ТРУДЫ ИНСТИТУТА МАТЕМАТИКИ И МЕХАНИКИ УрО РАН

Том 21 № 3 2015

УДК 519.16 + 519.85

ТОЧНЫЙ АЛГОРИТМ С ЛИНЕЙНОЙ ТРУДОЕМКОСТЬЮ ДЛЯ ОДНОЙ ЗАДАЧИ ОБХОДА МЕГАПОЛИСОВ[1]

А. Г. Ченцов, М. Ю. Хачай, Д. М. Хачай

Исследуется задача обхода мегаполисов с фиксированным числом "входов" и специальным образом заданными отношениями предшествования, являющаяся естественным обобщением классической задачи коммивояжера (TSP). Для поиска оптимального решения задачи приводится схема динамического программирования, эквивалентная методу поиска кратчайшего пути в подходящем бесконтурном ориентированном взвешенном графе. Обосновываются условия, при которых трудоемкость алгоритма полиномиально, в частности, линейно зависит от числа мегаполисов.

Ключевые слова: задача коммивояжера (TSP), NP-трудная задача, динамическое программирование, отношения предшествования.

A. G. Chentsov, D. M. Khachai, M. Yu. Khachai. An exact algorithm with linear complexity for a problem of visiting megalopolises.

A problem of visiting megalopolises with a fixed number of "entrances" and precedence relations defined in a special way is studied. The problem is a natural generalization of the classical traveling salesman problem. For finding an optimal solution we give a dynamic programming scheme, which is equivalent to a method of finding a shortest path in an appropriate acyclic oriented weighted graph. We justify conditions under which the complexity of the algorithm depends on the number of megalopolises polynomially, in particular, linearly.

Keywords: traveling salesman problem, NP-hard problem, dynamic programming, precedence relations.

Введение

В различных прикладных задачах возникает необходимость в использовании математических моделей, предусматривающих маршрутизацию, посещения конечной системы мегаполисов при условиях предшествования. Последние могут быть связаны с ограничениями технологического характера и являются по смыслу условиями на маршрут "в целом". Кроме того, нередко возникает потребность в оценивании перемещений между мегаполисами и работ в пределах данных мегаполисов с учетом возможной зависимости от списка заданий (в одних случаях речь идет об уже выполненных заданиях, а в других — о тех, что еще предстоит выполнить).

Так, в частности, в известной задаче о демонтаже оборудования АЭС, выводимого из эксплуатации (см., например, [1]), упомянутая особенность функции стоимости имеет следующий вид: исполнитель в каждый момент времени подвергается радиационному воздействию тех и только тех фрагментов оборудования, которые на данный момент еще не демонтированы.

В другой инженерной задаче, связанной с маршрутизацией перемещений инструмента при высокоточной листовой резке на станках с числовым программным управлением (ЧПУ), напротив, по соображениям, связанным с введением штрафов за несоблюдение технологических ограничений (жесткости листа), имеет смысл использовать функции стоимости, допускающие зависимость от уже выполненных заданий. Отметим, что при должной формализации оба упомянутых варианта, связанных с усложнением функции стоимости, можно объединить, что и было сделано в [2–4] и ряде других работ. Важно отметить, что в этих исследованиях доминирующую роль играют не вполне традиционные для дискретной оптимизации осложнения, связанные с вопросами качественного характера (большое число разнородных ограничений, усложненные функции стоимости и др.).

[1]Исследования поддержаны Российским научным фондом, грант № 14-11-00109.

В то же время и для вышеупомянутых постановок сохраняют свое значение трудности, связанные с вычислительной реализацией. Все исследуемые нами задачи имеют в качестве прототипа известную труднорешаемую задачу коммивояжера (TSP) [5; 6], вопросам проектирования и обоснования точных и приближенных алгоритмов для которой, а также для ее обобщений, посвящено большое количество работ (см., например, [7–11]).

