УДК 519.8

Ю.А. Зак

Методы случайного поиска в решении задач теории расписаний

Сформулированы общие подходы, установлены свойства и параметры алгоритмов глобального случайного поиска, приемлемые для решения широкого класса задач теории расписаний в условиях ограничений на сроки выполнения заданий.

General approaches are formulated, the properties and parameters of the algorithms of the global random search are established which are acceptable for solving a wide class of problems of the theory of schedule under conditions of the limits on assignments.

Сформульовано загальні підходи, встановлено властивості і параметри алгоритмів глобального випадкового пошуку, прийнятні для розв’язання широкого класу задач теорії розкладів за умов обмежень на терміни виконання завдань.

Введение. Характерной особенностью методов случайного поиска есть то, что на каждой итерации процесса выбор нового решения проводится не детерминированно, а в результате реализации некоторого случайного процесса, причем выбираемая стратегия может зависеть от предыстории поиска [1].

Для большинства таких алгоритмов при достаточно большом количестве реализаций случайных значений вектора переменных в задаче доказана сходимость в окрестность точки локального или глобального экстремума с вероятностью, равной единице, т.е.

$$P\left\{\lim_{k\to\infty}\left\|X^k - X^*\right\| \le \varepsilon\right\} = 1,$$

$$P\left\{\lim_{k\to\infty}\left\|F(X^k) - F(X^*)\right\| \le \delta\right\} = 1, \qquad (1)$$

где ε и δ – любые наперед заданные неотрицательные числа; X^* и $F(X^*)$ – соответственно значение вектора переменных и функции цели в точке экстремума; X^k, $F(X^k)$, $k = 1, 2, \ldots, K$ – последовательность векторов переменных и значений функции цели на различных итерациях алгоритма.

Данные методы могут применяться как при решении задач локальной оптимизации, так и при поиске глобального экстремума.

В алгоритмах направленного случайного поиска результаты вычислений, выполненных на предыдущих итерациях, используются для выбора направлений дальнейших испытаний и изменений вектора оптимизируемых переменных.

Алгоритмы локального случайного поиска [2] сводятся, по сути, к случайному перебору на каждом k-м шаге векторов X из некоторой окрестности $\| X^k - X \| \le \delta$, проверке эффективности каждого из этих векторов (пробного шага) $\{F(X) - F(X^k)\}$ и к выбору соответствующей реакции на исход эксперимента. Различные алгоритмы данного класса отличаются друг от друга реализацией отдельных этапов выбора экспериментальных точек и методами принятия решений. Наибольшее распространение сегодня получили монотонные алгоритмы случайного поиска, в которых переход к следующему значению вектора переменных задачи происходит лишь в том случае, если $F(X^{k+1}) > F(X^k)$ в задаче максимизации целевой функции или в случае, если $F(X^{k+1}) < F(X^k)$ – в задаче минимизации.

Решение задач теории расписаний, в самом общем случае, состоит из двух этапов. На первом этапе решения проводится разбиение некоторого множества объектов с заданным набором характеристик на непересекающиеся подмножества, а на втором – упорядочение объектов каждого из сформированных подмножеств в соответствии с выбранным критерием оптимальности и обеспечивающим выполнение заданной системы ограничений.

Ключевые слова: разбиение на подмножества и упорядочение заданий, глобальный случайный поиск, оптимальные расписания выполнения работ, ограничения на сроки выполнения заданий.

В рассматриваемых задачах теории расписаний все граничные значения времени выполнения операций предполагаются целочисленными величинами. Не допускается прерывание работ в процессе их выполнения.

Известно большое количество публикаций, в которых методы случайного поиска применяются для решения различных частных задач теории расписаний. Подробный обзор таких методов изложен в монографиях [3, 4]. Цель статьи – формирование некоторых общих подходов и установление общих свойств и параметров алгоритмов, приемлемых для решения широкого класса задач теории расписаний. Предложенные алгоритмы иллюстрируются при построении алгоритмов решения этими методами классических задач разбиения на подмножества и упорядочения: синхронизация работы конвейерных линий и минимизация суммы штрафов при построении допустимых расписаний на одной машине.

Общая вычислительная схема методов глобального случайного поиска

После выполнения каждой большой итерации глобального случайного поиска сохраняется одно допустимое расписание выполнения заданий и соответствующее ему значение критерия оптимальности, наилучшее среди всех построенных на предыдущих итерациях генерирования расписаний.

Опишем итерацию алгоритма глобального случайного поиска применительно к решению задачи теории расписаний. Необходимо осуществить разбиение некоторого подмножества, состоящего из N объектов, на m непересекающихся подмножеств и затем упорядочить элементы каждого из образованных подмножеств.

Пусть $\widetilde{J} = \{i_1, i_2, ..., i_l, ..., i_N\}$ – множество всех N объектов, упорядоченных по возрастанию их индексов.

Каждый шаг итерации состоит из двух последовательных этапов:

• разбиение случайным образом всех N объектов на m непересекающихся подмножеств, соответствующих подмножествам заданий, выполняемых на различных машинах;

• упорядочение случайным образом элементов каждого из этих подмножеств, т.е. построение допустимых последовательностей выполнения этих заданий на различных машинах.

