НАХОЖДЕНИЕ КРИТИЧЕСКИХ УЗЛОВ ТРАНСПОРТНОЙ СЕТИ НА ОСНОВЕ ПОСТРОЕНИЯ ЗАМКНУТОЙ ОБЛАСТИ

Крыгин А. А.[1], **Куприянов Б. В.**[2]
(*ФГБУН Институт проблем управления им. В.А. Трапезникова РАН, Москва*)

Рассматривается задача поиска критических узлов транспортной сети, решаемая с помощью максимизации обобщенной стоимости проезда, которая зависит от потребности в движении и стоимости поездки между каждой парой узлов сети. Предлагаемый в работе метод является улучшением полного перебора, основная сложность которого состоит в многократном вычислении матрицы минимальных стоимостей поездок. Метод заключается в выделении замкнутого в определенном смысле множества вершин исходного графа. Выделение замкнутого множества вершин графа позволяет осуществить редуцирование графа, декомпозицию соответствующих ему матриц и раздельные вычисления подматриц. Данные преобразования позволили сократить вычисления при переборе вариантов. Построен общий алгоритм нахождения критических узлов и проведена его оптимизация. Замкнутое множество разделено на внутреннее и граничное подмножества. Показано, что алгоритм работает наиболее быстро при минимальной мощности граничного подмножества и оптимальной мощности внутреннего подмножества, для определения которой предложен соответствующий алгоритм. Также предложен алгоритм построения и расширения замкнутого множества. на его основе построен приближенный алгоритм нахождения оптимального замкнутого множества. Показано, что сложность нахождения оптимального замкнутого множества во много раз меньше сложности улучшенного метода полного перебора.

Ключевые слова: транспортные сети, поиск критических узлов.

1. Введение

В научной литературе существует класс задач, связанных с нахождением критических узлов в транспортных сетях. В данной работе рассматриваются автотранспортные сети, одним из видов узлов сети являются однородные и неветвящиеся участки дороги. Критические узлы ищутся среди узлов указанного вида. При решении таких задач автотранспортные сети моделируют-

[1] *Андрей Александрович Крыгин, к.т.н.* (*andreyakr@yandex.ru*).
[2] *Борис Васильевич Куприянов, к.т.н.* (*kuprianovb@mail.ru*).

ся в виде нагруженного графа, ребрам которого и соответствуют однородные неветвящиеся участки дорог. Задача решается с помощью оценки ущерба сети при удалении из графа сети подмножества ребер и нахождения критического подмножества, при котором ущерб достигает оптимума. Работы, посвященные этой задаче можно разделить на два направления:

1. Работы, в которых предлагаются различные методы оценки ущерба или метрики, количественно оценивающие общую надежность и уязвимость сети до и после возникновения на ней повреждений.

2. Работы, в которых предлагаются методы нахождения критических узлов сети на основе той или иной метрики.

В [11] вводятся две группы метрик: одна отражает «перспективу равных возможностей», а другая — «перспективу социальной эффективности». Метрика в [9] связана с оценкой риска, в ней учитывается как вероятность наступления инцидента, так и его последствия, причем распространение последствий от инцидента моделируется динамически во времени. В [15] предлагается алгоритм оценки риска с помощью машинного обучения, основанный на теории перколяции сетей и кластеризации на основе данных о трафике в реальном времени.

В [14] предлагается метрика, основанная на изолированной уязвимости, и метод сведения задачи к задаче линейного целочисленного программирования для нахождения критических узлов на ее основе.

В [16] используется потоковая модель сети и на основе данных поездок такси определяется потребность в движении между парами узлов. Далее, с помощью метода энтропийных весов и метода предпочтения порядка вводится характеристика важности (критичности) узла и оценивается структурный и функциональный ущерб сети при выходе из строя критических узлов.

Авторы [6] отмечают проблему вычислительной сложности нахождения всех критических узлов, предлагают свою метрику и метод сокращения вычислений, основанный на введенном ими понятии «зоны воздействия». Для каждого объекта сети опреде-

ляется множество поставок (зона воздействия), маршруты которых изменятся при возникновении повреждений на объекте. При поиске критических узлов основной вычислительной проблемой является определение значения уязвимости или надежности по выбранной метрике. Авторы предлагают сократить вычисления, рассматривая не всю сеть, а объединение зон воздействия для выбранного подмножества узлов.

Понятно, что независимо от метрики наибольшая практическая ценность будет от решения задачи нахождения всех критических узлов для заданной сети, т.е. всех наборов ее элементов, при удалении которых значение ущерба будет максимальным среди наборов той же мощности. Во многих из перечисленных и других рассмотренных работ [4, 5, 7, 8, 10, 12, 13] используется сценарный подход. В них предлагается метод оценки ущерба для заданного набора удаляемых элементов. Соответственно, задача нахождения полного множества критических узлов предлагаемыми методами эквивалентна использованию метода полного перебора. В других работах, таких как [16], фактически используется жадный алгоритм: строится характеристика узла и определяется ее значение. Критическими (будем называть их «важными») считаются узлы с наибольшими или наименьшими (в зависимости от характеристики) ее значениями. Такая группа методов считается приближенными и обычно применяется для сетей большой размерности, когда нахождение критических узлов занимает слишком большое время. Стандартные недостатки такого подхода заключаются в следующем:

– невозможно сказать, является ли конкретный важный узел критическим, т.е. существует ли набор критических узлов, содержащий важный узел;

– также нельзя утверждать, что не существует набора критических узлов, каждый из которых не является важным.

Подход, используемый в таких работах, как [6], базируется на эмпирическом допущении, что наибольший ущерб при возникновении повреждения на узле приходится на небольшую локальную зону вокруг этого узла. Для транспортных сетей и одно-

го узла это допущение является приемлемым, но при возникновении повреждений на нескольких узлах можно только утверждать, что зона наибольшего ущерба включает в себя объединение локальных зон каждого поврежденного узла. Несложно построить пример, в котором при удалении нескольких узлов ущерб будет приходиться на значительно большую зону, чем объединение локальных зон узлов. В [1] построена модель транспортной автодорожной сети, предложены и обоснованны методики определения всех ее параметров. Также предложен метод сведения задачи нахождения критических узлов к эквивалентной задаче линейного программирования. Данная работа является продолжением [1] и базируется на той же математической модели. Для метода полного перебора предложен алгоритм сокращения вычислений.

Количество вариантов удаления q ребер из графа, содержащего m ребер определяется как число сочетаний из m по q. Трудоемкой частью оценки величины ущерба является вычисление матрицы минимальных стоимостей путей для каждой комбинации удаления ребер из графа.

Как правило, плотность автодорог внутри области существенно больше чем плотность межобластных дорог, см. рис. 1. На данном графе можно выделить три подграфа, которые являются изолированными в некотором смысле от остальной части графа.

В данной статье рассматривается метод выделения замкнутого множества вершин в графе и связанных с ними ребер. Данное выделение замкнутого множества позволяет декомпозировать и сократить вычисление матрицы путей минимальной стоимости.

