на последние. Была представлена идея динамического назначения между коммутатором и контроллером и предложен эффективный алгоритм для балансировки нагрузки и миграции коммутаторов. Однако в исследовании не было учтено существенное влияние первоначального размещения контроллеров на их загрузку.

В [8] авторы модифицировали задачу k -центра для повышения эффективности распределения ресурсов между клиентами. Клиентам выделяются ограниченные ресурсы с учетом ограничений на пропускную способность соответствующих объектов. Формулируется оптимизационная задача и вычисляется решение с помощью линейного и смешанного целочисленного программирования, когда в заданном радиусе доступно необходимое количество объектов с достаточной пропускной способностью.

В [9] авторы предложили метаэвристические решения для размещения контроллеров с использованием алгоритмов оптимизации «роя частиц» (PSO, аббр. от англ. Particle Swarm Optimization) и светлячков и сравнили результаты со случайным размещением контроллеров. Результаты моделирования показывали, что оба алгоритма работают лучше с точки зрения задержки и более быстрой сходимости. В приведенном анализе результатов

для индийской сети TATA число контроллеров составляет 20. Однако оптимальность наблюдаемого числа контроллеров не была определена.

В работе [10] авторы представили решения на основе алгоритма игры с ненулевой суммой для оптимального размещения нескольких контроллеров. В игре с ненулевой суммой каждый контроллер имеет механизм оптимизации, который вычисляет функцию вознаграждения и сравнивает свое собственное значение вознаграждения для экономии затрат и улучшения качества обслуживания (QoS, аббр. от англ. Quality of Service) путем оптимизации расположения контроллеров. Аналогично, в [11] авторы представили игровую модель для изучения размещения нескольких контроллеров. Эта модель учитывает несколько метрик, которые включают задержку связи между контроллерами и коммутаторами, накладные расходы на связь между контроллерами и нагрузку на них. На основе этих метрик в статье сформулирована оптимизационная задача с двумя противоречивыми целями: минимизация задержки связи и накладных расходов на связь.

В [12] авторы предложили использовать теорию очередей для размещения нескольких контроллеров. Был использован CSSA для решения задачи оптимизации. Авторы провели сравнение разработанного алгоритма с другими метаэвристическими алгоритмами. Результаты моделирования показали, что предложенный алгоритм превосходит другие метаэвристические алгоритмы и алгоритм на основе теории игр по своим характеристикам.

В [13] авторы сформулировали задачу оптимизации для размещения нескольких контроллеров, учитывая сценарий отказа одного контроллера в SDN. В работе был предложен метаэвристический алгоритм имитационного отжига для достижения глобального оптимального решения. Однако предварительное определение числа контроллеров привело к неоптимальному их размещению в динамически изменяющейся сети реального времени.

В [14] авторы предложили алгоритм разбиения сети, основанный на лувенской эвристической методике определения сообществ. Для идентификации сообществ они вычислили гаверсово расстояние между ребрами сети и присвоили ребрам вес, обратно пропорциональный расстоянию. Для определения расположения контроллеров были реализованы отдельные алгоритмы, которые находят компромисс между средней задержкой и нагрузкой на контроллер. Однако сложность алгоритмов значительно возрастает с размером сети.

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

определенные допущения и приближения. В целях разработки простых в вычислительном отношении решений авторы работы [15] предложили разделение сети и оптимизированную кластеризацию k -means для разделения сети на k подсетей, когда речь идет о задержке. Техника разбиения сети применяется для упрощения проблемы размещения контроллеров. По сравнению с приведенными выше схемами подход на основе кластеризации k -средних снижает вычислительную сложность. Однако для алгоритма требуются предопределенные входные параметры, что делает его неприменимым в сценариях сетей, работающих в реальном времени.

В [16] авторы применили иерархическую кластеризацию k -means для разделения сети и расширили оптимизированную кластеризацию k -means, предложенную в [15], учитывая как задержку, так и балансировку нагрузки. Было использовано расстояние кратчайшего пути между узлами вместо евклидова расстояния, используемого в оптимизированном k -means. Алгоритм итеративно объединяет k' начальных центров и в итоге получает k центров. Предварительно заданные значения k и k' в иерархической кластеризации делают алгоритм неприменимым для глобальных вычислительных сетей (WAN, аббр. от англ. Wide Area Network) на базе SDN.

В [17] авторы разработали систему BalanceFlow, которая представляет собой типичное решение кластеризации контроллеров, основанное на иерархическом развертывании. Основным преимуществом этого метода является гибкая настройка запросов потока, обрабатываемых каждым контроллером, без введения дополнительных задержек распространения. Он следует функции мультиконтроллеров в OpenFlow 1.2. Все контроллеры в BalanceFlow поддерживают свою собственную информацию о нагрузке и периодически публикуют ее друг другу через систему межконтроллерной связи. При изменении состояния трафика один из контроллеров BalanceFlow выбирается в качестве суперконтроллера, который разделяет трафик и перераспределяет различные настройки потока между соответствующими контроллерами.

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

3. Иерархическая кластеризация для мультиконтроллерных сетей SDN

Предложенный алгоритм иерархической кластеризации выполняет ее в динамическом режиме, основываясь на сетевом трафике и запросах потоков. Плоскость управления делится на кластеры SDN-контроллеров с головным SDN-контроллером для каждого. На рисунке 1 показана структура кластерной сети SDN. Сеть SDN состоит из централизованного альфа-контроллера (C_α) и кластеров обычных SDN-контроллеров с равным их числом в каждом кластере. В каждом кластере есть головной узел, который представляет бета-контроллер (C_β) и отвечает за настройку кластера и балансировку нагрузки между контроллерами, входящими в него. Основная цель разработанного алгоритма кластеризации – сбалансировать нагрузку между контроллерами плоскости управления и избежать отказов контроллеров. Это снижает стоимость сети и повышает ее общую доступность и надежность.

