

УДК 004.021

doi: 10.15622/rcai.2025.101

ДЕЦЕНТРАЛИЗОВАННАЯ МНОГОКРИТЕРИАЛЬНАЯ МАРШРУТИЗАЦИЯ

Н.П. Фомин (*heartmarshall@yandex.ru*)^A

К.С. Яковлев (*yakovlev@isa.ru*)^{A,B}

^A Санкт-Петербургский государственный университет,
Санкт-Петербург

^B Федеральный исследовательский центр
«Информатика и управление» РАН, Москва

Рассматривается задача децентрализованной многокритериальной маршрутизации в компьютерных сетях, где каждый узел самостоятельно формирует маршрутные решения на основе локально доступной информации. Современные протоколы маршрутизации зачастую ограничены алгоритмами поиска пути, минимизирующими значение скалярной целевой функции, что может приводить к неоптимальному выбору маршрута в сложных сетевых сценариях, когда имеется несколько равнозначных критериев качества. В работе предложены два новых децентрализованных метода маршрутизации, основанных на современных алгоритмах многокритериального поиска. Первый, метод моделирования поведения узлов, позволяет каждому узлу строить локальную таблицу маршрутизации с учетом узла-отправителя, гарантируя соблюдение заданных ограничений стоимости пути и детерминированность. Второй, жадный метод, принимает решение о следующем переходе, основываясь только на адресате пакета, предлагая компромисс между вычислительной сложностью и строгостью ограничений. Экспериментально подтверждена эффективность обоих подходов и представлены их сравнительные характеристики по качеству решений и ресурсоемкости.

Ключевые слова: многокритериальная маршрутизация, Парето-оптимальность, алгоритмы поиска пути, компьютерные сети.

Введение

Компьютерные сети являются важнейшей инфраструктурой современного мира, и от их производительности и стабильности зависит качество многих сфер жизнедеятельности человека. Ключевую роль в функциони-

ровании сетей играет алгоритм маршрутизации пакетов данных, который определяет пути их следования от отправителя к получателю. В большинстве сетей решения о маршрутизации принимаются децентрализованно: каждый узел (маршрутизатор) самостоятельно определяет куда отправить пакет на основе локально доступной ему информации и заложенного в него алгоритма. Выбор оптимального маршрута критически важен, так как каждый транзитный узел вносит задержку и потенциально увеличивает риск ошибок передачи данных.

Традиционные протоколы маршрутизации основаны на адаптации к децентрализованному сценарию работы таких известных (централизованных) алгоритмов поиска пути как, например, Дейкстры [Dijkstra, 1959] или Беллмана-Форда [Bellman, 1958]. Важной особенностью этих (и подобных им) алгоритмов является тот факт, что они являются однокритериальными. Такой подход, с одной стороны, упрощает реализацию, с другой – может приводить к выбору неоптимальных маршрутов на практике, особенно при наличии разнородных требований к качеству соединения, т.е. когда речь идет о необходимости учета нескольких критериев качества маршрута.

Подобный учет возможен в современных алгоритмах многокритериального поиска кратчайшего пути [Stewart et al., 1991], [Mandow et al., 2010], [Hernández Ulloa et al., 2020], [Hernández et al., 2023], способных находить множество Парето-оптимальных решений, т.е. осуществлять многокритериальную оптимизацию. Однако, все эти алгоритмы являются централизованными и, следовательно, напрямую неприменимыми в реальных задачах маршрутизации пакетов в компьютерных сетях, когда каждый узел принимает локальное решение о пересылке пакета на один из смежных узлов, обладая лишь локальной информацией. На устранение этого недостатка и направлена данная работа, целью которой является разработка и исследование новых методов децентрализованной многокритериальной маршрутизации, которые используют возможности современных централизованных алгоритмов поиска, в частности, алгоритма BOD [Hernández et al., 2023].

1. Постановка задачи

Сеть представляется в виде ориентированного графа $G = (S, E)$, где S – множество узлов, E – множество ребер. Каждому ребру $e \in E$ сопоставлен d -мерный вектор положительных стоимостей $c(e) = (c_1(e), \dots, c_d(e))$. Путь $\pi = (s_0, \dots, s_k)$ – последовательность вершин графа, между которыми существуют рёбра. Каждый путь имеет стоимость $C(\pi)$ равную сумме стоимостей рёбер между вершинами пути.