Пусть на каком-то s-м шаге $s = 1, \dots, N$ малой итерации первого этапа построены: $\bar{J}^{s,k} = \left\{ j_1^{sk}, j_2^{sk}, ..., j_r^{sk} \right\}$, $k = 1, \dots, m$ – подмножества заданий, назначенных для выполнения на k-й машине; $\bar{I}^s = \{i_1, ..., i_{n(s)}\}$ – подмножество заданий, еще не включенных ни в одно из подмножеств $\bar{J}^{s,k}$, $\bar{I}^s = \bar{I} / \left\{ \bigcup_{k=1}^{m} \bar{J}^{s,k} \right\}$. Элементы в подмножестве $\bar{I}^s$ упорядочены в последовательность $\widetilde{I}^s$ по возрастанию их индексов.

Выполнение второго этапа начинается, когда сформированы все подмножества элементов $\bar{I}^1, ..., \bar{I}^m$, каждое из которых включает соответственно $N_1, \dots, N^m$ членов, причем

$$\sum_{j=1}^{m} N^j = N, \ \bigcup_{j=1}^{m} \bar{I}^j = \bar{I}, \ \bar{I}^j \cap \bar{I}^l = \varnothing,$$
$$j, l = 1, ..., m; \ j \neq l. \qquad (2)$$

Пусть на каком-то g-м шаге малой итерации второго этапа уже сформированы m подпоследовательностей $\widetilde{V}^{kg} = \{i_1^{gk}, i_2^{gk}, ..., i_{N^{gk}}^{gk}\}$. Каждая из них содержит соответственно N^{gk} ($N^{gk} \leq N^k$) членов. В каждом из подмножеств $\bar{I}^{g1}, ..., \bar{I}^{gk}, ...$ $..., \bar{I}^{gm}$ еще осталось $n^{gk} = N^k - N^{gk}$, не включенных в эти последовательности элементов, каждый из которых стоит на определенном месте в последовательности $\widetilde{U}^{gk} = \{u_1^{gk}, u_2^{gk}, ..., u_{n^{gk}}^{gk}\}$, $k = 1, \dots, m$. Пусть $m(g) \leq m$ – количество последовательностей, для которых справедливо соотношение $N^k > N^{gk}$.

При отсутствии ограничений на сроки завершения выполнения заданий на каждой большой итерации первого этапа выполняем следующий объем вычислений.

Алгоритм 1. 1) Реализуем случайное число, распределенное по некоторому закону распределения Ψ_1^s на интервале чисел $\xi_1^s \in [0, m]$. Определим значение

$$\mu_1^s = \begin{cases} \xi_1^s, & \text{если} \quad \xi_1^s - \left|\xi_1^s\right| \le 0,5, \\ (\xi_1^s + 1), & \text{если} \quad \xi_1^s - \left|\xi_1^s\right| > 0,5. \end{cases} \quad (3)$$

2) Реализуем случайное число, распределенное по некоторому закону распределения на интервале чисел $\xi_2^s \in [0, n(s)]$, где k – номер шага на первом этапе итерации алгоритма, а $n(s)$ определяет число элементов в последовательности $\widetilde{I}^s$, до настоящего момента еще не отнесенных к какому-либо из подмножеств. Вычислим значение

$$\eta_2^s = \begin{cases} \xi_2^s, & \text{если} \quad \xi_2^s - \left|\xi_2^s\right| \le 0,5, \\ (\xi_2^s + 1), & \text{если} \quad \xi_2^s - \left|\xi_2^s\right| > 0,5. \end{cases} \quad (4)$$

3) Определим элемент, стоящий на η_2^s-м месте последовательности $\tilde{I}^s$. Пусть это будет элемент $i(\eta_2^s)$. Включим этот элемент в подмножество, номер которого равен μ_1^s, и исключим его из подмножества $\bar{I}^s$ и из последовательности $\widetilde{I}^s$, уменьшив тем самым количество членов этой последовательности на единицу. В зависимости от количества членов, включенных на данном этапе в каждое из подмножеств, скорректируем закон распределения случайных чисел Ψ_1^s. Если $\widetilde{I}^s \neq \varnothing$, т.е. $n(s) = 0$, то возвращаемся к выполнению пп. 1–3. В противном случае переходим к выполнению вычислений второго этапа.

Объем вычислений на втором этапе каждой большой итерации заключается в следующем.

4) Реализуем случайное число, распределенное по равномерному закону распределения Ψ_2^g на интервале чисел $\xi_3^g \in [0, m(g)]$. Вычислим значение

$$\mu_3^g = \begin{cases} \xi_3^g, & \text{если} \quad \xi_3^g - \left|\xi_3^g\right| \le 0,5, \\ (\xi_3^g + 1), & \text{если} \quad \xi_3^g - \left|\xi_3^g\right| > 0,5, \end{cases} \quad (5)$$

которое определяет номер подпоследовательности k, на последнее место в которой будет установлен выбираемый в дальнейшем элемент.

