ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Январь—июнь 2000. Серия 2. Том 7, № 1, 75–82

УДК 519.8

ЭФФЕКТИВНО РАЗРЕШИМЫЙ СЛУЧАЙ ЗАДАЧИ КАЛЕНДАРНОГО ПЛАНИРОВАНИЯ С ВОЗОБНОВИМЫМИ РЕСУРСАМИ

В. В. Сервах

Рассматривается задача календарного планирования с ограничениями на ресурсы и порядок выполнения работ. Эта задача NP-трудна в сильном смысле и является обобщением задач job-shop, flow-shop и др. Предложена ее геометрическая интерпретация, на основе которой разработан алгоритм построения оптимального расписания выполнения работ. Доказано, что если ширина графа редукции частичного порядка выполнения работ ограничена константой, то задача является псевдополиномиально разрешимой. Выделен полиномиально разрешимый случай задачи.

Введение

Пусть задано N работ, при выполнении которых используется M видов возобновимых ресурсов (станки, процессоры, исполнители и т. д.). Имеется V^m единиц ресурса вида m $(m = 1, 2, \ldots, M)$. Работа n характеризуется длительностью $\tau_n \in Z^+$ и потребностью v_n^m в m-м виде ресурсов $(n = 1, 2, \ldots, N;\ m = 1, 2, \ldots, M)$. Для каждого $t > 0$ и каждого m суммарная по всем работам, выполняемым в момент t, потребность в ресурсе m не должна превосходить V^m. Прерывания работ не допускаются. На множестве всех работ задан частичный порядок их выполнения: если работа i предшествует работе j, то выполнение работы j не может начаться раньше окончания работы i. Необходимо выполнить все работы за минимальное время.

Эта задача NP-трудна в сильном смысле [8]. Она является обобщением задачи построения конвейерного расписания [9], задачи с параллельными приборами [13], задачи календарного планирования с единичными длительностями работ. В статье предложена ее геометрическая интерпретация, на основе которой разработан алгоритм построения оптимального расписания выполнения заданных работ. При условии, что ширина графа редукции частичного порядка выполнения работ ограничена константой, доказана псевдополиномиальная разрешимость

этой задачи. Отсюда, в частности, вытекает псевдополиномиальная разрешимость задачи job-shop при фиксированном числе деталей, что также следует из результатов работы [11]. Кроме того, выделен полиномиально разрешимый случай рассматриваемой задачи.

1. Геометрическая интерпретация задачи

Для простоты изложения будем предполагать, что $\tau_n > 0$ для каждой работы n. Обозначим через s_n время начала выполнения работы n. Совокупность величин $\{s_n\}_{n=1,\dots,N}$ называется *расписанием*. Из условия неразрывности работ следует, что при заданном расписании работа n выполняется во временном интервале $(s_n, s_n + \tau_n]$. Расписание называется *допустимым*, если соблюдается заданный порядок выполнения работ и в каждый момент времени не превышены ограничения на ресурсы. *Длиной* расписания называется величина $\max\limits_{n=1,\dots,N}(s_n + \tau_n) - \min\limits_{n=1,\dots,N} s_n$. Допустимое расписание минимальной длины называется *оптимальным*.

В работах [7, 10] была предложена геометрическая интерпретация задачи обработки деталей на станках. Позднее она была развита в работах [2, 5, 6]. В настоящей статье предлагается описание такого подхода для решения задачи календарного планирования с ограничениями на ресурсы возобновимого типа.

Последовательность работ $n_1, n_2, \dots, n_l$ называется *цепью*, если работа n_i предшествует работе n_{i+1} $(i = 1, \dots, l-1)$. По определению цепи множество входящих в нее работ линейно упорядочено.