2. Определения и постановка задачи

Пусть имеется неориентированный граф $G = (V, E)$, $n = |V|$, $m = |E|$ с взвешенными ребрами. Множеству ребер E соответствуют однородные и неветвящиеся участки дорог: мосты, туннели, автострады. Ребро графа, соединяющее вершины v_i и v_j, обозначим как (i, j). Множеству вершин V соответству-

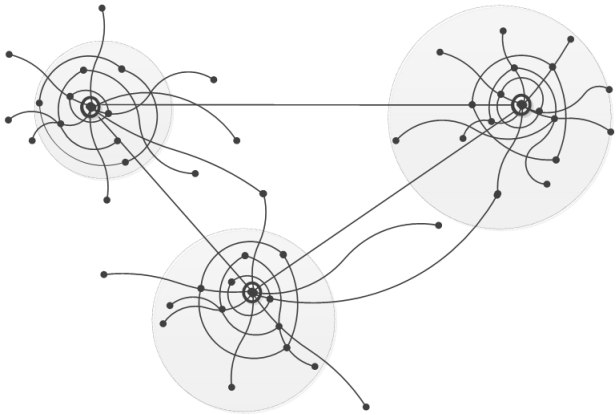

Рис. 1. Пример графа автотранспортной сети с тремя связанными областями

ют населенные пункты, перекрестки и границы смены категории дорог. Каждой паре вершин v_i и v_j ставится в соответствие два числа: $c_{i,j}$ и $d_{i,j}$. Величина $c_{i,j}$ характеризует стоимость поездки от узла v_i к узлу v_j, а $d_{i,j}$ характеризует потребность в движении (спрос на поездку), т.е. количество поездок от узла v_i к узлу v_j за единицу времени. В работе [1] приведена методика определения $c_{i,j}, d_{i,j}$. Величина $d_{i,j}$ не меняется при удалении ребер из графа; будем считать ее заданной. Величина $c_{i,j}$ определяется следующим образом. Для каждого ребра графа (i, j) задается его вес $w_{i,j}$, характеризующий длину соответствующего участка дороги. Путем в графе, соединяющим вершины v_i и v_j, является последовательность ребер $(i, l_1), (l_1, l_2), \ldots, (l_{q-1}, l_q), (l_q, j)$. Длиной пути является сумма весов всех его ребер. Путем минимальной стоимости, соединяющим вершины v_i и v_j, является такой путь, что длина любого другого пути, соединяющего эти вершины, не меньше пути минимальной стоимости. Значение $c_{i,j}$ определяется как длина пути минимальной стоимости. Существуют эффективные алгоритмы вычисления минимальных стоимостей для всех пар вершин графа, например [2].

Рассматривается задача нахождения подмножества критиче-

ских ребер Q для заданного $q = |Q|$. Будем считать, что задача решается методом полного перебора следующим образом. Для каждой из C_m^q комбинаций q ребер, составляющих множество Q, вычисляется стоимость

$$S(G(V, E\backslash Q)) = \sum_{i=1}^{n} \sum_{j=1}^{i-1} d_{i,j} c_{i,j},$$

где $c_{i,j}$ – минимальная стоимость пути между вершинами v_i, v_j при удалении из графа комбинации; $d_{i,j}$ – потребность в движении. Находится комбинация Q_0, при которой $S(G(V, E\backslash Q_0))$ достигает своего максимума. Основная часть этой операции – вычисление матрицы C для графа $G(V, E\backslash Q)$ – выполняется C_m^q раз. Задачей настоящего исследования является сокращение числа операций при вычислениях матрицы C.

Использование метода полного перебора в качестве базового было выбрано по следующим причинам.

1. При нахождении точного решения задачи известны только два метода: метод полного перебора и сведение этой задачи к эквивалентной задаче линейного программирования. Причем эти методы равноправны в том смысле, что для одних параметров задачи (т.е. заданного графа и величины q) выгоднее использовать метод полного перебора, а для других – решать задачу линейного программирования. Ниже это будет показано.

2. При оптимизации предлагаемого метода удобно проводить его сравнение с методом полного перебора.

Покажем выполнение первого пункта: для этого достаточно показать, что существуют параметры, при которых сложность метода полного перебора меньше сложности решения задачи ЛП. Будем рассматривать следующую последовательность задач нахождения критических узлов: граф сети является регулярным графом степени 4 (район Манхэттен в Нью-Йорке), имеющим вид квадрата, и количество вершин у графов этой последовательности неограниченно возрастает. При этом величина q фиксирована и постоянна для каждого элемента последовательности. Покажем, что для этой последовательности сложность задачи линейного программирования растет экспоненциально, а сложность

полного перебора – полиномиально от количества вершин. В [1] приводятся оценки для количества переменных и ограничений эквивалентной задачи. Общее количество неравенств в задаче ЛП составит $O(n^2k^2 + n^2kl)$, а общее количество переменных – $O(n^2k + m)$, где k – среднее количество путей между двумя вершинами, а l – средняя длина пути. Ключевая величина здесь – среднее количество путей, ее можно грубо оценить следующим образом. Известно, что у рассматриваемых графов количество кратчайших путей между двумя вершинами, отстоящими друг от друга на a ребер по горизонтали и b по вертикали, равно C^b_{a+b}. Тогда среднее количество ребер между парами вершин как по вертикали, так и по горизонтали можно оценить как $\frac{\sqrt{n}}{2}$, и

$$C^{\lceil \frac{\sqrt{n}}{2} \rceil}_{\lceil \sqrt{n} \rceil} = O((\sqrt{n})^{\frac{\sqrt{n}}{2}}) = O(e^{\frac{\ln(n)\sqrt{n}}{4}}).$$

То есть количество ограничений, соответствующих только кратчайшим путям, растет экспоненциально. Также известно, что сложность решения задачи линейного программирования растет полиномиально от количества ограничений. В итоге сложность задачи линейного программирования растет экспоненциально. Ниже будет показано, что сложность алгоритма полного перебора можно оценить как $O(n^{4+q})$, т.е. имеем полиномиальный рост.

3. Декомпозиция вычислений на основе замкнутого множества вершин

Выделим в графе подмножество вершин $S \subset V$. Определим подмножество Z, состоящее из внутреннего множества S и граничного множества P т.е. $Z = S \cup P$, следующим образом. Для любого ребра (i, j), где $v_i \in S \ \rightarrow \ v_j \in Z$, Z таково, что все вершины S смежны только вершинам из Z. Соответственно, $P = Z \backslash S$ и обозначим $H = V \backslash S$.

Пример графа показан на рис. 2. Некоторое подмножество вершин $Z = \{4, 5, 6, 7, 8, 9\}$ выделено пунктиром. Сама пунктирная линия проходит по граничным вершинам Z. Таким образом,

вершины множества $S = \{6, 8\}$ – образуют внутреннее множество, а $P = \{4, 9, 5, 7\}$ образует граничное множество в Z. Пусть $n_z = |Z|,\ n_p = |P|,\ n_s = |S|$, в примере $n_z = 6,\ n_p = 4$ и $n_s = 2$. Очевидно, что $n_z = n_s + n_p$.

Переупорядочим номера вершин таким образом, что вершины из $V \backslash Z$ будут пронумерованы от 1 до $(n - n_z)$, далее будут идти вершины из P с номерами вершин от $(n - n_z + 1)$ до $(n - n_z + n_p) = (n - n_s)$, а в конце – вершины из S с нумерацией от $(n - n_s + 1)$ до n. Таким образом, будет следующее разбиение: $1, .., (n - n_z), (n - n_z + 1), .., (n - n_s), (n - n_s + 1), .., n$.

