ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Январь—июнь 2007. Серия 2. Том 14, № 1. 43–58

УДК 519.854.2

МЕТОД ДЕКОМПОЗИЦИИ ДЛЯ ЗАДАЧИ О p-МЕДИАНЕ НА НЕСВЯЗНОМ ГРАФЕ*)

И. Л. Васильев

Рассматривается задача оптимального замещения, которая формулируется как задача о p-медиане. Практическое приложение данной задачи возникает в автомобильной промышленности, где существуют примеры, которые определяются на графах, состоящих из нескольких десятков тысяч вершин и нескольких миллионов дуг. Отличительной особенностью любой такой задачи является то, что граф состоит из нескольких компонент связности. Эта структура графа используется для декомпозиции исходной задачи в серию подзадач о p-медиане меньшей размерности. Предложенная декомпозиция естественным образом позволяет использовать параллельные вычисления при программной реализации метода. Эффективность предложенного подхода иллюстрируется численными экспериментами на практических примерах.

Введение

Задача оптимального замещения (ЗОЗ) в автомобильной промышленности была предложена в [7]. Обычно каждый легковой автомобиль может комплектоваться различным набором опций. Такой набор будем называть конфигурацией. Покупатель по своему желанию может дополнительно к базовой комплектации выбрать установку подушек безопасности, системы ABS, кондиционера и т. д. В соответствии с набором опций должен быть установлен определённый набор электропроводки, гарантирующий подключение этих дополнительных устройств. В современном автомобиле число опций может достигать 30–50, а число их комбинаций может превышать несколько тысяч. В силу технологических особенностей каждый набор электропроводки изготавливается как единое целое, и только ограниченное количество этих наборов может быть доступно на сборочном конвейере. Если необходимая конфигурация

*) Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 05–01–00110) и НАТО (проект 981258).

электропроводки недоступна, то она может быть замещена совместимой. Под совместимой подразумевается такая конфигурация, которая содержит по крайней мере электропроводку для подключения всех необходимых опций. При такой замене часть электропроводки не используется (дарится покупателю), и в связи с этим возникают избыточные производственные издержки, которые хотелось бы минимизировать.

Эта задача может быть сформулирована в виде задачи о p-медиане. Рассмотрим ориентированный граф, в котором каждой вершине сопоставлена некоторая конфигурация электропроводки; две вершины u и v соединены дугой uv, если конфигурация, соответствующая v-й вершине, может быть замещена конфигурацией, которая соответствует u-й вершине. Вес дуги равен избыточным издержкам, возникающим при замещении. Таким образом, если может быть использовано только ограниченное количество конфигураций (скажем, p), то задача заключается в нахождении p медианных вершин, минимизирующих сумму весов дуг до остальных вершин.

Задача о p-медиане — известная NP-трудная задача, сформулированная в [10]. Обширный литературный обзор по этой задаче можно найти в [12], а также в [4, 13]. Отдельно хочется отметить последние работы, касающиеся решения задачи о p-медиане большой размерности: VNS [9], GRASP эвристики [13], метод ветвей, отсечений и оценок [4].

В статье рассматривается случай несвязного графа. Данное свойство используется при разработке метода декомпозиции. Показано, что такая задача может быть сведена к решению серии подзадач меньшей размерности. Предложенный подход позволил работать с примерами на графах более чем с 80 000 вершинами и 6 000 000 дугами. Для решения подзадач использовались жадная эвристика и эвристика Лагранжа, адаптированные для данного случая. В результате численного эксперимента были получены решения с погрешностью до 1% за приемлемое время на персональном компьютере.

При декомпозиции получается серия подзадач, которые можно решать на параллельных процессорах. В статье представлена параллельная реализация предложенного подхода, которая позволяет значительно сократить время счёта.

Статья состоит из 5 разделов. В разделе 1 описана постановка задачи. Декомпозиция и её обоснование изложены в разделе 2. В разделе 3 дано описание общей вычислительной схемы, описание отдельных компонент и схемы распараллеливания. Результаты численных экспериментов приводятся в разделе 4. Наконец, в разделе 5 сделаны общие заключения.

1. Постановка задачи

Пусть при сборке некоторого промышленного продукта, например, при комплектации легковых автомобилей электропроводкой, имеется множество опций $S = \{1, \dots, s\}$. Некоторое подмножество опций $S_u \subseteq S$ будем называть u-й *конфигурацией*. Пусть дано множество конфигураций $V = \{1, \dots, n\}$, соответствующие им наборы опций $S_1, \dots, S_n$, а также соответствующие цены $c_1, \dots, c_n$ и рыночный спрос $d_1, \dots, d_n$. Некоторая конфигурация u может заменить конфигурацию v, если она содержит все опции v-й конфигурации, т. е. $S_v \subset S_u$. Возникающие при замене избыточные затраты выражаются формулой

$$w_{uv} = (c_u - c_v)d_v.$$

Без потери общности можно предположить, что все конфигурации различны, т. е. $S_u \neq S_v$ при $u \neq v$. На практике существуют также дополнительные условия по замене конфигураций, но они в этой статье не рассматриваются, так как не изменяют свойств задачи. Технические детали практических приложений этой задачи можно найти в [3, 7]. Задача оптимального замещения заключается в выборе таких конфигураций, которые минимизируют избыточные затраты, возникающие при замещении остальных конфигураций. Эта задача может быть сформулирована как задача о p-медиане.

