УДК 519.854.2

МЕТОД «НАПОЛНЕНИЯ МНОЖЕСТВ» РЕШЕНИЯ ЗАДАЧ ТЕОРИИ РАСПИСАНИЙ ДЛЯ ОДНОГО ПРИБОРА[*]

Д.И. Архипов[1], А.А. Лазарев[1,2,3,4], F. Werner[5]
[1]Институт проблем управления РАН, Москва, Россия;
[2]Московский государственный университет имени М.В. Ломоносова;
[3]НИУ «Высшая школа экономики»;
[4]Московский физико-технический институт (ГУ);
[5]Otto-von-Guericke University Magdeburg, Germany.

В работе предлагается алгоритм решения класса задач теории расписаний для одного прибора. На одном приборе необходимо обслужить множество из n требований, для каждого из которых заданы момент поступления, директивный срок и функция штрафа $\varphi_j(t)$. Кроме того, заданы интервалы доступности прибора. Время обслуживания каждого из требований зависит от момента начала обслуживания и задаётся функцией $p(t)$. Предлагается алгоритм нахождения Парето-множества расписаний для критериев C_{max} и φ_{max}. Трудоёмкость алгоритма составляет $O(n^3 \max\{\log n, P, H\})$ операций, где P и H – трудоёмкости нахождения значения $p(t)$ и оценки $\varphi_j(t)$ соответственно.

Введение

Множество требований N={1,…, n} необходимо обслужить на одном приборе. Для каждого требования $j \in N$ определены **момент поступления** r_j и **строгий директивный срок** D_j. Одновременное обслуживание двух требований запрещено. Все требования обслуживаются без прерываний. Будем называть **расписанием** π множество моментов **начала обслуживания** $S_j(\pi)$, заданных для каждого требования j, таких что $S_j(\pi) \geq r_j$ и $S_j(\pi) \geq S_k(\pi) + p(S_k(\pi))$ для любого требования k, где $S_k(\pi) < S_j(\pi)$. Определим **момент окончания обслуживания** требования j при расписании π через $C_j(\pi) = S_j(\pi) + p(S_j(\pi))$. Будем называть расписание π **допустимым**, если для любого требования $j \in N$ при расписании π выполняется неравенство $C_j(\pi) < D_j$. Множество допустимых расписаний обозначим через $\Pi(N)$. **Время обслуживания** каждого из требований $j \in N$ определяется функцией $p(t)$, значение которой может быть вычислено за P операций для любого момента начала обслуживания t. Причем $t + p(t)$ — монотонно неубывающая функция для любого t. Заметим, что при этом множество моментов начала об-

[*] Работа выполнена при поддержке грантов РФФИ 15-07-07489, 15-07-03141 и DAAD A/1400328.

служивания требований будет однозначно задавать последовательность их обслуживания.

Функции штрафа $\varphi_j(t)$, определенные для каждого из требований $j \in N$ монотонно не убывают и для любого значения y, для нахождения момента времени t', такого, что $t' = \min\{t \,|\varphi_j(t) \geq y\}$ требуется не более чем H операций.

В соответствии с обозначениями, принятыми в теории расписаний и предложенными в статье [1], данная задача может быть обозначена как $1|r_j, p_j = p(t), D_j|\varphi_{max}$. Для её решения требуется найти расписание $\pi \in \Pi(N)$, удовлетворяющее критерию
$\min_{\pi \in \Pi(N)} \max_{j \in N} \varphi_j(C_j(\pi))$.
Данная обобщённая постановка включает в себя следующие задачи теории расписаний для одного прибора:

Задача	*Значение* $\varphi_j\ (t)$
$1|r_j, p_j = p(t), D_j|C_{max}$	t
$1|r_j, p_j = p(t)|L_{max}$	$t - d_j$
$1|r_j, p_j = p(t)|T_{max}$	$\max\{0, t - d_j\}$
$1|r_j, p_j = p(t), w_j \geq 0|wL_{max}\}$	$w_j(t - d_j)$
$1|r_j, p_j = p(t), w_j \geq 0|wT_{max}\}$	$\max\{0, w_j(t - d_j)\}$

где w_j — вес (ценность) требования j.

Заметим, что с помощью функции $p(t)$ могут быть заданы интервалы доступности прибора.

В данной работе предлагается алгоритм построения Парето-множества расписаний для решения задачи $1|r_j, p_j = p(t), D_j|\varphi_{max}, C_{max}$.