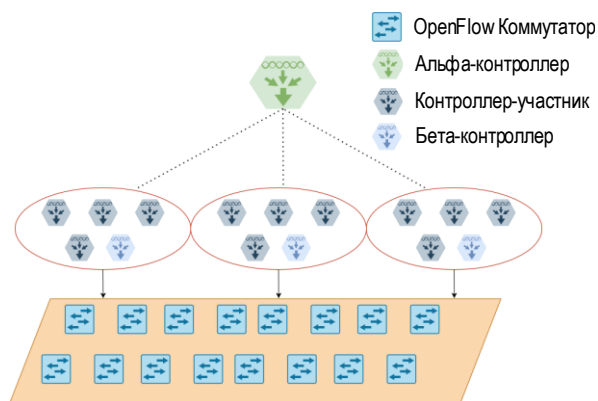


Рис. 1. Общая структура кластерной мультиконтроллерной сети SDN

Fig. 1. A Clustered Multi-Controller SDN Network's General Structure

Процесс кластеризации делится на фазы 1) настройки контроллера, 2) соединения коммутаторов и 3) установившегося состояния. На этапе создания кластера альфа-контроллер формирует SDN-кластеры, выбирая бета-контроллеры и контроллеры-участники каждого кластера. Контроллер с минимальной ожидаемой нагрузкой выбирается в качестве бета-контроллера и берет на себя роль руководителя членов кластера. Все кластеры формируются однородно, т.е. количество SDN-контроллеров в каждом кластере одинаково. После формирования кластеров фаза настройки контроллеров заканчивается, и начинается фаза настройки соединений коммутаторов. На этапе установки соединений коммутаторов вызывается разработанный CSSA, представленный в разделе 5.

Каждому кластеру назначается группа OpenFlow-коммутаторов, и, кроме того, CSSA динамически распределяет коммутаторы между каждым контроллером-участником таким образом, чтобы достичь оптимальной эффективности задержки и

стоимости, как CAPEX, так и OPEX. На этом этапе каждый SDN-контроллер принимает роль, т.е. либо члена кластера, либо бета, и назначает оптимальные соединения с OpenFlow-коммутаторами, определяемые CSSA. Затем начинается фаза установившегося состояния. Каждый контроллер-член кластера управляет подключенными OpenFlow-коммутаторами и берет на себя роль таблицы потоков таких коммутаторов. Каждый бета-контроллер управляет назначенными OpenFlow-коммутаторами как контроллеры-члены и, кроме того, берет на себя роль руководителя своего кластера.

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

Работа сети продолжается до тех пор, пока нагрузка между кластерами не станет несбалансированной. Альфа-контроллер получает отчеты от бета-контроллеров и отслеживает трафик между различными кластерами; когда он обнаруживает дисбаланс нагрузки между ними, он выполняет новый раунд общей кластеризации, формируя новые кластеры с помощью ранее введенных фаз. Эта схема поддерживает балансировку нагрузки между распределенными SDN-контроллерами, что позволяет достичь оптимального использования сети.

4. Математическое моделирование кластерной мультиконтроллерной сети SDN

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

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

где N – общее количество развернутых SDN-контроллеров.

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

$$C_{\beta} = \{C_{\beta_1}, C_{\beta_2}, C_{\beta_3}, \dots, C_{\beta_M}\},$$

$$C_{\beta} \subset C \quad \forall C_{\beta_i} \in C. \quad (2)$$

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

$$C_{mi} = \{C_{mi,1}, C_{mi,2}, C_{mi,3}, \dots, C_{mi,L}\},$$

$$C_{mi} \subset C \quad \forall C_{mi,j} \in C. \quad (3)$$

В плоскости данных сети SDN развернуто k OpenFlow-коммутаторов, распределенных между кластерами контроллеров. Каждый коммутатор имеет соединение с SDN-контроллером, который распределяется с помощью разработанного алгоритма размещения контроллеров, представленного в следующем разделе. Набор развернутых OpenFlow-коммутаторов определяется выражением:

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

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

$$S_{mi} = \{S_{mi,1}, S_{mi,2}, S_{mi,3}, \dots, S_{mi,R}\},$$

$$S_{mi} \subset S \quad \forall S_{mi,j} \in S. \quad (5)$$

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

Пример матрицы коммутации представлен в виде:

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

Одним из способов проверки производительности контроллера является оценка временного отклика контроллера, на который в основном влияет задержка в очереди. Контроллеры могут быть смоделированы с помощью многосерверной модели очередей $M/M/s$, где предполагается, что каждый контроллер имеет s ядер [18]. Передаваемые пакеты поступают на контроллер с определенной скоростью, соответствующей процессу Пуассона, образуя единую очередь на контроллере.