Рассмотрим орграф $G(V, A)$, в котором каждая вершина соответствует определённой конфигурации. Дуга соединяет вершины u и v, если конфигурация v может быть замещена на u, т. е.

$$A = \{uv |\ S_v \subset S_u\}.$$

Веса дуг равны соответствующим избыточным издержкам w_{uv}. Из определения графа следует, что граф $G(V, A)$ является неполным, ацикличным и совпадает со своим транзитивным замыканием.

Введём бинарные переменные y_u (x_{uv}), которые будут соответствовать вершинам $u \in V$ (дугам $uv \in A$). Переменная y_u (x_{uv}) равна 1, если соответствующая вершина (дуга) принадлежит допустимой точке. Обозначим через $\delta^-(u) = \{vu |\ vu \in A\}$ множество дуг, входящих в вершину u, $\delta^+(u) = \{uv |\ uv \in A\}$ — множество дуг, исходящих из u. В этих обозначениях задача о p-медиане может быть сформулирована как задача целочисленного программирования: найти

$$\min_{x,y} \sum_{uv \in A} w_{uv} x_{uv} \tag{1}$$

при ограничениях

$$\sum_{u\in\delta^-(v)} x_{uv} + y_v = 1 \text{ при любой вершине } v \in V, \tag{2}$$

$$x_{uv} \leqslant y_u \text{ при любой дуге } uv \in A,\ u \in V, \tag{3}$$

$$\sum_{v\in V} y_v = p, \tag{4}$$

$$y_u \in \{0,\ 1\} \text{ при любой вершине } v \in V, \tag{5}$$

$$x_{uv} \in \{0,1\} \text{ при любой дуге } uv \in A. \tag{6}$$

Ограничения (2) гарантируют, что каждая вершина v является либо медианой, либо имеет только одну входящую дугу из медианной вершины. Неравенства (3) исключают существование выходящих дуг из немедианных вершин. Число медиан определяется уравнением (4).

Как видно из определения множества дуг из A, граф $G(V,A)$ неполный. Следовательно, нет гарантии, что задача имеет решение при любом значении p. Поэтому возникает вопрос, при каком минимальном значении $p_{\min}$ допустимое множество задачи не пусто. Это значение может быть найдено решением следующей задачи о покрытии: найти

$$p_{\min} = \min_{(x,y)} \left\{ \sum_{u\in V} y_u \middle|\ \text{ при ограничениях (2), (3), (5), (6)} \right\}. \tag{7}$$

Можно доказать, что решение задачи (7) состоит из вершин, в которые не входят дуги. Такие вершины будем называть *корнями.*

Утверждение 1. *Пусть* $V^0 \subset V$ *— множество корней, т. е.* $\{(u,v) \in A|\ v \in V^0\} = \varnothing$, *и пусть* $p_{\min} = |V^0|$. *Тогда задача о* p*-медиане имеет решение тогда и только тогда, когда* $p \geqslant p_{\min}$.

Доказательство непосредственно следует из того, что в графе $G(V,A)$ выполняется свойство транзитивности: если $ut \in A$ и $tv \in A$, то $uv \in A$.

Если $p = p_{\min}$, то оптимальное решение задачи находится по следующему правилу:

$$y_u = \begin{cases} 1, \text{ если } u \in V^0, \\ 0, \text{ если } u \in V \backslash V^0, \end{cases} \quad x_{uv} = \begin{cases} 1, \text{ если } v \in V \setminus V^0,\ u = \operatorname*{argmin}_{t\in V^0} w_{tv}, \\ 0 \text{ в противном случае.} \end{cases}$$

Для решения задачи (1)–(6) в [7] предложен подход, основанный на релаксации Лагранжа и специальных процедурах уменьшения размерности. Далее применяются коммерческие решатели для поиска оптимального решения. Этот подход показал свою эффективность для многих задач,

но известны также примеры, для которых оптимальное решение найдено не было.

2. Декомпозиция задачи

Рассмотрим задачу (1)–(6), в которой граф $G(V, A)$ состоит из m компонент связности $G_i(V_i, A_i)$, $i = 1, \ldots, m$. Под компонентой связности ориентированного графа будем понимать слабо связный подграф максимально возможного размера. Предполагается, что $p > p_{\min}$.

Для i-й компоненты связности графа определим множество корней $V_i^0 = V^0 \cap V_i$. Число медиан в i-й компоненте связности не может быть меньше $p_{\min}^i = |V_i^0|$ и не может превосходить

$$p_{\max}^i = \min\{|V_i|,\ p - (p_{\min} - p_{\min}^i)\},$$

так как $(p_{\min} - p_{\min}^i)$ – минимальное число медиан, содержащихся в остальных компонентах связности.