5) Реализуем случайное число, распределенное по некоторому закону распределения на интервале чисел $\xi_4^g \in [0, n^{gk}]$. Вычислим значение

$$\eta_3^{gk} = \begin{cases} \xi_4^g, & \text{если} \quad \xi_4^g - \left|\xi_4^g\right| \le 0,5, \\ (\xi_4^g + 1), & \text{если} \quad \xi_4^g - \left|\xi_4^g\right| > 0,5. \end{cases} \quad (6)$$

Это число определяет элемент, стоящий в последовательности $\widetilde{U}^{gk}$ на соответствующем месте. Пусть это будет элемент $i(\eta_3^{gk})$. Установим этот элемент на последнее место в подпоследовательности $\widetilde{V}^{gk}$ и исключим его из $\widetilde{U}^{gk}$. Скорректируем значения $N^{g+1,k} = N^{gk} + 1$, $n^{l,t+1} = n^{lt} - 1$. Если $n^{g+1,k} = 0$, то полагаем $m(g+1) = m(g) - 1$. Если $m(g+1) = 0$, то большая итерация алгоритма заканчивает работу. В противном случае – переходим к выполнению пп. 4, 5.

Если в результате выполненной большой итерации получено допустимое расписание $\widetilde{P}^{SG}$ и $F(\widetilde{P}^{SG}) < F_{\min}(P)$, то полагаем $F_{\min}(P) = F(\widetilde{P}^{SG})$ и переходим к следующей большой итерации случайного поиска.

При наличии ограничений на сроки завершения выполнения заданий T_i, $i = 1, \ldots, N$, объем вычислений на каждой большой итерации алгоритма увеличивается. Упорядочим все задания $\bar{I} = \{1, \ldots, N\}$, а также подмножества заданий $\bar{I}^l = \{1, \ldots, N^l\}$, $l = 1, \ldots, m$, выполняемых на каждой машине по возрастанию значений T_i:

$$\tilde{W} = \left\{ i_1, i_2, \ldots, i_N \mid T_{i_1} \le T_{i_2} \le \ldots \le T_{i_N} \right\},$$

$$\tilde{W}^l = \left\{ i_{i_1}^l, i_{i_2}^l, \ldots, i_{i_N}^l \mid T(i_{i_1}^l) \le T(i_{i_2}^l) \le \ldots \le T(i_{i_N}^l) \right\}, \quad (7)$$

$$l = 1, \ldots, m.$$

Упорядочим подмножества заданий $\bar{J}^{s,k}$ в последовательность (7) в порядке возрастания значений $T(i_p^{sk})$ – $\tilde{V}^{s,k} = \left\{ v_1^{sk}, v_2^{sk}, \ldots, v_r^{sk} \mid T(v_1^{sk}) \le \right.$ $\left. \le T(v_2^{sk}) \le \ldots \le T(v_r^{sk}) \right\}$. Пусть $t^{k,\min}(v_p^{sk})$ – минимальное суммарное время, необходимое для переналадки и выполнения i_l-го задания на k-й машине, $k = 1, \ldots, m$.

Автором показано [3], что если выполняется хотя бы одно из системы неравенств

$$\theta^k + \sum_{l=1}^{p} t^{\min}(v_l^{s,k}) > T(v_p^{s,k}),$$

$$p=1,...,r\ ,\ k=1,...,m\ , \qquad (8)$$

то на k-й машине не может быть построено расписание выполнения заданий, удовлетворяющее всем ограничениям задачи.

Пусть на k-й машине построена некоторая подпоследовательность выполнения заданий $\widetilde{V}^{kg}=\{i_1^{gk},i_2^{gk},...,i_{N^{gk}}^{gk}\}$. $\overline{T}\{\widetilde{V}^{kg}\}$ – время завершения выполнения этой подпоследовательности заданий $\widetilde{V}^{kg}$ на k-й машине. Пусть на следующее место в этой последовательности (после задания $i_{N^{gk}}^{gk}$) устанавливается задание $i_{N+1}^{gk}=$ $=i(\eta_3^{gk})$ и время выполнения этого задания с учетом потерь времени на переналадку k-й машины составляет $t(i_{N+1}^{gk})$. Если

$$\overline{T}\{\widetilde{V}^{kg}\}+t(i_{N+1}^{gk})>T(i_{N+1}^{gk})\,,\ k=1,\ldots,m, \qquad (9)$$

то задание $i_{N^{gk}}^{gk}$ не может быть установлено на данное место, а следовательно, и на любое другое более дальнее место последовательности выполнения заданий на k-й машине.

Следовательно, строящаяся последовательность выполнения заданий на данной машине недопустима. Поэтому на каждом g-м шаге малой итерации второго этапа выбор заданий для установки на $(N^{gk}+1)$-е место в последовательности $\widetilde{V}^{kg}$ может выбираться только из некоторого подмножества $\widetilde{W}^{gk}\subseteq\widetilde{U}^{gk}$, где

$$\widetilde{W}^{gk}=\left\{\overline{u}_p^{gk}\in\widetilde{U}^{gk},T\{\widetilde{V}^{gk}\}+t(\overline{u}_p^{gk})\le T(\overline{u}_p^{gk}),\,p=1,...,n^{gk}\right\},$$
$$k=1,\ldots,m. \qquad (10)$$

Рассмотрим также некоторую другую модификацию описанного алгоритма.