Отметим однако, что упомянутые ранее ограничения предшествования могут сыграть и положительную роль, исключая из рассмотрения значительное число вариантов. Такой подход был предложен и развит в работах [2–4] для процедур типа динамического программирования (ДП); отметим в этой связи применение аппарата ДП для решения TSP в [12; 13]. Упомянутая экономия вычислений реализуется на этапе построения функции Беллмана: построение всего массива значений последней подменяется построением системы слоев (существенных фрагментов). Уравнение Беллмана порождает при этом рекуррентную процедуру, финалом которой является определение глобального экстремума. При этом слои функции Беллмана имеют каждый своей областью определения соответствующий слой пространства позиций. Последнее при наличии условий предшествования оказывается, однако, существенно менее "широким" множеством в сравнении с общим случаем, что и обеспечивает необходимую экономию как числа вычислительных операций, так и объема используемой памяти.

В настоящей работе получена новая оценка трудоемкости алгоритма динамического программирования для одного частного случая задачи обхода мегаполисов, характеризуемого специальным видом ограничений предшествования и восходящего к весьма актуальной инженерной проблеме. Речь идет об оптимизации маршрута при листовой резке на станках с ЧПУ в режиме холостого хода. Предполагается завершенной процедура раскроя, а контуры вырезаемых деталей снабжаются эквидистантами, позволяющими осуществлять резку с некоторым припуском. В реальных задачах возле эквидистант располагаются пары близких точек (врезки и выключения инструмента), образующие в своей совокупности непустые конечные множества, именуемые мегаполисами. Структуру последних на уровне рассматриваемой модели упрощаем, имея в виду близость самих "парных" точек и их близость к эквидистанте. С учетом последнего обстоятельства осуществляем всякий раз "слияние" упомянутых парных точек и отвечающей данной паре точки на эквидистанте, определяющей начало и конец реза. В результате мы приходим к модели, в рамках которой требуется посетить какую-либо одну точку мегаполиса. При этом, конечно, мы не учитываем время, затрачиваемое на резку соответствующих контуров, поскольку оно остается одним и тем же для всех маршрутов, а, стало быть, может быть исключено из математической постановки экстремальной задачи. При этом все перемещения между "укрупненными" упомянутым способом точками осуществляются в режиме холостого хода, т. е. с выключенным инструментом. Разумеется, данная модель вкладывается и в более общую постановку [2–4] посредством применения диагональных отношений, характеризующих возможные варианты внутренних работ. В модели, применяемой ниже, стоимости данных работ зануляются, что соответствует предположению о близости "парных" точек и отвечающих им точек на эквидистанте.

В данной работе мы приводим оценки трудоемкости алгоритма точного решения задачи обхода мегаполисов для специального семейства ограничений, опираясь на аналогичные результаты, полученные ранее в работах [14; 15] для классической задачи коммивояжера.

1. Постановка задачи

Рассмотрим следующую частную постановку задачи оптимальной маршрутизации (рис. 1). Дано: конечные дизъюнктные множества $M_1, \ldots, M_n$, называемые далее *мегаполисами*, и стартовая точка x_0, не принадлежащая ни одному из них. Без ограничения общности, полагаем множества M_j равномощными и определяемыми равенствами

$$M_j = \{g_{j1}, \ldots, g_{jp}\}.$$

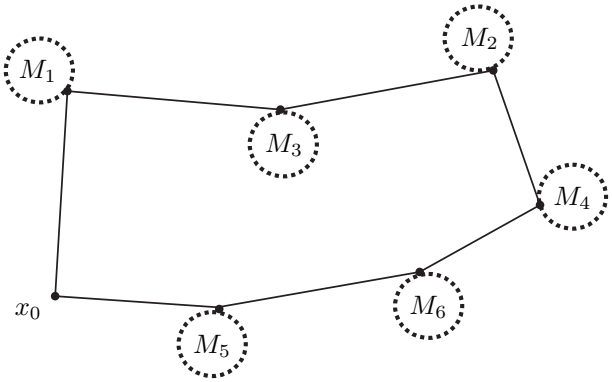

Рис. 1. Пример постановки с ограничениями предшествования (7).

