ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Май—июнь 2014. Том 21, № 3. С. 64–75

УДК 519.863

ТОЧНЫЙ АЛГОРИТМ РЕШЕНИЯ ДИСКРЕТНОЙ ЗАДАЧИ ВЕБЕРА ДЛЯ k-ДЕРЕВА

А. В. Панюков, Р. Э. Шангин

Аннотация. Рассматривается известная NP-трудная задача размещения взаимосвязанных объектов — дискретная задача Вебера. Предлагается последовательный детерминированный алгоритм, находящий точное решение задачи для k-дерева и конечного множества позиций размещения. Алгоритм использует идею динамического программирования на основе дерева декомпозиции. Проведён вычислительный эксперимент по анализу эффективности предложенного алгоритма в сравнении с пакетом IBM ILOG CPLEX.

Ключевые слова: задача Вебера, k-дерево, динамическое программирование, дерево декомпозиции, точный алгоритм.

Введение

Рассматривается задача Вебера в дискретной постановке [7, 8, 19], когда размещаемый граф имеет вид k-дерева. Математической моделью данной задачи могут быть описаны многие задачи проектирования логистической инфраструктуры предприятий, размещения единиц технологического оборудования в цехах, а также иные задачи создания сложных систем, состоящих из взаимосвязанных элементов. Приведём формулировку исследуемой задачи.

Пусть $G = (J, E)$ — k-дерево [1], где J — множество вершин графа (размещаемые объекты), E — множество рёбер графа G (связи между размещаемыми объектами). Пусть V — конечное множество позиций (точек), предназначенных для размещения вершин графа G. *Размещением* вершин графа G назовём однозначное отображение $\pi : J \to V$, т. е. вершина $i \in J$ размещается в позицию $\pi(i) \in V$. Множество всех однозначных отображений множества вершин J в множество позиций V обозначим через $\Pi = \{\pi \mid J \to V\}$. Пусть $p(i, \vartheta)$ — функция стоимости размещения вершины $i \in J$ в позиции $\vartheta \in V$ и $c([i, j], \vartheta, \kappa)$ — функция стоимости размещения ребра $[i, j] \in E$ на V^2 при размещении его концевых вершин $i, j \in J$ в позициях $\vartheta, \kappa \in V$ соответственно.

Необходимо разместить вершины графа G в позициях множества V так, что суммарная стоимость размещения вершин и рёбер графа G минимальна:

$$F(\pi) = \sum_{[i,j]\in E} c([i,j],\pi(i),\pi(j)) + \sum_{i\in J} p(i,\pi(i)) \to \min_{\pi\in\Pi}. \tag{1}$$

В общем случае дискретная задача Вебера NP-трудна [6] и представляет собой релаксацию квадратичной задачи о назначениях [3, 4, 9], где условие инъективности отображения множества вершин графа J в конечное множество позиций размещения V снимается, т. е. в дискретной задаче Вебера в одну позицию возможно размещение нескольких вершин графа [7, 8, 12].

Задача Вебера исследовалась в различных постановках, в том числе для непрерывной области размещения [10], в многокритериальной постановке [23] и др. Известны полиномиально разрешимые частные случаи. Для решения задачи Вебера на древовидной сети в непрерывной постановке, т. е. когда допускается размещение объектов на дугах, разработаны полиномиальные алгоритмы [5]. В [11] предложен полиномиальный алгоритм решения минимаксной задачи Вебера на дереве. Предложен полиномиальный алгоритм для решения задачи Вебера для корневого дерева и конечного множества мест размещения [19].

1. Дерево декомпозиции k-дерева

Для полноты изложения приведём известное определение k-дерева $G = (J, E)$ [1].

Определение 1. Связный неориентированный граф G называется *k-деревом*, если его построение возможно осуществить рекурсивно по правилам: полный граф из $k + 1$ вершин есть k-дерево; k-дерево с $i + 1$ вершинами получается из k-дерева с i вершинами добавлением в него новой вершины j и k рёбер таким образом, что новая вершина j смежна со всеми вершинами некоторой клики размера k.