Алгоритм 2. Пусть на некотором s-м шаге формирования допустимого расписания выполнения N заданий на m машинах построены допустимые подпоследовательности $\widetilde{V}^{ks}=\{i_1^{ks},i_2^{ks},...$ $...,i_{N^{ks}}^{ks}\}$ выполнения некоторого подмножества заданий $\overline{I}^{ks}$ на каждой машине. Обозначим $\overline{T}\{\widetilde{V}^{kg}\}$ – время завершения выполнения этой подпоследовательности заданий $\widetilde{V}^{ks}$ на k-й машине, $k=1,\ldots,m$. Пусть $\overline{J}^s$ – подмножество заданий, не включенных ни в одну из последовательностей $\widetilde{V}^{ks}$. Определим и упорядочим по возрастанию значений T_i подмножество заданий $\overline{\Omega}^{ks}=\{j_1^{ks},...,j_n^{ks}\mid T_{j_1^{ks}}\le T_{j_2^{ks}}\le...\le T_{j_n^{ks}}\}$, которые без нарушения сроков завершения их выполнения могут быть установлены на $(N^{ks}+1)$-е место в $\widetilde{V}^{ks}$. Пусть n^{ks} – количество элементов подмножества $\overline{\Omega}^{ks}$.

Следовательно,

$$\overline{T}\{\widetilde{V}^{ks}\}+a(i_{N^{ks}}^{ks},j_{l^{ks}}^{ks})+t(j_{l^{ks}}^{ks})\le T_{j_l^{ks}}\ ,\ j_n^{ks}\in\overline{\Omega}^{ks}\,.$$

Определим $\overline{M}^s=\{k_1^s,k_2^s,...,k_\mu^s\mid\overline{\Omega}^{ks}\ne\varnothing\}$ – подмножество машин, упорядоченное по возрастанию индексов этих машин, последовательность выполнения заданий на которых может быть продолжена, количество таких машин равно $m_1^s\le m$.

Если $\overline{\Omega}^{ks}=\varnothing$, $k=1,...,m$ и при этом $\overline{J}^s=\varnothing$, то построено допустимое расписание выполнения заданий. Если при этом $F(\widetilde{P}^s)<$ $<F_{opt}(\widetilde{P})$, то полагаем $F_{opt}(\widetilde{P})=F(\widetilde{P}^s)$. Если $\overline{\Omega}^{ks}=\varnothing$, $k=1,\ldots,m$ и при этом $\overline{J}^s\ne\varnothing$, то на данной итерации не может быть построено допустимое расписание выполнения заданий. Как в первом, так и во втором случае переходим к следующей большой итерации случайного поиска, положив $s=0$, $\overline{J}^s=\overline{I}=\{1,...,k,...,m\}$, $m_1^s=m$; $i_{N^{ks}}^{ks}=0$, $\overline{T}\{\widetilde{V}^{ks}\}=\theta^k$, $k=1,\ldots,m$ и определив соответствующим образом множества $\overline{\Omega}^{ks}$. Если $m_1^s\ge1$ и $\overline{\Omega}^{ks}\ne\varnothing$, $k\in\overline{M}^s$, то

– на каждом s-м шаге большой итерации алгоритма реализуем случайное число $\xi_1^s\in[0,m_1^s]$, с помощью которого определяем номер машины, последовательность выполнения заданий на которой $\widetilde{V}^{ks}$ должна быть продолжена;

– реализуем случайное число $\xi_2^s\in[0,n^{ks}]$, на основании которого выбираем номер задания $j_l^{ks}\in\overline{\Omega}^{ks}$, которое устанавливается на $(N^{ks}+1)$-е место в последовательности $\overline{\Omega}^{ks}$;

– полагаем $s:=(s+1)$, корректируем подмножества $\widetilde{V}^{ks}$, $\overline{\Omega}^{ks}$, $\overline{M}^s$, значения $\overline{T}\{\widetilde{V}^{ks}\}$, m_1^s и переходим к выполнению п. 1 большой итерации алгоритма.

Если в процессе выполнения достаточно большого количества итераций $S \to \infty$ алгоритма не получено ни одного допустимого решения задачи, то на основании этого делается заключение, что система ограничений задачи несовместна. В противном случае наилучшее из полученных решений $F_{\text{opt}}(\tilde{P})$ принимается в качестве оптимального решения задачи.

Синхронизация работы сборочных конвейеров

Пусть последовательность технологических операций сборки задана некоторым графом. Вершины графа $i = 1, \ldots, n$ определяют технологические операции, длительность которых равна t_i, где t_i – целые числа, а дуги определяют последовательность их выполнения.

Рассматриваемая задача заключается в разбиении всего множества выполняемых операций $\widetilde{I}$ на m $(m < n)$ непересекающихся подмножеств $\widetilde{I}_1, \widetilde{I}_2, ..., \widetilde{I}_m$ (здесь m – количество рабочих станций) таким образом, что

$$\bigcup_{k=1}^{m} \widetilde{I}_k = \widetilde{I}\ ,\ \widetilde{I}_k \cap \widetilde{I}_p = \varnothing\ ,\ k, p = 1, ..., m\ , \quad (11)$$

а также в определении последовательности выполнения операций на каждом рабочем посту и порядка расстановки рабочих постов, обеспечивающих допустимую последовательность выполнения операций сборки и экстремальное значение некоторого критерия оптимальности. В качестве критерия оптимальности рассмотрим минимизацию времени такта или цикла сборки (максимальной производительности) при заданном количестве рабочих станций.

