МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ М.В.ЛОМОНОСОВА
ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И КИБЕРНЕТИКИ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Романюк Полина Дмитриевна

Маршрутизация с повышенным уровнем безопасности в ad hoc сетях на основе обучения с подкреплением

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
Черепнев М.А., профессор кафедры ИБ, д. физ.-мат. наук

Москва, 2024

Оглавление

Введение

В современном мире большой проблемой является обеспечение целостности и конфиденциальности коммуникаций. Главной причиной этого является простота перехвата сообщения злоумышленником в широко распространенных сейчас централизованных сетях. Они полагаются на ведущий сервер, без которого два устройства в централизованных сетях не могут связаться между собой. Очевидно, что если злоумышленник контролирует центральный узел, то он может получить сообщение целиком и дешифровать его. Сеть ad hoc - это способ беспроводного общения группы устройств друг с другом без центрального контроллера. Это пример одноранговой сети, в которой все устройства общаются друг с другом напрямую[1].

В настоящий момент сети ad hoc не используются массово, в основном только при различных чрезвычайных ситуациях [2], в том числе и при социальных волнениях [3]. «Специально разработанные с использованием «ячеистой топологии» приложения для мобильных устройств целенаправленно использовались, показав высокую эффективность в ходе «Арабской весны» в Тунисе, Египте и Йемене. Именно поэтому уже в ходе последней Цветной Революции в Гонконге тысячи протестующих жителей общались между собой, минуя сети операторов, с помощью мобильного мессенджера FireChat, который использует только Wi-Fi и Bluetooth» [4].Firechat был создан выпускником мехмата МГУ Станиславом Шалуновым, одним из основателей фирмы OpenGarden.

Использование обучения с подкреплением для создания алгоритмов маршрутизации в ad hoc сетях является активно развиваемым направлением [5].

Цель — построить алгоритм маршрутизации, равномерно распределяющий трафик по всем узлам сети. Это снизит нагрузку на сеть и не позволит нарушителю после нескольких наблюдений за сетью выявить наиболее загруженные узлы и атаковать их.

1. Обучение с подкреплением

Обучение с подкреплением [6] (англ. reinforcement learning, RL) — это наука о принятии решений. Речь идет об обучении оптимальному поведению в окружающей среде для получения максимального вознаграждения. Марковский процесс принятия решений (англ. Markov decision process, MDP), дает нам возможность формализовать последовательное принятие решений. Эта формализация является основой структурирования задач, которые решаются с помощью обучения с подкреплением. В MDP у нас есть лицо, прини-

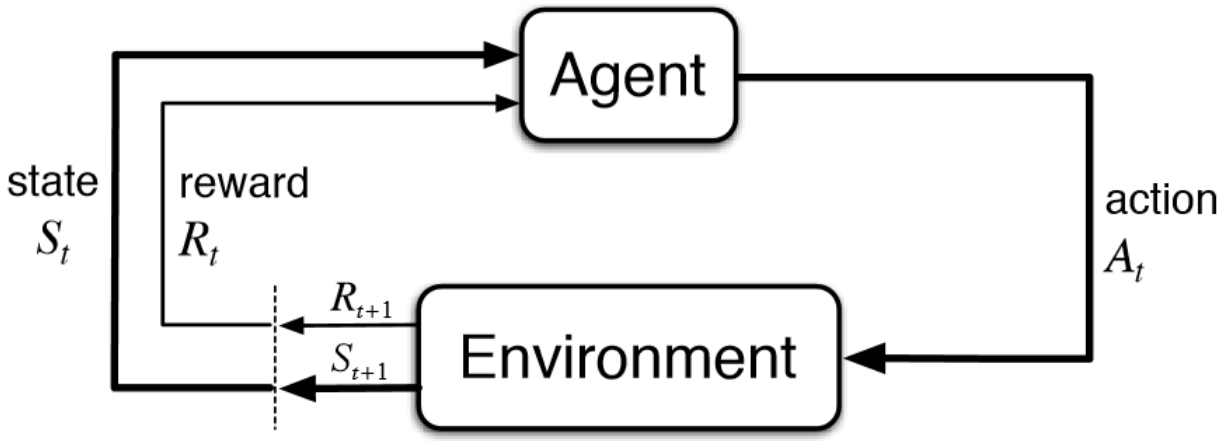

Рис. 1. Схема обучения с подкреплением

мающее решения, называемое агентом, которое взаимодействует со средой, в которой оно находится, как это показано на рисунке выше. Эти взаимодействия происходят последовательно во времени. На каждом временном этапе агент будет получать некоторое представление о состоянии среды. Учитывая это представление, агент выбирает действие, которое необходимо предпринять. Затем среда переводится в новое состояние, и агент получает вознаграждение как следствие предыдущего действия. Компоненты RL, используемые в MDP:

1) Агенты (A): Агенты являются центральным понятием в области RL. Агент RL — это автономная сущность или вычислительная система, которая взаимодействует со средой для обучения и принятия решений для достижения конкретных целей или максимизации совокупного вознаграждения. Эти Агенты могут быть реализованы в различных областях: от робототехники до игр и систем рекомендаций. Например, Агентом

может быть игрок в игре, мотивом которого является победа в игре или получение максимального вознаграждения.

2) Среда (E): Среда — это четко определенная и структурированная система, с которой агент RL взаимодействует на каждом временном этапе. Она представляет область, в которой действия агента,определенные ниже, имеют последствия, и предоставляет агенту обратную связь на основе этих действий.

3) Состояния (S): Среда имеет набор возможных состояний, которые представляют собой различные ситуации, конфигурации или условия, в которых она может находиться. Состояния предоставляют информацию о текущем контексте Среды.

4) Действия (A): среда определяет набор допустимых действий, которые может предпринять Агент RL. Действия — это решения, принимаемые агентом с целью повлиять на состояние окружающей среды.Выбор Действий Агентом определяет стратегия. Стратегия — это функция, которая отображает состояние на действие. В зависимости от контекста и решаемой задачи стратегия может быть детерминированной или стохастической. Детерминированная стратегия — это стратегия, которая точно сопоставляет каждое состояние с одним действием. Другими словами, Агент всегда будет предпринимать одно и то же Действие при заданном состоянии. Стохастическая стратегия — это стратегия, которая сопоставляет каждое состояние с распределением вероятности действий. Основное различие между детерминированной и стохастической стратегиями заключается в том, как они выбирают действия. Детерминированная стратегия выбирает одно действие для каждого Состояния, в то время как стохастическая стратегия выбирает из вероятностного распределения действий для каждого Состояния.

5) Награды (R): после каждого Действия Среда предоставляет Агенту RL числовой сигнал, называемый вознаграждением. Награда указывает на немедленную желательность или качество предпринятого действия. Награды получаются агентом после выполнения определенного действия в определенном состоянии. Обычно оно представляется функцией воз-

награждения R(s, a,s), которая присваивает числовое значение каждой тройке состояние-действие-состояние.