Класс k-деревьев введён в 1970-х гг. в [21, 22]. Свойства k-дерева подробно рассмотрены в [1, 13, 18, 20–22].

Способ построения k-дерева при $k = 2$ представлен на рис. 1.

Широкий интерес к изучению графов такого вида вызван тем, что некоторые NP-трудные задачи теории графов полиномиально разрешимы, если модельный граф является k-деревом. В данных условиях задача может быть решена методом динамического программирования на основе *дерева декомпозиции* модельного графа [14, 15]. Развернутое определение дерева декомпозиции и исчерпывающий анализ его свойств можно найти в [1, 2, 15, 16]. Ниже приведено определение дерева декомпозиции k-дерева с применением используемых в статье обозначений.

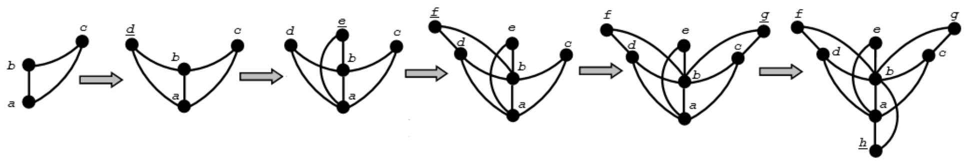

Рис. 1. Способ построения 2-дерева

Деревом декомпозиции k-дерева $G = (J, E)$ назовём $T = (M, W)$ с множествами узлов $M = K \cup S$, где K — множество клик графа G размера $k+1$, $S = \{X \cap Z \mid X, Z \in K, |X \cap Z| = k\}$, и рёбер $W = \{[X, Y] \mid X \in K, Y \in S, |X \cap Y| = k\}$. Вершины дерева T будем называть *узлами* для избежания путаницы с вершинами исходного графа G. Узел $X \in K$ в дальнейшем будем называть *узлом-кликой*. Поскольку каждое множество $Y \in S$ является сепаратором графа G [1, 2] (т. е. при удалении вершин, принадлежащих $Y \in S$, а также всех смежных с ними рёбер из графа G, увеличивается число компонент связности графа G), соответствующий узел Y будем называть *узлом-сепаратором*. Стоит заметить, что ребро $[X, Y] \in W$ существует только между узлом-кликой $X \in K$ и узлом-сепаратором $Y \in S$ и только тогда, когда $Y \subset X$ [17].

Пример 2-дерева и его дерева декомпозиции представлены на рис. 2.

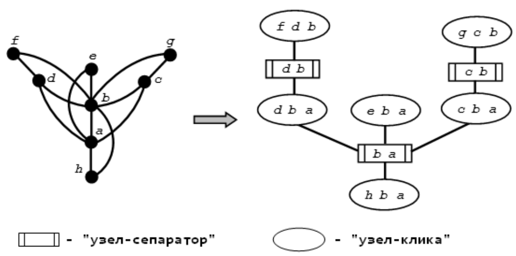

Рис. 2. Дерево декомпозиции 2-дерева

Построение дерева декомпозиции $T = (M, W)$ k-дерева G может быть осуществлено, например, с помощью известного полиномиального алгоритма [17], имеющего оценку вычислительной сложности $O(|J|^3)$.

2. Точный алгоритм для решения задачи Вебера для k-дерева

Обозначим через (G, V, F) исследуемый частный случай дискретной задачи Вебера, где $G = (J, E)$ — k-дерево, V — конечное множество позиций размещения и F — функция стоимости размещения графа G. Предлагается алгоритм kTrWPA (k-tree Weber Problem Algorithm), использующий идею динамического программирования (ДП) на основе дерева декомпозиции, находящий оптимальное решение задачи (G, V, F). Идея предлагаемого алгоритма kTrWPA заключается в разбиении исходной сложной задачи (G, V, F) на конечное число более простых комбинаторных задач, последовательно решая которые, алгоритм строит точное решение исходной задачи.