Время такта сборки θ, т.е. промежуток времени между завершением сборки r-го и $(r+1)$-го изделия или обратная ему величина – производительность сборочного конвейера (количество изделий, выпущенных линией за время T), определяется максимальной длительностью выполнения всех технологических операций на наиболее напряженном рабочем посту конвейерной линии

$$\theta = \max_{1 \le k \le m} \sum_{i \in \widetilde{I}_k} t_i + \tau\ , \quad (12)$$

где τ – время транспортировки собираемого изделия с одного поста на другой; $(\tau_k = \tau_{k+1} = \tau$, $k = 1, \ldots, m-1)$, $\widetilde{I}_k$ – подмножество операций выполняемых на k-м сборочном посту.

Для исследования свойств оптимальных решений и описания алгоритмов решения задачи введем следующие обозначения:

$\widetilde{U}$ – множество всех дуг графа; $A(i)$ (непосредственно после i) – множество всех дуг $j \in \widetilde{I}$, для которых $(i, j) \in \widetilde{U}$; $B(i)$ (непосредственно перед i) – множество всех дуг $j \in \widetilde{I}$, для которых $(j, i) \in \widetilde{U}$,

$$A(i) = \{ j \in \widetilde{I} \mid (i, j) \in \widetilde{U} \}\ ,$$
$$B(i) = \{ j \in \widetilde{I} \mid (j, i) \in \widetilde{U} \}\ . \quad (13)$$

Определим также следующие показатели:

$\overline{A}(i)$ – множество операций, которые могут выполняться только после завершения операции i;

$\overline{B}(i)$ – множество операций, которые должны быть выполнены перед началом операции i.

Нижняя граница величины такта работы конвейерной линии определяется выражением

$$\xi(\theta) = \max\left\{ \left| \frac{1}{m} \cdot \sum_{i=1}^{n} t_i \right|;\ \max_{1 \le i \le n} t_i \right\} + \tau\ . \quad (14)$$

Здесь и в дальнейшем $|\cdot|$ – целая часть частного от деления двух величин.

Пусть на некотором этапе решения определено назначение и последовательность выполнения подмножества $\widetilde{I}^1 \subset \widetilde{I}$ – операция на первых $m_1 < m$ рабочих станциях, т.е. определены подмножества операций $\widetilde{I}_k$, $k = 1, ..., m_1$ и вычислено значение

$$T(\widetilde{I}^1) = \max_{1 \le k \le m_1} \sum_{i \in \widetilde{I}_k} t_i\ . \quad (15)$$

Необходимо распределить оставшееся подмножество $\hat{I}^2 = \widetilde{I} / \widetilde{I}^1$ операций на последующих m_2, $k = m_1 + 1,\ m_1 + 2, \ldots, m = m_1 + m_2$ рабочих постах. В данном случае нижняя граница величины такта работы конвейерной линии определяется по формуле

$$\xi(\theta \mid \widetilde{I}^1) = \max\left\{ \left| \frac{1}{m_2} \cdot \sum_{i \in \hat{I}^2} t_i \right|;\ \max_{i \in \hat{I}^2} t_i;\ T(\widetilde{I}^1) \right\} + \tau\ . \quad (16)$$

Рассматривается алгоритм решения задачи методами глобального случайного поиска, использующими установленные формулами (14)–(16) оценки.

Алгоритм решения состоит из некоторого количества $p = 1, \ldots, P$ больших итераций, в каждой из которых S ($s = 1, \ldots, S$) шагов.

Введем следующие обозначения:

$s = 1, \ldots, S$ – номер шага итеративного процесса;

$\tilde{I}^{1s}$, $\hat{I}^{2s}$ – соответственно множество операций, выполнение которых на s-м шаге назначено и не назначено ни на одной из рабочих станций;

$\tilde{I}_k^s$ – множество операций, выполнение которых на s-м шаге итеративного процесса назначено на k-й рабочей станции;

$\tilde{U}^s \subseteq \hat{I}^{2s}$ – множество операций на s-м шаге, не имеющих предшественников,

$$\tilde{U}^s = \left\{ i \in \hat{I}^{2s} \mid B(i) = \varnothing \right\}; \qquad (17)$$

$T_k^s = \sum_{i \in \tilde{I}_k^s} t_i$, $k = 1, \ldots, m_1$ – суммарное время выполнения всех операций на k-й рабочей станции на s-м шаге алгоритма;

$\sigma_k^s = \xi(\theta \mid \tilde{I}^1) - \sum_{i \in \tilde{I}_k^s} t_i - \tau$ – свободный ресурс времени k-й рабочей станции на s-м шаге итеративного процесса;

$k(s)$ – номер рабочей станции, рассматриваемой на s-м шаге итеративного процесса.

$\overline{U}_{k1}^s$ – подмножество операций, выполнение которых на s-м шаге может быть назначено на k-й рабочей станции

$$\overline{U}_{k1}^s = \{ i \in \tilde{U}^s \mid t_i \le \sigma_k^s \}. \qquad (18)$$

Пусть $\theta(P_1)$ – наилучшее из допустимых решений, полученных в результате выполнения $P_1 \le P$ больших итераций.

В начале выполнения каждой p-й большой итерации положим

$$s = 1,\ \hat{I}^{2s} = \tilde{I}^s = \{1, \ldots, n\};\ \tilde{I}_k^s = \varnothing,\ \sigma_k^s = \theta,$$

$$k = 1, \ldots, m;\ k(s) = 1,\ \tilde{U}^s = \{ i \in \tilde{I} \mid B(i) = \varnothing \}. \qquad (19)$$

Значение $\theta(P_1) = \theta$ и вычисляем по формуле (12).