6) Условие завершения: Некоторые задачи RL имеют определенное условие завершения, которое определяет, когда заканчивается эпизод (последовательность взаимодействий). Условие завершения может быть основано на достижении определенного состояния, определенного количества временных шагов или других критериев.

7) Наблюдения. Используются Агентом как вспомогательный критерий для принятия решений.

Цель MDP обычно состоит в том, чтобы найти стратегию π^*, которая максимизирует ожидаемое совокупное вознаграждение с течением времени. Таким образом, марковский процесс принятия решений обеспечивает формальную основу для моделирования последовательных задач принятия решений, включающих состояния, действия, вероятности перехода, вознаграждения и политики (Обучение с подкреплением). Он широко используется в различных областях, включая искусственный интеллект, исследования операций, экономику и робототехнику, для моделирования и решения задач принятия решений в условиях неопределенности.

2. Описание структуры модели

Моделирование осуществляется на статичной топологии из 10-ти узлов, расположенных по схеме кольцо (рисунок 2). Каждый узел реализует протокол маршрутизации MANET, суть которого состоит в том, что узел, обладая информацией о своих соседях, определяет, кому из соседей передать исходящий или транзитный пакет данных, чтобы тот в конце достиг точки назначения. Опрашивая своих соседей пакетами служебного трафика, узел определяет топологию сети вокруг себя. Получая транзитный пакет данных узел соизмеряет адрес получателя и тот маршрут, который пакет уже прошёл и принимает решение, какому из своих узлов-соседей передать пакет дальше.

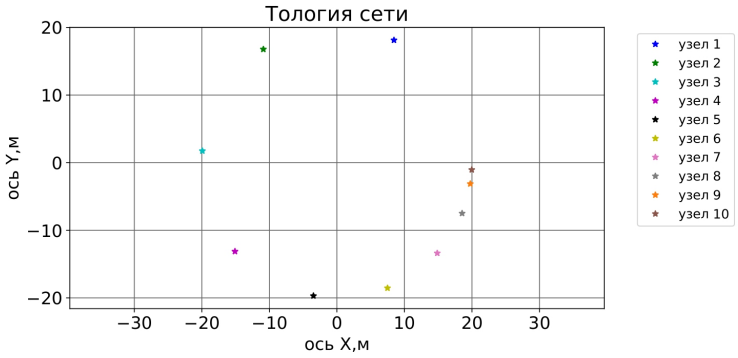

Рис. 2. Топология сети

Задача построения маршрутизации состоит в решении ряда связанных задач:

— Построить и держать актуальной информацию о топологии сети для каждого узла сети, при этом, не допуская излишнего роста служебного трафика;

— Построить маршрут, по которому по сети проходит пакет данных от отправителя до получателя, при том, что на каждом шаге маршрута решение принимает каждый узел изолированно на основе тех данных о топологии сети, который у него есть. Чем короче получившийся маршрут, тем лучше;

— Обеспечить приемлемое время передачи пакетов данных от отправителя к получателю. Конкретные узлы сети могут быть загружены дру-

гими пакетами данных, служебным траффиком, что приведёт к созданию очередей, когда пакеты данных не передаются от узла к узлу, а ожидают освобождения канала связи.

Данные три задачи не могут быть решены одновременно, поскольку чем больше узел знает о топологии сети, тем эффективнее он может построить маршрут с минимальным числом транзитных узлов, но при этом тем больше служебного траффика будет вынужден узел создавать и передавать, что приведёт к перегрузке узлов сети, созданию очередей и увеличению времени передачи полезного пакета данных по сети.

Сеанс моделирования состоит в моделировании 1500 тактов работы сети со случайными событиями —– инициациями узлами-отправителями пакетов данных для узлов получателей. Моменты, когда пакет данных будет создан в сети — случайные, адреса отправителя и получателя — случайные. Служебный траффик, создаваемый узлами сети в модели —– неслучаен и определяется протоколом MANET и выбранной стратегией маршрутизации узлов в текущем сеансе моделирования.

По результатам моделирования сеанса определяется, сколько пакетов данных было передано. Чем больше, тем лучше. Но, из-за случайной природы каждого сеанса моделирования, данное число варьируется.

Стратегия маршрутизации состоит в том, что узел в каждый момент времени принимает решение о создании пакета служебных данных для получения информации о своём окружении, или бездействии. При получении пакета данных узел принимает решение, какому из своих соседей его передать далее. Таким образом, за сеанс моделирования каждый узел принимает решения не менее 1500 раз, более, если через него проходил полезный траффик данных. При каждом принятии решения узел оперирует той информацией о топологии своего окружения, которая ему доступна в момент принятия решения. Доступная информация, в свою очередь, определяется теми инициативами по созданию и передаче пакетов служебного траффика, которые узел принял в предыдущие момент времени. Таким образом, за сеанс моделирования приходится более 15 000 событий, состоящих в принятии решений узлами сети в каждый такт моделирования, и каждое такое решение принимается по имеющейся у узла информации о топологии. Завершение сеанса моделирования показывает, сколько пакетов данных было передано

сетью в целом, что и является мерой эффективности стратегии маршрутизации, реализуемой узлами.

Стратегию агента моделирует нейронная сеть, которая на вход принимает указанные события и стремится максимизировать функцию вознаграждения. Функция вознаграждения — это количество доставленных пакетов данных. Как уже отмечалось выше, функция вознаграждения является совокупностью ряда компромиссных решений, которые напрямую зависят от стратегии маршрутизации, реализуемой нейронной сетью.

3. MDP в математическом виде

В данном разделе будет формулироваться MDP в математическом виде для модели [7] к статье [8], из которой будут взяты формулы для данной задачи.

Сначала определим количество пакетов $d_u[t]$, успешно доставленных UE[1] u в течение t-го временного интервала. Введем обозначения k — идентификатор кодовой книги, используемой TX[2], T^{slot} — вся длительность временного слота, T^{meas} — продолжительность тестирования каждой пары лучей, эффективный коэффициент C_k^{normal} — отношение времени фазы DT к T^{slot}. Тогда:

$$C_k^{normal} = 1 - \left\lceil \frac{N_k^{beam}}{N_{AP}^{arr}} \right\rceil \left\lceil \frac{N_1^{beam}}{N_{UE}^{arr}} \right\rceil \frac{T^{meas}}{T^{slot}} \tag{3.1}$$

Рассматривая все пространственное пространство как полный круг, выполняется слежение за лучом в пределах кругового сектора фиксированного размера. Этот круговой сектор (он же область отслеживания луча) делится поровну на направления, связанные с текущей парой лучей. Затем квантуется размер этого кругового сектора в радиусе и обозначаем его как φ^{track}. Далее обозначим F_k^{track} как минимальное подмножество лучей в k-й кодовой книге, которое может покрыть область отслеживания целевого луча. Соответственно, имеется $F_k^{track} = \left\lceil \frac{\varphi^{track}}{2\pi} N_k^{beam} \right\rceil$ В таком случае эффективный коэффициент следящего слота C_k^{track} равен:

$$C_k^{track} = 1 - |F_k^{track}||F_1^{track}| \frac{T_{meas}}{T_{slot}} \tag{3.2}$$