Задача минимизации общего времени выполнения требований с одинаковыми временами обслуживания $1|r_j, p_j = p, D_j|C_{max}$ была рассмотрена в работах [2] и [3]. Подробный обзор задач с одинаковыми временами обслуживания представлен в [4]. Алгоритм линейного программирования для задачи $P|r_j, p_j = p, D_j|\ \varphi_{\max}$, где $\varphi_j(t)$ — неубывающая функция, представлен в [5]. Решение двукритериальной задачи $1|r_j, p_j = p\ |L_{max}, C_{max}$ представлено в [6].

Вспомогательная задача

Пусть $O_1(\pi), \ldots, O_n(\pi)$ — последовательность требований, в которой они выполняются при расписании π, и $O(j, \pi)$ — порядковый номер требования $j \in N$ при расписании π, т.е., $O(j, \pi) = i \iff O_i(\pi) = j$.

Определим семейство множеств $F = \{N_0, N_1, \dots, N_n\}$ такое, что для любого $i = 0,1, \dots, n$, во множество N_i входят требования, порядковый номер обслуживания которых не превосходит i. Данное семейство множеств удовлетворяет свойству $N_0 \subseteq N_1 \subseteq \dots \subseteq N_n$. Для любого $i = 0, \dots, n$, определим дополнение к множеству N_i как $\overline{N}_i = N \setminus N_i$.

Будем говорить, что расписание π **удовлетворяет** семейству множеств $F = \{N_0, N_1, \dots, N_n\}$, если для любого $i = 1, \dots, n$ и для каждого требования $j \in N_i$ выполняется неравенство $O(j, \pi) \leq i$. Таким образом, для любого $i = 1, \dots, n$, только требование из множества $\overline{N}_{i-1}$ могут обслуживаться под порядковым номером i, т.е., $O_i(\pi) \in \overline{N}_{i-1}$. Очевидно, что $N_0 = \emptyset$, $N_n = N$ и для любого $i = 0, \dots, n$, количество требований, принадлежащих множеству N_i, не превосходит i, т.е., $|N_i| \leq i$.

Для любого $j \in N$ определим $N(j, F) = i$, если $j \in N_i$ и $j \notin N_{i-1}$, т.е., $N(j, F) = \min_{i=0,\dots,n} \{i \mid j \in N_i\}$.

Пусть $\Phi(N, F, y) \subseteq \Pi(N)$ — множество расписаний π, удовлетворяющих семейству F, и для каждого требования $j \in N$, выполнено неравенство $\varphi_j(\pi) < y$. Заметим, что для семейства $F^0 = \{\emptyset, \dots, \emptyset, N\}$, множество $\Phi(N, F^0, +\infty)$ совпадает с множеством допустимых расписаний $\Pi(N)$.

Сформулируем вспомогательную задачу и алгоритм её решения.

Вспомогательная задача. Найти расписание $\pi(F, y) \in \Phi(N, F, y)$ удовлетворяющее критерию
$\min_{\pi \in \Pi(N)} \max_{j \in N} \{ C_j(\pi) | \varphi_j(\pi) < y, C_j(\pi) < D_j \}$.

Алгоритм 1

1. Входные данные: $N, F, N(1, F), \dots, N(n, F), y$.
2. Выполняем операции:
   a) $m := 0$;
   b) для каждого $j \in N$ $D_j(y) := \min\{\min\{t \,|\, \varphi_j(t) \geq y\}, D_j\}$;
   c) для всех $i = 0, \dots, n$ присваиваем $N_i^m := N_i$.
3. Последовательно назначаем требования на порядковые номера обслуживания требований $i = n, \dots, 1$ по правилу наибольшего момента поступления с учётом семейства множеств F:
   $O_i(\pi^m) := \arg\max_{j \in \overline{N}_{i-1}^m} r_j$.
4. Устанавливаем моменты начала обслуживания по формулам:
   $S_{O_1(\pi^m)}(\pi^m) := r_{O_1(\pi^m)}$;
   $S_{O_i(\pi^m)}(\pi^m) := \max\{r_{O_i(\pi^m)}, S_{O_{i-1}(\pi^m)}(\pi^m) + p(S_{O_{i-1}(\pi^m)}(\pi^m)\}$,
   для значений $i = 2, \dots, n$.
5. Для требований $j \in N$ и порядковых номеров $i = N(j, F^m), \dots, 0$, проверяем неравенства

$C_{O_i(\pi^m)}(\pi^m) \geq D_j(y)$.

a) Если для некоторой пары j, i, неравенство не выполняется, включаем требование j во множество N_{i-1}^m, изменяем $N(j, F^m) := i - 1$, и продолжаем проверку для других значений j, i.

b) Если неравенство верно и $j \neq n$, то продолжаем проверку.

c) Если $j = n$ и для всех пар j, i, где $i = O(j, \pi)$ неравенства выполняются, то **return{π^m}**.

d) Если $j = n$ и для какой-то пары j, i, где $i = O(j, \pi)$ неравенство неверно, то переходим на шаг 6.