Состояние выполнения работ цепи $n_1, n_2, \dots, n_l$ можно отобразить на числовой прямой. Начиная с точки $x = 0$ на этой прямой последовательно отложим отрезки, равные длительностям работ цепи. Работе n_1 соответствует отрезок $(0, \tau_{n_1}]$, работе n_2 — отрезок $(\tau_{n_1}, \tau_{n_1} + \tau_{n_2}]$ и т. д. Точка $x \in \left(\sum\limits_{j=1}^{i-1} \tau_{n_j}, \sum\limits_{j=1}^{i-1} \tau_{n_j} + \tau_{n_i} \right]$ задает такое состояние, когда работы $n_1, n_2, \dots, n_{i-1}$ уже завершены и выполняется работа n_i, причем ее часть длительностью $x - \sum\limits_{j=1}^{i-1} \tau_{n_j}$ уже выполнена. Дальнейшему непрерывному выполнению работ этой цепи в течение времени τ отвечает отрезок $(x, x + \tau]$. Точка $x = \sum\limits_{j=1}^{l} \tau_{n_j}$ соответствует состоянию, при котором все работы цепи выполнены.

Исходное множество работ разобьем на непересекающиеся подмножества, каждое из которых является цепью. При этом работы, входящие в разные цепи, могут находиться в отношении предшествования. Пусть получилось K цепей. Обозначим через S_k^i суммарную длительность первых i работ, а через S_k суммарную длительность всех работ цепи

k ($k = 1, 2, \ldots, K$). Цепи k поставим в соответствие k-ю координатную ось в K-мерном пространстве. Обозначим $\mathbf{0} = (0, 0, \ldots, 0)$, $\mathbf{S} = (S_1, S_2, \ldots, S_K)$, $\mathscr{K} = \{1, 2, \ldots, K\}$, $X = \{\mathbf{x} = (x_1, x_2, \ldots, x_K) \mid 0 \leqslant x_k \leqslant S_k,\ k \in \mathscr{K}\}$. Для любой точки $\mathbf{x} \in X$ определим K текущих работ — по одной для каждой цепи $k \in \mathscr{K}$ — и для каждого k находим номер $i = i(k, \mathbf{x})$ такой, что $S_k^{i-1} < x_k \leqslant S_k^i$. (Если $x_k = 0$, то полагаем $i(k, \mathbf{x}) = 0$.) Работу, стоящую на i-м месте в цепи k, будем обозначать через $j(k, i)$ и называть *текущей* для точки $\mathbf{x}$. Тогда всякая точка $\mathbf{x} \in X$ задает текущее состояние выполнения работ каждой цепи. При $\mathbf{x} = \mathbf{0}$ ни одна работа не начала выполняться, а при $\mathbf{x} = \mathbf{S}$ все N работ уже завершились.

Определим понятия допустимых и недопустимых состояний. Состояние $\mathbf{x}$ считается *недопустимым* в следующих двух случаях.

1. Несогласованность с частичным порядком.

Работы каждой цепи выполняются последовательно. Однако, как было отмечено выше, в отношении предшествования могут находиться работы разных цепей. Пусть, например, i_1-я работа цепи k_1 предшествует i_2-й работе цепи k_2. Если точка $\mathbf{x}$ задает состояние, когда первая из этих работ еще не завершена ($x_{k_1} < S_{k_1}^{i_1}$), а вторая уже начала свое выполнение ($x_{k_2} > S_{k_2}^{i_2-1}$), то в силу заданного на множестве работ порядка состояние $\mathbf{x}$ недопустимо.

2. Несоблюдение ограничений на ресурсы.

Если для точки $\mathbf{x}$ и некоторого k выполнены неравенства $S_k^{i-1} < x_k < S_k^i$, то работа $j = j(k, i)$ находится в состоянии выполнения и требует v_j^m единиц ресурса вида m. Суммируя эту потребность по всем $k \in \mathscr{K}$, получим общие затраты ресурсов, отвечающие точке $\mathbf{x}$. Если эти затраты превосходят V^m хотя бы для одного m, то состояние $\mathbf{x}$ недопустимо.

Остальные состояния $\mathbf{x} \in X$ считаются допустимыми.