Предположим, что можно найти решение подзадачи о p-медиане в каждой компоненте связности графа $G_i(V_i, A_i)$ при любом

$$p_i \in P_i = \left[p_{\min}^i, p_{\max}^i\right], \quad i = 1, \ldots, m.$$

Соответствующие оптимальные решения обозначим через $(\bar{x}^{ip_i}, \bar{y}^{ip_i})$, а оптимальные значения задачи — через c_{ip_i}. Возникает вопрос: можно ли найти решение исходной задачи, основываясь на этих данных? Идея поиска решения заключается в том, что решение исходной задачи должно состоять из таких решений подзадач, что их суммарное значение будет минимально при условии, что общее число медиан равно p. Эти рассуждения могут быть сформулированы в виде задачи целочисленного программирования следующим образом.

Введём двоичные переменные z_{ip_i}, где $i = 1, \ldots, m$ и $p_i \in P_i$ такое, что $z_{ip_i} = 1$, если из i-й компоненты связности выбирается решение подзадачи с p_i медианами; иначе $z_{ip_i} = 0$. В этом случае задача примет вид: найти

$$\min_z \left\{ \sum_{i=1}^{m} \sum_{p_i \in P_i} c_{ip_i} z_{ip_i} \right\} \tag{8}$$

при ограничениях

$$\sum_{i=1}^{m} \sum_{p_i \in P_i} p_i z_{ip_i} = p, \tag{9}$$

$$\sum_{p_i \in P_i} z_{ip_i} = 1 \text{ при } i = 1, \ldots, m \tag{10}$$

$$z_{ip_i} \in \{0, 1\} \text{ при } i = 1, \ldots, m,\ p_i \in P_i. \tag{11}$$

Целевая функция (8) гарантирует выбор решений подзадач с минимальной суммарной стоимостью. Условия (9) определяют общее число медиан. Равенства (10) обеспечивают выбор по одному решению из каждой компоненты связности. Эта задача является известной задачей многовариантного рюкзака [11].

Теорема 1. *Пусть* $(\bar{x}^{ip_i}, \bar{y}^{ip_i})$ *— допустимые точки подзадач о p-медиане в компонентах связности* $G_i(V_i, A_i)$ *и* $p_i \in P_i$, $i = 1, \ldots, m$. *Пусть* $\bar{z} = \{\bar{z}_{ip_i}\}$ *— допустимая точка задачи* (8)–(11) *и*

$$(\bar{x}, \bar{y}) = \sum_{\substack{i = 1, \ldots, m \\ p_i : \bar{z}_{ip_i} = 1}} (\bar{x}^{ip_i}, \bar{y}^{ip_i}).$$

Тогда $(\bar{x}, \bar{y})$ *– допустимая точка исходной задачи о p-медиане* (1)–(6); *она является оптимальной, если* $(\bar{x}^{ip_i}, \bar{y}^{ip_i})$ *и* $\bar{z}$ *являются оптимальными решениями соответствующих подзадач.*

Доказательство. Допустимость точки $(\bar{x}, \bar{y})$ непосредственно следует из определения рассматриваемых задач.

Для доказательства оптимальности предположим, что существует допустимая точка $(\hat{x}, \hat{y})$, лучшая чем $(\bar{x}, \bar{y})$, т. е.

$$\sum_{uv \in A} w_{uv} \hat{x}_{uv} < \sum_{uv \in A} w_{uv} \bar{x}_{uv}.$$

Для каждого $i = 1, \ldots, m$ рассмотрим точку $(\hat{x}^i, \hat{y}^i)$ такую, что

$$(\hat{x}^i, \hat{y}^i) = \begin{cases} \hat{y}^i_u = \hat{y}_u, & \text{если } u \in V_i \text{ и } \hat{y}^i_u = 0 \text{ при } u \notin V_i, \\ \hat{x}^i_{uv} = \hat{x}_{uv}, & \text{если } uv \in A_i \text{ и } \hat{x}^i_{uv} = 0 \text{ при } uv \in A_i. \end{cases}$$

Пусть $\hat{p}_i = \sum\limits_{i \in V_i} \hat{y}_i$. Очевидно, что $(\hat{x}^i, \hat{y}^i)$ – допустимая точка подзадачи в i-й компоненте связности при $p_i = \hat{p}_i$. Также рассмотрим

$$\hat{z} = \begin{cases} \hat{z}_{ip_i} = 1, & \text{если } p_i = \hat{p}_i \text{ при любом } i = 1, \ldots, m, \\ \hat{z}_{ip_i} = 0 & \text{в противном случае.} \end{cases}$$

Легко видеть, что $\hat{z}$ является допустимой точкой в задаче (8)–(11). Значения $c_{i\hat{p}_i}$ являются оптимальными в соответствующих подзадачах. По-

этому $c_{i\hat{p}_i} \leqslant \sum\limits_{uv \in A_i} w_{uv} \hat{x}_{uv}$. Следовательно,

$$\sum_{i=1}^{m} \sum_{p_i \in P_i} c_{ip_i} \hat{z}_{ip_i} \leqslant \sum_{i=1}^{m} \sum_{uv \in A_i} w_{uv} \hat{x}_{uv}^i = \sum_{uv \in A} w_{uv} \hat{x}_{uv}$$
$$< \sum_{uv \in A} w_{uv} \bar{x}_{uv} = \sum_{i=1}^{m} \sum_{uv \in A_i} w_{uv} \bar{x}_{uv}^i = \sum_{i=1}^{m} \sum_{p_i \in P_i} c_{ip_i} \bar{z}_{ip_i},$$

что противоречит оптимальности $\bar{z}$ в задаче (8)–(11). Теорема 1 доказана.