Ш а г 1. 1) Для рабочей станции $k(s)$ определяем в соответствии с выражениями (17) и (18) подмножества операций $\tilde{U}^s$ и $\overline{U}_{k1}^s$. Пусть $m_{k1}^s = \xi(\overline{U}_{k1}^s)$ – количество элементов в множестве $\overline{U}_{k1}^s$.

2) Разобьем весь диапазон изменения $D \in [0; 1{,}0]$ на m_{k1}^s равномерных интервалов, длиной $1/m_{k1}^s$ каждый. Упорядочим все элементы подмножества $\overline{U}_{k1}^s$, например, в последовательность $\tilde{V}_{k1}^s$ по возрастанию величин t_j. Реализуем случайное число, равномерно распределенное в диапазоне $d \in [0; 1{,}0]$. Выбираем j-й элемент последовательности $\tilde{V}_{k1}^s$ в соответствии с тем, в какой из рассматриваемых интервалов попало это случайное число.

Для выбранной операции $j \in \overline{U}_{k1}^s$, время выполнения которой равно t_j, определяем

$$\hat{I}^{2s} = \{ \hat{I}^{2s} / j \},\ \tilde{I}^{1s} = \{ \tilde{I}^{1s} \cup j \},$$

$$T_k^s = T_k^s + t_j,\ \sigma_k^s = \sigma_k^s - t_j. \qquad (20)$$

Определяем подмножество $\overline{U}_{k1}^s$ в соответствии с выражением (18). Если $\overline{U}_{k1}^s \ne \varnothing$, вновь выполняем шаг 1. В противном случае переходим к шагу 2.

Ш а г 2. Полагаем $s := s + 1$, $k := k + 1$, а также $m_1 := m_1 + 1$, $m_2 := m_2 - 1$, вычислим значение $\xi(\theta \mid \tilde{I}^1)$ по формуле (16). В дальнейшем выполняем следующий объем вычислений:

1) Если $s = S$, если $\xi(\theta \mid \tilde{I}^1) < \xi(P_1)$, то полагаем $\xi(P_1) = \xi(\theta \mid \tilde{I}^1)$. Запоминаем все подмножества операций $\tilde{I}_k^s(P_1)$ $k, s = 1, \ldots, m$ и соответствующие им последовательности выполнения на каждой машине $\overline{W}_k^s(P_1)$, удовлетворяющие условиям технологической последовательности выполнения операций. Определяем $P_1 := P_1 + 1$. Если

$$\frac{\xi(P_1) - \theta}{\theta} \le \varepsilon \ \text{ либо } P_1 = P, \qquad (21)$$

где ε – установленная точность решения задачи, то алгоритм завершает работу. Множества

$\widetilde{I}_k^s(P_1)$ и последовательности $\overline{W}_k^s(P_1)$ и есть решение задачи. Если одно из условий (21) не выполняется, то, приняв значения всех величин согласно выражению (19), вновь переходим к выполнению шага 1.

2) Если $s < S$, то переходим к шагу 1.

Автором выполнен вычислительный эксперимент, в процессе которого решались задачи размерностью $n = 100 - 250$ операций и $m = 8 - 25$ рабочих станций. Матрица смежности графа последовательности технологических операций содержала 15–20 процентов единиц. Время выполнения технологических операций в диапазоне $t_i \in [3, 10]$ выбиралось случайным образом. Было решено более 70 задач, исходные данные которых выбирались случайным образом. В результате выполненного вычислительного эксперимента установлено, что за количество больших итераций, равное $P \leq 300$, как правило, получено решение, значение критерия оптимальности которого отличалось от нижней границы функции цели, вычисленной по формуле (12), не более чем на 8–13,5 процентов.

В работах автора [3, 5] приведены алгоритмы получения точных решений этой задачи последовательными алгоритмами оптимизации.

Минимизация суммы штрафов, связанных со сроками завершения выполнения заданий на одной машине

Пусть заданы время начала выполнения θ_i, а также время выполнения t_i и произвольные нелинейные функции штрафов, связанных со сроками завершения выполнения каждого из заданий $f_i(T_i)$, $i = 1, \ldots, n$. Если определена некоторая последовательность выполнения заданий $\overline{U} = \{u_1, u_2, ..., u_{i-1}, u_i, ..., u_n\}$, то время выполнения задания u_i определяется по формуле

$$T_i = \max\left[T_{i-1} + 1,\ \theta_i\right] + t_i,\ i = 1, ..., n. \qquad (22)$$

В качестве примеров функций $f_i(T_i)$ могут встречаться нелинейные функции вида

$$f_i(T_i) = \varphi_i(T_i),\ f_i(T_i) = \begin{cases} 0, & \text{если } T_i \leq B_i, \\ \varphi_i(T_i) & \text{если } T_i > B_i, \end{cases}$$

$$f_i(T_i) = \begin{cases} 0, & \text{если } T_i \leq B_{1i}, \\ c_{1i}, & \text{если } B_{1i} < T_i \leq B_{2i}, \\ & \ldots\ldots \\ c_{ri}, & \text{если } B_{r,i} < T_i, \end{cases} \qquad (23)$$

где $\varphi_i(T_i)$ – произвольные нелинейные функции, которые могут быть недифференцируемыми и разрывными. На сроки выполнения некоторых заданий могут быть наложены ограничения вида $T_i \leq b_i$. Необходимо найти последовательность выполнения заданий, минимизирующую сумму штрафов

$$F(\overline{U}) = \sum_{i=1}^{n} f_i(T_i) \to \min. \qquad (24)$$

В случае нелинейных функций штрафов не существует простого решающего правила, приведенного например в [3, 4, 6], позволяющего построить оптимальную последовательность выполнения заданий за полиномиальное время, и рассматриваемая задача относится к классу *NP*-полных задач. Для ее решения могут быть использованы описанные методы глобального случайного поиска.