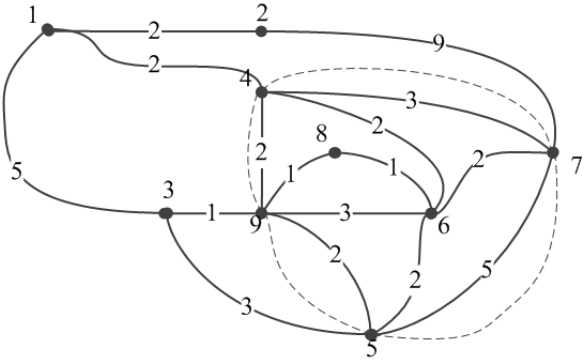

Рис. 2. Пример графа и некоторой области вершин Z

С учетом данной нумерации построим матрицу смежности W графа G:

$$W_{i,j} = \begin{cases} \text{вес ребра } (i,j), & \text{если ребро } (i,j) \in E; \\ 0, & \text{если ребро } (i,j) \notin E. \end{cases}$$

Например, для графа на рис. 2 матрица смежности W представлена в таблице 1.

Матрица W будет иметь структуру, показанную в двух разрезах на рис. 3.

В соответствии с данным разбиением будем рассматривать подматрицы матрицы W: $W_H,\ W_M,\ W_Z,\ W_M^T$. Подматрицы $W_H,\ W_Z$ будут включать строки и столбцы, соответствующие граничным вершинам. Квадратная матрица W_H будет иметь

Таблица 1. Матрица смежности W

	1	2	3	4	5	6	7	8	9
1	0	2	5	2	0	0	0	0	0
2	2	0	0	0	0	0	9	0	0
3	5	0	0	0	3	0	0	0	1
4	2	0	0	0	0	2	3	0	2
5	0	0	3	0	0	2	5	0	2
6	0	0	0	2	2	0	2	1	3
7	0	9	0	3	5	2	0	0	0
8	0	0	0	0	0	1	0	0	1
9	0	0	1	2	2	3	0	1	0

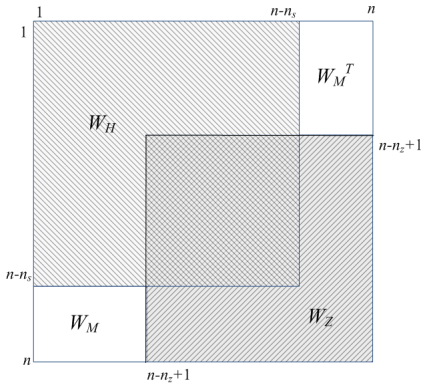

Рис. 3. Структура матрицы W

размерность $n - n_s$, матрица W_M будет иметь размерность $(n - n_z, n_s)$ и квадратная матрица W_Z будет иметь размерность n_z. Матрица W симметрична, поэтому в дальнейшем матрицу W_M^T рассматривать не будем, так как она является результатом транспонирования матрицы W_M.

Рассматривая веса ребер как стоимости, вычислим минимальные стоимости путей (далее просто минимальные стоимости) для всех пар вершин графа на основе матрицы W, например алгоритмом Флойда – Уоршалла [2]. Пусть C – матрица мини-

мальных стоимостей путей, вычисленная на основе матрицы W. Матрице C будет соответствовать 4 «вырезанные из нее» подматрицы $C_H,\ C_M,\ C_M^T,\ C_Z$. Ввиду симметрии матрицы C подматрицу C_M^T рассматривать не будем. В дальнейшем в статье будут рассматриваться граф $G = (V, E)$ и соответствующие ему подмножества вершин H, Z, P, S в описанном выше смысле.

Определение 1. *Пусть граф $F = (V_F, E_F)$ является подграфом G, т.е. $V_F \subset V$ и $E_F \subset E$. В этом случае назовем W_F квадратной вырезкой из матрицы W, если номера вершин V_F являются индексами строк и столбцов матрицы W. То есть $W_F = ||W_{i,j}||$б если $v_i, v_j \in V_F$. Определим функцию вырезки $Cut_F(W)$, т.е. $W_F = Cut_F(W)$.*

Определение 2. *Пусть для графа G определена матрица смежностей с весами ребер. Определим функцию $\mathcal{M}s(W)$, вычисляющую матрицу минимальных стоимостей для матрицы W.*

Определение 3. *Будем называть в графе $G = (V, E)$ подмножество вершин $Z \subset V$ замкнутым, если для каждой пары вершин множества Z существует путь минимальной стоимости, такой что все его вершины принадлежат множеству Z.*

Рассмотрим матрицу $C_Z = Cut_Z(\mathcal{M}s(W))$, т.е. вырезку внутренней области Z из матрицы C (вычисленной по матрице W). Матрица минимальных стоимостей C представлена в таблице 2, а матрица C_Z – в таблице 3.

Если на основании матрицы $W_Z = Cut_Z(W)$, представленной в таблице 4, вычислить матрицу C, то она совпадет с матрицей C_Z. Это справедливо в общем случае благодаря следующему утверждению.

***Утверждение* 1.** *Пусть дан граф $G = (V, E)$ и подмножество вершин $Z \subset V$. Чтобы множество Z было замкнутым, необходимо и достаточно, чтобы*

$$\mathcal{M}s(Cut_Z(W)) = Cut_Z(\mathcal{M}s(W)).$$

Доказательство. Заметим, что при поэлементном сравнении

$$\mathcal{M}s(Cut_Z(W)) \geqslant Cut_Z(\mathcal{M}s(W)),$$

Таблица 2. Матрица минимальных стоимостей C

	1	2	3	4	5	6	7	8	9
1	0	2	5	2	6	4	6	5	4
2	2	0	7	4	8	6	7	7	6
3	5	7	0	3	3	3	5	2	1
4	2	4	3	0	4	2	3	3	2
5	6	8	3	4	0	2	4	3	2
6	4	6	3	2	2	0	2	1	2
7	6	7	5	3	4	2	0	3	4
8	5	7	2	3	3	1	3	0	1
9	4	6	1	2	2	2	4	1	0

Таблица 3. Вырезка C_Z из матрицы минимальных стоимостей C

	4	5	6	7	8	9
4	0	4	2	3	3	2
5	4	0	2	4	3	2
6	2	2	0	2	1	2
7	3	4	2	0	3	4
8	3	3	1	3	0	1
9	2	2	2	4	1	0

Таблица 4. Матрица W_Z

	4	5	6	7	8	9
4	0	0	2	3	0	2
5	0	0	2	5	0	2
6	2	2	0	2	1	3
7	3	5	2	0	0	0
8	0	0	1	0	0	1
9	2	2	3	0	1	0

так как в левой части минимальная стоимость вычисляется по подграфу. Докажем утверждение необходимости. Пусть Z – замкнутое подмножество. Предположим, что

$$\mathcal{M}s(Cut_Z(W)) \neq Cut_Z(\mathcal{M}s(W)),$$

т.е. для некоторых $v_i, v_j \in Z$ соответствующие элементы матриц $\mathcal{M}s(Cut_Z(W))$ и $Cut_Z(\mathcal{M}s(W))$ различны. В этом случае в графе G между вершинами v_i, v_j найдется путь меньшей стоимости, чем в подграфе $G_Z = (V_Z, E_Z)$, т.е. Z не замкнутое множество.

Аналогично доказывается достаточность. Утверждение доказано.

Матрица C однозначно определяется через матрицы C_H, C_M, C_Z. Покажем, как сократить вычисления при нахождении каждой из этих матриц.

Так как Z замкнутое подмножество, то

$$C_Z = \mathcal{M}s(Cut_Z(W)),$$

т.е. вычисления проводятся по матрице меньшего размера.