Теорема 1 показывает, что задача о p-медиане может быть сведена к решению $m(p - p_{\min})$ подзадач меньшей размерности и решению одной задачи (8)–(11). С вычислительной точки зрения могут быть полезны следующими замечания.

Замечание 1. Предположим, что нельзя получить оптимальные решения подзадач о p-медиане на компонентах связности, но имеются нижние и верхние оценки. Подставляя эти оценки в коэффициенты целевой функции задачи (8)–(11), получим нижние и верхние оценки исходной задачи о p-медиане.

Замечание 2. Если необходимо решить задачу о p-медиане для всех значений p из интервала $[p_{\min}, p_{\max}]$, то можно положить $p^i_{\max} = p_{\max} - p_{\min} + p^i_{\min}$, вследствие чего отпадает необходимость решать повторяющиеся подзадачи в компонентах связности при разных значениях p. Таким образом, для того чтобы решить $p_{\max} - p_{\min}$ задач о p-медиане, необходимо решить $m(p_{\max} - p_{\min})$ подзадач и $p_{\max} - p_{\min}$ задач (8)–(11).

3. Стратегия решения

В этом разделе описывается общая схема решения, которая базируется на результатах, представленных в предыдущем разделе. В схеме можно выделить три основных этапа:

1. Нахождение разложения графа $G(V, A)$ на компоненты связности $G_i(V_i, A_i)$, $i = 1, \ldots, m$.

2. Для каждой компоненты связности решение подзадач о p-медиане при всех $p_i \in P_i$. Анализ практических примеров показал, что несмотря на снижение размерности подзадачи остаются достаточно большими, и применение точных методов решения не представляется возможным. Поэтому для рассмотренной задачи адаптированы жадная эвристика и эвристика Лагранжа. Эти алгоритмы описываются в пунктах 3.1 и 3.2.

3. Решения подзадач второго этапа используются в формулировке задачи (8)–(11). Эта задача решается методом ветвей и границ, описываемом в пункте 3.3.

Некоторые подзадачи второго этапа не взаимосвязаны и могут быть решены на различных процессорах одновременно. Распараллеливание представлено в пункте 3.4.

3.1. Жадная эвристика

Как уже говорилось во введении, возникающие при решении практических задач графы могут иметь более 80000 вершин и 6000000 дуг. Следовательно, эти графы очень разрежены. После декомпозиции в некоторых компонентах связности может оказаться более 10000 вершин, при этом плотность графа не превосходит 5%. Разреженность графов используется при реализации жадной эвристики.

В [8] была представлена жадная эвристика для решения задачи о p-медиане. Были предложены различные способы ускорения работы алгоритма, в которых используются рекурсивные соотношения для обновления списка кандидатов [15] или матрицы пар кандидатов [14] в медианы на следующей итерации. Как показывает практика, эти операции значительно ускоряют жадный алгоритм при больших значениях p (когда p близко или больше $|\mathrm{V}|/2$) на полных графах. Однако они теряют своё преимущество при малых p и на сильно разреженных графах. Поэтому при реализации используется прямая схема алгоритма и специальный вид представления графов.

Граф хранится в виде списка смежности входящих дуг для каждой вершины, причём этот список упорядочен по возрастанию весов дуг, т. е. для любого v из V хранится такой список $u_1, \ldots, u_{k_v}$ (k – число дуг, входящих в вершину v), что $w_{u_1 v} \leqslant w_{u_2 v} \leqslant \cdots \leqslant w_{u_k v}$. Также хранится матрица смежности графа. Так как для каждого элемента этой матрицы используется один бит, то она не занимает большого объема памяти.

Предложенная структура хранения графа позволяет эффективно реализовать жадный алгоритм, представленный ниже, где T_p обозначает текущее множество медиан, а Z_p — значение целевой функции.