Процесс выполнения работ изображается с помощью пути, идущего из точки $\mathbf{0}$ в точку $\mathbf{S}$ и представляющего собой ломаную линию, состоящую из отрезков. Каждый отрезок $(\mathbf{x}, \mathbf{y}]$ удовлетворяет соотношению $\mathbf{y} = \mathbf{x} + \tau\delta$, где $\delta = (\delta_1, \delta_2, \ldots, \delta_K) \in \{0, 1\}^K$ — булевый вектор, $\tau > 0$. Таким образом, каждому отрезку $(\mathbf{x}, \mathbf{y}]$ пути из $\mathbf{0}$ в $\mathbf{S}$ поставлен в соответствие булевый вектор δ. Отрезок $(\mathbf{x}, \mathbf{x} + \tau\delta]$ соответствует одновременному выполнению в течение времени τ работ тех и только тех цепей, для которых $\delta_k = 1$.

Длиной отрезка $(\mathbf{x}, \mathbf{y}]$ будем называть величину $\max\limits_{k \in \mathscr{K}} |y_k - x_k| = \tau$. *Длина пути* определяется как сумма длин составляющих его отрезков.

Отрезок $(\mathbf{x}, \mathbf{x} + \tau\delta]$ считается *недопустимым*, если он содержит хотя бы одну недопустимую точку, либо не удовлетворяет требованию непрерывности выполнения работ, т. е. когда для некоторых k и i

выполняется одновременно $\delta_k = 0$ и $S_k^{i-1} < x_k < S_k^i$ (что соответствует простою уже начатой, но не завершенной работы $j(k,i)$).

Путь из $\mathbf{0}$ в $\mathbf{x}$, не содержащий недопустимых отрезков, называется *допустимым*. Очевидно, что каждому допустимому пути из $\mathbf{0}$ в $\mathbf{S}$ отвечает единственное допустимое расписание без простоев. (Под простоем понимается такой ненулевой интервал времени, в течение которого не выполняется ни одна работа.) Верно и обратное: любому допустимому расписанию без простоев отвечает единственный допустимый путь из $\mathbf{0}$ в $\mathbf{S}$. Таким образом, для построения оптимального расписания достаточно найти кратчайший допустимый путь из $\mathbf{0}$ в $\mathbf{S}$. Такой путь назовем *оптимальным*.

2. Теоретические основы алгоритма

Прежде всего отметим простой факт.

Утверждение 1. *Если все* τ_n $(n = 1, 2, \ldots, N)$ *целочисленны, то существует оптимальный путь из* $\mathbf{0}$ *в* $\mathbf{S}$, *в котором длина каждого отрезка целочисленна.*

Это утверждение доказано в [6] для случая $\tau_n = 1$ и легко обобщается на случай произвольных целочисленных τ_n.

Согласно утверждению 1 оптимальное решение задачи при целочисленных длительностях достаточно искать среди расписаний, в которых каждая работа начинается и заканчивается в целочисленные моменты времени. В таких расписаниях при любом целом t на временном интервале $(t-1, t]$ множество выполняемых работ не изменяется. В геометрической интерпретации интервалу $(t-1, t]$ соответствует отрезок единичной длины $(\mathbf{x} - \delta, \mathbf{x}]$, где $\mathbf{x}$ — целочисленная точка, а δ — булевый вектор, в котором $\delta_k = 1$, если в интервале $(t-1, t]$ работа цепи k выполняется, и $\delta_k = 0$ в противном случае. Таким образом, любой интересующий нас путь из $\mathbf{0}$ в $\mathbf{S}$ является последовательностью отрезков единичной длины вида $(\mathbf{x} - \delta, \mathbf{x}]$.

Введем следующие обозначения:

$\Delta_{\mathbf{x}}$ — множество таких булевых векторов δ, что отрезок $(\mathbf{x} - \delta, \mathbf{x}]$ является допустимым;

$L(\mathbf{x})$ — длина кратчайшего допустимого пути от $\mathbf{0}$ до $\mathbf{x}$.