Модифицируем матрицу W_H, заменив ее элементы, относящиеся к области P, т.е. $w_{i,j}$ такие, что $v_i, v_j \in P$, на соответствующие элементы матрицы C_Z, как показано на рис. 4.

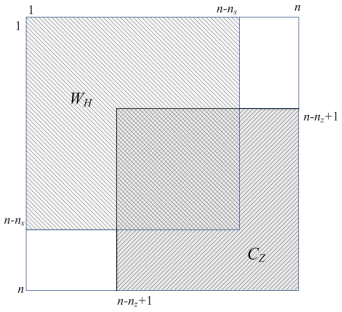

Рис. 4. Модификация матрицы W_H

***Утверждение* 2.** *Для модифицированной описанным образом матрицы W'_H выполняется равенство:*

$$\mathcal{M}s(W'_H) = Cut_H(\mathcal{M}s(W)).$$

Доказательство. Если кратчайший путь между вершинами $v_i, v_j \in V \backslash Z$ не проходит через вершины множества Z, то соответствующие элементы матриц $\mathcal{M}s(W'_H)$ и $Cut_H(\mathcal{M}s(W))$ совпадут. Рассмотрим случай, когда все кратчайшие пути между этими вершинами проходят через вершины множества Z.

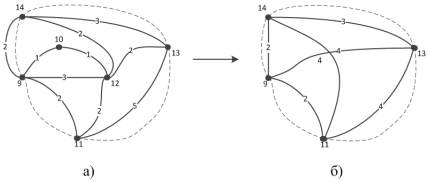

Рис. 5. Преобразование замкнутого множества

Преобразуем граф G так, что все вершины множества S и смежные с ними ребра удаляются из графа, а граничные вершины дополняются ребрами до полного графа со стоимостями ребер, равными соответствующим минимальным стоимостям путей исходного графа, как на примере рис. 5. Пусть путь минимальной стоимости в исходном графе G проходит в замкнутой области через вершины $(a, b, \ldots, c, d)$, где a и d граничные вершины. В преобразованном графе этот путь проходит по ребру (a, d), поэтому его минимальная стоимость не изменится. С другой стороны матрица смежности преобразованного графа совпадает с W'_H. Утверждение доказано.

Это позволяет рассчитать матрицу C_H по матрице W'_H меньшей размерности.

Покажем, как уменьшить вычисления при нахождении матрицы C_M: матрица кратчайших путей между вершиной $v_i \in (V \backslash Z)$ и вершиной $v_j \in S$. Определим функцию $minp(i, j)$, вычисляющую минимальную стоимость пути между вершинами v_i и v_j. Любой путь между этими вершинами проходит через вершины из P, поэтому

$$minp(v_i, v_j) = \min_{w \in P}(minp(v_i, w) + minp(w, v_j)). \quad (1)$$

Все значения $minp(v_i, w)$ и $minp(w, v_j)$ вычислены и имеются в матрицах C_H и C_S.

4. Алгоритм построения замкнутого множества

Алгоритм 1 (Построения замкнутого множества)**.**

Исходные данные алгоритма: матрица смежностей W и начальное множество S_0 (внутренних вершин). Алгоритм строит замкнутое множество Z такое, что $S_0 \subseteq S$.

1. Построить множество P_0 – множество смежных с S_0 вершин, такое что $P_0 \cap S_0 = \emptyset$. Определим $Z = S_0 \cup P_0$.

2. Вычислить матрицу минимальных стоимостей $C = \mathcal{M}s(W)$ и матрицу $C_Z = Cut_Z(C)$.

3. Вычислить матрицу $C'_Z = \mathcal{M}s(Cut_Z(W))$.

4. Если $C_Z = C'_Z$, то множество Z является замкнутым и перейти к п. 1.

5. Пусть $C'_Z(i,j) \neq C_Z(i,j)$. На основании матрицы C вычислим множество вершин пути минимальной стоимости (см. алгоритм [2]) $\{v_i, \tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_k, v_j\}$. Включим множество вершин $\{\tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_k\}$ в множество Z. Перейти к п. 1.

Для найденного замкнутого множества Z определяется внутреннее множество S. Перенумеруем номера вершин так, чтобы индексы вершины Z превышали индексы остальных вершин. Вершина $v_k \in Z$ является внутренней, если $W(k,i) = 0$ для любого $1 \leqslant i \leqslant (n - n_z)$.

Конец алгоритма.

Так как по построению множество S_0 окружено границей P_0, то $S_0 \subseteq S$.

Данный алгоритм находит решение не более, чем за $|V|$ итераций, однако в худшем случае множество S может совпадать с множеством V.

Пример 1. Рассмотрим в качестве исходных данных граф на рис. 2. Матрица смежностей представлена в таблице 1. Соответствующая матрица минимальных стоимостей представлена в таблице 2. Выберем в качестве начального внутреннего множества $S_0 = \{8\}$. Построим множество $P_0 = \{6, 9\}$ – граничных вершин, $Z = \{6, 8, 9\}$. Матрицы $\mathcal{M}s(Cut_Z(W))$ и $Cut_Z(\mathcal{M}s(W))$

совпадают и равны

	6	8	9
6	0	1	2
8	1	0	1
9	2	1	0

Поэтому Z– замкнутое множество и после перенумерации вершин (п. 1) вершина 8 определяется как внутренняя. •

5. Алгоритм нахождения критических ребер

Будем считать, что вычисление матрицы путей минимальной стоимости (кратчайших путей) C осуществляется с помощью алгоритма Флойда – Уоршалла [2]. Сложность этого алгоритма зависит только от количества вершин, поэтому в дальнейшем фразу «вычисление матрицы C по матрице W размера n» будем понимать в смысле применения алгоритма Флойда – Уоршалла для матрицы смежности W размера $n \times n$ и получение в результате его работы матрицы C того же размера.

Алгоритм 2 (Нахождение критических ребер)**.** Исходными данными алгоритма являются матрица весов W, матрица потребности в движении D и количество удаляемых ребер q.

1. Выполнить алгоритм вычисления замкнутого множества вершин, в частности алгоритм 1 вычислит первоначальную матрицу C для графа $G(V, E)$.

2. Выполнить в графе преобразование замкнутого множества и получить матрицу W'_H.

3. Для каждой из C_m^q комбинаций q ребер, составляющих множество Q, вычисляется обобщенная стоимость поездок

$$S(G(V, E\backslash Q)) = \sum_{i=1}^{n} \sum_{j=1}^{i-1} d_{i,j} c_{i,j}.$$

Основная часть этой операции – вычисление матрицы C для графа $G(V, E\backslash Q)$ – распадается на два случая.

3.1) если хотя бы одно из q ребер является ребром из замкнутого множества. В этом случае матрица C вычисляется обычным способом по матрице W размера n;

3.2) если ни одно из q ребер не принадлежит замкнутому множеству. В этом случае матрица C для графа $G(V, E\backslash Q)$ однозначно определяется с помощью уже вычисленной матрицы C для графа $G(V, E)$ и вычислением матрицы C_H по матрице $W'_H(V, E\backslash Q)$ размера $n - n_s$.

4. Среди всех комбинаций определяются комбинации, имеющие максимальное значение обобщенной стоимости поездок. Ребра, составляющие эти комбинации, являются критическими.

Конец алгоритма.