`Шаг 0. Инициализировать` $p = p_{\min}$, $T_p = V_0$,
$$\bar{T}_p = V \setminus T_p, \quad Z_p = \sum_{u \in \bar{T}_p} \min_{u \in T_p} w_{uv}.$$

`Шаг 1. Вычислить` $\forall u \in \bar{T}_p$
$$\Delta_u = \sum_{uv \in \delta^+(u):\ v \in \bar{T}_p} \min\left\{0,\ w_{uv} - \min_{t \in T_p} w_{tv}\right\} - \min_{v \in T_p} w_{vu}.$$

```
Шаг 2. Найти $\bar{u} \in \bar{T}_p$:
       $\Delta_{\bar{u}} = \min\limits_{u \in \bar{T}_p} \Delta_u$.
Шаг 3. Установить $T_{p+1} = T_p \cup \{\bar{u}\}$, $\bar{T}_{p+1} = \bar{T}_p \setminus \{\bar{u}\}$,
       $Z_{p+1} = Z_p + \Delta_{\bar{u}}$, $p = p + 1$.
Шаг 4. Если $p = p_{\max}$, то перейти на Шаг 5,
       иначе перейти на Шаг 1.
Шаг 5. $T_p$ - решение задачи.
```

Заметим, что только при $p = p_{\min} + 1$ жадная эвристика всегда находит оптимальное решение. В остальных случаях этого утверждать нельзя.

3.2. Эвристика Лагранжа

Полученное жадной эвристикой решение можно попытаться улучшить эвристикой Лагранжа. Использовалась эвристика, описанная в [4].

Пусть $p > p_{\min} + 1$. Функция Лагранжа $\theta(\lambda)$ получается ослаблением ограничений (2):

$$\theta(\lambda) = \min_{(x,y)} \left\{ \sum_{v \in V} \left[\sum_{uv \in \delta^-(v)} (w_{uv} - \lambda_v)\, x_{uv} - \lambda_v y_v \right] + \sum_{u \in V} \lambda_u \ \middle| \ \text{при ограничениях (3)–(6)} \right\}.$$

Если множители Лагранжа $\bar{\lambda}$ заданы, то значение $\theta(\bar{\lambda})$ формируется следующим образом [6]:

a) для каждого $u \in V$ подсчитывается приведённая оценка Лагранжа:

$$\rho(u) = \sum_{uv \in \delta^+(u)} \min\{0,\ (w_{uv} - \bar{\lambda}_v)\} - \bar{\lambda}_u;$$

b) вершины упорядочиваются по возрастанию $\rho(u)$ и полагается, что $T = \{u_1, \ldots, u_p\}$;

c) решение Лагранжа $(x(\bar{\lambda}), y(\bar{\lambda}))$ строится по правилу:

$$y_u(\bar{\lambda}) = \begin{cases} 1, \text{если } u \in T, \\ 0 \text{ в противном случае,} \end{cases}$$

$$\bar{x}_{uv} = \begin{cases} 1, \text{если } \bar{y}_u = 1 \text{ и } w_{uv} - \bar{\lambda}_v < 0, \\ 0 \text{ в противном случае.} \end{cases}$$

Значение функции Лагранжа $\theta(\bar{\lambda})$ определяется формулой

$$\theta(\bar{\lambda}) = \sum_{u \in T} \rho(u) + \sum_{u \in V} \bar{\lambda}_u.$$

Используя элементы множества T в качестве медиан, можно построить допустимую точку и найти верхнюю оценку. Для максимизации функции Лагранжа используется стандартный субградиентный метод из [6], с выбором шага по субградиенту на k-й итерации

$$h_k = \frac{1{,}05\, UB - \theta(\lambda^k)}{||d^k||^2} \varphi_k,$$

где UB — лучшая верхняя оценка, d^k — субградиент в точке λ^k, φ_k — параметр, который изначально выбирается равным 2 и на каждой итерации делится на величину γ. Эвристика Лагранжа позволяет находить верхнюю и нижнюю оценки. Поэтому есть возможность оценивать качество полученных решений. Выбирая различные значения величины γ, можно или уменьшить время счёта, или увеличить качество получаемого решения за счёт лучшей сходимости алгоритма. На каждой итерации субградиентного метода также применялись алгоритмы снижения размерности из [5, 7].

3.3. Метод ветвей и границ

Задача (8)–(11) не является «узким местом» общей вычислительной схемы. Существуют различные методы её решения, обзор которых можно найти в [11]. Более того, так как все коэффициенты задачи целочисленны и $p < n$, то такую задачу можно решить точно методом динамического программирования с полиномиальной трудоёмкостью. В рассматриваемом случае для её решения был реализован метод ветвей и границ, основанный на нижней оценке, которая была получена ослаблением ограничения (9). Верхние оценки подсчитывались из решения Лагранжа при помощи жадного алгоритма.

Тесты на практических примерах показали, что задача (8)–(11) имеет малую размерность. Задача решается за несколько миллисекунд, что существенно меньше общего времени счёта. Поэтому внимание на решении этой задачи не заостряется.

3.4. Параллелизация метода

Из пункта 3.1 следует, что для получения решения при некотором $p = k$ необходимо иметь решение при $p = k-1$. Таким образом, в предложенном алгоритме решения подзадач о p-медиане для каждой компоненты связности графа взаимосвязаны. Но для разных компонент связности они независимы. Поэтому за единицу распараллеливания было выбрано решение серии подзадач для компоненты связности, т. е. для некоторой i-компоненты связности серия подзадач при всех $p_i \in P_i$ рассматривается как одна параллельная задача. В результате имеется m параллельных задач.