Вектор p доминирует вектор q (обозначается $p < q$), если $\exists i: p_i < q_i$ и $\forall j: p_j \leq q_j$. Путь π является Парето-оптимальным, если его стоимость $C(\pi)$ не доминируется стоимостью любого другого пути между теми же

начальным и конечным узлами. Задача многокритериального поиска кратчайшего пути заключается в нахождении множества всех Парето-оптимальных путей от s_{start} к s_{target} . В постановке “от одного ко многим” результатом для каждого узла s является Парето-оптимальное множество $sols(s)$, содержащее все недоминируемые векторы стоимостей путей из s_{start} в s .

В децентрализованной системе каждый узел $s \in S$ определяет следующий переход $nextHop(p) \in Succ(s)$ для пакета $p = (s_{source}, s_{dest})$, где $Succ(s)$ – множество смежных с s узлов. Совокупность функций $rules = \{nextHop | s \in S\}$ описывает правила маршрутизации. Трассировка $trace(p, rules)$ определяет путь пакета в сети.

Критерии для оценки множества правил $rules$:

1. **Ограниченность:** если \exists путь π от s_{source} к s_{dest} с $C(\pi) \leq C_{limit} = (C_1, \dots, C_d)$, то $C(trace(p, rules)) \leq C_{limit}$.
2. **Детерминированность:** $trace(p, rules)$ всегда приводит к единственному пути.

Экземпляр задачи – пара (G, C_{limit}) . Рассматриваются две постановки для $nextHop_s(p)$: зависимость этой функции от пары (s_{source}, s_{dest}) или только от s_{dest} .

2. Анализ существующих подходов

Задача многокритериальной маршрутизации активно исследуется в контексте обеспечения качества соединения в компьютерных сетях. Существующие подходы можно условно разделить на несколько категорий.

Наиболее распространенным методом является агрегация нескольких метрик в единую скалярную функцию стоимости, как правило, путем их взвешенной суммы. На основе этой объединенной метрики затем выполняется алгоритм поиска кратчайшего пути, например, Дейкстры. Хотя такие протоколы, как OSPF, являются децентрализованными, сам подход со взвешенной суммой имеет фундаментальные недостатки, рассмотренные в обзорных работах по QoS-маршрутизации [Wang & Crowcroft, 1996]. Выбор весовых коэффициентов нетривиален и, по сути, является глобальным решением, плохо адаптирующимся к динамическим условиям сети. Кроме того, любой фиксированный набор весов приводит к потере информации о Парето-оптимальных маршрутах.

Другой класс подходов формулирует задачу как поиск кратчайшего пути с ограничениями. В этом случае одна метрика минимизируется при условии, что остальные не превышают заданных порогов. Эта задача в общем случае является NP-трудной [Jaffe, 1984]. Из-за этого исследования в данной области часто фокусируются на разработке эвристических алгоритмов, которые находят приближенные решения за приемлемое время [Korkmaz & Krungz, 2001]. Однако эвристики не гарантируют нахождения оптимального пути, а их децентрализованная реализация остается сложной задачей.

Именно эту проблему решает предложенный в работе Метод Моделирования. Он является полностью децентрализованным, но, в отличие от стандартных эвристик, путем моделирования принятия решений другими узлами обеспечивает строгие гарантии соблюдения сквозных ограничений. Жадный метод, в свою очередь, представляет собой практический децентрализованный компромисс: он не предоставляет полных гарантий, но, в отличие от простой агрегации метрик, учитывает Парето-оптимальность на локальном уровне. Таким образом, данное исследование предлагает новые механизмы именно для децентрализованной многокритериальной маршрутизации, анализируя компромисс между полнотой гарантий соблюдения ограничений и ресурсоемкостью.

2. Предлагаемые методы

Для решения поставленной задачи децентрализованной многокритериальной маршрутизации предлагаются два метода, использующие алгоритм BOD для нахождения Парето-оптимальных множеств путей $sols(s)$. Методы различаются объемом используемой информации для формирования правил $nextHop_s(p)$ и, как следствие, обеспечиваемыми гарантиями ограниченности и детерминированности в обмен на вычислительную сложность.