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

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

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

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

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

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

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

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

Скорость прибытия контроллера может быть рассчитана как сумма средних скоростей прибытия коммутаторов, подключенных к контроллеру:

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

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

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

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

5.1. Формулировка проблемы

В связи с динамическим изменением нагрузки на сеть проблема размещения контроллера должна решаться динамически – чтобы достигались определенные показатели сети. В этом разделе эта проблема сформулирована с точки зрения задержки, использования и стоимости сети. Задача размещения контроллеров при этом направлена на получение динамически оптимального числа SDN-контроллеров, которое обеспечивает требуемую задержку между распределенными OpenFlow-коммутаторами и SDN-контроллерами, а также минимальную стоимость сети. Это оптимальное количество динамически изменяется в зависимости от нагрузки на сеть. Более того, задача направлена на определение оптимальных соединений между коммутаторами и SDN-контроллерами таким образом, чтобы

достичь требуемой латентности и стоимости. Модифицируем ранее разработанную нами задачу размещения контроллеров с учетом задержки и стоимости, представленную в [12], для кластерной сети SDN.

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

Задача формулируется следующим образом:

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

Ограничения:

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

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

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

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

Задача представляет собой многоцелевую оптимизацию с множеством ограничений, которая может быть решена соответствующим метаэвристическим алгоритмом с тремя ограничениями.

Первое указывает, что среднее время ответа T_i^j контроллера C_i , принадлежащего кластеру j , должно быть меньше порогового значения T_{thr} , которое является предопределенным значением; это имеет место для всех SDN-контроллеров в наборе доступных контроллеров $[\gamma]$ для набора развернутых кластеров $[\beta]$. T_{thr} предопределено таким образом, чтобы удовлетворить требуемое QoS.

Второе вводится для поддержания уровня использования каждого SDN-контроллера. Показатель использования U_i^j контроллера C_i в кластере j должен находиться в пределах, с одной стороны, нижней границы использования SDN-контроллера U_{lb} (т. е. минимального значения использования SDN-контроллера, ниже которого он должен быть

отключен для достижения экономической эффективности), а с другой – верхней (т. е. максимального значения использования SDN-контроллера, выше которого контроллер может быть перегружен). Верхний и нижний пределы U_{ub} и U_{lb} предопределены таким образом, чтобы достичь требуемого QoS. Этот показатель использования SDN-контроллера применяется для сопоставления с показателями использования мощности, хранения и обработки данных.

Третье учитывает поддержание общего индекса использования кластера j между максимальным (U_{C-ub}) и минимальным пределами (U_{C-lb}). Это ограничение введено для поддержания баланса нагрузки между кластерами и для предотвращения выхода из строя раздела сети.

5.2. Использование системы

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

$$U: V \rightarrow R. \quad (16)$$

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

Временная полезность SDN-контроллера C_i , принадлежащего кластеру j , равна U_{T-Ci} и определяется следующим образом:

$$U_{T-Ci}^j = \{\alpha + \delta (T_{\text{thr}} - T_i^j(\lambda))^2\}, \quad (17)$$

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

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

Эти константы могут быть определены следующим образом:

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

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

Для $T_{\text{thr}} = 70 \%$ время использования может быть пересчитано следующим образом:

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

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

Функция использования времени кластера $j - U_T^j$ может быть рассчитана как нормализованное среднее значение использования времени каждого члена кластера (21).

Второй функцией полезности, которую следует рассмотреть, является функция полезности затрат, которая отображает стоимость используемых контроллеров. Под стоимостью в основном понимаются оба термина: CAPEX и OPEX, связанные с разверну-

тыми контроллерами. Квадратичная функция потерь также представляет собой подходящую функцию для выражения утилизации затрат. Стоимость использования каждого контроллера в наборе доступных контроллеров может быть определена по выражению (22), где U_{C-Ci} – функция использования затрат контроллера C_i в кластере j , а ρ – константа, которая не влияет на решение, независимо от того, какое значение ей присвоено. Правильное значение ρ может быть определено следующим образом, для U_{C-Ci} между 0 и 1 (23). Функция использования затрат в кластере $j - U_C^j$ и может быть рассчитана как нормализованное среднее значение использования затрат каждого члена кластера (24).

$$U_T^j = \frac{\sum_{N_C} U_{T-Ci}^j}{N_C} = \frac{\left(\sum_{N_C} \{0,7 + \frac{0,3}{T_{thr}^2} (T_{thr} - T_i^j(\lambda))^2, \quad T_i^j(\lambda) \leq T_{thr} \quad 0, \quad T_i^j(\lambda) > T_{thr} \quad \forall i \in \gamma \wedge j \in \beta \right)}{N_C}, \quad (21)$$

$$U_{C-Ci}^j = \{\rho(U_{ub} - U_i^j)^2 \quad \forall U_i^j \in [U_{lb}, U_{ub}] \quad 0 \quad \forall U_i^j \notin [U_{lb}, U_{ub}] \quad \forall i \in \gamma \wedge j \in \beta, \quad (22)$$

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

$$U_C^j = \frac{\sum_{N_C} U_{C-Ci}^j}{N_C} = \frac{\left(\sum_{N_C} \{\rho(U_{ub} - U_i^j)^2 \quad \forall U_i^j \in [U_{lb}, U_{ub}] \quad 0 \quad \forall U_i^j \notin [U_{lb}, U_{ub}] \quad \forall i \in \gamma \wedge j \in \beta \right)}{N_C} \quad (24)$$

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

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

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

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

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

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

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

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

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

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

$$U^j = \delta_C U_C^j + \delta_T U_T^j. \quad (29)$$

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

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

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

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

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

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

В этом подразделе представляется процедура оптимизации на основе CSSA для решения задачи размещения контроллеров в кластерной сети SDN с в постановке, сформулированной в подразделе 5.1.

Разработанный алгоритм является модифицированной версией CSSA, представленного в [12].