Пусть I_{tx} — идентификатор TX, I_{rx} — идентификатор RX[3], $C_{k,I_{tx},I_{rx}}^{outage}[t]$ — отношение продолжительности отключения канала к продолжительности всей фазы DT(Это случайная переменная, поскольку она зависит от подвижности приемопередатчиков и состояния канала), $C^{eff-main}[t]$ — итоговый эффективный коэффициент для основной линии, $C^{eff-d2d}[t]$ — итоговый эффективный коэффициент для D2D[4] линии, $I_{main-rx}[t]$ —идентификатор RX в

[1]U пользователей мобильных устройств, англ U mobile users

[2]Передатчик (англ. transmitter)

[3]Приемник (англ. receiver)

[4]Устройство-устройство (англ.Device-to-device)

основном канале, $I_{main-dest}[t]$ — идентификатор назначения пакетов, передаваемых по основному каналу. Тогда:

$$C^{eff-main}[t] = (1-C^{outage}_{I_{cb}[t],0,I_{main-rx}[t]}[t])\times[C^{normal}_{I_{cb}[t]}(1-1^T b^{track}[t])+C^{track}_{I_{cb}[t]}1^T b^{track}[t]] \tag{3.3}$$

$$C^{eff-d2d}[t] = (1 - C^{outage}_{1,I_{d2d-tx}[t],I_{d2d-rx}[t]}[t]) \tag{3.4}$$

где $1^T b^{track}[t] = I_{track}[t-1]$ Обозначим j_{tx} и j_{rx} — это идентификаторы лучей, используемых TX и RX соответственно, $F[t]$ — набор лучей, используемых TX, $rss_{main}[t]$ — максимальное значение RSS, полученное после сканирования всех пар лучей, rss_m — минимальный RSS, необходимый для поддержки MCS m, $R_{main}[t]$ — мгновенная скорость передачи данных по основному каналу связи, $R_{d2d}[t]$ — мгновенная скорость передачи данных по каналу D2D. Значит:

$$rss_{main}[t] = \max_{j_{tx}\in F[t], j_{rx}\in[N_1^{beam}]^+} f(t, I_{cb}[t], 0, I_{main-rx}[t], j_{tx}, j_{rx}) \tag{3.5}$$

где $F[t] = [N^{beam}_{I_{cb}[t]}]^+$ для обычного временного интервала и $F[t] = F^{track}_{I_{cb}[t]}$ для временного интервала отслеживания.

$$R_{main}[t] = \max_{m\in[M]} \mathbb{1}\{rss_{main} \geqslant rss_m\}R_m \tag{3.6}$$

$$rss_{d2d}[t] = \max_{j_{tx}\in F[t], j_{rx}\in[N_1^{beam}]^+} f(t, 1, cI_{d2d-tx}[t], I_{d2d-rx}[t], j_{tx}, j_{rx}) \tag{3.7}$$

$$R_{d2d}[t] = \max_{m\in[M]} \mathbb{1}\{rss_{d2d} \geqslant rss_m\}R_m \tag{3.8}$$

Пусть $d_{main}[t]$ — количество пакетов, отправленных по основному каналу связи, $d_{d2d}[t]$ — количество пакетов, отправленных по каналу связи D2D.

$$d_{main}[t] = \left\lfloor \frac{R_{main}[t]C^{eff-main}[t]}{S^{pkg}} \right\rfloor \tag{3.9}$$

$$d_{d2d}[t] = \left\lfloor \frac{R_{d2d}[t]C^{eff-d2d}[t]}{S^{pkg}} \right\rfloor \tag{3.10}$$

И наконец

$$d_u[t] = \begin{cases} d_{main}[t]\mathbb{1}\{I_{main-dest}[t] = I_{main-rx}[t]\} & u = I_{main-dest}[t] \\ min(d_{main}[t-1], d_{d2d}[t])b^{d2d}_u[t] & u = I_{d2d-rx}[t] \\ 0 & \text{иначе} \end{cases} \tag{3.11}$$

Теперь сформулируем задачу MDP в математическом виде для модели [7].

1) Агенты(A): Агентом является акторная сеть. Акторная сеть состоит из NN[5], параметризованного θ, и дополнительного вычислительного слоя, который назван обработчиком ограничений (Constraint handler). Выходом NN является вектор вероятностей, обозначаемый как $\tilde{\pi}_\theta$. Обработчик ограничений генерирует булев вектор, который имеет тот же размер, что и $\tilde{\pi}_\theta$, и обозначается как $m(s)$. Обработчик ограничений устанавливает элементы $m(s)$ в ноль или единицу по правилу: элемент равен нулю, если его индекс соответствует одному из невыполнимых действий, нарушающих ограничения расписания. Напомним, что ограничения таковы:

   а) Если UE запланирован как TX или RX в канале D2D, этот UE не может быть запланирован как получатель (т.е. RX) в основном канале на тот же временной слот.

   б) Если контроллер решает в t-м временном слоте отслеживать UE в последовательном временном слоте, то этот UE должен быть запланирован как пункт назначения (т. е. RX) в основном канале в (t + 1)-м временном слоте.

   В результате на выходе сети агентов получается стохастическая стратегия, представленная в виде $\pi_\theta(a|s) \sim \tilde{\pi}_\theta(a|s) \odot m(s)$, где $\pi(a|s)$ — вероятность выбора действия a при наблюдении состояния s.

2) Среда (E): Средой является сеть ретрансляторов. Определим для неё функции значения состояния $v^\pi[s[t]]$, значения состояния-действия $Q^\pi[s[t], a[t]]$ и преимущества $A^\pi[s[t], a[t]]$ [9]:

$$v^\pi[s[t]] = \mathbb{E}_{a[t],s[t+1],a[t+1]\ldots}\{G[t]|s[t]\} \tag{3.12}$$

$$Q^\pi[s[t], a[t]] = \mathbb{E}_{s[t+1],a[t+1]\ldots}\{G[t]|s[t], a[t]\} \tag{3.13}$$

$$A^\pi[s[t], a[t]] = Q^\pi[s[t], a[t]] - v^\pi[s[t]] \tag{3.14}$$

   где кумулятивное дисконтированное вознаграждение за t-й временной интервал, обозначаемое $G[t]$, определяется как [10]:

$$G[t] = \sum_{l=t}^{=+\infty} \gamma^{l-t} r[l] \tag{3.15}$$

[5]Нейронная сеть (англ. neural network)

3) Состояния(S): $s[t] = (q[t], b^{d2d}[t], b^{track}[t], l^{block}[t])$ — наблюдаемое состояние для t-го временного интервала, где:

— $q[t] = [q_1[t], ..., q_U[t]]^T$ — вектор состояния очереди, где $q_u[t]$ — количество пакетов в очереди UE u в начале t-го временного слота.

$$q[t+1] = q[t] - d[t]^+ + z[t] \tag{3.16}$$

где:

— U — количество поддерживаемых точкой доступа очередей для соответствующей буферизации пакетов для UE.