Докажем справедливость этого алгоритма. При удалении хотя бы одного ребра из замкнутого множества уже нельзя гарантировать, что это множество останется замкнутым, т.е. что все пути минимальной стоимости лежат внутри замкнутой области. Именно поэтому в п. 3.1 матрица C вычисляется обычным способом. Остается показать справедливость п. 3.2. В зависимости от того, к какому множеству относятся вершины i и j, существует 6 вариантов;

1. $i \in V\backslash Z, j \in V\backslash Z$;
2. $i \in V\backslash Z, j \in P$;
3. $i \in V\backslash Z, j \in S$;
4. $i \in P, j \in P$;
5. $i \in P, j \in S$;
6. $i \in S, j \in S$.

Значения элементов $c_{i,j}$ матрицы C в вариантах 4, 5, 6 уже получены при вычислении первоначальной матрицы C в п.1. Варианты 1, 2, 3 соответствуют элементам матриц C_H и C_M, которые вычисляются описанным выше способом.

6. Оптимизация алгоритма

Отметим, что единственным параметром, на который можно влиять при реализации предложенного алгоритма, является мощность множества S – внутренних вершин замкнутого множества. Действительно, можно построить замкнутое множество, добавить к внутреннему множеству еще одну вершину (например, одну из граничных) и построить следующее замкнутое мно-

жество большего размера, пользуясь тем же алгоритмом. Также понятно, что чем больше будет мощность множества S, тем меньшего размера будет матрица W_H и меньшее число операций потребуется для вычисления матрицы C. А с другой стороны, тем меньше будет количество комбинаций ребер, не входящих в замкнутое множество, т.е. тем чаще придется вычислять матрицу C обычным способом по п. 3.1. Эти соображения позволяют предположить, что существует оптимальное значение n_s. Логично искать это оптимальное значение, оценивая количество операций алгоритма и минимизируя ее по n_s. В общем виде это сделать невозможно по следующей причине. Для конкретного графа и заданного множества S множество P определяется однозначно (если оно существует), при этом его мощность невозможно оценить, так как она зависит и от графа и от S. В то же время, как будет видно ниже, сложность алгоритма является функцией от мощностей реберных и вершинных, внутренних и граничных множеств, а сами эти величины не являются независимыми и их зависимость неизвестна. Тем не менее некоторый анализ функции сложности оказывается возможным.

Определим ребра замкнутого множества как ребра, соединяющие вершины из замкнутого множества и, по аналогии, ребра внутреннего и граничного множества. Введем следующие обозначения: m_s – количество ребер внутреннего множества; m_p - количество ребер граничного множества; m_z – количество ребер множества Z. При этом $m_z > m_s + m_p$, так как еще учитываются ребра, соединяющие вершины из S с вершинами из P.

В большинстве современных процессоров операции сложения и умножения занимают одинаковое количество тактов, поэтому будем оценивать только общее количество этих операций. Переменные $m_s,\ n_p,\ m_p,\ n_z,\ m_z$ и переменная n_s, относительно которой ищется минимум сложности всего алгоритма, «участвуют» в пунктах 1, 3.1 и 3.2, поэтому будем оценивать сложность только этих пунктов. Ниже будет часто встречаться выражение сложности вычисления матрицы C, поэтому определим для него отдельную функцию. Обозначим $M(l),\ l \in N$, – количество опе-

раций сложения и умножения, необходимых для нахождения матрицы кратчайших путей размера $l \times l$.

В алгоритме Флойда – Уоршалла элементы матрицы C определяются с помощью операции

$$\tilde{c}_{i,j} = \min_{k=1..l} (c_{i,j}, c_{i,k} + c_{k,j}),$$

которая проводится в тройном цикле, т.е. l^3 раз. Вся операция состоит из одной операции сложения, поэтому $M(l) = l^3$.

Оценим сложность алгоритма вычисления замкнутого множества. Будем рассматривать наихудший случай и считать, что вершины в п. 5 добавляются по одной и первоначальное множество состоит из одной внутренней и одной граничной вершины. По окончании выполнения алгоритма мощность множества Z равно n_z, т.е. количество операций для вычисления матрицы C'_Z равно $\sum_{i=2}^{n_z} M(i)$. Необходимо оценить сложность операции в п. 5 по определению множества вершин пути минимальной стоимости между вершинами (x, y). Так как мы рассматриваем наихудший случай и вершины добавляются по одной, то количество ребер в этом пути равно двум. Стоимость (длина) этого пути известна из исходной матрицы C (п. 1) и в среднем необходимо перебрать половину вершин, соседних и с x и с y, чтобы найти добавляемую вершину. Это вершина w такая, что сумма длин ребер $c_{x,w}$ и $c_{w,y}$ равна стоимости кратчайшего пути. Тогда среднее количество операций сложения можно оценить как половину средней степени вершины. И общая сложность алгоритма вычисления замкнутого множества (обозначим ее Σ_1) оценивается как

$$\Sigma_1 = \sum_{i=2}^{n_z} (M(i) + \frac{m}{n}) = \frac{m(n_z - 1)}{n} + \sum_{i=2}^{n_z} M(i).$$

Рассмотрим пункты 3.1 и 3.2. Количество комбинаций ребер, соответствующее пункту 3.2 равно $C^q_{m-m_z}$, а пункту 3.1 – $(C^q_m - C^q_{m-m_z})$. Тогда общая сложность п. 3.1 (обозначим ее Σ_2) оценивается как $\Sigma_2 = (C^q_m - C^q_{m-m_z})M(n)$. Сложность п. 3.2 состоит из вычисления матрицы C_H по матрице $W'_H(V, E \backslash Q)$ размера $n - n_s$ и кроме одной группы случаев все элементы матрицы

C определяются без дополнительных расчетов. Эта группа соответствует комбинациям $i \in V\backslash Z,\ j \in S$, и количество таких комбинаций вершин равно $(n - n_z)n_s$. Для каждой комбинации необходимо найти

$$\min_{v_1 \in P} (minp(i, v_1) + minp(v_1, j))$$

по всем граничным вершинам. То есть необходимо выполнить n_p операций сложения. Общую сложность п. 3.2 (обозначим ее Σ_3) можно оценить так: $\Sigma_3 = C^q_{m-m_z}(M(n-n_s)+(n-n_z)n_sn_p)$. Суммируя результаты, получим, что количество операций сложения и умножения в рассматриваемых пунктах алгоритма нахождения критических ребер можно оценить так. $\Sigma = \Sigma_1 + \Sigma_2 + \Sigma_3$, где

$$\begin{aligned}
\Sigma_1 &= \frac{m(n_z - 1)}{n} + \sum_{i=2}^{n_z} M(i), \\
\Sigma_2 &= (C^q_m - C^q_{m-m_z})M(n), \\
\Sigma_3 &= C^q_{m-m_z}(M(n - n_s) + (n - n_z)n_sn_p),\ M(l) = l^3.
\end{aligned}$$

Также отметим, что сложность алгоритма полного перебора (обозначим ее $\hat{\Sigma}$) можно по аналогии оценить как $\hat{\Sigma} = C^q_m M(n)$. Покажем, что $\hat{\Sigma} = O(n^{4+q})$. Граф транспортной сети является разреженным графом и $m = O(n)$, тогда $C^q_m = O(n^q).M(n)$ – это последовательная сумма кубов, которая оценивается как $O(n^4)$.

