

**Гончар Д.Р.**

## **Комбинированный алгоритм планирования групп вычислительных работ на основе метода ветвей и границ**

**Аннотация:** Рассматривается минимаксная задача составления расписания минимальной длины без прерываний для многопроцессорной системы. Для решения данной задачи предложен параллельный алгоритм на основе сочетания метода ветвей и границ ограниченной глубины с эвристикой и алгоритмы для проверки качества полученного решения.

**Ключевые слова:** многопроцессорная система, работы без прерываний, расписание минимальной длины

### **Введение**

Задачи построения оптимальных расписаний весьма широко встречаются на практике при планировании производственной деятельности, управлении энергетическими объектами, подготовке и проведении испытаний сложных технических систем (например, авиационных), работе различных систем отслеживания (в т.ч. экологических и медицинских систем) и множества других. Различные области применения в ряде случаев обуславливают и разные требования к точности и скорости получения искомого решения, адекватное удовлетворение которых позволяет повысить безопасность управляемой (отслеживаемой) системы в целом.

Предложенная версия алгоритма сочетает использование подхода метода ветвей и границ для обхода заданной части дерева решений с назначением части оставшихся заданий согласно эвристическому алгоритму, что дает дополнительные возможности при выборе наиболее подходящего к условиям решения данной задачи пути решения.

### **1. Постановка задачи**

Пусть имеется множество работ  $N = \{1, 2, \dots, n\}$ , которое необходимо выполнить с помощью  $m$  процессоров, составляющих вычислительную систему для их обработки. Длительность выполнения работы  $i$  на процессоре  $j$  равно  $t_{ij}$  ( $i = 1, 2, \dots, n; j = 1, 2, \dots, m$ ). Переключения с одного процессора на другой и прерывания при выполнении работ не допускаются. В каждый

момент времени каждая работа может выполняться не более чем одним процессором, а каждый процессор может выполнять не более одной работы.

Расписание выполнения работ  $N$  определим как разбиение множества  $N$  на  $m$  непересекающихся подмножеств  $N_1, N_2, \dots, N_m$

( $N = \bigcup_{j=1}^m N_j$ ;  $N_{j_1} \cap N_{j_2} = \emptyset$  при  $j_1 \neq j_2$ ). Работы из множества  $N_j$

приписываются процессору  $j$  и выполняются на нем одна за другой в произвольном порядке. Под загруженностью процессора  $j$  ( $j = 1, 2, \dots, m$ ) будем понимать величину  $Q_j = \sum_{i \in N_j} t_{ij}$ , а  $\max_{j=1, 2, \dots, m} Q_j$  –

это длина расписания. Задача заключается в построении оптимального по быстродействию расписания, т.е. расписания минимальной длины.

Подобные задачи широко освещены в литературе. При их решении применяются, например, такие методы, как случайный и исчерпывающий поиск, методы математического программирования [1], метод ветвей и границ [2, 6], муравьиные алгоритмы, поиск с запретами, вероятностные алгоритмы, генетические алгоритмы [3], метод имитации отжига, различные эвристические алгоритмы [4, 5], алгоритмы агрегирования и др.

## 2. Метод ветвей и границ

Для решения поставленной задачи предлагается метод ветвей и границ, основанный на результатах работы [2].

### 2.1. Ветвление

Множество всех расписаний (их число равно  $m^n$ ) будем описывать в виде дерева расписаний. На нулевом уровне дерева находится корень, который соответствует множеству всех расписаний. На первом уровне находится  $m$  вершин, каждая из которых соответствует множеству всех расписаний, в которых первая работа назначена на определенный процессор. На втором уровне дерева находится  $m^2$  вершин, каждая из которых соответствует множеству всех расписаний, в которых первые две работы назначены на один или два определенных процессора. На  $n$ -м уровне дерева расписаний находится  $m^n$  листьев, каждый из

которых соответствует некоторому расписанию выполнения множества работ  $N$ .