При этом стоит отметить невозможность нахождения детерминированного ограниченного метода при зависимости $nextHop_s(p)$ только от s_{dest} (т.е., без учета s_{source}). Доказательством служит следующий сценарий, иллюстрируемый на рис. 1.

Рассмотрим узел s_{cur} , который получил два пакета, оба адресованные узлу s_{goal} , но пришедшие по разным предшествующим путям π и π' (из разных s_{source} , как показано на рис. 1,б). Эти пути имеют различные накопленные стоимости $c(\pi)$ и $c(\pi')$ (рис. 1а). Если s_{cur} принимает решение о следующем переходе, основываясь только на адресате s_{goal} , он должен выбрать один и тот же следующий узел (например, с ребром стоимости (g_1, g_2) или с ребром (g'_1, g'_2) на рис. 1,а) для обоих пакетов. Однако, как видно из рис. 1,а, выбор одного ребра может привести к тому, что суммарная стоимость пути для одного пакета $(c(\pi) + (g_1, g_2))$ уложится в заданные ограничения C_{limit} , в то время как для другого пакета $(c(\pi') + (g_1, g_2))$ эти ограничения будут превышены. Аналогичная ситуация может возникнуть при выборе другого ребра. Таким образом, не существует единого детерминированного выбора следующего перехода для s_{cur} , зависящего только от s_{dest} , который гарантировал бы соблюдение ограничений C_{limit} для обоих пакетов несмотря на то, что для каждого из них по отдельности может существовать допустимый маршрут при учете s_{source} .

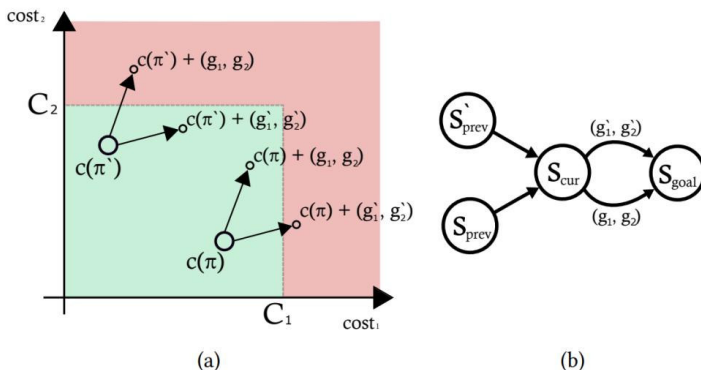


Рис. 1. (a) Пример возможных исходов при продолжении некоторых путей π и π' из вершины s_{cur} и (b) рисунок подграфа, в котором содержится эта вершина

Метод моделирования поведения узлов (Modeling Method)

Данный метод нацелен на построение полного множества правил маршрутизации *rules*, удовлетворяющего критериям ограниченности и детерминированности. Каждый узел *cur_node* формирует свою локальную таблицу маршрутизации $nextHop_{cur_node(p)}$, учитывая как узел-отправитель s_{source} , так и узел-адресат s_{dest} пакета p . Это достигается путем моделирования принятия решений другими узлами для предсказания полных путей и их соответствия ограничениям C_{limit} .

Процесс построения таблицы маршрутизации для *cur_node* включает:

1. Определение множества локально достижимых из *cur_node* узлов (далее R_{loc}), пути к которым удовлетворяют $limit$, и запись начальных правил для них.
2. Определение множества потенциальных узлов-отправителей S_{pot} , для которых *cur_node* достижим в пределах $limit$. Для этого используется BOD на графе с инвертированными ребрами.
3. Для каждой пары $(s_i, S_{pot}, s_j, R_{loc})$ моделируется поиск пути из s_i в s_j . Если оптимальный путь проходит через *cur_node*, то определяется следующий узел после *cur_node* и соответствующее правило $(s_i, s_j, next_hop)$ добавляется в таблицу *cur_node*.
4. Опциональная оптимизация таблицы путем замены набора правил $(s_k, t, next_hop_common)$ на одно правило $(*, t, next_hop_common)$, если $next_hop_common$ одинаков для всех s_k .

Псевдокод метода представлен в листинге 1.