Выражение для Σ зависит от четырёх переменных, которые, в свою очередь, являются зависимыми друг от друга и эта зависимость неизвестна. Понятно, что методы математического анализа в такой ситуации неприменимы.

Построим график Σ, приняв следующее допущение. Будем считать, что в исходном графе транспортной сети отношения количества ребер к количеству вершин всего графа и его достаточно большого подграфа примерно равны. Так, на рис. 1 это отношение для всего графа примерно равно отношению для подграфа, представляющего собой одну из областей. Отношение для всего графа равно $\frac{229}{127} = 1{,}80$, а для подграфов, представляющих собой области, эти отношения равны $\frac{82}{48} = 1{,}71$, $\frac{83}{45} = 1{,}84$ и $\frac{60}{34} = 1{,}76$.

Обозначим это отношение $\xi = m/n = m_s/n_s = m_p/n_p$ и перепишем, используя также равенство $n_z = n_s + n_p$, выражение для Σ относительно n_s и n_p – количества вершин во внутренней и граничной области. При этом необходимо через n_s и n_p оценить величину $m - m_z$. Это множество ребер состоит из ребер, соединяющих внешние вершины, количество которых оценивается как $\xi(n - n_s - n_p)$, и множества ребер, соединяющих внешнюю и граничную вершины. Оценим второе множество в наихудшем случае. С одной стороны, чем меньше будет размер внешней области, тем меньше будет количество комбинаций q ребер, лежащих во внешней области. С другой стороны, каждая граничная вершина по определению имеет хотя бы одно ребро, связанное с внешней вершиной. Поэтому оценим количество таких ребер величиной n_p. В результате получаем оценку $m - m_z \approx \xi(n - n_s - n_p) + n_p$.

После подстановки Σ_1 остается без изменений,

$$\Sigma_2 = (C_m^q - C^q_{\xi(n-n_s-n_p)+n_p})M(n),$$

$$\Sigma_3 = C^q_{\xi(n-n_s-n_p)+n_p}(M(n-n_s) + (n - n_s - n_p)n_s n_p).$$

В итоге получаем функцию $\Sigma(n_s, n_p)$ с параметрами n, q и ξ.

На рис. 6 и 7 показаны графики этой функции в двух ракурсах при различных значениях параметров.

Показанные на рисунках графики – типичные, т.е. и при других значениях параметров наблюдается схожая картина: наличие «оврага», «дно» которого проходит примерно параллельно плоскости (Σ, n_p). Примем это обстоятельство в качестве еще одного допущения. С его помощью становится возможным решить проблему зависимости переменных n_s и n_p. Обозначим n_s^0 – значение n_s, зависящее от $n,\ q$ и ξ, в котором $\Sigma(n_s, n_p)$ достигает своего минимума при $n_p = 1$. В силу принятого допущения для других фиксированных значений n_p минимум $\Sigma(n_s, n_p)$ также будет достигаться при $n_s = n_s^0$. Тогда можно утверждать, что алгоритм нахождения критических ребер будет работать наиболее быстро, если замкнутое подмножество вершин удовлетворяет следующим условиям:

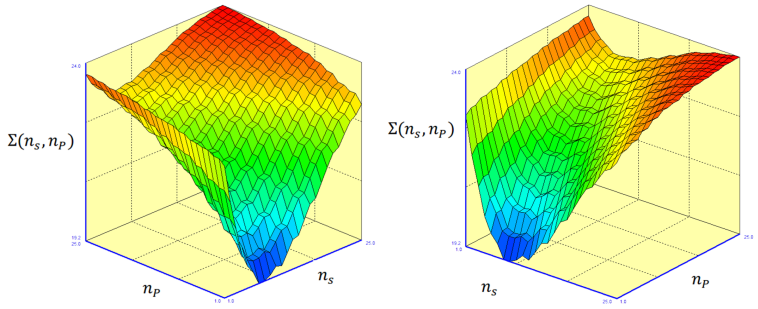

Рис. 6. График $\Sigma(n_s, n_p)$ *при* $n = 50,\ q = 4,\ \xi = 1{,}7$ *в двух ракурсах*

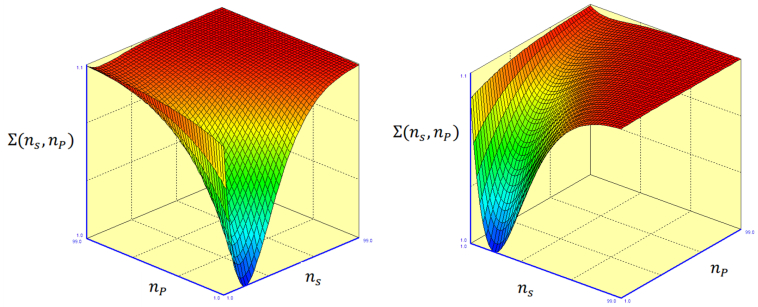

Рис. 7. График $\Sigma(n_s, n_p)$ *при* $n = 200,\ q = 10,\ \xi = 1{,}5$ *в двух ракурсах*

1. Мощность внутреннего множества примерно равна n_s^0.
2. Мощность граничного множества минимальна.

Рассмотрим, как можно добиться выполнения каждого условия.

6.1. Мощность внутреннего множества

По заданному графу и величине q несложно определить значения n и ξ и численными методами найти n_s^0. Чтобы ускорить эту операцию, был проведен анализ интервала значений n_s^0 при

различных параметрах n, q и ξ с целью его уменьшения. Для этого фиксировался один из параметров и рассматривался трехмерный график относительно двух других. Было установлено, что интервал значений n_s^0 практически не зависит от ξ. Диапазон изменения параметра ξ можно оценить интервалом (1, 2): если граф сети представляет собой дерево, то ξ близко к единице; для наиболее плотных дорожных сетей (например, район Манхэттен в Нью-Йорке), граф которых практически является регулярным графом степени 4, $\xi = 2$.

На рис. 8 показаны типичный график зависимости n_s^0 от n и q и для наглядности – зависимость относительного значения $\frac{n_s^0}{n}$ от этих же параметров.

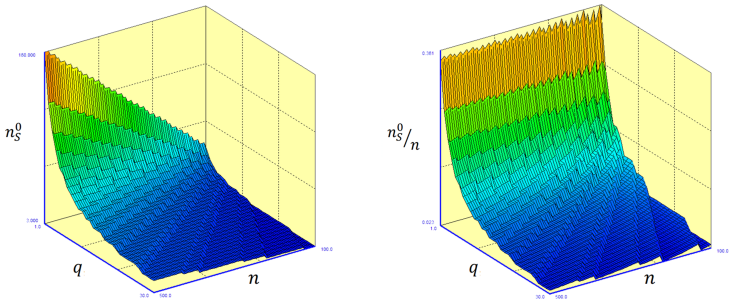

Рис. 8. График зависимости n_s^0 *и* $\frac{n_s^0}{n}$
при $\xi = 1{,}6, n = 100, ..., 500, q = 1, ..., 30$

Анализ этих и схожих графиков привел к следующим выводам:

1. При любых значениях n, q и ξ, величина n_s^0 не превосходит $n/2$, причем максимальное значение достигается при $q = 1$. Этот вывод был также подтвержден полным перебором значений $\Sigma(n_s, n_p)$ при различных значениях параметров.

2. График зависимости $\Sigma(n_s, 1)$ от n_s при фиксированных n, q, ξ и $n_p = 1$ имеет локальные минимумы. На рис. 9 показан пример такого графика. Для наглядности ось ординат имеет отно-

сительную шкалу: вместо абсолютного значения Σ используются значения $\frac{\Sigma-\Sigma_{\min}}{\Sigma_{\min}}$.