Пусть  $x_k$  – некоторый узел уровня  $k$  дерева расписаний,  $R(x_k)$  – множество всех расписаний, соответствующих этому узлу (т.е. множество расписаний, в которых работы  $1, 2, \dots, k$  назначены на определенные процессоры),  $x_{k+1}^j$  – узел уровня  $k+1$  ( $k < n$ ), связанный с узлом  $x_k$  ребром, соответствующим процессору  $j$ . Наша цель – вычисление нижней и верхней оценок минимальной длины расписания на множестве  $R(x_k)$ . Имея эти оценки, можно применить стандартную схему метода ветвей и границ [6] (например, одностороннего или фронтального ветвления).

## 2.2. Нижняя оценка

Пусть  $T_j$  ( $j = 1, \dots, m$ ) – загруженность процессора  $j$  после назначения первых  $k$  работ (т.е.  $T_j$  – это суммарная длительность работ из числа  $1, 2, \dots, k$ , назначенных на процессор  $j$ ). Нижнюю оценку  $L(x_k)$  минимальной длины расписания на множестве  $R(x_k)$  будем вычислять следующим образом:

$$L(x_k) = \max(L_1(x_k), L_2(x_k), L_3(x_k)),$$

где  $L_1(x_k)$ ,  $L_2(x_k)$ ,  $L_3(x_k)$  – это нижние оценки, вычисленные тремя различными способами.

Величина  $L_1(x_k)$  вычисляется как следующий максимум:  $L_1(x_k) = \max_{j=1,2,\dots,m} T_j$ . При хранении величины  $T_1, T_2, \dots, T_m$  в виде

обычного массива сложность вычисления  $L_1(x_k)$  составляет  $\theta(m)$ .

Величина  $L_2(x_k)$  вычисляется как следующий максимум:

$$L_2(x_k) = \max_{i=k+1,\dots,n} \min_{j=1,\dots,m} (T_j + t_{ij})$$

При использовании для этого двумерного массива  $A$  с элементами  $a_{ij} = T_j + t_{ij}$ ,  $i = k+1, \dots, n$ ;  $j = 1, 2, \dots, m$  сложность вычисления величины  $L_2(x_k)$  составляет  $\theta(mn)$ .

Величина  $L_3(x_k)$  вычисляется по формуле.

$$L_3(x_k) = \frac{1}{m} \left( \sum_{j=1}^m T_j + \sum_{i=k+1}^n \min_{j=1,\dots,m} t_{ij} \right)$$

Величину  $\min_{j=1,\dots,m} t_{ij}$  вычислим сразу для всех  $i = 1, 2, \dots, n$  до начала вычисления нижних оценок. Тогда сложность вычисления

величины  $L_3(x_k)$  составляет  $O(n + m)$ . Перейдем в дереве расписаний от узла  $x_k$  к узлу  $x_{k+1}^{j_0}$ ,  $k < n$  (т.е. будем считать, что работа  $k+1$  назначена на процессор  $j_0$ ).

Тогда

$$L_3(x_{k+1}^{j_0}) = \frac{1}{m} \left( \sum_{j=1}^m T_j + t_{k+1, j_0} \right) + \sum_{i=k+1}^n \min_{j=1, \dots, m} t_{ij}$$

Вычислим разность

$$L_3(x_{k+1}^{j_0}) - L_3(x_k) = \frac{1}{m} (t_{k+1, j_0} - \min_{j=1, \dots, m} t_{k+1, j})$$

Таким образом,

$$L_3(x_{k+1}^{j_0}) = L_3(x_k) + \frac{1}{m} (t_{k+1, j_0} - \min_{j=1, \dots, m} t_{k+1, j})$$

и с помощью данного рекуррентного соотношения, используя  $L_3(x_k)$ , величина  $L_3(x_{k+1}^{j_0})$  вычисляется за время  $O(1)$ .