Для ясности описания логики работы алгоритма введём следующие обозначения. Пусть N — мощность множества M. Выбор узла $X_N \in M$ в качестве корня дерева $T = (M, W)$ индуцирует на множестве узлов M отношение частичного порядка

$$L = \{(X, Y) \mid Y \text{ принадлежит цепи в } T \text{ между } X \text{ и } X_N\}, \qquad (2)$$

где $X, Y \in M$. В дальнейшем будем считать, что

$$M = \{X_i\}_{i=1}^{N}, \quad (X_l, X_m) \in L \Rightarrow l < m.$$

Для каждого узла $X_i \in M$ определяется множество D_i узлов, являющихся *прямыми* потомками узла X_i, т. е. $[X_i, X_j] \in W$ для любого $X_j \in D_i$ и $i > j$. Также для каждого узла $X_i \in M$ определяется множество $\overline{D}_i$ узлов, являющихся потомками узла X_i, т. е. для каждого $X_j \in \overline{D}_i$ найдётся цепь $l = \{X_i, \ldots, X_j\}$, соединяющая узлы X_i и X_j, такая, что для любого $X_m \in l : X_m \neq X_i$ справедливо неравенство $i > m$. Пусть G_i — подграф G, индуцированный вершинами, принадлежащими множеству $X_i \cup \overline{D}_i$.

Пусть $\Pi(X_i) = \{\pi_i \mid X_i \to V\}$ — множество всех однозначных отображений вершин клики $X_i \in M$ в множестве позиций V, где $\pi_i = \{\pi(j) \mid j \in X_i\}$ — некоторый способ размещения вершин клики X_i в множестве позиций V. Отметим, что для любых узлов-клик $X_i \in K$ имеет место равенство $|\Pi(X_i)| = |V|^{k+1}$, а для любых узлов-сепараторов $Y_i \in S$ справедливо равенство $|\Pi(Y_i)| = |V|^k$. Размещения π_i и π_j будем обозначать через $\pi_i \bowtie \pi_j$ тогда и только тогда, когда справедливо равенство $|\pi_i \cap \pi_j| = k$. Пусть $P(\pi_i)$ — функция стоимости способа размещения π_i вершин клики X_i в множестве позиций V:

$$P(\pi_i) = \sum_{j \in X_i,\ \pi(j) \in \pi_i} p(j, \pi(j)).$$

Пусть $C(\pi_i)$ — функция стоимости размещения рёбер подграфа графа G, индуцированного множеством вершин X_i, когда размещение множества вершин клики X_i в позициях V равно π_i:

$$C(\pi_i) = \sum_{j,l \in X_i,\, \pi(j),\pi(l) \in \pi_i} c([j,l], \pi(j), \pi(l)).$$

Процесс решения задачи (G, V, F) разбивается на $N+2$ этапа процесса ДП. Обозначим через $V_i = \{V(\pi_i) \mid \pi_i \in \Pi(X_i)\}$ множество состояний процесса ДП на этапе i, где под состоянием $V(\pi_i)$ понимается способ оптимального размещения вершин подграфа G_i в множестве позиций V, когда размещение вершин его клики X_i в множестве V равно π_i. Определяется функция Беллмана $f_i(\cdot)$ для процесса ДП решения задачи (G, V, F). Значение функции $f_i(\pi_i)$, вычисленное на этапе i для некоторого состояния $V(\pi_i) \in V_i$, есть стоимость оптимального размещения подграфа G_i в множестве позиций V, когда размещение вершин его клики X_i в множестве V равно π_i.

АЛГОРИТМ kTrWPA

Input: k-дерево $G = (J, E)$, множество V, функции $p(\cdot)$ и $c(\cdot)$.

Output: оптимальное отображение π^*_G вершин графа G в V.

Method:

Этап 0 (начальный) процесса ДП

Шаг 1. Построить дерево декомпозиции $T = (M, W)$ k-дерева G.