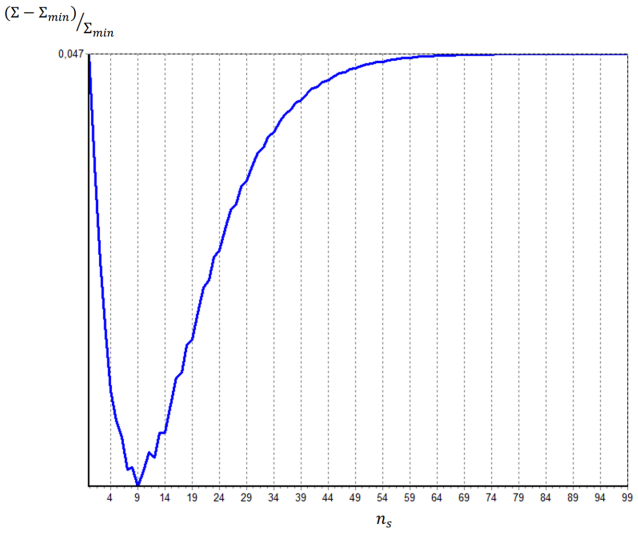

Рис. 9. График зависимости $\Sigma(n_s, 1)$ *от* n_s *при* $\xi = 1{,}6, n = 200, ..., 500, q = 20$

Поэтому логично определять n_s^0 численными методами, вычисляя значения $\Sigma(n_s, 1)$, начиная с $n_s = 1$ до $n_s = n/2$.

6.2. Мощность граничного множества

Итак, необходимо построить замкнутое множество такое, чтобы мощность внутреннего множества приблизительно равнялась заданной величине n_S^0, а мощность граничного множества была как можно меньше. Создание эффективного алгоритма решения этой подзадачи требует дополнительного исследования по следующим причинам. Логично адаптировать и использовать один из многочисленных алгоритмов разбиения графа на сообщества и подобрать сообщество, мощность которого близка к n_S^0. Но тут возникнут трудности связанные с тем, что кратчайший путь между двумя вершинами определяется как сумма длин ребер пути. Если две вершины в сообществе связаны ребром, то

это не означает, что кратчайший путь между ними проходит по этому ребру, и сообщество не обязательно является замкнутым множеством. Здесь приводится простой и, возможно, не самый эффективный алгоритм.

Для обоснования алгоритма был проведен анализ области применимости предлагаемого метода. Для этого фиксировались параметры $n,\ q,\ \xi$ и рассматривалась плоскость, на которой оси абсцисс соответствует значение n_s, а оси ординат – n_p. На этой плоскости отмечались точки (синим цветом), для которых выгоднее использовать метод полного перебора, т.е. $\hat{\Sigma}(n_s, n_p) < \Sigma(n_s, n_p)$. На рис. 10 показан пример такой раскраски. Коричневой линией отмечено значение n_s^0.

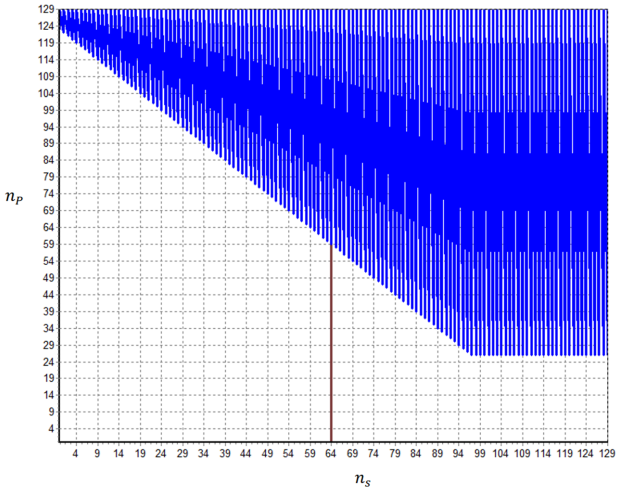

Рис. 10. Область применимости метода при $n = 130, q = 11, \xi = 1{,}6$

Во всех подобных графиках синяя область представляла собой прямоугольную трапецию, хотя во многих случаях она отсутствовала. То есть значение $\hat{n}_p$ такое, что для всех $n_p > \hat{n}_p$ выгоднее использовать метод полного перебора, превосходило n. Также из этих графиков видно, что если значение n_s мало, то использование предлагаемого метода выгоднее, чем метод пол-

ного перебора, так как в транспортных сетях не встречается ситуация, когда граничное множество для небольшого внутреннего множества состоит практически из всех вершин графа. Имеется в виду внутреннее множество, географическое расстояние между вершинами которого существенно меньше максимального географического расстояния между вершинами графа. Предлагаемый алгоритм нахождения замкнутого множества с требуемыми характеристиками основывается на методе перебора и состоит из следующих шагов.

1. Определяется величина n_s^0. Величина Σ_{min} инициализируется равной бесконечности. Множества Z_{min}, S_{min}, P_{min} инициализируются как пустые множества.

2. Для каждой вершины графа v строится замкнутое множество $Z = S \cup P$ с начальным множеством $S_0 = \{v\}$.

3. Вычисляются значения $\Sigma(n_s, n_p)$ и $\hat{\Sigma}(n_s, n_p)$. Если полученное Z находится в области применимости метода и $n_s \leqslant n_s^0$, то $\Sigma(n_s, n_p)$ сравнивается с Σ_{min}. Если $\Sigma(n_s, n_p) < \Sigma_{min}$, то $Z_{min} = Z$, $S_{min} = S$, $P_{min} = P$, $\Sigma_{min} = \Sigma(n_s, n_p)$.

4. Если $n_s > n_s^0$, то переход к следующей вершине и п. 2.

5. Если $n_s \leqslant n_s^0$, то область Z расширяется: строится замкнутое множество с начальным множеством $S_0 = S \cup \{u\}$, где u – произвольная вершина из граничного множества P. Переход к п. 3.

Конец алгоритма.

Очевидно, что алгоритм работает за конечное число шагов. В п. 2 для каждой вершины строится замкнутое множество, в котором эта вершина является внутренней. В этом случае мощность S невелика, и это замкнутое множество находится в области применимости, т.е. практически сразу определяется замкнутое множество, для которого предлагаемый метод работает быстрее метода полного перебора. Далее (в п. 5) замкнутое множество расширяется до тех пор, пока мощность внутреннего множества не превысит оптимального значения n_s^0. Алгоритм можно модифицировать и в п. 4 использовать условие $n_s \geqslant \beta n_s^0$, где β – константа, большая единицы. Но определение оптимального значе-

ния β также требует дополнительных исследований. Среди всех полученных замкнутых множеств для дальнейшей работы предлагаемого метода выбирается множество с минимальным значением $\Sigma(n_s, n_p)$. Из-за того, что расширение происходит через произвольную граничную вершину, алгоритм не перебирает все возможные замкнутые множества, поэтому найденное множество может не быть оптимальным. Тем не менее решение задачи предлагаемым методом с полученным замкнутым множеством происходит быстрее, чем полный перебор. Осталось показать, что выполнение этого алгоритма не приведет к заметному увеличению сложности решения всей задачи. Так как для каждой вершины строится замкнутое множество мощности n_s^0, то это утверждение эквивалентно выполнению неравенства $n\Sigma_1 \ll \Sigma_2 + \Sigma_3$. Неравенство несложно обосновать, используя соотношение порядков. Будем считать, что все величины: количество вершин и ребер различных множеств (внутреннего, граничного и т.д.) по порядку оцениваются как $O(n)$. Тогда $n\Sigma_1 = O(n^5)$, так как основное слагаемое в Σ_1 – это последовательная сумма кубов. Из суммы $\Sigma_2 + \Sigma_3$ рассмотрим слагаемое $C^q_{m-m_z} M(n - n_s)$. $C^q_{m-m_z} M(n - n_s) = O(n^q)O(n^4) = O(n^{4+q})$ и уже при $q > 1$ это слагаемое растет быстрее, чем $n\Sigma_1$.