Алгоритм решения состоит из некоторого количества $p = 1, \ldots, P$ больших итераций, каждая из которых состоит из S $(s = 1, \ldots, S)$ шагов. Пусть $F(P_1)$ – наилучшее из допустимых решений, полученных в результате выполнения $P_1 \leq P$ больших итераций.

Введем следующие обозначения:

$s = 1, \ldots, S$ – номер шага итеративного процесса; $\tilde{I}^{1s}$, $\hat{I}^{2s}$ – соответственно множество заданий, выполнение которых на s-м шаге назначено и не назначено на данной машине; $\overline{U}^{1s}$ – последовательность выполнения заданий подмножества $\widetilde{I}^{1s}$; T_{s-1} – время завершения всех операций подмножества $\widetilde{I}^{1s}$ в последовательности их выполнения $\overline{U}^{1s}$; $\overline{R}^s$ – наиболее раннее время начала выполнения операций подмножества $\hat{I}^{2s}$, которое вычисляется по формуле

$$\overline{R}^s = \max\left[T_{s-1} + 1,\ \max_{i \in \hat{I}^{2s}} \theta_i\right]; \qquad (25)$$

$F(\widetilde{I}^{1s}) = \sum_{i \in \widetilde{I}^{1s}} f_i(T_i)$ – сумма штрафов, связанная с завершением подмножества операций $i \in \widetilde{I}^{1s}$ в рассчитанные сроки T_i.

Упорядочим все задания $i \in \hat{I}^{2s}$ в последовательность $\overline{U}^{2s}$ по невозрастанию значений b_i

$$\overline{U}^{2s} = \left\{ u_1^s, u_2^s, ..., u_p^s, ..., u_R^s \mid u_p^s \in \hat{I}^{2s}, \right. \\ \left. r = 1,...R; \quad b(u_{r-1}^s) \le b(u_r^s) \right\} . \tag{26}$$

Если выполняется хотя бы одно из системы неравенств [3]

$$T(u_r^s) = \max\left[T(u_{r-1}^s) + 1, \theta(u_r^s) \right] + t(u_p^s) > b(u_r^s), \\ r = 1,...,R, \tag{27}$$

где $T(u_0^s) = \overline{R}^s$, $T(u_{r-1}^s) = \overline{R}^s + \sum_{l=1}^{r-1} t(u_l^s)$, R – количество элементов в последовательности $\overline{U}^{2s}$, то подпоследовательность выполнения заданий $\overline{U}^{1s}$ не имеет допустимых решений и может быть отброшена как неперспективная.

В начале выполнения каждой p-й большой итерации положим $F(P_1) = \infty$, а также

$$s = 1, \ \hat{I}^{2s} = \widetilde{I} = \{1,...,n\}; \\ \widetilde{I}^{1s} = \overline{U}^{1s} = \varnothing, \ F(\widetilde{I}^{1s}) = 0 . \tag{28}$$

Ш а г 1. Определим по формулам (26) последовательность $\overline{U}^{2s}$ и проверим выполнение неравенств (27). Если хотя бы для одного значения индекса $r = 1, \dots, R$ не выполняется неравенство (27), то $\overline{U}^{1s}$ не имеет допустимых решений. Если $s = 1$ и $p = 1$, то система ограничений несовместна и алгоритм с соответствующим сообщением завершает работу. Если $s > 1$, то устанавливаем (28), $p := (p+1)$. Если $p = P$, то в случае $F(P_1) < \infty$, алгоритм заканчивает работу, и последовательность $\overline{U}(P_1)$ – решение задачи, а $F(P_1)$– соответствующее этому решению значение критерия оптимальности. В случае если $F(P_1) <= \infty$, то алгоритм завершает работу с сообщением, что допустимых решений не получено. Если $p =< P$ и $s > 1$, то переходим ко второму шагу алгоритма.

Ш а г 2. Определим $t_l^s = \min_{i \in \widetilde{I}^{1s}} \left[\max\left(\overline{R}^s, \theta_i \right) + t_i \right]$,

$$\hat{J}^{2s} = \left\{ i \in \hat{I}^{2s} \mid \theta_i - t_l^s - 1 \le 0 \right\}, \ \hat{J}^{2s} \subseteq \hat{I}^{2s} . \tag{29}$$

Если $\hat{J}^{2s} \ne \varnothing$, то построим последовательность заданий

$$\overline{V}^{2s} = \left\{ v_1^s, v_2^s, ..., v_l^s, ..., v_L^s \mid u_l^s \in \hat{J}^{2s}, \right. \\ \left. L = 1,...,L \le R; \quad b(v_{l-1}^s) \le b(v_l^s) \right\}, \tag{30}$$

где L – количество элементов в множестве $\hat{J}^{2s}$ и последовательности $\overline{V}^{2s}$.