Шаг 2. Выбрать любой узел-клику $X_N \in M$ в роли корня дерева T. Задать отношение порядка (2) на множестве узлов M. Для каждого узла $X_i \in M$ определить множества D_i и $\overline{D}_i$.

Этап $i = 1, 2, \ldots, N$ процесса ДП

Шаг 1. Определить множество $\Pi(X_i) = \{\pi_i\}$.

Шаг 2. Если X_i — узел-клика, то выполнить шаги 2.1 и 2.2, затем перейти на шаг 4.

Шаг 2.1. Для каждого $\pi_i \in \Pi(X_i)$ вычислить значение функции $R(\pi_i)$ стоимости оптимального размещения подграфа G_i в множестве позиций V, когда размещение вершин его клики X_i размера $k+1$ в V равно π_i:

$$R(\pi_i) = P(\pi_i) + C(\pi_i) + \sum_{X_j \in D_i | \pi_j \bowtie \pi_i} f_j(\pi_j) - \sum_{X_j \in D_i | \pi_j \bowtie \pi_i} \left(P(\pi_j) + C(\pi_j)\right), \quad (3)$$

где четвёртое слагаемое в правой части формулы используется для исключения «повторного счёта» при сложении стоимостей размещения вершин и рёбер подграфа, индуцированного кликой X_i, и подграфов G_j таких, что $X_j \in D_i$.

Шаг 2.2. Для каждого $\pi_i \in \Pi(X_i)$ определить множество $V(\pi_i)$ оптимального размещения вершин подграфа G_i в множестве позиций V, когда размещение вершин клики X_i размера $k+1$ в V равно π_i:

$$V(\pi_i) = \pi_i \cup \Big(\bigcup_{X_j \in D_i,\, \pi_j \bowtie \pi_i} V(\pi_j) \Big).$$

Шаг 3. Если X_i — узел-сепаратор, то выполнить шаги 3.1 и 3.2, затем перейти на шаг 4.

Шаг 3.1. Для любых $\pi_i \in \Pi(X_i)$ и каждого t такого, что $X_t \in D_i$, определить функцию $\overline{R}(\pi_i, t)$ стоимости оптимального размещения подграфа G_t в множестве позиций V, когда размещение вершин его клики X_i размера k в V равно π_i, и определить соответствующее множество $\overline{V}(\pi_i, t)$ такого оптимального размещения вершин подграфа G_t по формулам:

$$\overline{R}(\pi_i, t) = \min_{\pi_t \in \Pi(X_t),\, \pi_t \bowtie \pi_i} \{R(\pi_t)\},$$

$$\overline{V}(\pi_i, t) = V(\pi_t^*), \quad \pi_t^* = \arg \min_{\pi_t \in \Pi(X_t),\, \pi_t \bowtie \pi_i} \{R(\pi_t)\}.$$

Шаг 3.2. Для каждого $\pi_i \in \Pi(X_i)$ вычислить значение функции Беллмана

$$f_i(\pi_i) = \sum_{t : X_t \in D_i} \overline{R}(\pi_i, t) \; - \; (P(\pi_i) + C(\pi_i))\,(|D_i| - 1)$$

и определить множество

$$V(\pi_i) = \bigcup_{t : X_t \in D_i} \overline{V}(\pi_i, t). \tag{4}$$

Шаг 4. Если узел X_i — корень дерева T, то перейти на конечный этап $N+1$ процесса ДП, иначе — на этап $i+1$.

Этап $N+1$ (конечный) процесса ДП. Найти оптимальное размещение π_G^* вершин графа G в множестве позиций V по формуле

$$\pi_G^* = V(\pi_N^*) : \pi_N^* = \arg \min_{\pi_N \in \Pi(X_N)} \{R(\pi_N)\}. \tag{5}$$

Стоп.

Теорема 1. *Алгоритм* kTrWPA *строит точное решение задачи Вебера* (G, V, F), *где* $G = (J, E)$ — k-*дерево,* V — *конечное множество позиций размещения. Оценки его вычислительной и пространственной сложности равны* $O(|V|^{k+1} \cdot |J|^2)$.