Листинг 1

Алгоритм моделирования поведения узлов

Input: A single routing problem $(S, E, \vec{c}, C_1, C_2, s_{cur_node})$
Output: *route_table*

```

1  sols  $\leftarrow BOD^1(S, E, \vec{c}, s_{cur\_node})$ 
2  for each  $t \in S$  do
3      if sols( $t$ )  $\neq \emptyset$  then
4          route_table.write( $s_{cur\_node}, t, first\_step(choose(sols(t)))$ )
5          reacheble  $\leftarrow t$ 
6  sols  $\leftarrow BOD^2(S, reverse(E), \vec{c}, s_{cur\_node})$ 
7  for each  $t \in S$  do
8      if sols( $t$ )  $\neq \emptyset$  then
9          possible_senders  $\leftarrow p$ 
10 for each  $t \in possible\_senders$  do
11     sols  $\leftarrow BOD^3(S, E, \vec{c}, t)$ 
12     for each  $p \in reacheble$  do
13         if next_node(sols( $p$ ))  $\neq \emptyset$  then
14             route_table.write( $t, p, next\_node(choose(sols(p)))$ )
15 for each target  $\in reacheble$  do
16     senders_target  $\leftarrow \emptyset$ 
17     for each sender  $\in possible\_senders$  do
18         Add route_table.get_next_hop(sender, target) to senders_target
19     if  $|senders\_target| \leq 1$  then
20         // Заменяем все записи, содержащие target в поле адресата пакета на
           строку с универсальным переходом (target, *, next_hop)
21         route_table.prune(target)
22 return route_table

```

Метод жадной маршрутизации (Greedy Method)

Жадный метод предлагает упрощенный подход, где узел *cur_node* формирует правила *nextHopcur_node(p)*, основываясь только на адресате *sdest* и Парето-оптимальных путях из *cur_node*. Метод не моделирует поведение других узлов.

Алгоритм состоит из одного запуска BOD из *cur_node* для определения *sols*(t) для всех достижимых t . Правило *nextHopcur_node*(*, t) формируется применением функции *choose*(*sols*(t)) для выбора первого шага пути. Псевдокод представлен в листинге 2.

Алгоритм жадной маршрутизации

Input: A single routing problem $(S, E, \vec{c}, C_1, C_2, s_{start})$
Output: *route_table*

```

1  $sols \leftarrow BOD^1(S, E, \vec{c}, s_{start})$ 
2 for each  $t \in S$  do
3   if  $sols(t) \neq \emptyset$  then
4      $route\_table.write(s_{start}, t, first\_step(choose(sols(t))))$ 
5 return route_table

```

Исследованы три варианта функции $choose(sols(s))$ для двухкритериального случая ($d=2$):

1. **Ближайший к «идеальной» точке:** среди решений, удовлетворяющих *limit* (или всех, если таких нет), выбирается минимизирующее сначала евклидово расстояние до $(0,0)$, затем до прямой $g_1=g_2$. См. рисунок №2.
2. **Мин. стоимость по критерию 1:** выбор решения с минимальной первой компонентой стоимости g_1 (с приоритетом для удовлетворяющих *limit*).
3. **Мин. стоимость по критерию 2:** аналогично варианту 2, но для g_2 .

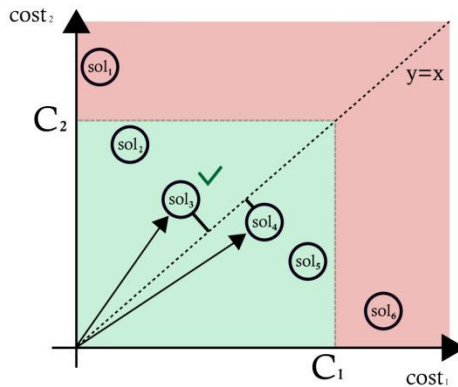


Рис. 2. Демонстрация работы варианта №1 функции $choose$ на Парето-множестве из 6 решений: выбрано решение sol_3

Жадный метод уступает методу моделирования в гарантиях ограниченности и детерминированности (последнее зависит от реализации функции $choose$), но значительно выигрывает в скорости выполнения.

3. Экспериментальные исследования

Для оценки эффективности и сравнения предложенных методов моделирования и жадной маршрутизации были проведены экспериментальные исследования.