Имеют место следующие рекуррентные соотношения (Беллмана):

$$L(\mathbf{0}) = 0; \quad L(\mathbf{x}) = \min_{\delta \in \Delta_{\mathbf{x}}} (L(\mathbf{x} - \delta) + 1), \ \mathbf{x} \in X \cap Z^K \setminus \{0\}.$$

3. Описание алгоритма и анализ его сложности

Предполагаем, что частичный порядок на множестве работ задан удобным для нас образом, позволяющим для любых двух работ за время

$O(1)$ установить факт предшествования одной работы другой. (Например, это можно сделать с помощью матрицы размера $N \times N$. Построение такой матрицы в случае задания частичного порядка в другом виде, например с помощью графа редукции частичного порядка, потребует не более $O(N^2)$ вычислительных операций.)

Предлагаемый алгоритм состоит из трех этапов.

Этап 1. Формирование цепей.

Этот этап опишем ниже, когда будут ясны критерии, по которым необходимо формировать цепи. Для найденных цепей вычислим значения величин S_k^i и S_k.

Этап 2. Вычисление длины кратчайшего пути из $\mathbf{0}$ в $\mathbf{S}$.

Полагается, что $L(\mathbf{0}) = 0$. Затем в порядке лексикографического возрастания перебираются все целочисленные точки $\{\mathbf{x} \mid \mathbf{0} < \mathbf{x} \leqslant \mathbf{S}\}$. Для каждой точки $\mathbf{x} = (x_1, x_2, \ldots, x_K)$ выполняются следующие шаги.

Шаг 1. Полагается, что $L(\mathbf{x}) = \infty$.

Шаг 2. Находится множество текущих работ, отвечающих рассматриваемой точке $\mathbf{x}$. Это делается так. Для каждого k такого, что $x_k > 0$, находится такое i, что $S_k^{i-1} < x_k \leqslant S_k^i$. Это множество легко формируется при лексикографическом переборе, когда переход по каждой оси осуществляется либо к соседней целочисленной точке, либо к нулевому значению. Таким образом, зная множество текущих работ для одной точки, легко найти такое множество и для следующей точки.

Шаг 3. Если для какой-то пары координат $k_1, k_2 \in \mathscr{K}$ при $i_1 \doteq i(k_1, \mathbf{x})$ и $i_2 \doteq i(k_2, \mathbf{x})$ выполнено неравенство $i_1 < n_{k_1}$ и работа $j(k_1, i_1+1)$ предшествует работе $j(k_2, i_2)$, то точка $\mathbf{x}$ в этом случае недопустима; рассматривается следующая точка.

Шаг 4. В булевом векторе $z_0 \in \{0,1\}^K$ формируются запреты для координат вектора δ принимать значение 0. Для этого просматриваются $k \in \mathscr{K}$; если для $i = i(k, \mathbf{x})$ выполнены неравенства $S_k^{i-1} < x_k < S_k^i$, то полагается $z_0(k) = 1$. Это означает запрет на выполнение равенства $\delta_k = 0$ (запрет на прерывание начатой работы $j(k,i)$); в противном случае $z_0(k) = 0$.

Шаг 5. В булевом векторе $z_1 \in \{0,1\}^K$ формируются запреты на равенство $\delta_k = 1$. Сначала полагается $z_1(k) = 0$ для каждого $k \in \mathscr{K}$. Затем просматриваются всевозможные пары координат $k_1, k_2 \in \mathscr{K}$; если для $i_1 \doteq i(k_1, \mathbf{x})$ и $i_2 \doteq i(k_2, \mathbf{x})$ работа $j(k_1, i_1)$ предшествует работе $j(k_2, i_2)$, то полагается $z_1(k_1) = 1$. Это означает запрет на выполнение равенства $\delta_{k_1} = 1$.

Шаг 6. Просматривается $k \in \mathscr{K}$; если $z_0(k) = 1$ и $z_1(k) = 1$, то множество $\Delta_{\mathbf{x}}$ допустимых для точки $\mathbf{x}$ векторов δ пусто и осуществляется переход к следующей точке $\mathbf{x}$.

Шаг 7. Перебираются все такие булевы векторы $\delta \in \{0,1\}^K \setminus \{\mathbf{0}\}$, что $\delta_k = 1$, если $z_0(k) = 1$, и $\delta_k = 0$, если $z_1(k) = 1$.