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

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

$$X_i^1 = \{F_i + C_1((ub_i - ul_i)C_2 + lb_i),$$

$$C_3 \geq 0 \quad F_i - C_1((ub_i - ul_i)C_2 + lb_i), \quad C_3 < 0, \quad (33)$$

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

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

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

где t – текущая итерация; T_{\max} – максимальное число итераций.

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

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

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

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

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

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

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

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

Для NP-трудной задачи, смоделированной в подразделе 5.1, мы стремимся определить оптимальное количество SDN-контроллеров и кластеров, а также оптимальное соединение для каждого коммутатора в наборе коммутаторов S . Путем итераций несколько роев сальп параллельно ищут оптимальные решения, которые представляют собой оптимальное количество контроллеров, и кластеров, а также оптимальное распределение контроллеров между коммутаторами. Следовательно, для получения оптимальных решений вводятся три вложенных алгоритма, основанных на ранее представленном хаотическом сальпе.

Алгоритм 1 представляет псевдокод CSSA, разработанного для задачи размещения контроллеров, поставленной в подразделе 5.1, где каждая сальпа представляет контроллер в сети. Выходом этого алгоритма является оптимальное количество SDN-контроллеров. Алгоритм 2 представляет псевдокод CSSA, разработанного для оптимального соединения SDN OpenFlow-коммутаторов на основе оптимального количества контроллеров, рассчитанного алгоритмом 1. Каждая сальпа в алгоритме 2 представляет все доступные соединения для всех коммутаторов с их выделенными контроллерами, и это M -мерный вектор, каждое измерение которого представляет коммутатор. Выходом второго алгоритма является наилучшее распределение контроллеров между коммутаторами. Алгоритм 3 представляет собой псевдокод CSSA, разработанного для оптимальной кластеризации. Выходом алгоритма 3 является оптимальное число кластеров.

Алгоритм 1. CSSA для поиска оптимального числа контроллеров

```

1: Initialize  $ub, lb, T_{\max}, d, n$ 
2: Initialize positions of salps  $x_i$  ( $i = 1, 2, 3, \dots, n$ )
3: While ( $t \leq T_{\max}$ )
4:   Calculate the fitness function of each salp position using Eq. (15)
5:    $F =$  The best salp position
6:   Update the value of  $C_1$  using (23)
7:   Get the value of chaotic map (Logistic)  $w(t)$ 
8:   For ( $i = 1; i \leq n$ ) do
9:     if ( $i == 1$ )
10:      Update the position of the leading salp using (22, 23, 25, 26)
11:     else
12:      Update the position of the follower salps using (24)
13:     end if
14:   end for
15:   Adjust salps based on the upper and lower bounds
16:   Calculate the best connections of switches for the best salp (call Algorithm 2)
17:   Update the best salp based on the results of Algorithm 2
18:   Calculate the best number of clusters (call Algorithm 3)
19:    $t \leftarrow t+1$ 
20: Return  $F$ 

```

Алгоритм 2. CSSA для оптимального соединения переключателей с контроллерами

```

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

```

Алгоритм 3. CSSA для оптимальной кластеризации

```

1: Initialize  $ub, lb, T_{max}, d, n$ 
2: Initialize positions of salps  $x_i$  ( $i = 1, 2, 3, \dots, n$ )
3: While ( $t \leq t_{max}$ )
4:   Calculate the fitness function of each salp position using (19)
5:    $F$  = The best salp position
6:   Update the value of  $C1$  using (23)
7:   Get the value of chaotic map (Logistic)  $w(t)$ 
8:   For ( $i = 1: i \leq n$ ) do
9:     if ( $i == 1$ )
10:      Update the position of leading salp using (22, 23, 25, 26)
11:     else
12:      Update the position of follower salp using (24)
13:     end if
14:   end for
15:   Adjust salps based on the upper and lower bounds
16:    $t \leftarrow t+1$ 
17: Return  $F$ 

```

6. Оценка эффективности

6.1. Настройка моделирования

Разработанная оптимизированная схема кластеризации смоделирована в среде Matlab с использованием процессора Intel Core i7 и оперативной памяти 8 Гб. Для оценки производительности рассмотрим реальные топологии крупномасштабных сетей WAN, чтобы лучше проиллюстрировать эффективность предложенной схемы. Для моделирования выбрано 15 приближенных реальных топологий из набора данных Internet Topology Zoo. Эти топологии рассматриваются как крупномасштабные сети с массивными устройствами пересылки. Эти выбранные топологии представлены в таблице 1 с их характеристиками. Каждая точка присутствия в топологии сети рассматривается как сетевой коммутатор, т.е. устройство пересылки. Предполагается, что поток является случайной величиной, подчиняющейся распределению Пуассона. В таблице 2 представлены рассматриваемые параметры моделирования. В качестве SDN-контроллера используется NOX-контроллер.