Доказательство. Разобьём исходную задачу (G, V, F) на последовательность задач (G_i, V, F), где $1 \leqslant i \leqslant N$. Каждая задача (G_i, V, F), в свою очередь, разбивается на ряд подзадач $(G(\pi_i), V, F)$ для любых $\pi_i \in \Pi(X_i)$, где некоторая подзадача $(G(\pi_i), V, F)$ заключается в оптимальном размещении вершин подграфа G_i в множестве позиций V, когда размещение вершин его клики X_i в V равно π_i.

Любая подзадача $(G(\pi_i), V, F)$, $\pi_i \in \Pi(X_i)$, когда X_i — узел-клика, решается оптимально, поскольку в силу однозначности отображения π оптимальное решение подзадачи $V(\pi_i)$ строится путём тривиального объединения множеств размещения π_i и $V(\pi_j)$, $X_j \in D_i$, $\pi_j \bowtie \pi_i$, найденными на предыдущих этапах.

В свою очередь, каждая подзадача $(G(\pi_i), V, F)$, когда X_i — узел-сепаратор, в силу однозначности отображения π также решается оптимально, поскольку соответствующее оптимальное решение $V(\pi_i)$ определяется по формуле (4), а оптимальное решение $V(\pi_j)$ любой подзадачи $(G(\pi_j), V, F)$, $\pi_j \in \Pi(X_j)$, для каждого $X_j \in D_i$ известно и построено на предыдущих этапах.

Таким образом, рекуррентно решая подзадачи $(G(\pi_i), V, F)$ на этапах $1 < i < N$ процесса ДП, алгоритм на этапе N, где X_N — корень дерева T, для каждого $\pi_N \in \Pi(X_N)$ находит оптимальное решение соответствующей подзадачи $(G(\pi_N), V, F)$. Так как задача (G_N, V, F) соответствует исходной задаче (G, V, F), алгоритм на конечном этапе $N + 1$ процесса ДП по вычисленному оптимальному решению $V(\pi_N)$ каждой подзадачи $(G(\pi_N), V, F)$, $\pi_N \in \Pi(X_N)$, находит согласно формуле (5) оптимальное решение π^*_G исходной задачи (G, V, F).

Докажем оценки пространственной и вычислительной сложности. Для входных данных алгоритма требуется порядка $O(|J|^2|V|^2)$ памяти, поскольку $O(|J|^2)$ памяти необходимо для хранения графа G, $O(|J||V|)$ памяти требуется для функции $p(\cdot)$ и $O(|J|^2|V|^2)$ памяти необходимо для функции $c(\cdot)$. Для хранения данных на этапе i требуется $O(|J||V|^{k+1})$ памяти, так как количество множеств $V(\cdot)$, $\overline{V}(\cdot)$ и значений функций $R(\cdot)$, $\overline{R}(\cdot)$ и $f_i(\cdot)$ не превосходит $O(|V|^{k+1})$, при этом для хранения каждого такого множества требуется $O(|J|)$ памяти, а для хранения каждого значения соответствующих функций необходимо $O(1)$ памяти. Так как в общем случае для построения оптимального решения требуется на каждом этапе i хранить решения, полученные на всех предыдущих этапах, а количество этапов алгоритма не превосходит $O(|J|)$, пространственная сложность алгоритма равна $O(|V|^{k+1}|J|^2)$ памяти. Поскольку множества $V(\cdot)$, $\overline{V}(\cdot)$ и значения функций $R(\cdot)$, $\overline{R}(\cdot)$ и $f_i(\cdot)$ вычисляются однократно, вычислительная сложность алгоритма не превосходит его пространственной сложности и равна $O(|V|^{k+1}|J|^2)$ операций. Теорема 1 доказана.