Вектор δ является недопустимым для точки $\mathbf{x}$ в следующих трех случаях:

a) $x_k - \delta_k < 0$ для некоторого $k \in \mathscr{K}$;

b) $L(\mathbf{x} - \delta) = \infty$;

c) суммарный объем ресурса m, потребляемый работами на временном интервале $(\mathbf{x} - \delta, \mathbf{x}]$, превосходит V^m хотя бы для одного m.

Если вектор δ допустим, то полагается $L(\mathbf{x}) = \min\{L(\mathbf{x}), L(\mathbf{x}-\delta)+1\}$; вектор δ фиксируется как $\delta(\mathbf{x})$, если минимум достигается на втором члене.

Этап 3. Восстановление оптимального расписания.

Первоначально полагается $\mathbf{x} = \mathbf{S}$. Множество таких работ, что $\delta_k(\mathbf{x}) = 1$, является множеством работ, выполняемых во временном интервале $(L(\mathbf{x})-1, L(\mathbf{x})]$. Для всех $k \in \mathscr{K}$ таких, что $x_k - \delta_k(\mathbf{x}) = S_k^{i(k,\mathbf{x})-1}$, полагается $s_{j(k,i(k,\mathbf{x}))} = L(\mathbf{x}) - 1$. Далее осуществляется переход к точке $\mathbf{x} - \delta(\mathbf{x})$ и процесс повторяется до тех пор, пока $\mathbf{x}$ не станет равным $\mathbf{0}$.

Оценим время работы алгоритма. Наиболее трудоемким является этап 2. На нем для каждой целочисленной точки $\mathbf{x} \in X$ вычисляется $L(\mathbf{x})$. Временная сложность шагов 1–6 не превосходит $O(K^2)$, а выполнение шага 7 требует перебора $2^K - 1$ булевых векторов. При этом проверка ограничений по ресурсам требует порядка $O(KM)$ операций. Таким образом, число операций на шагах 1–7 не превосходит $O(K^2 + 2^K KM) \leqslant O(2^K KM)$. Поэтому суммарное число T_2 операций этапа 2 составляет

$$T_2 = O(2^K KM|X \cap Z^K|) = O\left(2^K KM \prod_{k=1}^{K}(S_k + 1)\right).$$

Вернемся теперь к этапу 1. Понятно, что цепи необходимо формировать таким образом, чтобы минимизировать величину выписанного выражения. На самом деле для доказательства приведенного ниже основного результата данной статьи нам достаточно рассмотреть такой алгоритм, при котором минимизируется количество цепей K. Такая задача сводится к построению максимального паросочетания в двудольном графе [3] и, следовательно, полиномиально разрешима.

Рассмотрим двудольный ориентированный граф (V_1, V_2, E), в котором множества вершин V_1 и V_2 каждой доли содержат по N вершин, а дуга (i,j) $(i \in V_1,\ j \in V_2)$ принадлежит множеству E, если работа i предшествует работе j. В этом графе находим максимальное паросочетание. После этого совмещаем соответствующие вершины двух долей.

Получаем граф с N вершинами. Дуга (i, j) является дугой нового графа, если соответствующая дуга (i, j) входит в паросочетание двудольного графа. Очевидно, что построенный граф состоит из $K = N - P$ цепей, где P — число дуг в паросочетании. При максимальном P число цепей минимально.

Паросочетание в двудольном графе строится за $O(N^{5/2})$ операций [4, с. 231]. Таким образом, временная сложность T_1 этапа 1 есть полином от N и не зависит от K. Остается сделать анализ временной сложности в зависимости от величины K. По теореме Дилуорса [1, с. 459] минимальное число цепей K, на которое разбивается частичный порядок, совпадает с максимально возможным числом независимых работ (этот параметр называется *шириной частичного порядка*). С ростом ширины частичного порядка K временная сложность $T_1 + T_2$ предложенного алгоритма растет экспоненциально, а с ростом параметров M и N — полиномиально. Тем самым справедлива

Теорема. *Задача календарного планирования с ограниченными ресурсами при условии, что ширина заданного на множестве работ частичного порядка ограничена константой, разрешима за псевдополиномиальное время.*

Отсюда можно сделать важный вывод, что сильная NP-трудность рассматриваемой задачи «заключена» в ширине частичного порядка.