Заданы стоимости $\hat{c}(x_0, g_{j\tau})$ и $\check{c}(g_{j\tau}, x_0)$ перемещения из точки x_0 в каждую из точек $g_{j\tau}$ (и обратно), стоимости перемещения $c(g_{l\sigma}, g_{j\tau})$ для произвольных $j, l \in \mathbb{N}_n = \{1, \ldots, n\}$, $j \neq l$ и $\sigma, \tau \in \mathbb{N}_p$, а также стоимости посещения мегаполисов (внутренних работ) $c'(g_{j\tau})$.

Требуется найти перестановку $\pi\colon \mathbb{N}_n \to \mathbb{N}_n$, задающую порядок посещения мегаполисов, называемую далее *маршрутом*, и конечную последовательность $g_{\pi(1)\tau(1)}, \ldots, g_{\pi(n)\tau(n)}$, именуемую *трассой* так, чтобы

$$\hat{c}(x_0, g_{\pi(1)\tau(1)}) + \sum_{i=1}^{n-1} \bigl(c'(g_{\pi(i)\tau(i)}) + c(g_{\pi(i)\tau(i)}, g_{\pi(i+1)\tau(i+1)})\bigr) + \check{c}(g_{\pi(n)\tau(n)}, x_0) \to \min. \qquad (1)$$

Традиционный подход к решению задачи (1) восходит к работам [12;13] и связан с использованием той или иной схемы динамического программирования. Пусть оптимальная трасса, выйдя из точки x_0, посетив за первые $i-1$ шагов мегаполисы с номерами из подмножества $J \subset \mathbb{N}_n$, в i-й позиции посещает мегаполис M_j в точке $g_{j\tau(i)} \in M_j$. Обозначим стоимость этой части трассы через $C(J, i, j, g_{j\tau(i)})$.

Справедливы следующие рекуррентные соотношения:

$$C(\varnothing, 1, j, g_{j\tau(1)}) = \hat{c}(x_0, g_{j\tau(1)}), \qquad (2)$$

$$C(J, i, j, g_{j\tau(i)}) = \min_{l \in J} \min_{g_{l\tau(i-1)} \in M_l} \bigl\{C(J\setminus\{l\}, i-1, l, g_{l\tau(i-1)}) + c(g_{l\tau(i-1)}, g_{j\tau(i)}) + c'(g_{j\tau(i)})\bigr\}, \qquad (3)$$

причем оптимальное значение задачи (1) может быть вычислено по формуле

$$C^* = \min_{j \in \mathbb{N}_n} \bigl(C(\mathbb{N}_n\setminus\{j\}, n, j, g_{j\tau(n)}) + \check{c}(g_{j\tau(n)}, x_0)\bigr). \qquad (4)$$

Отметим, что рассматриваемая в данной работе "прямая" схема динамического программирования, вообще говоря, отличается от исследованной в работах [1–4].

Рекуррентная процедура (2)–(4) допускает эквивалентную формулировку в терминах теории графов. В самом деле, пользуясь известным подходом, сопоставим исходной задаче задачу поиска кратчайшего s–t пути в подходящем реберно-взвешенном $(n+2)$-дольном ориентированном графе $G^*[p] = (V^*[p], A^*[p], w^*[p])$. Через $V_i^*[p]$ обозначим подмножество вершин, составляющих i-ю долю, задаваемое равенствами

$$V_0^*[p] = \{s\}, \quad V_{n+1}^*[p] = \{t\},$$

$$V_i^*[p] = \{(J, i, j, \tau)\colon j \in \mathbb{N}_n \setminus J,\ g_{j\tau} \in M_j,\ J \subset \mathbb{N}_n,\ |J| = i-1\} \quad (i \in \mathbb{N}_n).$$

При этом вершины s и t соответствуют стартовой точке x_0, а произвольная вершина (J, i, j, τ) соответствует части трассы длины i, посещающей мегаполисы с номерами из множества $J \cup \{j\}$, последним среди которых — мегаполис M_j (в точке $g_{j\tau}$).