7. Заключение

В работе рассматривается задача нахождения критических узлов транспортной сети и метод ее решения с помощью вычисления аргументов, при которых функция обобщенной стоимости поездок достигает своего максимума. Наиболее трудоемкая операция в этом методе заключается в многократном вычислении матрицы длин кратчайших путей между всеми вершинами. Введено понятие замкнутой области и предложен метод сокращения вычислений этой матрицы. Метод основан на декомпозиции и последующих вычислений подматриц. Была приведена оценка количества операций и предложен метод построения замкнутой области, минимизирущей количество вычислений при решении задачи нахождения критических узлов.

Литература

1. КРЫГИН А.А., КУПРИЯНОВ Б.В. *Определение критических узлов транспортной сети.* – М.: Изд-во «УБС», 2023. – 24 с.
2. ЛИПСКИЙ В. *Комбинаторика для программистов.* – М.: «Мир», 1988.
3. ПАПАДИМИТРИУ Х.Х., СТАЙГЛИЦ К. *Комбинаторная оптимизация. Алгоритмы и сложность.* – М.: «Мир», 1982. –512 с.
4. BELL M.G.H., CASSIR C. *Reliability of Transport Networks.* – Baldock, Herts: Research Studies Press, 2000.
5. BELL M.G.H., IIDA Y. *Transportation Network Analysis.* – Chichester: John Wiley and Sons, 1997.
6. BI Y.C., WILLIAM H.K., LAM A.S., QINGQUAN L. ET AL. *Vulnerability analysis for large-scale and congested road networks with demand uncertainty* // Transportation Research Part A: Policy and Practice. – 2012. – Vol. 46, Iss. 3. – P. 501–516. – DOI: https://doi.org/10.1016/j.tra.2011.11.018. – URL: https://www.sciencedirect.com/science/article/pii/S096585641100187X).
7. D'ESTE G.M., TAYLOR M.A.P. *Modelling network vulnerability at the level of the national strategic transport network* // Journal of the Eastern Asia Society for Transportation Studies. –2001. Vol. 4(2). – P. 1–14.
8. D'ESTE G.M., TAYLOR M.A.P. *Network vulnerability: an approach to reliability analysis at the level of national strategic transport networks* // In: The Network Reliability of Transport / Eds.: Y. Iida, M.G.H. Bell. – Oxford: Pergamon-Elsevier, 2003. – P. 23–44.
9. ESFEH M.A., KATTAN L., WILLIAM H.K., LAM M.S. ET AL. *Road network vulnerability analysis considering the probability and consequence of disruptive events: A spatiotemporal incident impact approach* // Transportation Research Part C: Emerging Technologies. – 2021. – Vol. 136. –

103549 – DOI: https://doi.org/10.1016/j.trc.2021.103549. – URL: https://www.sciencedirect.com/science/article/pii/S0968090X21005313.

10. GUZE S. *Graph Theory Approach to the Vulnerability of Transportation Networks* // Algorithms. – 2019. – Vol. 12(12). – P. 270.
11. JENELIUS E., PETERSEN T., MATTSSON L.G. *Importance and exposure in road network vulnerability analysis* // Transportation Research Part A: Policy and Practice. – 2006. – Vol. 40, Iss. 7. – P. 537–560. – DOI: https://doi.org/10.1016/j.tra.2005.11.003. – URL: https://www.sciencedirect.com/science/article/pii/ S096585640500162X.
12. JENELIUS E., PETERSEN T., MATTSSON L-G. *Importance and exposure in road network vulnerability analysis* // Transportation Research Part A Policy and Practice. – February 2006. – Vol. 40(7). – P. 537–560.
13. NEUMANN T., BEHRISCH M. *Terminal reliability of road networks with multiple destination options* // Int. J. of Safety and Security Eng. – 2018. – Vol. 8, No. 3. – P. 426–437.
14. SUGIURA S., KURAUCHI F. *Isolation vulnerability analysis in road network: Edge connectivity and critical link sets* // Transportation Research Part D: Transport and Environment. – 2023. – Vol. 119. – P. 103768. – DOI: https://doi.org/10.1016/j.trd.2023.103768. – URL: https://www.sciencedirect.com/science/article/pii/S1361920923001657.
15. VIVEK S., CONNER H. *Urban road network vulnerability and resilience to large-scale attacks* // Safety Science. – 2022. – Vol. 147. – P. 105575. – DOI: https://doi.org/10.1016/j.ssci.2021.105575. – URL: https://www.sciencedirect.com/science/article/pii/S0925753521004173.
16. WANG S., CHEN C., ZHANG J., GU X., HUANG X. *Vulnerability assessment of urban road traffic systems based on traffic flow* // Int. Journal of Critical Infrastructure

Protection. – 2022. – Vol. 38. – P. 100536. – DOI: https://doi.org/10.1016/j.ijcip.2022.100536. – URL: https://www.sciencedirect.com/science/article/pii/S1874548222000269.

FINDING CRITICAL NODES IN A TRANSPORTATION NETWORK BASED ON CONSTRUCTING A CLOSED REGION

Andrey Krygin, V.A. Trapeznikov Institute of Control Sciences of RAS, Moscow, Cand.Sc. (andreyakr@yandex.ru).
Boris Kupriyanov, V.A. Trapeznikov Institute of Control Sciences of RAS, Moscow, Cand.Sc. (kuprianovb@mail.ru).

Abstract: The problem of finding critical nodes in a transportation network is considered, which is solved by maximizing the generalized cost of travel, which depends on the demand for movement and the cost of travel between each pair of network nodes. The proposed method in the paper is an improvement of exhaustive search, the main difficulty of which lies in the repeated computation of the matrix of minimum travel costs. The method consists of extracting a closed set of vertices from the original graph. The extraction of a closed set of vertices allows for graph reduction, decomposition of the corresponding matrices, and separate computations of sub-matrices. These transformations have helped reduce computations during the search for options. A general algorithm for finding critical nodes has been constructed and optimized. The closed set is divided into an internal and boundary subset. It has been shown that the algorithm works fastest with a minimum size of the boundary subset and an optimal size of the internal subset, for which a corresponding algorithm has been proposed to determine. An algorithm for constructing and expanding the closed set is also proposed, based on which an approximate algorithm for finding the optimal closed set is constructed. It has been shown that the complexity of finding the optimal closed set is much lower than the complexity of the improved exhaustive search method.

Keywords: transport networks, critical object search.

УДК 338.49
ББК 39.311
DOI: 10.25728/ubs.2023.106.10

Статья представлена к публикации членом редакционной коллегии Я.И. Квинто.

Поступила в редакцию 27.07.2023.
Дата опубликования 30.11.2023.