Следствие 1. *Задача календарного планирования с ограниченными ресурсами и единичными длительностями работ разрешима за полиномиальное время, если ширина частичного порядка, заданного на множестве работ, ограничена константой.*

В качестве примера, иллюстрирующего теорему, рассмотрим задачу job-shop минимизации времени обработки N деталей на M станках. Каждая деталь проходит свой технологический маршрут обработки. Этот маршрут можно представить как цепь последовательно выполняемых работ. Работа заключается в обработке одной детали на одном станке. Частичный порядок на множестве работ представляет собой совокупность N цепей. В качестве ресурсов рассматриваются станки.

Следствие 2. *При фиксированном числе деталей задача* job-shop *псевдополиномиально разрешима.*

Этот результат также вытекает из результата работы [11]. Полиномиального алгоритма при числе деталей больше двух не существует [12] (в предположении P$\neq$NP).

Автор сердечно благодарит С. В. Севастьянова за огромную помощь в работе над статьей.

ЛИТЕРАТУРА

1. **Айгнер М.** Комбинаторная теория. М.: Мир, 1982.
2. **Глебов Н. И.** Некоторые случаи сводимости m-станочной задачи Джонсона к задачам с двумя станками // Управляемые системы: Сб. науч. тр. Новосибирск: Ин-т математики СО АН СССР, 1978. Вып. 17. С. 46–51.
3. **Косточка А. В.** Дискретная математика: Учеб. пособие. Ч. 2. Новосибирск: Изд-во Новосиб. ун-та, 1985.
4. **Пападимитриу П., Стайнглиц К.** Комбинаторная оптимизация. Алгоритмы и сложность. М.: Мир, 1985.
5. **Сервах В. В.** О задаче Акерса–Фридмана // Управляемые системы: Сб. науч. тр. Новосибирск: Ин-т математики СО АН СССР, 1983. Вып. 23. С. 70–81.
6. **Сервах В. В.** Алгоритм решения задачи календарного планирования с единичными длительностями работ // Дискретная оптимизация и анализ сложных систем. Новосибирск: ВЦ СО АН СССР, 1989. С. 98–107.
7. **Akers S. E.** A graphical approach to production scheduling problems // Oper. Res. 1956. V. 4, N 2. P. 244–245.
8. **Garey M. R., Johnson D. S.** Complexity results for multiprocessor scheduling under resource constraints // SIAM J. Comput. 1975. V. 4, N 4. P. 397–411.
9. **Garey M. R., Johnson D. S., Sethi R.** The complexity of flow shop and job shop scheduling // Math. Oper. Res. 1976. V. 1, N 2. P. 117–129.
10. **Hardgrave W. W., Nemhauser G. L.** A geometric model and a graphical algorithm for a sequencing problem // Oper. Res. 1963. V. 11, N 6. P. 889–900.
11. **Meyer W.** Geometrische Methoden zur Lösung von Job-Shop Problemen und deren Verallgemeinerungen. PhD Diss. Fachbereich Mathematik / Informatik, Univ. Osnabrück, 1992.
12. **Sotskov Y. N., Shakhlevich N. V.** NP-hardness of shop-scheduling problems with three jobs // Discrete Appl. Math. 1995. V. 59, N 3. P. 237–266.
13. **Ullman J. D.** NP-complete scheduling problems // J. Comput. System Sci. 1975. V. 10, N 4. P. 384–393.

Адрес автора:

Омский филиал
Института математики
им. С. Л. Соболева СО РАН,
ул. Певцова, 13,
644099 Омск, Россия.
E-mail: svv@iitam.omsk.net.ru

Статья поступила
15 апреля 1999 г.,
переработанный вариант —
23 февраля 2000 г.