ТАБЛИЦА 1. Топологии сети, рассмотренные для моделирования

TABLE 1. Network Topologies Considered for Simulation

Ссылка. Номер	Топология	Количество
1	IBM	18
2	Oxford	20
3	FCCN	23
4	AGIS	25
5	Viatel	83
6	GEANT	27
7	TATA	169
8	GTS CE	187
9	PalmettoNet	49
10	OTEGlobe	78
11	DFN	47
12	GARR	36
13	ULAKNET	76
14	RNP	28
15	Carnet	43

ТАБЛИЦА 2. Параметры моделирования

TABLE 2. Simulation Parameters

Параметр	Описание	Количество
λ_s	Средняя скорость запроса коммутатора	[1500, 3000]
μ	Скорость обслуживания контроллера (запрос/сек)	30000
T_{thr}	Порог задержки (мс)	2
U_{ub}	Верхняя граница индекса использования контроллера	0.9
U_{lb}	Нижняя граница индекса использования контроллера	0.7
U_{c-ub}	Верхняя граница индекса использования кластера	0.9
U_{c-lb}	Нижняя граница индекса использования кластера	0.7
$T_{max}(1)$	Максимальное количество итераций для алгоритма 1	50
$T_{max}(2)$	Максимальное количество итераций для алгоритма 2	30
$T_{max}(3)$	Максимальное количество итераций для алгоритма 3	30
δ_c	Весовой коэффициент затрат	18
δ_t	Весовой коэффициент времени	25

6.2. Результаты моделирования

В интересах оценки производительности рассматриваются три сценария моделирования для исследования влияния на оптимальные топологические решения (число SDN-контроллеров) следующих параметров: 1) порога среднего времени отклика SDN-контроллера T_{thr} (Сценарий I), 2) верхней границы индекса использования U_{ub} (Сценарий II), верхней границы индекса использования кластера U_{c-ub} (Сценарий III). Для Сценариев I и III – U_{ub} установлена на 0.9; для Сценариев II и III – T_{thr} установлен на 2 мс; для Сценариев I и II – U_{c-ub} установлена на 0.9. Разработанная оптимизированная схема кластеризации реализуется для каждой из пятнадцати топологий глобальной сети для четырех случаев (наборов параметров), представленных в таблице 3.

ТАБЛИЦА 3. Четыре набора параметров для трех сценариев моделирования

TABLE 3. Four Sets of Parameters for Three Simulation Scenarios

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

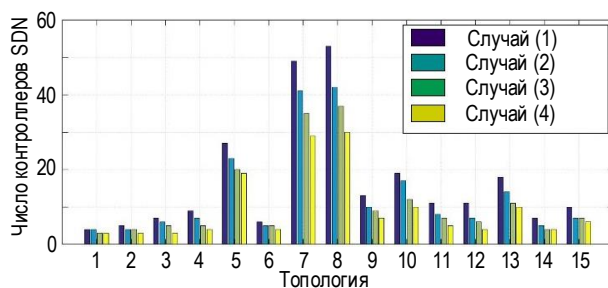
Результаты показывают, что оптимальное количество активных SDN-контроллеров уменьшается по мере увеличения порогового времени отклика и индекса максимального использования каждого SDN-контроллера, а значит, уменьшается и оптимальное количество кластеров. С увеличением порогового времени отклика и индекса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

Разработанная оптимизированная схема кластеризации реализована для случайной сети с

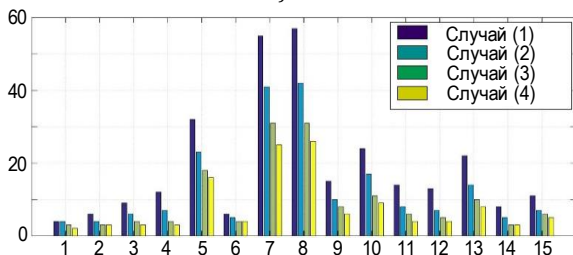
плоскостью данных из 100 коммутаторов, чтобы оценить производительность CSSA для сетей большого масштаба. Основной целью данного сценария моделирования является оценка влияния вариации параметров сети на производительность оптимизированной схемы кластеризации. Производительность системы измеряется в широком диапазоне порогового времени обработки SDN-контроллера T_{thr} верхнего индекса использования SDN-контроллера U_{ub} и верхней границы индекса использования кластера U_{C-ub} .

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

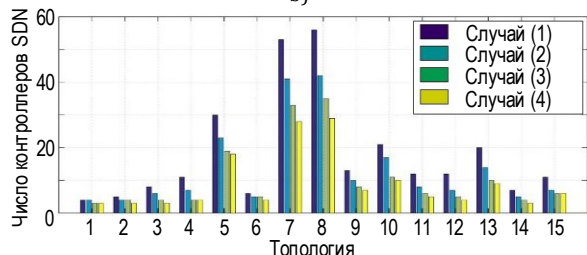
Далее для моделирования рассматриваются три сценария с десятью случаями. В первом сценарии оценивается влияние изменения порогового времени обработки SDN-контроллера T_{thr} , во втором сценарии – верхнего индекса использования SDN-контроллера U_{ub} , а в третьем сценарии – верхней границы индекса использования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



a)



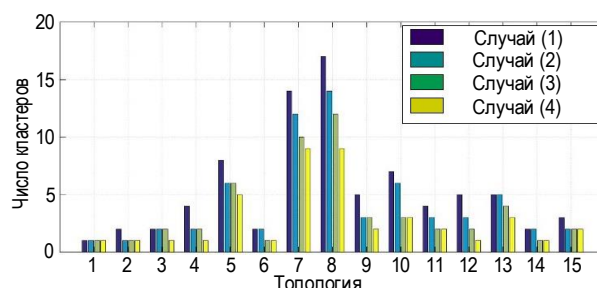
b)



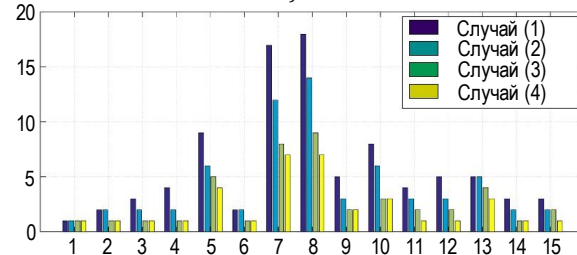
c)