В параллельной реализации использовалась стандартная схема Master-Slave, в которой Master производит декомпозицию исходной задачи и составляет очередь параллельных задач. В дальнейшем он отсылает из очереди задачи на свободные Slave процессы, которые решают эти задачи и отсылают решения обратно. Это продолжается до тех пор, пока не будут решены все задачи. После этого Master решает задачу (8)–(11) и выдаёт окончательный результат.

В рассмотренных примерах число компонент связности графа составляло лишь несколько десятков, т. е. число параллельных задач было относительно невелико, причём размерности этих задач могли значительно различаться. Для задач такого рода актуальной является проблема сбалансированной загрузки параллельных процессоров, которая гарантирует минимальное абсолютное время счёта. Данная задача известна как задача планирования параллельных машин.

Пусть даны k параллельных одинаковых процессоров, на которых необходимо решить m задач, и пусть t_j – время решения j-й задачи. Пусть τ – это абсолютное время решения, $x_{ij} = 1$, если i-й процессор выполняет j-ю задачу, и $x_{ij} = 0$ в противном случае. Тогда задача минимизации времени сводится к задаче календарного планирования параллельных машин [1]: найти

$$\min_{\tau,x} \tau \tag{12}$$

при ограничениях:

$$\sum_{j=1}^{m} t_j x_{ij} \leqslant \tau \text{ при всех } i = 1,\dots,k, \tag{13}$$

$$\sum_{i=1}^{k} x_{ij} = 1 \text{ при всех } j = 1,\dots,m, \tag{14}$$

$$x_{ij} \in \{0,1\} \text{ при всех } i = 1,\ldots,m,\ j = 1,\ldots,k. \qquad (15)$$

Задача (12)–(15) решалась аналогично задаче (8)–(11): применялся метод ветвей и границ с использованием оценок Лагранжа путем ослабления неравенств (13). Предугадать точное время решения параллельных задач затруднительно, но можно оценить их сложность по размерности соответствующей компоненты связности графа. Поэтому предполагалось, что время решения параллельной задачи пропорционально размерности соответствующей компоненты связности. Как и в случае задачи (8)–(11), численный эксперимент показал, что рассмотренные примеры задач (12)–(15) не являются сложными с вычислительной точки зрения. Результаты вычислений, представленные в следующем разделе, показали, что схема загрузки, полученная из решения задачи (12)–(15), позволяет улучшить абсолютное время счёта на практике.

4. Численные результаты

В этом разделе представлены результаты численного эксперимента на двух примерах, которые определены на графах с 54742 и 80759 вершинами и несколькими миллионами дуг. Остальные параметры примеров даны в таблице 1.

Т а б л и ц а 1. Параметры тестовых примеров

	Пример 1	Пример 2
$\|V\|$ (число вершин)	54742	80759
$\|A\|$ (число дуг)	5007158	6667069
$p_{\min}$ ($\|V^0\|$)	135	210
$p_{\max}$	178	253
m (число компонент в графе)	42	56

Численный эксперимент проводился на вычислительном кластере МВС-1000/16 с 32 процессорами Intel Xeon 2.6 GHz (остальные характеристики кластера можно найти в Интернете по адресу mvs.icc.ru). При распараллеливании использовался интерфейс передачи сообщений MPICH 1.2 [2].

Т а б л и ц а 2. Время работы на одном процессоре

Алгоритм	Пример 1		Пример 2	
	t_d	t_{wd}	t_d	t_{wd}
Greedy	196	1631	259	3238
LH ($\gamma = 1.04$)	1188	8957	1480	13562
LH ($\gamma = 1.01$)	3774	25203	4699	32695
LH ($\gamma = 1.005$)	7085	55669	8666	74848

В таблице 2 представлены результаты счёта (в секундах) только для жадной эвристики (Greedy) и в комбинации с эвристикой Лагранжа (LH)

при различных значениях параметра γ. В столбце t_d указано время счёта с использованием декомпозиции, в столбце t_{wd} — без декомпозиции. Как видно из представленных результатов, использование декомпозиции значительно сокращает время счёта. Но это не единственное преимущество декомпозиции. Она также позволяет находить более качественные решения. Это видно из таблиц 4–7. В этих таблицах используются следующие обозначения: p — число искомых медиан, UB — верхняя оценка, RA — относительная погрешность, вычисленная по формуле $\frac{UB-LB}{UB}100\%$, где LB — нижняя оценка (*opt* означает, что найдено оптимальное решение). Например, при $\gamma = 1{,}005$ средняя относительная погрешность для примера 2 составляет 3,35% без использования декомпозиции и 0,56% — с декомпозицией. Этот факт можно объяснить тем, что на задачах небольшой размерности субградиентный метод сходится значительно лучше.