— $\{\cdot\}^+ \triangleq max(\cdot, 0)$

— $d[t] = [d_1[t], ..., d_U[t]]^T$ — вектор отправления пакетов.

— $z[t] = [z_1[t], ..., z_U[t]]^T$ — соответствующий прибытию пакетов случайный процесс, где $z_u[t]$ — количество прибывших пакетов для UE u в течение t-го временного слота. Он независимо и идентично распределен по временным слотам, а его ожидание E z[t] равно $\lambda = [\lambda_1, ..., \lambda_U]$.

— $b^{d2d}[t] = b_1^{d2d}[t], ..., b_U^{d2d}[t]^T$ — вектор состояния D2D, где $b_u^{d2d}[t]$ — двоичная переменная, которая равна 1, когда UE u активирован в канале D2D в t-м временном слоте.

$$b_u^{d2d}[t+1] = \mathbb{1}\{I_{main-dest}[t] \neq I_{main-rx}[t]\} \times \tag{3.17}$$

$$\times \quad \mathbb{1}\{I_{main-dest}[t] = u \quad or \quad I_{main-rx}[t] = u\} \tag{3.18}$$

— $b^{track}[t] = [b_1^{track}[t], ..., b_U^{track}[t]]^T$ — вектор состояния слежения, где $b_u^{track}[t]$ — двоичная переменная, которая равна 1, когда t-й временной слот является слотом отслеживания для UE u.

$$b_u^{track}[t+1] = I_{track}[t]\mathbb{1}\{I_{main-rx}[t] = u\} \tag{3.19}$$

где $I_{track}[t]$ — двоичная переменная, которая равна 1, когда система решает, что временной интервал $t + 1$ должен стать интервалом отслеживания.

— $l^{block}[t] = [l_1^{block}[t], ..., l_U^{block}[t]]^T$ — вектор состояния блокировки, где $l_u^{block}[t]$ — количество временных интервалов, прошедших с момента последнего наблюдаемого блокирования UE.

Пространство состояний S состоит из всех возможностей $s[t]$. В S существует счетное бесконечное множество дискретных состояний, так как $q[t]$ и $l^{block}[t]$ могут состоять из любых неотрицательных целых чисел. Предполагается, что начальное состояние $s[0]$ соответствует распределению D_0.

4) Действия (A): пространство действий A состоит из всех выполнимых действий, которые потенциально может предпринять агент.
$a[t] = (I_{main-dest}[t], I_{main-rx}[t], I_{cb}[t], I_{track}[t])$ — действие в t-й временной слот, где $I_{cb}[t]$ — идентификатор кодовой книги, используемой точкой доступа для t-го временного слота,
$I_{main-dest}[t] \in [U]^+, I_{main-rx}[t] \in [U]^+, I_{cb}[t] \in [K]^+ = 1, \ldots, K, I_{track}[t] \in \{0, 1\}$. Стоит отметить, что $I_{track}[t]$ должен быть равен 0, когда $I_{main-dest}[t] \neq I_{main-rx}[t]$, что связано с тем, что слежение за ретранслятором в исследуемой системе не разрешено. В результате размер пространства действий A вычисляется как $|A| = U^2K + UK$, где U^2K соответствует сценарию, когда слежение за лучом не активировано ($I_{track}[t] = 0$), а UK соответствует другому сценарию ($I_{track}[t] = 1$).

5) Награды (R): R выбирается как количество пакетов, доставленных в пункт назначения, т. е. $r[t] = 1^T d[t]$, где 1 — вектор из всех единиц с контекстно-зависимой размерностью. Так апроксимируется исходную задачу минимизации средней задержки пакетов максимизацией ожидаемого кумулятивного дисконтированного вознаграждения, т.е. $\mathbb{E}\{\sum_{l=0}^{=+\infty} \gamma^l r[l]\}$, где $\gamma \in [0, 1]$ — коэффициент дисконтирования, определяющий важность текущих и будущих вознаграждений.

6) Наблюдения. Для нашей задачи роль наблюдателя выполняет критическая сеть. Критическая сеть — это NN, параметризованная ω, которая выдает оценку функции состояния-значения $v^{\pi_\theta}(s)$, обозначаемую $v_\omega^{\pi_\theta}(s)$. Опускается индекс начального слота и обозначим эти T шагов через $\tilde{t} \in [0, T-1]$. В GAE оценивается $Q^\pi(s[\tilde{t}], a[\tilde{t}])$ с помощью линей-

ной комбинации T-шаговых бутстрапов, приведенных ниже [11]:

$$\hat{Q}^{\pi} = r[\tilde{t}] + \gamma r[\tilde{t}+1] + ... + \gamma^{T-\tilde{t}-1} r[T-1] + \gamma^{T-\tilde{t}} v_{\omega}^{\pi_\theta}(s[T]) \tag{3.20}$$

Соответственно, оценка для $A^{\pi}(s[\tilde{t}], a[\tilde{t}])$ может быть получена как [20]:

$$\hat{A}^{\pi} = \hat{Q}^{\pi} - v_{\omega}^{\pi_\theta}(s[\tilde{t}]) = \beta[\tilde{t}] + \gamma\beta[\tilde{t}+1] + ... + \gamma^{T-\tilde{t}-1}\beta[T-1] \tag{3.21a}$$

$$\beta[\tilde{t}] = r[\tilde{t}] + \gamma v_{\omega}^{\pi}(s[\tilde{t}+1] - v_{\omega}^{\pi_\theta}(s[\tilde{t}]) \tag{3.21b}$$

где $\beta[\tilde{t}]$ в литературе по RL называется ошибкой временной разницы (TD). Теперь определим функции потерь для обучения NN. Для начала введём обозначения $\hat{A}[t]$ — это оценка функции преимущества $A^{\pi_\theta}(s[t], a[t])$, π_θ — параметризованная θ стратегия, θ_{old} — параметр последней выученной стратегии, $\rho_\theta = \frac{\pi_\theta(a[t],s[t])}{\pi_{\theta_{old}}(a[t],s[t])}$, $clip(\rho_\theta, 1-\varepsilon, 1+\varepsilon)$ — это функция обрезания, которая ограничивает отношение ρ_θ $1+\varepsilon$ когда $\hat{A}[t] > 0$ и $1-\varepsilon$ когда $\hat{A}[t] < 0$. Тогда:

$$L^{clip}(\theta) = \mathbb{E}\{min(\rho_\theta \hat{A}[t], clip(\rho_\theta, 1-\varepsilon, 1+\varepsilon)\hat{A}[t])\} \tag{3.22}$$

$$L^{entropy}(\theta) = \frac{1}{T}\sum_{\tilde{t}=0}^{T}\sum_{a\in A}\pi_\theta(a|s[\tilde{t}])log(a|s[\tilde{t}]) \tag{3.23}$$

$$L^{actor}(\theta) = -\hat{L}^{clip}(\theta) - c_e L^{entropy}(\theta) \tag{3.24}$$

$$L^{critic}(\omega) = \frac{1}{T}\sum_{\tilde{t}=0}^{T-1}\left(v_{\omega}^{\pi_\theta}(s[\tilde{t}]) - \sum_{l=\tilde{t}}^{T-1}\gamma^{l-\tilde{t}} r[l] - \gamma^{T-\tilde{t}} v_{\omega}^{\pi_\theta}(s[T-1])\right)^2 \tag{3.25}$$

где c_e — перестраиваемый коэффициент. Приближенное значение $L^{clip}(\theta)$ в (3.22), обозначаемое как $\hat{L}^{clip}(\theta)$, может быть получено путем взятия ожидания по наблюдаемой траектории $\tilde{t} \in [0, T-1]$.