Смежными в графе $G^*[p]$ являются лишь вершины последовательных долей $V_i^*[p]$ и $V_{i+1}^*[p]$. Дуги $G^*[p]$ соединяют вершину s с произвольной вершиной доли $V_1^*[p]$, произвольную вершину доли $V_n^*[p]$ — с вершиной t, а также произвольные вершины (J, i, l, σ) и $(J', i+1, j, \tau)$, удовлетворяющие условиям

$$|J| = i - 1, \quad J' = J \cup \{l\}, \quad j \notin J', \quad \sigma, \tau \in \mathbb{N}_p. \tag{5}$$

Множество дуг, соединяющих вершины, принадлежащие $V_i^*[p]$ и $V_{i+1}^*[p]$, договоримся обозначать через $A_{i,i+1}^*[p]$.

Веса дуг определяются соотношениями

$$w^*[p](s, (\varnothing, 1, j, \tau)) = \hat{c}(x_0, g_{j\tau}), \quad w^*[p]((\mathbb{N}_n \setminus \{j\}, n, j, \tau), t) = \check{c}(g_{j\tau}, x_0),$$

$$w^*[p]((J, i, l, \sigma), (J', i+1, j, \tau)) = c(g_{l\sigma}, g_{j\tau}) + c'(g_{j\tau}).$$

Нетрудно убедиться в том, что множество допустимых трасс в задаче (1) взаимно однозначно отображается на множество s–t путей в графе $G^*[p]$. Как следствие, кратчайшей трассе (оптимальному решению задачи (1)) соответствует s–t путь наименьшего веса. Более того, схема (2)–(4) эквивалентна модификации известного алгоритма Форда—Беллмана для поиска кратчайшего пути в бесконтурной сети (см., например, [16]), обладающего трудоемкостью $O(|A^*|)$.

Универсальность описанного подхода к поиску точного решения неоднократно отмечалась ранее (см, например, [17]). Нетрудно видеть, что величины $\hat{c}(x_0, g_{j\tau})$, $\check{c}(g_{j\tau}, x_0)$, $c(g_{l\sigma}, g_{j\tau})$ и $c'(g_{j\tau})$ могут, вообще говоря, зависеть от целого ряда дополнительных параметров, среди которых списки посещенных (или непосещенных) к данному моменту мегаполисов. При этом и схема метода, задаваемая формулами (2)–(4), и сведение к задаче поиска кратчайшего s–t пути остаются неизменными. В дальнейшем для упрощения записи договоримся эти дополнительные параметры опускать.

К сожалению, число дуг в орграфе $G^*[p]$ в общем случае экспоненциально растет с ростом числа мегаполисов n, что влечет экспоненциальную же оценку трудоемкости метода динамического программирования для задачи (1) в общей постановке. В самом деле, для произвольного $n \geq 2$

$$|V^*[p]| > |V_2^*[p] \cup \ldots \cup V_n^*[p]| \geq pn2^{n-2}.$$

При этом входящая степень произвольной вершины $(J, m, j, \tau) \in V_m^*[p]$ при $m \geq 2$ удовлетворяет соотношению $\deg^-(J, m, j, \tau) = (m-1)p \geq p$. Таким образом, для числа дуг графа $G^*[p]$ справедлива нижняя оценка[2] $|A^*[p]| = \Omega(np^2 2^n)$.

Тем не менее, введение в условие задачи дополнительных ограничений, например, ограничений предшествования на множестве мегаполисов, может существенно снизить суммарную трудоемкость процедуры поиска оптимального решения (см., например, [19;20]). Ниже мы воспользуемся подходом, предложенным в работах [14;15] для описания подкласса задачи обхода мегаполисов, для которого схема (2)–(4) обладает линейной по n трудоемкостью.

2. Ограничения предшествования

Пусть для некоторого $k \in \mathbb{N}_n$ на множестве мегаполисов $\{M_1, \ldots, M_n\}$ заданы дополнительные ограничения предшествования: перестановка $\pi\colon \mathbb{N}_n \to \mathbb{N}_n$ является допустимым порядком обхода мегаполисов (маршрутом), если

$$\forall i, j \in \mathbb{N}_n \ (j \geq i + k) \Rightarrow (\pi(i) < \pi(j)). \tag{6}$$

[2]Заметим, что из результатов [18] следует справедливость и верхней оценки $|A^*[p]| = O(n^2 p^2 2^n)$.