Разобьем весь диапазон изменения $D \in [0; 1,0]$ на L интервалов, длина каждого из которых d_l пропорциональна величине $1/b(v_l^s)$, т.е. чем меньше граничное время завершения задания, тем длина интервала больше. Реализуем случайное число равномерно распределенное в диапазоне $D \in [0; 1,0]$. Выбираем тот k-й элемент последовательности в соответствии с тем, в какой из рассматриваемых интервалов попало это случайное число.

Пусть это будет задание с индексом $i_k \in \hat{J}^{2s}$. Переходим к шагу 3.

Ш а г 3. Полагаем $\widetilde{I}^{1s} = \widetilde{I}^{1s} \bigcup i_k$, $\hat{I}^{2s} = \hat{I}^{2s} / i_k$. Включаем элемент i_k последним членом строящейся последовательности $\overline{U}^{1s}$. Вычисляем значения

$$T_{i_k} = \max\left(R^s, \theta_{i_k} \right) + t_{i_k} - 1, \\ F(\widetilde{I}^{1,s+1}) = F(\widetilde{I}^{1s}) + f_{i_k}(T_{i_k}) . \tag{31}$$

Полагаем $s := (s+1)$. Если $s := S+1$, т.е. $\widetilde{I}^{1s} = \widetilde{I}$ и $\hat{I}^{2s} = \varnothing$, то полагаем $F(p) = F(\widetilde{I}^{1,S}) = F(\widetilde{I}^{1s})$. Если $F(P) < F(P_1)$, то определяем $F(P_1) = F(p)$ и запоминаем последовательность $\overline{U}^{1s}$, т.е. $\overline{U}(P_1) = \overline{U}^{1S} = \overline{U}^{1s}$. Полагаем $p := (p+1)$.

Если $p = P$ и $F(P_1) \ne \infty$, то получено решение задачи, которое определяется расписанием выполнения заданий $\overline{U}(P_1)$ со значением критерия оптимальности $F(P_1)$.

Если $p = P$ и $F(P_1) = \infty$, то алгоритм заканчивает работу с сообщением, что допустимых решений не получено.

Если $p < P$, то полагаем $s := 1$, $\widetilde{I}^{1s} = \overline{U}^{1s} = \varnothing$, $\hat{I}^{2s} = \widetilde{I}$, $F(\widetilde{I}^{1s}) = 0$ и переходим к первому шагу алгоритма.

Заключение. Проведен вычислительный эксперимент, в процессе которого решались задачи размерностью $n = 20 - 25$ заданий с нелинейными разрывными функциями штрафов. Время выполнения заданий выбиралось случайным образом и варьировалось в диапазоне $t_i \in [5, 12]$. Для 30–50 процентов заданий устанавливались граничные сроки завершения их выполнения. Процесс решения включал 200–250 больших итераций. В процессе выполненных расчетов в большинстве случаев факт несовместности системы ограничений устанавливался уже на первой большой итерации алгоритма в результате проверки выполнения системы неравенств (27). Наилучшее решение, которое на 12–20 процентов хуже значения достаточно грубой нижней границы оптимального решения задачи, достигалось уже на 120–140 большой итерации и в дальнейшем ходе вычислений не улучшалось.

Отметим, что в монографии автора [3] приведен алгоритм решения этой задачи методом ветвей и границ, который позволяет получить точное решение задачи за экспоненциальное время.

Разработанные методы иллюстрируются при построении алгоритмов решения различных классических задач разбиения на подмножества и упорядочения, имеющих прикладное значение в проблемах календарного планирования производства, маршрутизации перевозок и организации технического и сервисного обслуживания объектов.

Описанные методы глобального случайного поиска в сочетании с различными эвристиками и оценками нижних границ оптимального решения могут успешно применяться и для решения других задач теории расписаний: *Job–Shop–Problem*, *Flow–Shop–Problem*, построение расписаний работы на параллельных машинах и др. Некоторые из этих алгоритмов описаны в [2, 3, 6].

1. *Жиглявский А.А.*, *Жилинскас А.Г.* Методы поиска глобального экстремума. – М.: Наука, 1991. – 205 с.
2. *Растригин Л.А.* Статистические методы поиска. – М.: Наука, 1968. – 376 с.
3. *Зак Ю.А.* Прикладные задачи теории расписаний и маршрутизации перевозок. – М.: Книжный дом «Либроком», 2012. – 393 с.
4. *Domschke W.*, *Scholl A.*, *Voß S.* Produktionsplanung. Ablauforganisatorische Aspekte. – Berlin, Heidelberg: Springer Verlag, 2005. – 456 p.
5. *Зак Ю.А.* Оптимальное распределение технологических операций на сборочном конвейере // Кибернетика. – 1990. – № 4. – С. 45–54.
6. *Brücker P.* Scheduling Algorithms. – Leipzig: Springer, 2007. – 371 p.

Поступила 12.08.2012
Тел. для справок: +49 241 543-255 (*Aachen*, *Deutschland*)
E-mail: *yuriy_zack@hotmail.com*
Сайт: *www.optimorum.de*

●