Т а б л и ц а 3. Время работы параллельного метода

Алгоритм	np	Пример 1		Пример 2	
		$V1$	$V2$	$V1$	$V2$
Greedy	1	196	196	259	259
	2	112	102	140	128
	4	61	55	72	65
	8	41	34	50	42
LH ($\gamma = 1,04$)	1	1188	1188	1480	1480
	2	623	605	778	756
	4	333	296	390	347
	8	238	217	247	226
LH ($\gamma = 1,01$)	1	3774	3774	4699	4699
	2	2032	1935	2433	2317
	4	1078	965	1200	1075
	8	729	656	757	682
LH ($\gamma = 1,005$)	1	7085	7085	8666	8666
	2	3885	3698	4579	4359
	4	2029	1820	2268	2035
	8	1361	1246	1621	1485

В таблице 3 представлены результаты счёта с распараллеливанием, которая описана в пункте 3.4. В столбце np приведено число Slave процессов, каждый из которых был запущен на отдельном процессоре, time — абсолютное время в секундах. Было рассмотрено два варианта: $V1$ — очередь параллельных задач формировалась без учета их трудоемкости, $V2$ — очередь формировалась из решения задачи (12)–(15). Как можно видеть, вариант, представленный в столбце $V2$, позволяет сократить время счёта. Результаты представлены только до 8 процессоров, так как использование бóльшего числа процессоров не приводит к снижению времени счёта. Это связано с крупнозернистостью и неоднородностью применяемой схемы распараллеливания. Действительно, при 8 процессорах абсолютное время счёта равно времени решения самой большой

параллельной задачи. До 8 процессоров параллельное ускорение почти линейно, что является неплохим результатом.

Т а б л и ц а 4. Результаты на примере 1 с декомпозицией

	Greedy	LH ($\gamma = 1,04$)		LH ($\gamma = 1,01$)		LH ($\gamma = 1,005$)	
p	UB	UB	RA	UB	RA	UB	RA
135	1760651,18	1760651,18	*opt*	1760651,18	*opt*	1760651,18	*opt*
140	1170519,60	1170519,60	6,55	1168162,65	1,56	1168162,65	0,43
145	1040327,74	1040327,74	6,34	1038373,79	1,70	1038373,79	0,55
150	950259,85	947857,45	5,39	943055,97	1,23	943055,97	0,04
155	894577,25	892174,85	5,32	886667,50	0,91	886667,50	0,17
160	844754,45	842352,05	5,62	836859,70	0,97	836859,70	0,22
165	795494,45	793092,05	5,38	790167,22	1,19	790167,22	0,43
170	760191,26	757788,86	5,14	753516,14	1,23	753019,80	0,38
175	731461,31	729911,96	5,00	725815,90	1,27	725319,56	0,43
178	715700,17	714641,91	5,05	710935,51	1,29	710439,17	0,43

Т а б л и ц а 5. Результаты на примере 1 без декомпозиции

	Greedy	LH ($\gamma = 1,04$)		LH ($\gamma = 1,01$)		LH ($\gamma = 1,005$)	
p	UB	UB	RA	UB	RA	UB	RA
135	1760651,18	1760651,18	*opt*	1760651,18	*opt*	1760651,18	*opt*
140	1170519,60	1170519,60	14,14	1170519,60	6,81	1170519,60	3,06
145	1040327,74	1040327,74	12,30	1040327,74	5,94	1040327,74	3,43
150	950259,85	950259,85	10,81	950259,85	4,67	950259,85	2,04
155	894577,25	894577,25	11,17	894577,25	4,83	894577,25	2,49
160	844754,45	844754,45	11,09	844754,45	5,11	844754,45	2,76
165	795494,45	795494,45	9,88	795494,45	4,19	795494,45	2,39
170	760191,26	760191,26	9,36	760191,26	3,66	760191,26	2,06
175	731461,31	731461,31	9,29	731461,31	3,51	731461,31	1,88
178	715700,17	715700,17	9,21	715700,17	3,58	715700,17	1,82

Т а б л и ц а 6. Результаты на примере 2 с декомпозицией

	Greedy	LH ($\gamma = 1,04$)		LH ($\gamma = 1,01$)		LH ($\gamma = 1,005$)	
p	UB	UB	RA	UB	RA	UB	RA
210	1337516,14	1337516,14	*opt*	1337516,14	*opt*	1337516,14	*opt*
215	786214,70	786214,70	6,31	786214,70	1,68	786214,70	1,02
220	676982,24	676982,24	8,31	676922,42	1,65	676922,42	0,63
225	621167,32	619758,71	6,84	616964,39	1,23	616964,39	0,27
230	585760,15	584351,54	6,86	582470,63	1,35	582470,63	0,37
235	553806,34	552397,73	7,22	550848,47	1,49	550848,47	0,46
240	528125,73	526860,52	7,38	525948,58	2,07	525880,74	0,80
245	507607,69	506638,08	7,11	505875,16	2,01	504862,45	0,79
250	489520,65	488980,62	7,16	487637,95	1,92	486198,91	0,70
253	479333,30	478904,76	7,27	477354,31	1,90	475867,14	0,56

5. Заключение

В статье рассмотрена задача оптимального замещения, сформулированная в виде задачи о p-медиане. В качестве примеров использовались практические задачи, возникающие в автомобильной промышленности

при комплектации автомобилей электропроводкой. Особенностью рассмотренного случая является то, что эти примеры определены на графах, которые состоят из нескольких компонент связности.