### 2.3. Верхняя оценка

В качестве верхней оценки  $H(x_k)$  минимальной длины расписания на множестве  $R(x_k)$  возьмем длину расписания, в котором работы  $1, 2, \dots, k$  назначены на процессоры в соответствии с вершиной  $x_k$  дерева расписаний, а работы  $k+1, \dots, n$  назначаются по следующему «жадному» алгоритму. Пусть уже назначены работы  $1, 2, \dots, p$  ( $k \leq p < n$ ),  $T_j$  – загруженность процессора  $j$  ( $j = 1, 2, \dots, m$ ) и  $\min(T_1 + t_{p+1, 1}, \dots, T_m + t_{p+1, m}) = T_{j_0} + t_{p+1, j_0}$ . Тогда работа  $p+1$  назначается на процессор  $j_0$ . Указанная процедура повторяется для  $p = k, k + 1, \dots, n - 1$ . Сложность процедуры вычисления величины  $H(x_k)$  составляет  $O(mn)$ .

## 3. Распараллеливание обхода дерева в методе ветвей и границ при реализации алгоритма в многопроцессорной системе

Дерево решения исходной задачи при применении метода ветвей и границ строится следующим образом: множества допустимых решений последовательно разбиваются на подмножества: новые подмножества на каждом следующем шаге создаются в итоге разбиения некоторых подмножеств, полученных на предыдущих шагах. Такое разбиение продолжается до тех пор,

пока для подмножеств, соответствующих концевым вершинам дерева, решение задачи уже не требует разбиения. В итоге разбиения начальная задача распадается на ряд подзадач, которые могут решаться в заметной степени независимо друг от друга.

Для оптимизации распараллеливания решения задачи применяются методы оптимизации загрузки процессов, минимизации обменов данными, а также распределения обменов по вычислительному пространству [6, 7].

При реализации метода ветвей и границ известны несколько стратегий обхода дерева, в частности, фронтальный обход, когда исследуются все узлы на каждом уровне дерева (последовательно или параллельно, в зависимости от реализации алгоритма и доступных вычислительных средств) или обход в глубину, когда на каждом следующем уровне выбирается на основе предпочтений какая-то одна из подветвей, при этом уточняются реально достижимые верхние и нижние оценки конкретного решения, пока оно не будет получено полностью. В каждом из подходов есть свои преимущества и недостатки.

Поскольку полная реализация метода ветвей и границ обладает высокими требованиями к объёму оперативной памяти и довольно трудоёмка в расчётах, для ряда задач, по мнению автора, имеет смысл применять комбинированный алгоритм, когда первые  $k$  уровней дерева решений получаемого расписания исследуются по методу ветвей и границ, а оставшиеся – на основе одного из известных эвристических алгоритмов [5, 6].

Это предложение было программно реализовано автором, с его помощью проведены расчёты, показавшие весьма высокую точность получаемых решений (по сравнению с длительностью рассчитываемой для соответствующих входных данных идеальной оценки длительности расписания).

Подробные итоги расчётов предполагается представить на конференции.

Литература:

1. *Алексеев О.Г.* Комплексное применение методов дискретной оптимизации. – М.: Наука, 1987. – 247 с.

2. *Фуругян М.Г.* Некоторые алгоритмы решения минимаксной задачи составления многопроцессорного расписания. // Изв. РАН, ТиСУ. – 2014, № 2. – С. 50–56.
3. *Костенко В.А., Смелянский Р.Л., Трекин А.Г.* Синтез структур вычислительных систем реального времени с использованием генетических алгоритмов // Программирование. – 2000. – № 5. – С. 63–72.
4. *Brucker P.* Scheduling Algorithms. Heidelberg, Springer, 2001.
5. *Гончар Д.Р.* Параллельная реализация мультиоценочного алгоритма составления многопроцессорного расписания без прерываний. // Некоторые алгоритмы планирования вычислений и методы многокритериальной оптимизации для многопроцессорных систем. – М.: ВЦ РАН, 2014. – С. 21–31.
6. *Посыпкин М.А., Сигал И.Х., Галимьянова Н.Н.* Алгоритмы параллельных вычислений для решения некоторых классов задач дискретной оптимизации. – М.: ВЦ РАН, 2005. – 44 с.
7. *Посыпкин М.А., Сигал И.Х., Галимьянова Н.Н.* Параллельные алгоритмы в задачах дискретной оптимизации: вычислительные модели, библиотека, результаты экспериментов. – М.: ВЦ РАН, 2006. – 50 с.