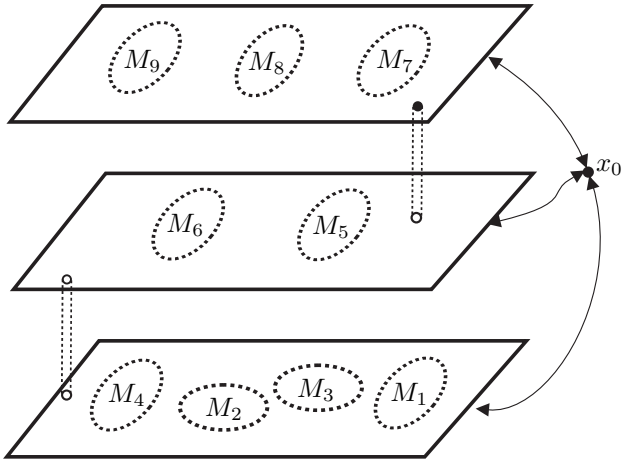

Рис. 2. Пример постановки с ограничениями предшествования (7).

Структура введенной системы ограничений существенно определяется значением параметра k. В самом деле, при $k = 1$ единственным допустимым маршрутом, очевидно, является тождественная перестановка $\pi(i) = i$, в то время, как при $k \geq n - 1$, любая перестановка допустима.

Ограничения (6) могут быть существенно ослаблены. Например, мы можем отказаться от равномерности в ограничениях, предположив для каждого i наличие собственного параметра $k(i)$ такого, что

$$\forall i, j \in \mathbb{N}_n \ (j \geq i + k(i)) \Rightarrow (\pi(i) < \pi(j)). \tag{7}$$

Очевидно, при $k(i) \equiv k$ ограничения (7) эквивалентны ограничениям (6).

Введенные ограничения предшествования на первый взгляд могут показаться излишне абстрактными. Приведем пример задачи обхода мегаполисов, отягощенной ограничениями вида (7), близкой по постановке к реальным задачам, решаемым на практике.

Допустим, исследуемая постановка (рис. 2) требует поиска наиболее экономичного плана работ, связанных с посещением семи помещений (мегаполисов), находящихся на трех уровнях обслуживаемого здания. В начальной позиции исполнитель находится в стартовой точке x_0, из которой он может быть доставлен на любой уровень с одинаковой пренебрежимо малой стоимостью. После проведения всех работ (но не ранее) исполнитель возвращается в стартовую точку с произвольного уровня. Особенность постановки состоит в том, что перемещения между уровнями могут быть произведены исключительно по лифтовым шахтам, отмеченным на рисунке, и стоимость каждого такого перемещения существенно превышает стоимость перемещений внутри уровня. В данной ситуации маршруты, начинающиеся и (или) заканчивающиеся на втором уровне, естественным образом доминируются маршрутами, начинающимися и заканчивающимися на первом или третьем уровнях, и могут быть исключены из рассмотрения. Данный факт может быть формализован в виде специфических ограничений предшествования вида (7), в которых

$$k(i) = \begin{cases} 5 - i, & \text{если } 1 \leq i \leq 4, \\ 7 - i, & \text{если } 5 \leq i \leq 6, \\ 10 - i, & \text{в противном случае.} \end{cases}$$

Легко видеть, что введение таких дополнительных ограничений фактически позволяет свести решение исходной задачи к решению семейства подзадач меньшей размерности.

Ниже мы обсудим модификацию стандартной схемы динамического программирования, позволяющую эффективно находить точное решение задачи (1), отягощенной дополнительными ограничениями предшествования вида (6) или (7).

3. Структура графа $G^*[p]$

Убедимся в том, что при произвольном $p \geq 1$ и фиксированных значениях параметров k или $k(i)$ структура графа $G^*[p]$ определяется структурой графа $G^*[1]$, соответствующего постановке задачи обхода одноточечных мегаполисов, эквивалентной классической задаче коммивояжера с ограничениями предшествования. Поскольку вид произвольной вершины графа $G^*[1]$, принадлежащей доле $V_i^*[1]$, однозначно определяется значениями первых трех параметров, договоримся обозначать каждую такую вершину через (J, i, j).