3.1. Условия эксперимента

Эксперименты проводились на синтетически сгенерированном направленном графе, моделирующем компьютерную сеть и состоящем из 1024 узлов. Средняя степень исхода вершины составляла четыре. Каждое ребро графа характеризовалось двумя критериями ($d=2$): задержкой (delay) и вероятностью потери пакета (loss), значения которых генерировались случайным образом в диапазонах $[0.00497, 0.02953]$ и $[0.00869, 0.04347]$ соответственно.

В качестве ограничений на суммарную стоимость пути $limit = (C_1, C_2)$ использовались три пары значений: (0.6, 0.6), (0.7, 0.7) и (0.8, 0.8).

Эксперимент состоял из двух основных этапов:

1. Построение таблиц маршрутизации: для каждого узла сети и для каждого из исследуемых методов (Modeling и трех вариантов Greedy, различающихся функцией *choose*) генерировалось полное множество правил маршрутизации. Для Modeling функция *choose* выбирала решение с минимальной первой компонентой стоимости среди удовлетворяющих $limit$ (или из всех, если таковых нет).
2. Трассировка пакетов: генерировался набор из 10000 тестовых пакетов – случайных пар отправитель-получатель. Для каждого пакета и для каждой построенной таблицы маршрутизации выполнялась трассировка для определения итогового пути и его стоимости.

Также для оценки ресурсоемкости Modeling метода было проведено сравнение с его версией, где оптимизации алгоритма BOD (используемые на шагах 1-3 метода, такие как ограничение глубины поиска по $limit$ или сохранение только первого/следующего шага вместо полных предков) были отключены (далее Modeling-no-opt).

Все эксперименты проводились на персональном компьютере с процессором AMD Ryzen 5 7600X и 64 ГБ оперативной памяти.

3.2. Показатели качества

Для сравнения методов использовались следующие метрики:

1. Процент недоставленных пакетов: доля пакетов, для которых в результате трассировки либо не был найден путь до адресата, либо стоимость найденного пути превысила ограничения $limit$. Эта метрика напрямую отражает соблюдение критерия ограниченности.
2. Среднее время построения таблицы маршрутизации для одного узла. Характеризует вычислительную сложность каждого метода на этапе подготовки.

3. Максимальное количество состояний в очереди OPEN алгоритма BOD: используется для оценки пикового потребления памяти алгоритмом BOD в ходе работы методов.

3.3. Результаты и их анализ

Основные результаты экспериментов представлены в табл. 1–3.

Таблица 1

Процент пакетов, для которых не найден путь, стоимость которого не превышает ограничения

Algorithm	Experiments		
	C=(0.6, 0.6)	C=(0.7, 0.7)	C=(0.8, 0.8)
	bad pockets (%)	bad pockets (%)	bad pockets (%)
Greedy	18	14	9
	14	11	6
	15	10	7
Modeling	0	0	0

Как видно из табл. 1, Modeling метод успешно доставил 100% пакетов в рамках заданных ограничений для всех исследованных $limib$ подтверждая свою способность обеспечивать ограниченность. Методы Greedy показали наличие недоставленных пакетов, причем их доля уменьшалась с ростом значений $limir$. Среди вариантов Greedy метода, наилучшие результаты показали варианты, выбирающие решение по минимальной стоимости одного из критериев: Greedy-2 и Greedy-3. Greedy-1 ("ближайший к идеальной точке") оказался наименее эффективным по данной метрике.

Таблица 2

Среднее время выполнения алгоритма

Algorithm	Experiments		
	C=(0.6, 0.6)	C=(0.7, 0.7)	C=(0.8, 0.8)
	avg time (s.)	avg time (s.)	avg time (s.)
Greedy	0.3	0.42	0.51
Modeling	95	186	305

Табл. 2 демонстрирует значительное различие в вычислительной сложности. Greedy требует на порядки меньше времени для построения таблиц, так как выполняет лишь один проход BOD для каждого узла. Modeling метод, включающий многократные запуски BOD для моделирования, существенно более ресурсоемок.