При практической реализации NN рассматривается акторную и критическую сети как единую объединенную сеть (взвешенную по θ и ω), которая выводит $\pi_\theta(a|s)$ и $v_{\omega}^{\pi_\theta}(s)$ с разными блоками. Поэтому конечная функция потерь $L^{event-loss}$ имеет вид:

$$L^{event-loss} = L^{critic}(\theta) + L^{critic}(\omega) \tag{3.26}$$

В этой работе сформулирована задача MDP для модели [7] в упрощенном виде. Типичная формулировка может быть сделана путем дальнейшего уточнения условных вероятностей перехода между состояниями, наблюдениями

и условных вероятностей наблюдений [12]. Исходный алгоритм модели, который кратко описан в Алгоритме 1.

Algorithm 1 Исходная модель

1: Входные данные: Общее количество шагов по времени T^{total} , размер батча T , параметр обрезки ε, коэффициент потери энтропии c_e, коэффициент масштабирования данных x, константа $\tilde{N}^{block}$.
2: Инициализация: рандомизация параметров NN θ и ω; $\theta_{old} \leftarrow \theta$; $s[0] \sim D_0$
3: **for** $t = 0...T^{total}$ **do**
4: Масштабируйте состояние $s[t]$ и введите его в акторные/критические сети
5: Сохраните в буфере оценку функции состояния-значения $(v_\omega^{\pi_\theta}(s[\tilde{t}])$, предоставляемая критической сетью
6: Выберите действие $a[t]$, следуя стохастической политике $\pi_{\theta_{old}}(a|s[t])$, предоставленной сетью акторов, и выполните $a[t]$ в системе (она же среда)
7: Получите новое состояние $s[t+1]$, предоставляемого системой
8: Сохраните в буфере масштабированное вознаграждение $r[t]$, предоставляемое системой
9: **if** $Mod(t+1, T) == 0$ **then**
10: Поменяйте индексы у сохраненных в буфере T временных шагов с $\{t, t+1, ..., t+T-1\}$ до $\{0, 1, ..., T-1\}$
11: Вычислите ошибку TD $\beta[\tilde{t}]$, $\tilde{t} \in [0, T-1]$ как в 3.21b.
12: Рассчитайте $\hat{A}^{\pi}$,$\tilde{t} \in [0, T-1]$ как в 3.21a
13: Оптимизируйте параметр NN θ (акторная сеть) по $\nabla_\theta L^{actor}(\theta)$ как в 3.24
14: Оптимизируйте параметр NN ω (критическая сеть) по $\nabla_\omega L^{critic}(\omega)$ как в 3.25
15: $\pi_{\theta_{old}} \leftarrow \pi_\theta$
16: Очистите буфер
17: **end if**
18: **end for**

В данной модели агенты учатся реализовывать маршрутизацию таким образом, чтобы максимизировать пропускную способность сети. При этом, естественно, через некоторые узлы идёт большой трафик, что позволит злоумышленнику, контролирующему такой узел, перехватывать сообщения. В таком случае необходимо обучать агентов так, чтобы они стремились избегать создавать такие узлы «концентраторы трафика». Этого можно добиться этого модификацией алгоритма обучения контроллера.

4. Модификация модели

Более безопасный алгоритм маршрутизации должен стремиться, по мере возможности, к передаче пакетов данных разными маршрутами, через разные транзитные узлы. Обеспечить указанное изменение стратегии маршрутизации можно за счёт модификации функции вознаграждения. Модифицированная функция вознаграждения должна поощрять передачу пакетов через те узлы, через которые на предыдущем такте моделирования пакеты не передавались и штрафовать в противном случае.

Для вычисления штрафной функции необходимо наблюдать за время сеанса моделирования за фактическими маршрутами передачи пакетов и накапливать статистику случаев, когда пакеты данных передавались через одни и те же транзитные узлы. Для этого происходит подсчёт количества узлов, загрузка которых на предыдущем шаге была выше пороговой. Порог установлен в 80%, и может корректироваться в зависимости от желаемой степени безопасности модифицированного алгоритма маршрутизации. Повышая порог вплоть до 100% штрафная функция теряет свой смысл, поскольку при 100% загрузке узел не пропускает через себя трафик и ставит его в очередь, что приводит к снижению общей пропускной способности сети (падение функции вознаграждения). Понижая порог ниже 80% алгоритм маршрутизации начинает искать более агрессивные стратегии маршрутизации, стремясь избегать узлов-концентраторов трафика, что в свою очередь приводит к кардинальному повышению доли служебного трафика, что увеличивает количество загруженных узлов и препятствует реализации агрессивной стратегии маршрутизации. Для модели из 10 узлов эмпирически была получена цифра в 80%, тогда как для больших и более сложных топологий поиск агрессивных стратегий за счёт понижения порога может быть обоснованным.

В исходный алгоритм внесены следующие изменения:

1) при подсчете функции вознаграждения $r[t]$ помимо её значения возвращается число $n^{overload}[t, \%max - load]$, где $\%max - load$ — максимальная загрузка канала D2D. В случае перегрузки выбранного канала оно равно числу пакетов в канале, иначе оно равно нулю.

2) в цикле if после смены индексов сохраненных в буфере T временных шагов с $\{t, t+1, ..., t+T-1\}$ до $\{0, 1, ..., T-1\}$ и до вычисления ошиб-

ки TD функция вознаграждения считается заново. От значения функции вознаграждения $r[t]$ зависит TD, от TD зависит $\hat{A}^{\pi}$, оценка для $A^{\pi}(s[\tilde{t}], a[\tilde{t}])$, преимущества. Таким образом, снижается ценность перегруженного маршрута.

Тестируемая функция вознаграждения имеет вид

$$r[\tilde{t}] = \left(1 - \frac{n^{overload}[\tilde{t}]}{num - nodes}\right) r[\tilde{t}] \tag{4.1}$$

Функция вознаграждения имеет представленный в 4.1 вид, потому что таким образом можно учесть при подсчете вознаграждения количество перегруженных узлов между двумя обновлениями стохастической стратегии, что в программной реализации исходного и измененного алгоритмов является обучением нейронной сети. Перегруженный узел — это такой узел, который не может в момент моделирования передавать через себя пакеты из-за перегрузки своих каналов связи с соседями. Пакеты в таком узле ставятся в очередь. Каждый узел за время сеанса моделирования может быть перегружен или нет 1500 раз, по разу за каждый такт моделирования.