Предложение 1. *Справедливы следующие соотношения*:

$$V_i^*[p] = V_i^*[1] \times \mathbb{N}_p \quad (i \in \mathbb{N}_n), \tag{8}$$

$$A_{0,1}^*[p] = A_{0,1}^*[1] \times \mathbb{N}_p, \quad A_{n,n+1}^*[p] = A_{n,n+1}^*[1] \times \mathbb{N}_p, \tag{9}$$

$$A_{i,i+1}^*[p] = A_{i,i+1}^*[1] \times \mathbb{N}_p^2 \quad (i \in \mathbb{N}_{n-1}). \tag{10}$$

Доказательство. Задавшись произвольным $p > 1$ и одним из ограничений предшествования, (6) или (7), рассмотрим отображение $\Gamma\colon V^*[p] \to V^*[1]$, задаваемое соотношениями

$$\Gamma(s) = s, \quad \Gamma(t) = t, \quad \Gamma((J, i, j, g_{j\tau})) = (J, i, j).$$

Поскольку смежность вершин в графе $G^*[p]$ при произвольном p определяется соотношением (5), отображение Γ является гомоморфизмом. Более того, вершины $(J, i, l, g_{l\sigma})$ и $(J \cup \{l\}, i+1, j, g_{j\tau})$ смежны тогда и только тогда, когда смежны вершины (J, i, l) и $(J \cup \{l\}, i+1, j)$ в графе $G^*[1]$. По построению

$$\Gamma^{-1}((J, i, j)) = \{(J, i, j, g_{j1}), \ldots, (J, i, j, g_{jp})\},$$

что влечет справедливость соотношений (8)–(10).

Предложение доказано.

Следствие 1. *Справедлива оценка* $|A^*[p]| \leq |A^*[1]|\, p^2$.

Справедливость последующих утверждений непосредственно следует из следствия 1 и свойств графа $G^*[1]$, обоснованных в работе [14].

4. Оценка трудоемкости схемы (2)–(4) при условии (6)

Пусть задача (1) отягощена ограничениями предшествования (6). Покажем, что в этом случае трудоемкость схемы динамического программирования (2)–(4), применяемой для поиска ее решения, линейно зависит от числа мегаполисов n.

Предложение 2 [14]. *Для произвольного маршрута* π *и* $i, j \in \mathbb{N}_n$ *таких, что* $i = \pi(j)$, *справедливы соотношения*

$$j - k + 1 \leqslant i \leqslant j + k - 1, \quad i - k + 1 \leqslant j \leqslant i + k - 1.$$

Таким образом, в любом допустимом маршруте мегаполис M_j может занимать лишь одну из $2k - 1$ допустимых позиций, равно как и позицию i может занимать лишь один из $2k - 1$ мегаполисов. Поскольку число мегаполисов, которые могут занимать в допустимом маршруте заданную позицию i, зависит исключительно от значения параметра k (а не от n, как в общем случае), структура графа состояний $G^*[p]$ существенно упрощается.

Введем обозначения

$$S^-(\pi, i) = \{l \in \mathbb{N}_n\colon l \geq i, \pi(l) \leq i - 1\}, \quad S^+(\pi, i) = \{h \in \mathbb{N}_n\colon h \leq i - 1, \pi(h) \geq i\}.$$

Предложение 3 [14]. *Для произвольного допустимого маршрута π и произвольной позиции $i \in \mathbb{N}_n$ справедливо соотношение*

$$|S^-(\pi, i)| = |S^+(\pi, i)| \le k/2.$$

Каждое множество J, используемое при описании вершин доли $V_i^*[p]$, может быть однозначно описано в терминах подмножеств $S^-(\pi, i)$ и $S^+(\pi, i)$:

$$J = \mathbb{N}_{i-1} \setminus S^+(\pi, i) \cup S^-(\pi, i).$$

Предложение 4 [14]. *Для произвольного $k \in \mathbb{N}_n$* $|A^*[1]| \le n\, k^2 2^{k-2}$.