6. Для всех множеств $N_i^m, i = 0, \dots, n$, проверяем мощности $|N_i^m| \leq i$.

   a) Если для всех множеств неравенство верно, присваиваем $m := m + 1$, и $N_i^m := N_i^{m-1}$ для всех $i = 0, 1, \dots, n$, а затем переходим на новую итерацию (шаг 3).

   b) Иначе, **return{$\emptyset$}**.

Алгоритм решения основной задачи

Теперь представим алгоритм построения Парето-множества расписаний для решения основной задачи.

Алгоритм 2

1. Входные данные: $N, \Omega(N) = \emptyset, y = +\infty$.
2. Присваиваем $s := 0, N_n^s := N$ и для всех $i = 0, \dots, n-1$: $N_i^s := \emptyset$. Для каждого требования $j \in N$ получаем $N(j, F^s) = N$.
3. Выполняем построения расписания $\pi_{s+1} = \pi(F^s, y)$ по алгоритму 1. После выполнения алгоритма 1 получено семейство множеств $F^{s+1} = \{N_0^{s+1}, N_1^{s+1}, \dots, N_n^{s+1}\}$.
4. Если $\pi_{s+1} \neq \emptyset$:
   a) присваиваем $\pi^*(N) := \pi_{s+1}, y := max_{j \in N} \varphi_j(\pi_{s+1})$;
   b) Если $C_{max}(\pi_s) < C_{max}(\pi_{s+1})$, добавляем π_{s+1} во множество $\Omega(N)$, увеличиваем $s := s + 1$ и переходим на новую итерацию (шаг 3).
   c) Если $C_{max}(\pi_s) = C_{max}(\pi_{s+1})$, заменяем π_s на π_{s+1} во множестве $\Omega(N)$, присваиваем $F^s := F^{s+1}$ и переходим на новую итерацию (шаг 3).
5. Если $\pi_{s+1} = \emptyset$, $\boldsymbol{return}\ \{\boldsymbol{\Omega(N)}, \boldsymbol{\pi^*(N)}\}$.

Лемма 1. *Мощность множества* $\Omega(N)$ *не превышает* $\frac{n(n-1)}{2}$.

Теорема 1*.* *Трудоёмкость алгоритма 2 составляет* $O(n^3 \max\{\log n, H, P\})$ *операций, где* n *— количество требований,* H *— трудоёмкость вычисления момента времени* t' *такого, что* $t' = \min\{t \mid \varphi_j(t) \geq y\}$*, и* P *— трудоёмкость вычисления* $p(t)$*.*

Теорема 2*.* *Множество расписаний* $\Omega(N)$*, полученное в результате выполнения алгоритма 2 является Парето-оптимальным по критериям* φ_{max} *и* C_{max}*, а расписание* $\pi^*(N)$ *оптимально по критерию* $\min_{\pi\in\Pi(N)} \max_{j\in N} \varphi_j(C_j(\pi))$*.*

Если $\Omega(N) = \emptyset$*, то допустимых расписаний не существует т.е.,* $\Pi(N) = \emptyset$*.*

Заключение

В данной работе представлен метод «наполнения множеств» для решения однокритериальных и двукритериальных минимаксных оптимизационных задач построения расписания для одного прибора. Основная идея данного подхода заключается в итерационном построении расписаний. На каждой итерации строится расписание со значением функции штрафа строго меньше, чем на предыдущем шаге с учётом свойств, полученных на предыдущей итерации, выражающихся в постепенном заполнении семейства контрольных множеств.

Список литературы

1. R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. // *Ann. Discrete Math.* V.5. 287-326 pp., 1979
2. B. Simons. A fast algorithm for single processor scheduling. // *Ann. Arbor Mich.*, 246 - 252 pp., 1978
3. M.R. Garey, D.S. Johnson, B.B. Simons and R.E. Tarjan. Scheduling unit-time tasks with arbitrary release times and deadlines. // *SIAM Journal on Computing*, Vol. 10, No. 2, 256 - 269 pp., 1981.
4. S. Kravchenko and F. Werner. Parallel machine problems with equal processing times: a survey. // *Journal of Scheduling*, Vol. 14, No. 5, 435 - 444 pp., 2011.
5. S. Kravchenko and F. Werner. On a parallel machine scheduling problem with equal processing times. // *Otto-von-Guericke-Universitat Magdeburg, FMA,* Preprint 26/07, 9 pp., 2007.
6. Alexander A. Lazarev, Dmitry I. Arkhipov, Frank Werner. Scheduling jobs with equal processing times on a single machine: minimizing maximum lateness and makespan. // Optimization Letters, 2016, DOI: 10.1007/s11590-016-1003-y.