Все внесенные в исходный алгоритм изменения представлены в алгоритме 2. Они выделены красным цветом.

Algorithm 2 Измененная модель

Входные данные: Общее количество шагов по времени T^{total} , количество узлов T, параметр обрезки ε, коэффициент потери энтропии c_e, коэффициент масштабирования данных x, константа $\tilde{N}^{block}$, максимальная загрузка узла $\%max - load$.
Инициализация: рандомизация параметров NN θ и ω; $\theta_{old} \leftarrow \theta$; $s[0] \sim D_0$
for $t = 0...T^{total}$ **do**
    Масштабируйте состояние $s[t]$ и введите его в акторные/критические сети
    Сохраните в буфере оценку функции состояния-значения $(v_\omega^{\pi_\theta}(s[\tilde{t}])$, предоставляемая критической сетью
    Выберите действие $a[t]$, следуя стохастической политике $\pi_{\theta_{old}}(a|s[t])$, предоставленной сетью акторов, и выполните $a[t]$ в системе (она же среда)
    Получите новое состояние $s[t+1]$, предоставляемого системой
    Сохраните в буфере масштабированное вознаграждение $r[t]$, $n^{overload}[t, \%max - load]$, предоставляемых системой
    if $Mod(t+1, T) == 0$ **then**
        Поменяйте индексы у сохраненных в буфере T временных шагов с $\{t, t+1, ..., t+T-1\}$ до $\{0, 1, ..., T-1\}$
        $num - nodes$ — число узлов
        for $\tilde{t} = 0...T-1$ **do**
            $r[\tilde{t}] = \left(1 - \frac{n^{overload}[\tilde{t}]}{num-nodes}\right) r[\tilde{t}]$
        end for
            Вычислите ошибку TD $\beta[\tilde{t}]$, $\tilde{t} \in [0, T-1]$ как в 3.21b
            Рассчитайте $\hat{A}^\pi = \hat{Q}^\pi$, $\tilde{t} \in [0, T-1]$ как в 3.21a
            Оптимизируйте параметр NN θ (акторная сеть) по $\nabla_\theta L^{actor}(\theta)$ как в 3.24
            Оптимизируйте параметр NN ω (критическая сеть) по $\nabla_\omega L^{critic}(\omega)$ как в 3.25
            $\pi_{\theta_{old}} \leftarrow \pi_\theta$
            Очистите буфер
        end if
    end for

Полученные после тестирования исходной и измененной моделей графики продемонстрированы на рисунке 4.

При выборе штрафной функции для модифицированного алгоритма были рассмотрены три варианта:

1) $r[\tilde{t}] = \left(1 - \frac{n^{overload}[\tilde{t}]}{num-nodes}\right) r[\tilde{t}]$

2) $r[\tilde{t}] = \frac{r[t]+\frac{n^{overload}[\tilde{t}]}{num-nodes}}{2}$

3) $r[t] = \sqrt{r_{prev}[t]} * \sqrt[3]{\left(1 - \frac{n^{overload}[\tilde{t}]}{num-nodes}\right) * r[t]}\ , r_{prev}[0] = 1$

Полученные после их тестирования графики продемонстрированы на рисунке 3.

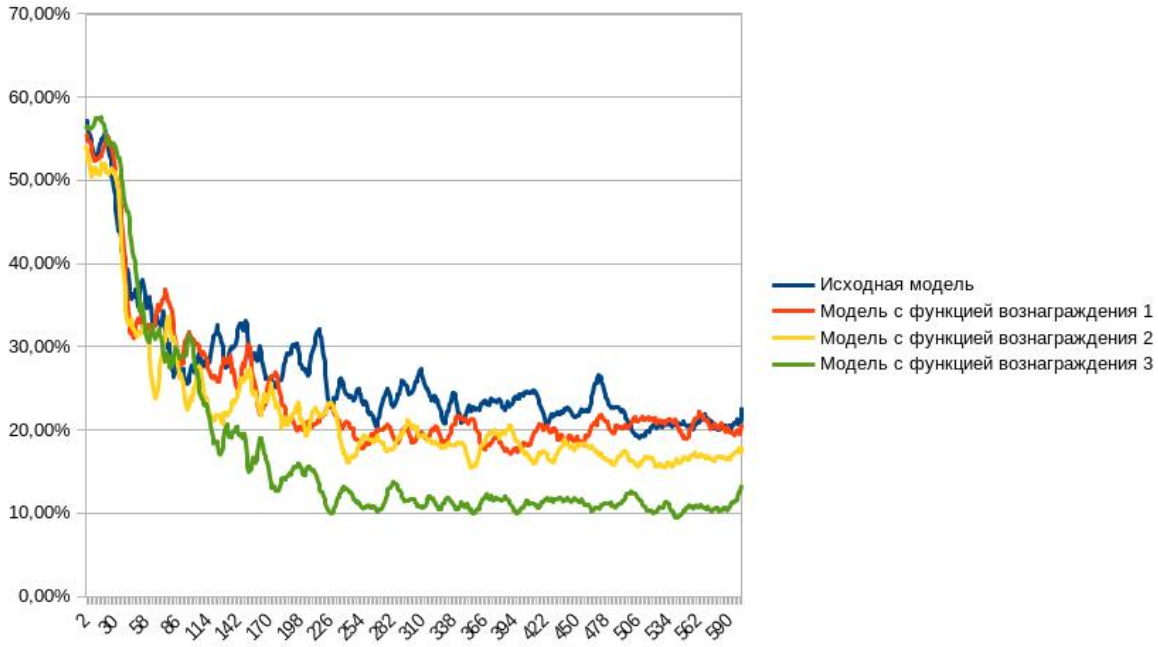

Рис. 3. Кривые обучения. Доля перегруженных узлов

Вариант 3 продемонстрировал более равномерно распределение трафика по узлам сети, что привело к заметному снижению доли нагруженных узлов, но ценой падения общей пропускной способности сети. Сильное падение общей пропускной способности сети было признано нежелательным и вариант был отвергнут. Из вариантов 2 и 1 был выбран вариант 2, как дающий немного более равномерное распределение трафика по сети без падения общей пропускной способности.

Проведём сравнительный анализ исходной и измененной моделей.

Графики близко сходятся после 299 итерации, так как обе модели научились эффективной маршрутизации, благодаря которой сеть не перегружена. На 199 итерации закончился тренд на понижение у графика измененной модели, на 225 итерации — у исходной модели. Значит закончилось эффективное обучение исходной модели на 225 итерации, измененной — 199 итерации, что равносильно тому, что исходная и измененная модели вышли на

стабильный режим на 199 и 225 итерациях соответственно. Поэтому точкой останова выбрана 199 итерация. Также отсюда следует вывод, что обучение модели было сокращено на 26 итераций.