Рис. 2. Оптимальное число SDN-контроллеров для каждой топологии Сценария I (a), Сценария II (b), Сценария III (c)

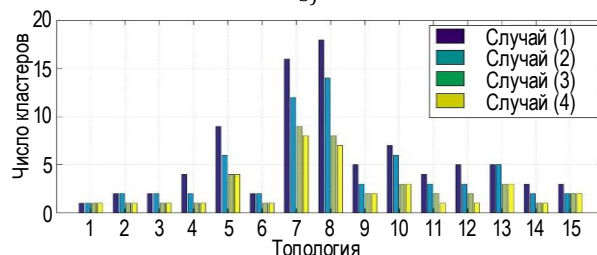
Fig. 2. The Ideal Number of SDN Controllers for Each Topology in a Simulation Scenario I (a), Scenario II (b), Scenario III (c)



a)



b)



c)

Рис. 3. Оптимальное число кластеров для каждой топологии Сценария I (a), Сценария II (b), Сценария III (c)

Fig. 3. The Ideal Number of Clusters for Each Topology in a Simulation Scenario I (a), Scenario II (b), Scenario III (c)

Для каждого сценария рассматривается десять значений каждого параметра, а каждое значение представляет собой случай моделирования. В таблице 4 приведены значения рассматриваемых параметров в каждом случае по каждому сценарию.

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

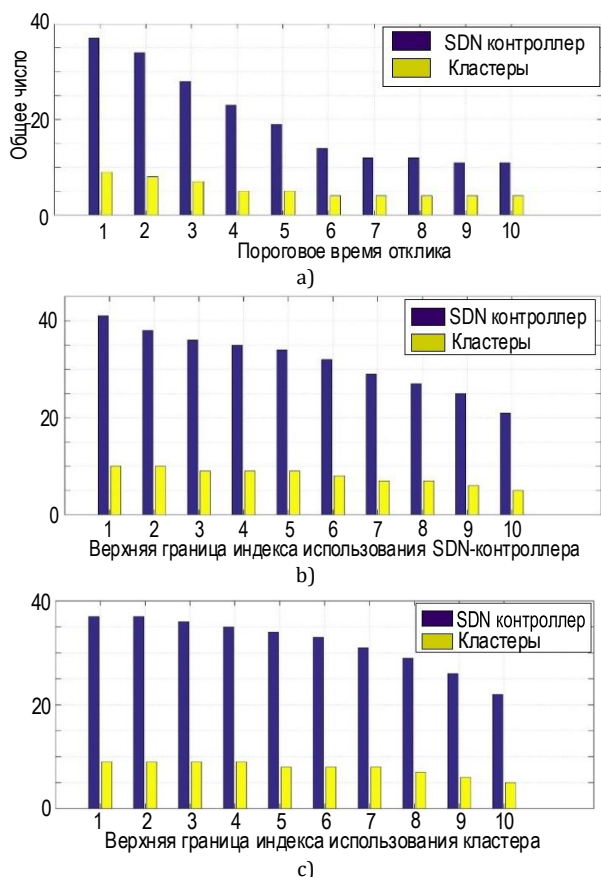


Рис. 4. Оптимальное число SDN-контроллеров и кластеров для Сценария I (а), Сценария II (б), Сценария III (с)

Fig. 4. The Ideal Number of SDN Clusters and Controllers for Simulated Scenario I (a), Scenario II (b), Scenario III (c)

ТАБЛИЦА 4. Десять наборов параметров для трех сценариев моделирования

TABLE 4. Ten Sets of Parameters for Three Simulation Scenarios

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

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

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

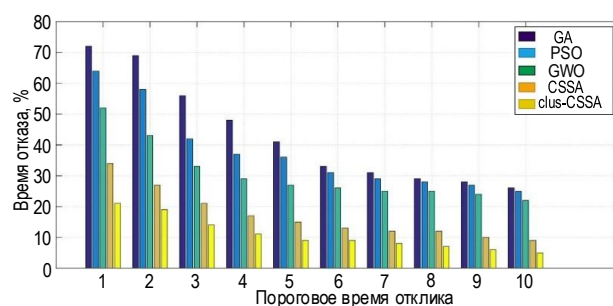


Рис. 5. Доля отказов в обслуживании со стороны контроллера при использовании сравниваемых алгоритмов
Fig. 5. The Percentage of Controller Denials of Service While Utilising Comparative Techniques

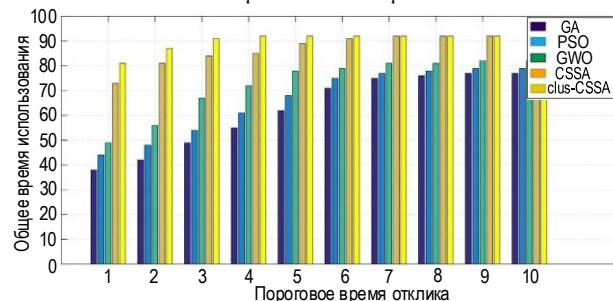


Рис. 6. Общее использование системы для сравниваемых алгоритмов

Fig. 6. System Usage in General for Comparing Methods

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

Заключение

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

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

методе построения сети. В-третьих, разработан модифицированный алгоритм CSSA для использования в иерархических кластерных сетях clus-CSSA.