Непосредственным следствием предложения 4 и следствия 1, а также оценки трудоемкости алгоритма поиска кратчайшего s–t пути в бесконтурной сети является следующее утверждение.

Следствие 2. *Для произвольных $k \in \mathbb{N}_n$ и p трудоемкость схемы* (2)–(4) *составляет* $O(n\, p^2 k^2 2^{k-2})$.

Заметим, что полученная оценка полиномиальна как по n, так и по p (и экспоненциальна по k). Оценка сохраняет полиномиальность по n при $p = O(\mathrm{poly}(n))$ и $k = O(\mathrm{poly}(\log n))$.

5. Случай более общих ограничений предшествования

Пусть задача (1) отягощена ограничениями предшествования (7), имеющими более общую природу, нежели ограничения, рассмотренные в предыдущем разделе. Введем обозначение: $k^*(i) = \max\{k(j)\colon i - k(j) + 1 \le j \le i\}$. Справедливо следующее утверждение.

Предложение 5 [14]. *Для произвольных значений $k(i) \in \mathbb{N}_n$*

$$|A^*[1]| \le \sum_{i=1}^{n} k^*(i)(k^*(i)+1)2^{k^*(i)-2}.$$

Применяя, как и в предыдущем разделе, результат, обоснованный в следствии 1, получаем окончательную оценку.

Следствие 3. *Трудоемкость схемы динамического программирования, применяемой для нахождения точного решения задачи* (1) *с ограничениями предшествования* (7)*, составляет*

$$O\big(p^2 \sum_{i=1}^{n} k^*(i)(k^*(i)+1)2^{k^*(i)-2}\big).$$

Заметим, что полученная оценка путем некоторого огрубления может быть сведена к оценке, представленной в следствии 2. Достаточно ввести обозначение $k = \max k^*(i)$. Видно, что в этом случае оценка сохраняет линейный порядок роста по n и квадратичный по p, будучи экспоненциальной по k.

Заключение

В настоящей работе обоснована эффективность схемы динамического программирования для решения задачи оптимального обхода мегаполисов при дополнительных ограничениях предшествования. Показано, что если эти ограничения имеют вид (6) или (7), то получаемый в результате алгоритм позволяет находить точное решение задачи за время, ограниченное сверху линейной функцией от числа мегаполисов n и квадрата их мощности p^2 (при произвольных

фиксированных значениях параметров ограничений предшествования k или $k(i)$). Показано, что найденные оценки трудоемкости алгоритма остаются справедливыми даже при условии, когда функции стоимости зависят от списков посещенных или непосещенных мегаполисов.

Отметим, что в общем случае задачи обхода мегаполисов обсуждаемый в работе алгоритм, конечно же, не является полиномиальным. Более того, из полученных в работе результатов следует, что в общем случае обхода мегаполисов даже задача нахождения маршрута, удовлетворяющего совместно с оптимальным маршрутом ограничению (6) или (7) для каких-либо k, $k^*(i) = O(\mathrm{poly}(\log n))$ NP-трудна.

Тем не менее, обсуждаемые в работе алгоритмы могут использоваться при построения эффективных эвристик для поиска и оценки приближенных решений задачи обхода мегаполисов в общем случае.