3. Экспериментальное исследование алгоритма kTrWPA

Алгоритм kTrWPA, находящий точное решение задачи (G, V, F), является квазиполиномиальным, поскольку при некотором фиксированном значении входного параметра k сложность предложенного алгоритма полиномиальна и ограничена полиномом порядка k. Отсюда следует, что практическое применение предложенного алгоритма, впрочем, как и любых алгоритмов, использующих идею ДП по дереву декомпозиции, ограничено высокой потребностью в памяти и большими затратами вычислительных операций в тех случаях, когда величина параметра k сравнительно велика.

Вследствие экспоненциального возрастания сложности предложенного алгоритма очевидно, что применение его для решения дискретной задачи Вебера перспективно при сравнительно малых значениях величины параметра k, при этом чем меньше величина k и чем больше количество вершин размещаемого графа, тем предлагаемый алгоритм более эффективен по сравнению с точными алгоритмами, являющимися вариациями полного перебора с отсевом заведомо бесперспективных подмножеств допустимых решений, например, алгоритма ветвей и границ, реализованного в пакете IBM ILOG CPLEX, так как вычислительная сложность алгоритма kTrWPA ограничивается полиномом $O(|V|^{k+1} \cdot (|J| - k))$, в то время как вычислительная сложность упомянутых точных алгоритмов в общем случае экспоненциальна. Очевидно, что чем ближе величина параметра k к числу вершин размещаемого графа, тем предлагаемый алгоритм менее эффективен по сравнению с алгоритмами, являющимися вариациями полного перебора. Например, при решении дискретной задачи Вебера для k-дерева $G = (J, E)$ с количеством вершин $|J| = k + 1$ время работы предложенного алгоритма kTrWPA сравнимо с временем работы алгоритма, использующего идею тривиального последовательного перебора всех допустимых решений.

Алгоритм kTrWPA был реализован на ЭВМ в среде MATLAB. Проведён вычислительный эксперимент по анализу его эффективности. Для оценки эффективности алгоритма использовался программный пакет IBM ILOG CPLEX Optimization Studio 12.2 (решение модели целочисленного линейного программирования (ЦЛП) дискретной задачи Вебера алгоритмом ветвей и границ с ограничением по времени работы).

Для проведения эксперимента был случайным образом с равномерным распределением сгенерирован класс тестовых задач, состоящий из серий, каждая из которых включала 30 задач одинаковой размерности. Вычисления проводились на ПК с процессором Intel Pentium 1.86 GHz.

Результаты вычислительного эксперимента приведены в табл. 1, где $\bar{t}_{\mathrm{alg}}$ — среднее время работы предложенного алгоритма, с; $\bar{t}_{\mathrm{ILP}}$ — среднее время работы модели ЦЛП, реализованной в среде IBM ILOG CPLEX Optimization Studio 12.2, с.

Т а б л и ц а 1

Результаты вычислительного эксперимента

k	Размерности задачи					
		$\lvert J\rvert = 5$ $\lvert V\rvert = 5$	$\lvert J\rvert = 10$ $\lvert V\rvert = 10$	$\lvert J\rvert = 20$ $\lvert V\rvert = 20$	$\lvert J\rvert = 40$ $\lvert V\rvert = 40$	$\lvert J\rvert = 100$ $\lvert V\rvert = 100$
1	$\bar{t}_{\mathrm{alg}}$	0,0018	0,0044	0,0258	0,2478	3,8826
	$\bar{t}_{\mathrm{ILP}}$	0,1527	0,6408	14,2633	—	—
2	$\bar{t}_{\mathrm{alg}}$	0,0223	0,028	0,6792	9,2275	398,22
	$\bar{t}_{\mathrm{ILP}}$	0,1764	0,606	14,9004	—	—
3	$\bar{t}_{\mathrm{alg}}$	0,1094	0,3628	12,1001	387,165	—
	$\bar{t}_{\mathrm{ILP}}$	0,1708	0,6408	19,9869	—	—
4	$\bar{t}_{\mathrm{alg}}$	0,1831	3,7897	42,7618	—	—
	$\bar{t}_{\mathrm{ILP}}$	0,1229	1,0105	17,8403	—	—
5	$\bar{t}_{\mathrm{alg}}$	0,2966	42,3036	—	—	—
	$\bar{t}_{\mathrm{ILP}}$	0,1656	1,0078	18,2815	—	—