Разработанный метод построения мультиконтроллерной сети позволяет уменьшить долю отказов в обслуживании со стороны контроллера и увеличить общее использование системы во всем диапазоне изменения задержки от 1 до 10 мс по сравнению как с широко известными метаэвристическими алгоритмами PSO и GWO, так и с предыдущей версией CSSA. При этом для наиболее сложного случая задержки величиной в 1 мс выигрыш по доле отказов и по общему использованию системы достигает значения более, чем в 2 раза.

В дальнейшем планируется реализовать алгоритм CSSA для сети Интернета Вещей высокой плотности.

Список источников

1. Кучерявый А.Е., Маколкина М.А., Киричек Р.В. Тактильный интернет. Сети связи со сверхмалыми задержками // *Электросвязь*. 2016. № 1. С. 44–46.
2. Бородин А.С., Кучерявый А.Е. Сети связи пятого поколения как основа цифровой экономики // *Электросвязь*. 2017. № 5. С. 45–49.
3. Кучерявый А.Е., Прокопьев А.В., Кучерявый Е.А. Самоорганизующиеся сети. СПб: Типография «Любавич», 2011. 312 с.
4. Атея А.А., Мутханна А.С., Кучерявый А.Е. Интеллектуальное ядро для сетей связи 5G и тактильного интернета на базе программно-конфигурируемых сетей // *Электросвязь*. 2019. № 3. С. 34–40.
5. Heller B., Sherwood R., McKeown N. The controller placement problem // *Proceedings of the Special Interest Group on Data Communication (SIGCOMM '12, Helsinki, Finland, 13 August–17 August 2012)*. Special October issue ACM SIGCOMM Computer Communication Review. New York: ACM, 2012. Vol. 42. Iss. 4. PP. 473–478. DOI:10.1145/2377677.2377767
6. Yao G., Bi J., Li Y., Guo L. On the Capacitated Controller Placement Problem in Software Defined Networks // *IEEE Communications Letters*. 2014. Vol. 18. Iss. 8. PP. 1339–1342. DOI:10.1109/LCOMM.2014.2332341
7. Dixit A., Hao F., Mukherjee S., Lakshmanet T.V., Kompella R. Towards an elastic distributed SDN controller // *Proceedings of the Special Interest Group on Data Communication (SIGCOMM '13, Hong Kong, China, 16 August 2013)*. Special October issue ACM SIGCOMM Computer Communication Review. New York: ACM, 2013. Vol. 43. Iss. 4. PP. 7–12. DOI: 10.1145/2534169.2491193
8. Ozsoy F.A., Pinar M.C. An exact algorithm for the capacitated vertex pcenter problem // *Computers & Operations Research*. 2006. Vol. 33. Iss. 5. PP. 1420–1436. DOI:10.1016/j.cor.2004.09.035
9. Sahoo K.S., Sarkar A., Mishra S.K., Sahoo B., Puthal D., Obaidat M.S., et al. Metaheuristic Solutions for Solving Controller Placement Problem in SDN-based WAN Architecture // *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) and 8th International Conference on Data Communication Networking (DCNET)*, Madrid, Spain, 15–23 July 2017. SciTePress Digital Library, 2017. PP. 15–23. DOI:10.5220/0006483200150023
10. Rath H.K., Revoori V., Nadaf S.M., Simha A. Optimal controller placement in Software Defined Networks (SDN) using a non-zero-sum game // *Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks (Sydney, Australia, 19 June 2014)*. IEEE, 2014. DOI:10.1109/WoWMoM.2014.6918987
11. Ksentini A., Bagaa M., Taleb T., Balasingham I. On using bargaining game for Optimal Placement of SDN controllers // *Proceedings of the International Conference on Communications (ICC, Kuala Lumpur, Malaysia, 22–27 May 2016)*. IEEE, 2016. DOI:10.1109/ICC.2016.7511136
12. Ateya A.A., Muthanna A., Vybornova A., Algarni A.D., Abuarqoub A., Koucheryavy Y., et al. Chaotic salp swarm algorithm for SDN multi-controller networks // *Engineering Science and Technology, an International Journal*. 2019. Vol. 22. Iss. 4. PP. 1001–1012. DOI:10.1016/j.jestch.2018.12.015
13. Killi B.P., Rao S.V. Capacitated Next Controller Placement in Software Defined Networks // *IEEE Transactions on Network and Service Management*. 2017. Vol. 14. Iss. 3. PP. 514–527. DOI:10.1109/TNSM.2017.2720699
14. Chen W., Chen C., Jiang X., Liu L. Multi-Controller Placement Towards SDN Based on Louvain Heuristic Algorithm // *IEEE Access*. 2018. Vol. 6. PP. 49486–49497. DOI:10.1109/ACCESS.2018.2867931
15. Wang G., Zhao Y., Huang J., Duan Q., Li J. A K-means-based network partition algorithm for controller placement in software defined network // *Proceedings of the International Conference on Communications (ICC, Kuala Lumpur, Malaysia, 22–27 May 2016)*. IEEE, 2016. DOI:10.1109/ICC.2016.7511441

16. Kuang H., Qiu Y., Li R., Liu X. A Hierarchical K-Means Algorithm for Controller Placement in SDN-Based WAN Architecture // Proceedings of the 10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA, Changsha, China, 10–11 February 2018). IEEE, 2018. PP. 263–267. DOI:10.1109/ICMTMA.2018.00070
17. Hu Y., Wang W., Gong X., Que X., Cheng S. BalanceFlow: Controller load balancing for OpenFlow networks // Proceedings of the 2nd International Conference on Cloud Computing and Intelligent Systems (Hangzhou, China, 30 October 2012–01 November 2012). IEEE, 2013. PP. 780–785. DOI:10.1109/CCIS.2012.6664282