СПИСОК ЛИТЕРАТУРЫ

1. Методы маршрутизации и их приложения в задачах повышения безопасности и эффективности эксплуатации атомных станций / В.В. Коробкин, А.Н. Сесекин, О.Л. Ташлыков, А.Г. Ченцов. М.: Новые технологии, 2012. 234 с.
2. **Ченцов А.Г.** К вопросу о маршрутизации комплексов работ // Вестн. Удмурт. ун-та. 2013. Вып. 1. С. 59–82. (Математика, механика, компьютер. науки).
3. **Ченцов А.Г., Ченцов А.А., Ченцов П.А.** Элементы динамического программирования в экстремальных задачах маршрутизации // Проблемы управления. 2013. № 5. С. 12–21.
4. **Ченцов А.Г.** Задача последовательного обхода мегаполисов с условиями предшествования // Автоматика и телемеханика. 2014. Вып. 4. С. 170–190.
5. **Karp R.** Reducibility among combinatorial problems // Complexity of Computer Computations: Proc. Sympos. / eds. R. E. Miller, J. W. Thatcher. New York: Plenum, 1972. P. 85–103.
6. **Garey M., Johnson D.** Computers and intractability: a guide to the theory of NP-completeness. San Francisco: W. H. Freeman and Co, 1979. 339 p. (A Ser. of Books in the Mathematical Sciences.)
7. **Christofides N.** Worst-case analysis of a new heuristic for the traveling salesman problem // Proc. of a Symposium on new directions and recent results in algorithms and complexity. New York: Academic Press, 1976. P. 441.
8. **Arora S.** Polynomial-time approximation schemes for Euclidean traveling salesman and other geometric problems // J. ACM. 1998. Vol. 45, № 5. P. 753–782.
9. **Гимади Э.Х., Перепелица В.А.** Асимптотически точный подход к решению задачи коммивояжера // Управляемые системы: сб. науч. тр. / Ин-т математики СО АН СССР. Новосибирск, 1974. Вып. 12. С. 35–45.
10. **Хачай М.Ю., Незнахина Е.Д.** Аппроксимируемость задачи о минимальном по весу цикловом покрытии графа // Докл. РАН. 2015. Т. 461, № 6. С. 644–649.
11. **Хачай М.Ю., Незнахина Е.Д.** PTAS для задачи Min-k-SCCP в евклидовом пространстве произвольной фиксированной размерности // Тр. Ин-та математики и механики УрО РАН. 2014. Т. 20, № 4. С. 297–311.
12. **Bellman R.** Dynamic programming treatment of the traveling salesman problem // J. Assoc. Comput. Mach. 1962. Vol. 9. P. 61–63.
13. **Held M., Karp R.** A dynamic programming approach to sequencing problems // J. Soc. Indust. Appl. Math. 1962. Vol. 10, no. 1. P. 196–210.
14. **Balas E.** New classes of efficiently solvable generalized traveling salesman problems // Ann. Oper. Res. 1999. Vol. 86. P. 529–558.
15. **Balas E., Simonetti N.** Linear time dynamic-programming algorithms for new classes of restricted TSPs: a computational study // INFORMS J. Comput. 2001. Vol. 13, no. 1. P. 56–75.
16. Introduction to algorithms / eds. T. Cormen [et al.]. 3 ed. Boston: MIT Press, 2009. 1312 p.
17. **Меламед И.И., Сергеев С.И., Сигал И.Х.** Задача коммивояжера. Точные методы // Автоматика и телемеханика. 1989. Вып. 10. С. 3–29.
18. The traveling salesman problem: a guided tour of combinatorial optimization / eds. E.L. Lawler [et al.]. New York: John Wiley & Sons, 1985. 476 p.

19. **Steiner G.** On the complexity of dynamic programming for sequencing problems with precedence constraints // Ann. Oper. Res. 1990. Vol. 26, no. 1-4. P. 103–123.
20. Динамическое программирование в обобщенной задаче курьера, осложненной внутренними работами / А.М. Григорьев, Е.Е. Иванко, С.Т. Князев, А.Г. Ченцов // Мехатроника, автоматизация, управление. 2012. № 7. С. 14–21.

Поступила 11.05.2015

Ченцов Александр Георгиевич
д-р физ.-мат. наук, профессор
чл.-корр. РАН
главный науч. сотрудник
Институт математики и механики им. Н. Н. Красовского УрО РАН,
Уральский федеральный университет им. Б. Н. Ельцина
e-mail: chentsov@imm.uran.ru

Хачай Михаил Юрьевич
д-р физ.-мат. наук, профессор
зав. отделом
Институт математики и механики им. Н. Н. Красовского УрО РАН,
Уральский федеральный университет им. Б. Н. Ельцина
e-mail: mkhachay@imm.uran.ru

Хачай Даниил Михайлович
студент
Уральский федеральный университет им. Б. Н. Ельцина
e-mail: daniil.khachay@gmail.com