Для дискретной задачи Вебера для k-дерева $G = (J, E)$ и конечного множества позиций размещения V размерности $|J| = 40, |V| = 40$ и выше не удалось получить решения с помощью пакета IBM ILOG CPLEX 12.2 за приемлемое время для любых значений параметра k при том, что среднее время решения задачи Вебера такой размерности для 1(2)-дерева с помощью предлагаемого алгоритма kTrWPA не превысило одной (десяти) секунд. Стоит отметить, что среднее время решения задачи Вебера размерности $|J| = 100, |V| = 100$ для 1-дерева с помощью алгоритма kTrWPA не превысило четырёх секунд.

Применение алгоритма kTrWPA для решения дискретной задачи Вебера для k-дерева при $k \geqslant 4$ нецелесообразно, так как среднее время работы алгоритма превосходит среднее время работы модели ЦЛП, реализованной в среде IBM ILOG CPLEX 12.2.

Заключение

Рассмотрена дискретная задача Вебера, когда размещаемый граф имеет вид k-дерева. Предложен детерминированный последовательный квазиполиномиальный алгоритм kTrWPA, основанный на методе динамического программирования по дереву декомпозиции. Определены оценки вычислительной и пространственной сложности предложенного алгоритма. Доказано, что предложенный алгоритм kTrWPA находит точное решение рассматриваемой задачи (G, V, F).

Проведено теоретическое и экспериментальное исследование эффективности предложенного алгоритма. Вычислительный эксперимент показал, что среднее время решения дискретной задачи Вебера размерности $|J| = 100, |V| = 100$ для 1-дерева и конечного множества позиций размещения с помощью алгоритма kTrWPA не превысило четырёх секунд, в то время как с помощью пакета IBM ILOG CPLEX 12.2 не удалось получить решения для задачи размерности $|J| = 40,\ |V| = 40$ и выше за приемлемое время при любых значениях параметра k.

Экспериментально установлено, что применение алгоритма kTrWPA целесообразно для решения дискретной задачи Вебера для k-дерева при сравнительно малых значениях величины k, так как при $k \geqslant 4$ время работы предлагаемого алгоритма существенно превосходит время работы модели ЦЛП, реализованной в среде IBM ILOG CPLEX 12.2. Также теоретически и экспериментально установлено, что чем меньше величина k и чем больше количество вершин размещаемого графа, тем предлагаемый алгоритм kTrWPA более эффективен по сравнению с точными алгоритмами, являющимися вариациями полного перебора с отсевом заведомо бесперспективных подмножеств допустимых решений, например, алгоритма ветвей и границ, реализованного в среде IBM ILOG CPLEX.