Т а б л и ц а 7. Результаты на примере 2 без декомпозиции

	Greedy	LH ($\gamma = 1,04$)		LH ($\gamma = 1,01$)		LH ($\gamma = 1,005$)	
p	UB	UB	RA	UB	RA	UB	RA
210	1337516,14	1337516,14	*opt*	1337516,14	*opt*	1337516,14	*opt*
215	786214,70	786214,70	21,76	786214,70	11,41	786214,70	6,72
220	676982,24	676982,24	17,10	676982,24	7,35	676982,24	3,79
225	621167,32	621167,32	15,89	621167,32	6,43	621167,32	3,08
230	579255,40	579255,40	15,19	579255,40	6,36	579255,40	3,46
235	547633,24	547633,24	14,24	547633,24	5,68	547633,24	2,79
240	523934,85	523934,85	14,27	523934,85	5,30	523934,85	2,56
245	503744,63	503744,63	14,12	503744,63	5,21	503744,63	2,40
250	486087,17	486087,17	14,41	486087,17	5,39	486087,17	2,51
253	479333,30	479333,30	14,58	479333,30	5,44	479333,30	2,48

Предложен декомпозиционный метод, который позволяет свести исходную задачу к серии подзадач о p-медиане для каждой компоненты связности графа. Для решения этих задач реализована адаптация жадного алгоритма и эвристики Лагранжа. Из решения подзадач получено приближённое решение исходной задачи с помощью вспомогательной задачи целочисленного программирования. С помощью этого подхода были найдены приближённые решения задач большой размерности за приемлемое время на одном процессоре (сопоставимом с персональным компьютером). Выбирая параметры метода, можно получить решение с малой относительной погрешностью за более длительное время. Предложенная декомпозиция также позволяет распараллелить метод решения, добиваясь линейного параллельного ускорения на 8 процессорах.

ЛИТЕРАТУРА

1. **Танаев В. С., Шкурба В. В.** Введение в теорию расписаний. М.: Наука, 1975.
2. **Argonne National Laboratory.** Installation and User's Guide to MPICH. Chicago: University of Chicago, 2003.
3. **Avella P., Boccia M., Di Martino C., Oliviero G., Sforza A., Vasil'ev I.** A decomposition approach for a very large scale optimal diversity management problem // 4OR. 2003. V. 3, N 1. P. 23–37.
4. **Avella P., Sassano A., Vasil'ev I.** Computational study of large-scale p-median problems // Mathematical Programming. 2007. V. 109, N 1. P. 89–114.
5. **Avella P., Sforza A.** Logical reduction tests for the p-median problem // Ann. Oper. Res. 1999. V. 86. P. 105–115.

6. **Beasley J. E.** Lagrangean heuristics for location problems // European J. Oper. Res. 1993. V. 65, N 3. P. 383–399.

7. **Briant O., Naddef D.** The optimal diversity management problem // Oper. Res.. 2004. V. 52, N 4. P. 515–526.

8. **Cornuejols G., Fisher M. L., Nemhauser G. L.** Location of bank accounts to optimize float : An analytic study of exact and approximate algorithms // Management Science. 1977. V. 23. P. 789–810.

9. **Hansen P., Mladenovic N., Perez-Brito D.** Variable neighbourhood decomposition search // J. Heuristics. 2001. V. 7, N 4. P. 335–350.

10. **Kariv O., Hakimi L.** An algorithmic approach to network location problems. II: The p-medians // SIAM J. Appl. Math. 1979. V. 37. P. 539–560.

11. **Martello S., Toth P.** Knapsack problems. Algorithms and computer implementations. Chichester: John Wiley & Sons, Ltd., 1990.

12. **Mladenovic N., Brimberg J., Hansen P., Moreno-Perez J. A.** The p-median problem: a survey of metaheuristic approaches // European J. Oper. Res. 2007. V. 179, N 3. P. 927–939.

13. **Resende M. G. C., Werneck R. F.** A grasp with path-relinking for the p-median problem. Technical Report TD-5E53XL, AT&T Labs, 2002.

14. **Resende M. G. C., Werneck R. F.** On the implementation of a swap-based local search procedure for the p-median problem. Technical Report TD-5E4QKA, AT&T Labs, 2002.

15. **Whitaker R. A.** A fast algorithm for the greedy interchange for large-scale clustering and median location problems // INFOR. Can. J. Oper. Res. Inf. Process. 1983. V. 21. P. 95–108.

Адрес автора:

Институт динамики систем
и теории управления СО РАН,
ул. Лермонтова, 134,
664033 Иркутск,
Россия.
E-mail: vil@icc.ru

Статья поступила
13 сентября 2006 г.

Переработанный вариант —
22 января 2007 г.