References

1. Koucheryavy A.E., Makolkina M.A., Kirichek R.V. Tactile internet. ULTRA-Low Latency Networks. *Electrosvyaz*. 2016;1:44–46. (in Russ.)
2. Borodin A.S., Koucheryavy A.E. Fifth generation networks as a base to the digital economy. *Electrosvyaz*. 2017;5(45–49).
3. Koucheryavy A.E., Prokopiev A.V., Koucheryavy Y.A. *Self-Organizing Network*. St. Petersburg: Lyubavich Printing House; 2011. 312 p. (in Russ.)
4. Ateya A.A., Muthanna A.S., Koucheryavy A.E. Intelligent core network for 5g and tactile internet systems based on software defined networks. *Electrosvyaz*. 2019;3:34–40. (in Russ.)
5. Heller B., Sherwood R., McKeown N. The controller placement problem. *Proceedings of the Special Interest Group on Data Communication, SIGCOMM '12, 13 August–17 August 2012, Helsinki, Finland. Special October issue ACM SIGCOMM Computer Communication Review*. New York: ACM; 2012;42(4):473–478. DOI:10.1145/2377677.2377767
6. Yao G., Bi J., Li Y., Guo L. On the Capacitated Controller Placement Problem in Software Defined Networks. *IEEE Communications Letters*. 2014;18(8):1339–1342. DOI:10.1109/LCOMM.2014.2332341
7. Dixit A., Hao F., Mukherjee S., Lakshmanet T.V., Kompella R. Towards an elastic distributed SDN controller. *Proceedings of the Special Interest Group on Data Communication, SIGCOMM '13, 16 August 2013, Hong Kong, China. Special October issue ACM SIGCOMM Computer Communication Review*. New York: ACM; 2013;43(4):7–12. DOI: 10.1145/2534169.2491193
8. Ozsoy F.A., Pinar M.C. An exact algorithm for the capacitated vertex pcenter problem. *Computers & Operations Research*. 2006;33(5):1420–1436. DOI:10.1016/j.cor.2004.09.035
9. Sahoo K.S., Sarkar A., Mishra S.K., Sahoo B., Puthal D., Obaidat M.S., et al. Metaheuristic Solutions for Solving Controller Placement Problem in SDN-based WAN Architecture. *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) and 8th International Conference on Data Communication Networking (DCNET), Madrid, Spain, 15–23 July 2017*. SciTePress Digital Library; 2017. p.15–23. DOI:10.5220/0006483200150023
10. Rath H.K., Revoori V., Nadaf S.M., Simha A. Optimal controller placement in Software Defined Networks (SDN) using a non-zero-sum game. *Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks, 19 June 2014, Sydney, Australia*. IEEE; 2014. DOI:10.1109/WoWMoM.2014.6918987
11. Ksentini A., Bagaa M., Taleb T., Balasingham I. On using bargaining game for Optimal Placement of SDN controllers. *Proceedings of the International Conference on Communications, ICC, 22–27 May 2016, Kuala Lumpur, Malaysia*. IEEE; 2016. DOI:10.1109/ICC.2016.7511136
12. Ateya A.A., Muthanna A., Vybornova A., Algarni A.D., Abuarqoub A., Koucheryavy Y., et al. Chaotic salp swarm algorithm for SDN multi-controller networks. *Engineering Science and Technology, an International Journal*. 2019;22(4):1001–1012. DOI:10.1016/j.jestch.2018.12.015
13. Killi B.P., Rao S.V. Capacitated Next Controller Placement in Software Defined Networks. *IEEE Transactions on Network and Service Management*. 2017;14(3):514–527. DOI:10.1109/TNSM.2017.2720699
14. Chen W, Chen C, Jiang X, Liu L. Multi-Controller Placement Towards SDN Based on Louvain Heuristic Algorithm. *IEEE Access*. 2018;6:49486–49497. DOI:10.1109/ACCESS.2018.2867931
15. Wang G, Zhao Y, Huang J, Duan Q, Li J. A K-means-based network partition algorithm for controller placement in software defined network. *Proceedings of the International Conference on Communications, ICC, 22–27 May 2016, Kuala Lumpur, Malaysia*. IEEE; 2016. DOI:10.1109/ICC.2016.7511441
16. Kuang H., Qiu Y., Li R., Liu X. A Hierarchical K-Means Algorithm for Controller Placement in SDN-Based WAN Architecture. *Proceedings of the 10th International Conference on Measuring Technology and Mechatronics Automation, ICMTMA, 10–11 February 2018, Changsha, China*. IEEE; 2018. p.263–267. DOI:10.1109/ICMTMA.2018.00070
17. Hu Y., Wang W., Gong X., Que X., Cheng S. BalanceFlow: Controller load balancing for OpenFlow networks. *Proceedings of the 2nd International Conference on Cloud Computing and Intelligent Systems, 30 October 2012–01 November 2012, Hangzhou, China*. IEEE; 2013. p.780–785. DOI:10.1109/CCIS.2012.6664282


Статья поступила в редакцию 02.05.2023; одобрена после рецензирования 10.05.2023; принята к публикации 17.05.2023.

The article was submitted 02.05.2023; approved after reviewing 10.05.2023; accepted for publication 17.05.2023.

Информация об авторе:

МУТХАННА
Аммар Салех Али

кандидат технических наук, доцент кафедры сети связи и передачи данных Санкт-Петербургского государственного университета телекоммуникаций им. проф. М.А. Бонч-Бруевича

 <https://orcid.org/0000-0003-0213-8145>