ЛИТЕРАТУРА

1. **Быкова В. В.** Вычислительные аспекты древовидной ширины графа // Прикл. дискрет. математика. — 2011. — № 3. — С. 65–79.
2. **Быкова В. В.** FTP-алгоритмы на графах ограниченной древовидной ширины // Прикл. дискрет. математика. — 2012. — № 2. — С. 65–78.
3. **Забудский Г. Г., Лагздин А. Ю.** Полиномиальные алгоритмы решения минимаксной квадратичной задачи о назначениях на сетях // Дискрет. анализ и исслед. операций. — 2011. — Т. 18, № 4. — С. 49–65.
4. **Забудский Г. Г., Лагздин А. Ю.** Динамическое программирование для решения квадратичной задачи о назначениях на дереве // Автоматика и телемеханика. — 2012. — № 2. — С. 141–155.
5. **Забудский Г. Г., Филимонов Д. В.** О минимакской и минисуммной задачах размещения на сетях // Тр. XII Байкальской междунар. конф. «Методы оптимизации и их приложения» (Иркутск, 24 июня–1 июля 2001 г.). — Иркутск: Изд-во ИГУ, 2001. — С. 150–155.
6. **Панюков А. В.** Модели и методы решения задач построения и идентификации геометрического размещения: дисс. ... д-ра физ.-мат. наук: 05.13.16. — Челябинск, 1999. — 260 с.
7. **Панюков А. В., Пельцвергер Б. Ф.** Оптимальное размещение дерева в конечном множестве // Журн. вычисл. математики и мат. физики. — 1988. — Т. 28. — С. 618–620.
8. **Панюков А. В., Пельцвергер Б. Ф., Шафир А. Ю.** Оптимальное размещение точек ветвления транспортной сети на цифровой модели местности // Автоматика и телемеханика. — 1990. — № 9. — С. 153–162.
9. **Сергеев С. И.** Квадратичная задача о назначениях. I // Автоматика и телемеханика. — 1999. — № 8. — С. 127–147.
10. **Трубин В. А.** Эффективный алгоритм для задачи Вебера с прямоугольной метрикой // Кибернетика. — 1978. — № 6. — С. 67–70.
11. **Филимонов Д. В.** Решение дискретной минимаксной задачи размещения на древовидной сети // Мат. ежегод. научн. семинара аспирантов. — Омск: Изд-во Омск. гос. ун-та, 2003. — С. 58–61.
12. **Шангин Р. Э.** Исследование эффективности приближенных алгоритмов решения одного частного случая задачи Вебера // Экономика, статистика и информатика. Вестн. УМО. — 2012. — № 1. — С. 163–169.
13. **Шангин Р. Э.** О некоторых свойствах n-последовательносвязной цепи // Вестн. ЮУрГУ. Сер. Вычисл. математика и информатика. — 2013. — Т. 2, № 1. — С. 106–113.
14. **Arnborg S.** Efficient algorithms for combinatorial problems with bounded decomposability — a survey // BIT Numerical Math. — 1985. — Vol. 25, N 1. — P. 1–23.
15. **Arnborg S., Proskurowski A.** Linear time algorithms for NP-hard problems restricted to partial k-trees // Discrete Appl. Math. — 1989. — Vol. 23. — P. 11–24.

16. **Bodlaender H. L.** Treewidth: algorithmic techniques and results // Lect. Notes Comput. Sci. — 1997. — Vol. 1295. — P. 19–36.
17. **Ibarra L.** The clique-separator graph for chordal graphs // Discrete Appl. Math. — 2009. — Vol. 157, N 8. — P. 1737–1749.
18. **McKee T. A.** On the chordality of a graph // J. Graph Theory. — 1993. — Vol. 17. — P. 221–232.
19. **Panyukov A. V., Pelzwerger B. V.** Polynomial algorithms to finite Weber problem for a tree network // J. Comput. Appl. Math. — 1991. — Vol. 35. — P. 291–296.
20. **Robertson N., Seymour P. D.** Graph minors. II. Algorithmic aspects of treewidth // J. Algorithms. — 1986. — Vol. 7. — P. 309–322.
21. **Rose D. J.** Triangulated graphs and the elimination process // J. Math. Anal. Appl. — 1970. — Vol. 32. — P. 597–609.
22. **Rose D. J.** On simple characterizations of k-trees // Discrete Math. — 1973. — Vol. 7. — P. 317–322.
23. **Zabudsky G. G., Filimonov D. V.** An algorithm for minimax location problem on tree with maximal distances // Proc. II Int. Workshop Discrete Optimization Methods in Production and Logistics (DOM 2004). — Omsk; Irkutsk: ISU Press, 2004. — P. 81–85.

Панюков Анатолий Васильевич,
e-mail: a_panyukov@mail.ru
Шангин Роман Эдуардович,
e-mail: shanginre@gmail.com

Статья поступила
3 сентября 2013 г.
Переработанный вариант —
31 января 2014 г.