Общее количество полезного трафика, пропускаемого обеими моделями оказалось примерно одинаковым, с учётом случайной природы каждого сеанса моделирования. Из этого можно сделать вывод, что при используемых параметрах моделирования обе модели нашли эффективные стратегии маршрутизации, не приводящие к заметной перегрузке сети. В небезопасном варианте (исходная модель) общий трафик составил 7459 пакетов, тогда как в безопасном варианте (модифицированная модель) 8292. Указаны не доставленные пакеты от отправителя до получателя, а именно общий трафик, с учётом транзитных передач пакетов. Видно, что при равном количестве доставленных пакетов модифицированный алгоритм порождает больше транзитного трафика, чем исходный. При этом, если изучить рис. 4 видно, что обе модели в ходе своего обучения сходятся на примерно 20% нагруженных узлов сети, причём исходная небезопасная модель в среднем даёт немного больше нагруженных узлов.

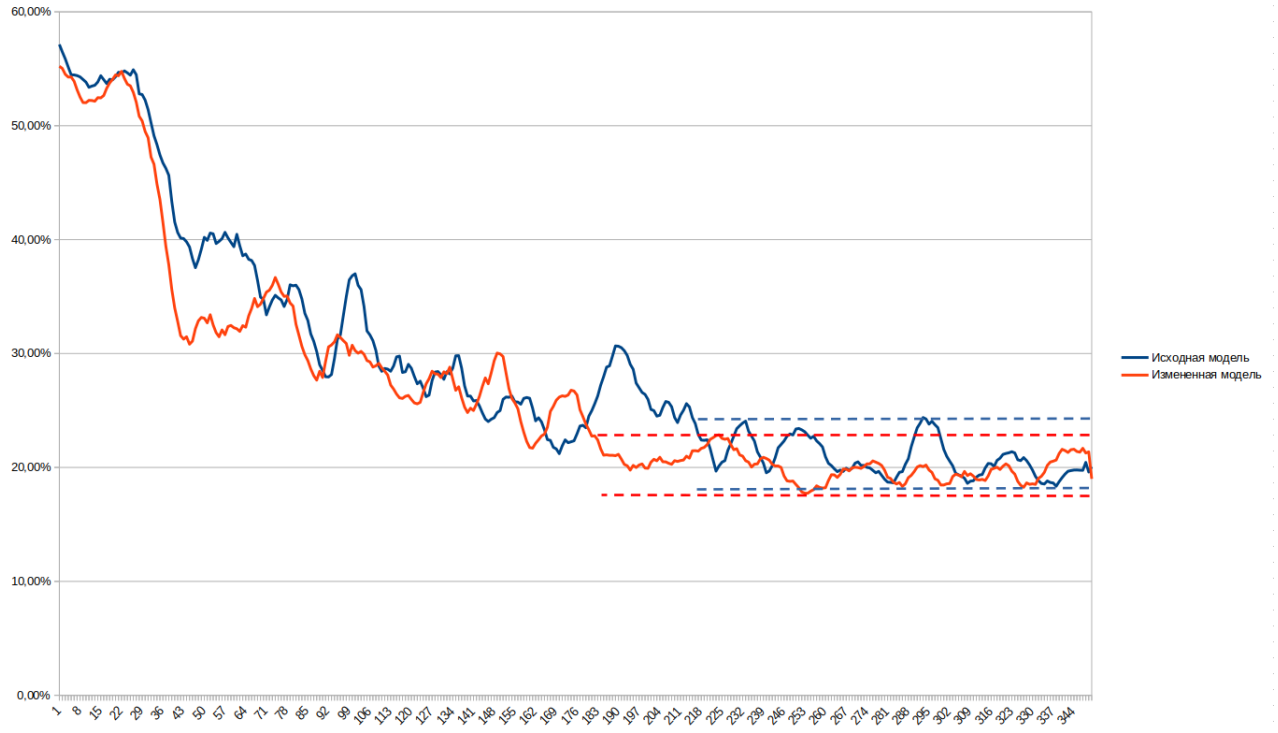

Рис. 4. Кривые обучения. Доля перегруженных узлов

Если при моделировании 20% узлов загружены, то оставшиеся 80% —

не загружены и дополнительный трафик в модифицированной модели передаётся через ненагруженные узлы. Что косвенно указывает на построение альтернативных маршрутов модифицированной сетью.
Проведём более детальный анализ движения трафика в модифицированной и исходной моделях.

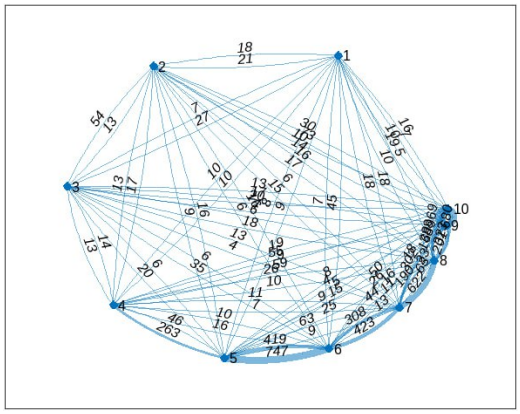

Рис. 5. Граф интенсивности обмена в небезопасном алгоритме

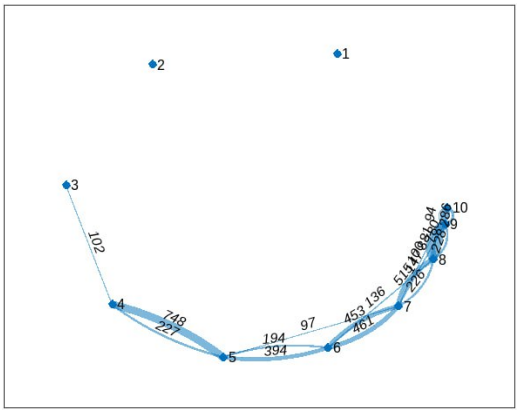

Рис. 6. Граф интенсивности обмена в небезопасном алгоритме. Скрыты ненагруженные связи.

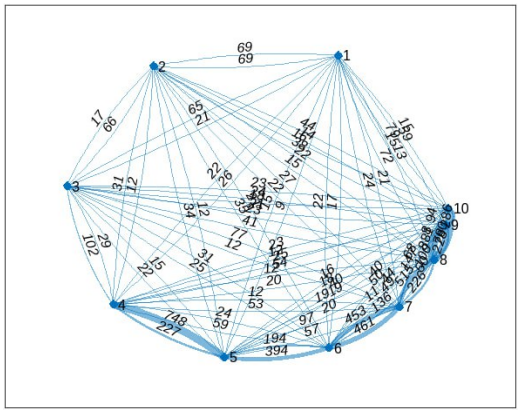

Рис. 7. Граф интенсивности обмена в безопасном алгоритме

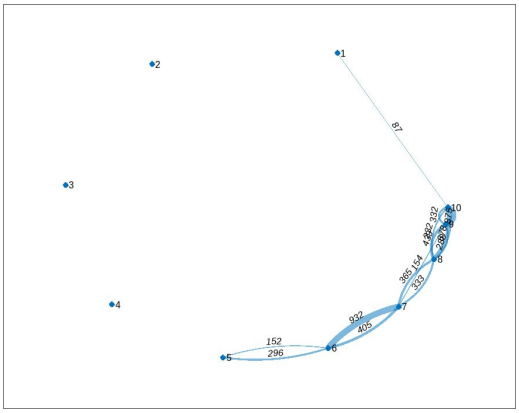

Рис. 8. Граф интенсивности обмена в безопасном алгоритме. Скрыты ненагруженные связи.