Таблица 3

Сравнение потребления ресурсов у методов моделирования
поведения узлов

\vec{C}	Algorithm	average time(s)	max OPENlen	solutions count
(0.6, 0.6)	Modeling	95	3406	7280
	Modeling_no_opt	81	3406	8032
(0.7, 0.7)	Modeling	186	3630	8290
	Modeling_no_opt	200	3630	8600
(0.8, 0.8)	Modeling	305	3910	9028
	Modeling_no_opt	285	3910	9240

Результаты, представленные в табл. 3, показывают эффективность оптимизаций, примененных в алгоритме BOD в рамках Modeling метода. Версия Modeling с оптимизациями требует несколько меньшего среднего времени и, что более важно, работает с меньшим или таким же пиковым размером очереди OPEN и количеством генерируемых решений, что подтверждает целесообразность внесенных модификаций в BOD для снижения ресурсоемкости без ущерба для качества основного метода.

В целом, экспериментальные результаты подтверждают теоретические предпосылки: Modeling обеспечивает более строгие гарантии ценой вычислительной сложности, тогда как Greedy предлагает более быстрый, но менее надежный в плане соблюдения ограничений подход. Выбор между методами зависит от конкретных требований к системе маршрутизации.

Заключение

В работе предложены новые методы децентрализованной многокритериальной маршрутизации, основанные на адаптации методов централизованного эвристического поиска. Проведены их экспериментальных исследования.

Проведенное исследование имеет ряд ограничений, которые определяют направления для дальнейшей работы. Так, экспериментальная оценка была проведена на графах с фиксированными параметрами. Для подтверждения обобщаемости полученных результатов возможно проведение исследований с расширенными наборами топологий графов. Также стоит провести анализ влияния ключевых топологических на производительность предложенных методов.

Другим важным направлением для будущих исследований является анализ эффективности алгоритмов в динамических сетевых условиях, где метрики каналов связи могут изменяться во времени. Такое исследование позволит оценить не только устойчивость найденных решений, но и накладные расходы, связанные с обновлением таблиц маршрутизации.

Список литературы

- [Bellman, 1958] Bellman R. On a routing problem // Quart. Appl. Math. – 1958. – Vol. 16. – P. 87-90. – doi: 10.1090/qam/102435.
- [Dijkstra, 1959] Dijkstra E.W. A note on two problems in connexion with graphs // Numerische Mathematik. – 1959. – Vol. 1, No. 1. – P. 269-271.
- [Hernández, 2023] Hernández C., Yeoh W., Baier J.A., Zhang H., Suazo L., Koenig S., Salzman O. Simple and efficient bi-objective search algorithms via fast dominance checks // Artificial Intelligence. – 2023. – Vol. 314. – P. 103807. – doi: 10.1016/j.artint.2022.103807.
- [Hernández Ulloa, 2020] Hernández Ulloa C., Yeoh W., Baier J.A., Zhang H., Suazo L., Koenig S. A Simple and Fast Bi-Objective Search Algorithm // ICAPS. – 2020. – P. 143-151. – doi: 10.1609/icaps.v30i1.6655.
- [Jaffe, 1984] Jaffe J.M. Algorithms for finding a collection of circuit-disjoint paths with certain constraints // Networks. – 1984. – Vol. 14, No. 1. – P. 95-114. – doi: 10.1002/net.3230140108.
- [Korkmaz & Krunz, 2001] Korkmaz T., Krunz M. A randomized algorithm for finding a path subject to multiple QoS constraints // Computer Networks. – 2001. – Vol. 36. No. 2-3. – P. 251-268. – doi: 10.1016/S1389-1286(00)00209-7.
- [Madow, 2010] Madow L., De La Cruz J.L.P. Multiobjective A* search with consistent heuristics // J. ACM. – 2010. – Vol. 57, No. 5. – P. 27:1-27:25. – doi: 10.1145/1754399.1754404.
- [Stewart, 1991] Stewart B.S., White III C.C. Multiobjective A* // J. ACM. – 1991. – Vol. 38, No. 4. – P. 775-814. – doi: 10.1145/115783.115786.
- [Wang & Crowcroft, 1996] Wang Z., Crowcroft J. Quality-of-service routing for supporting multimedia applications // IEEE Journal on selected areas in communications. – 1996. – Vol. 14, No. 7. – P. 1228-1234. – doi: 10.1109/49.536364.