На рисунках 5 и 7 показана средняя загрузка связей между каждыми узлами за последние 10 итераций перед точкой останова в безопасном и небез-

опасном алгоритмах. На этих рисунках некоторые связи нагружены выше среднего. Они вынесены отдельно на рисунки 6 и 8. Из них следует, что в обоих случаях компромиссным решением является реализация «хребта» маршрутизации через полукруг, передавая данные от узла к узлу последовательно. Для сравнения графов связей в обоих вариантах рассчитывается общий обмен по нагруженным и ненагруженным связям. В небезопасном варианте по ненагруженным связям обмен составляет 1456 пакетов, по нагруженным 6003 пакетов. В безопасном варианте 2414 и 5878, соответственно. Поэтому в безопасном варианте обмен по ненагруженным связям на (2414-1456)/1456*100% = 65% выше.

Чтобы убедиться, что альтернативные маршруты действительно строятся, необходимо проанализировать фактор динамики — распределение во времени количества проходящих по связи пакетов. Для учета этого фактора в процессе моделирования собирается статистика по загрузке узлов и связей на каждом такте моделирования. Она представлена в таблицах 1,2 где № обозначает номер итерации моделирования.

Таблица 1. Количество раз, когда узел был высокозагружен

	Количество раз, когда узел был высокозагружен в:		Количество раз, когда узел был высокозагружен 2 раза подряд в:	
№	Небезопасном алгоритме.	Безопасном алгоритме.	Небезопасном алгоритме.	Безопасном алгоритме.
189	448	301	138	60
190	416	350	123	79
191	430	336	117	64
192	460	334	131	75
193	414	307	103	67
194	507	339	187	82
195	488	285	159	65
196	485	297	159	72
197	493	318	150	60
198	455	305	140	61
199	429	238	114	42

Таблица 2. Количество раз, когда узел был высокозагружен

	Сокращение доли высокозагруженных..	
№	2 раза подряд узлов	узлов
189	32,81%	56,52%
190	15,87%	35,77%
191	21,86%	45,30%
192	27,39%	42,75%
193	25,85%	34,95%
194	33,14%	56,15%
195	41,60%	59,12%
196	38,76%	54,72%
197	35,50%	60,00%
198	32,97%	56,43%
199	44,52%	63,16%

Данные таблиц показывают следующее:

— Уменьшение высокозагруженных узлов, среднее значение за 10 последних итераций 31,84%;

— Уменьшение постоянно высокозагруженных узлов, среднее значение за 10 последних итераций 50,83%.

Учитывая вероятностный характер моделирования, были взяты результаты для нескольких сеансов моделирования — 10.

Уменьшение количества постоянно высоконагруженных узлов на 50,83% говорит о том, что в безопасном алгоритме половина из высоконагруженных узлов не нагружена два такта подряд. Несмотря на то, что основной поток данных идёт по высоконагруженной хорде, маршрутизация в безопасном режиме позволяет чередовать узлы и обеспечивать в 50% случаев передачу данных через узлы, не использованные в предыдущем такте. Таким образом, в безопасном режиме существует как минимум два альтернативных маршрута в высоконагруженной хорде, тогда как в небезопасном режиме — только один.

Уменьшение количества высоконагруженных узлов на 31,84% говорит о том, что общая нагрузка в безопасном режиме на нагруженной хорде на 31% меньше. Это означает, что 31% потока данных выталкивается из нагруженной хорды на альтернативные маршруты на ненагруженных связях (рис 6 и 8). Таким образом, долю потока данных, выстраиваемых по альтернативным маршрутам (как минимум двум разным) можно оценить как: 31% + (1-31%)*50% = 65%, что согласуется с процентом роста обмена по ненагруженным связям, полученным из рисунков 5, 7.

5. Заключение

За основу была использована модель, реализующая маршрутизацию в MANET сетях методом обучения с подкреплением. Функция вознаграждения данной модели была ориентирована на максимизацию пропускной способности сети. Для обеспечения построения более безопасной стратегии маршрутизации функция вознаграждения была модифицирована, штрафуя передачу трафика по узлам-концентраторам трафика. Модифицированная модель с функцией вознаграждения (4.1) при обучении стремится передавать трафик как минимум по двум альтернативным маршрутам, что и было показано сравнительным анализом моделей. Предложены критерии оценки — статичные, на основе долей перегруженных узлов, динамические — на основе долей перегруженных два такта подряд узлов. Предложены способы визуализации полученных алгоритмов маршрутизации.

Предложенные критерии оценки удовлетворительно согласуются друг с другом и показывают, что безопасный алгоритм маршрутизации действительно работает, выстраивая альтернативные маршруты.

Измененная модель обучается быстрее исходной, так как раньше выходит на стабильный режим.

Список литературы

1. *Linux*. Ad-hoc networking. — URL: archlinux.org. — (accessed: 01.03.2024).
2. Lifeline: Emergency Ad hoc Network //. — 2011.
3. *Баженов С., Баженова Е.* . Гибридная платформа «цветных революций» как системная внешняя угроза национальной безопасности России // Научные сообщения. — 2014.
4. *RBC*. Мессенджер выпускника мехмата МГУ стал инструментом протеста в Гонконге. — URL: https://www.rbc.ru/technology_and_media/30/09/2014/542a70d8cbb20f3bd7068c7a. — (accessed: 01.03.2024).
5. *a-pichugin*. Глубокое обучение (Deep Learning): обзор. — URL: https://habr.com/ru/companies/newprolab/articles/343834/. — (accessed: 01.03.2024).
6. *CENTRE M.* MDP (марковский процесс принятия решений) - RL (обучение с подкреплением). — URL: https://mlcentre.ru/articles/780186/. — (accessed: 09.03.2024).
7. *yzhang417*. DeepRL-mmWave-MANET. — URL: https://github.com/yzhang417/DeepRL-mmWave-MANET/tree/main. — (accessed: 01.03.2024).
8. *Zhang Y., Jr R. W. H.* Reinforcement Learning-Based Joint User Scheduling and Link Configuration in Millimeter-Wave Networks // IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. — 2023. — Т. 22, № 5. — С. 3038—3054.
9. *al V. M. et.* Asynchronous methods for deep reinforcement learning // Proc. Int. Conf. Mach. Learn. — 2016. — Т. 48. — С. 1928—1937.
10. *Sutton R. S., Barto A. G.* Reinforcement Learning: An Introduction. — The MIT Press, 1998.
11. *P. J. S., M. M. S. L., Abbeel I. J. P.* High-dimensional continuous control using generalized advantage estimation //. — 2016.

12. *Meng L.*, *Gorbet R.*, *Kulic´ D.* Memory-based Deep Reinforcement Learning for POMDPs //. — 2021.