Journal of Contemporary Applied Mathematics
V. 5, No 1, 2015, September
ISSN 2222-5498

Построение оптимального упорядочения для выполнения потока заданий в заданные сроки в вычислительной системе

Дж. К. Кязимов

Аннотация. Рассматриваемая работа посвящена построению оптимального упорядочения для выполнения потока заданий в заданные сроки в вычислительной системе. Используя интервал очередности и функции штрафа задается алгоритм определения оптимальной последовательности для выполнения заданий. Рассматривается случай, в котором интервал очередности не зависит от времени.

Key Words and Phrases: оптимальное расписание, упорядочение заданий, интервал очередности, суммарный штраф.

2000 Mathematics Subject Classifications: 68W40, 68Q25

1. Постановка задачи и предварительные замечания

Пусть необходимо выполнить множество $N = \{1, 2, \ldots, n\}$ заданий в вычислительной системе. Задание $k,\ k = \overline{1,n}$ поступает на выполнение в момент времени $d_k \geq 0$ и для выполнения требуется $t_k > 0$ единиц времени. Необходимо так организовать процесс выполнения, чтобы он в том или ином смысле был наилучшим. Процесс выполнения может быть описан заданием кусочно-постоянной, непрерывной слева функции $S = S(t)$, принимающей при $0 < t < \infty$ одно из значений $0, 1, 2, \ldots, n$. Если $S(t) = k \neq 0$, то в момент времени t процессор выполняет задание k, если $S(t) = 0$, то в момент времени t, ни одно из заданий не выполняется. Эта функция называется расписанием [3]. Расписание должно удовлетворять определенным условиям [4].

Каждому допустимому расписанию $S = S(t)$ соответствует вектор $T = (T_1, T_2, \ldots, T_n)$ времени завершения выполнения заданий при этом расписание. Значения $T_k,\ k = \overline{1, n}$, таковы, что $S(T_k) = k$ и $S(t) \neq k$ для всех $t > T_k$.

Предполагается, что качество расписания S определяется вектором T, т.е. каждому расписанию S ставится в соответствие значение некоторой скалярной функции $F(T)$ вектора T, определяемого расписанием S.

Функция $F(T)$ предполагается монотонно возрастающей кусочно-непрерывной относительно всех компонентов вектора T.

Следуя [1],[2], функцию $F(T)$ можно задавать следующими способами. Для каждого задания $k = \overline{1, n}$ задается неубывающая кусочно-непрерывная функция $\Phi_k(x)$, выражающая в количественном отношении "штраф", если выполнение этого задания завершается в момент времени x. В качестве $F(T)$ выбирается одна из функций $F_1(T) = \sum_{k=1}^{n} \Phi_k(T_k)$ или $F_2(T) = \overline{\max}\{\Phi_k(T_k)\}$. Функции $\Phi_k(x)$ называются функциями штрафа. Расписание, которое удовлетворяет всем условиям рассматриваемой задачи, называется оптимальным, если ему соответствует наименьшее значение $F(T)$. В работе [1],[2] в распределенных системах строится оптимальное расписание и показаны необходимые и достаточные условия для существования оптимального расписания. В рассматриваемой работе, задача состоит в следующем: надо найти оптимальную последовательность выполнения потоков заданий, где средний штраф принимает наименьшее значение.

2. Оптимальные упорядочения заданий

Рассмотрим решение задачи оптимального упорядочения заданий для некоторых случаев функции штрафа. Предполагается, что множество N заданий не упорядочено, все задания поступают в очередь на выполнение одновременно, в момент времени $d = 0$, критерий оптимальности расписания – суммарный штраф.

Пусть задания i и j выполняются непосредственно друг за другом и их выполнение начинается в момент времени $t \geq 0$. Если при этом первым выполняется задание i, то суммарный штраф за выполнение этих заданий можно определить следующим образом:

$$F_1(t, i, j) = \Phi_i(t + t_i) + \Phi_j(t + t_i + t_j). \tag{1}$$

Если первым выполняется задание j, то суммарный штраф за выполнение этих двух заданий определяется так:

$$F_2(t, j, i) = \Phi_j(t + t_j) + \Phi_i(t + t_i + t_j). \tag{2}$$

Определяем

$$R_{ij}(t) = F_1(t, i, j) - F_2(t, j, i). \tag{3}$$

Величина $R_{ij}(t)$ показывает, насколько изменяется суммарный штраф при переходе от последовательности, при которой задание i начинало выполняться в момент времени t непосредственно перед заданием j. Если $R_{ij}(t) < 0$,то первым в момент времени t следует выполнять задание i. Если $R_{ij}(t) > 0$, то первым в момент времени t следует выполнять задание j. Наконец, если $R_{ij}(t) = 0$, то порядок выполнения этих заданий безразличен.

Поскольку функции $\Phi_i(x)$ и $\Phi_j(x)$ предполагаются кусочно-непрерывными в интервале $(0,\ T = \sum_{k=1}^{n} t_k)$, то этот интервал может быть разбит на конечное число интервалов, в каждом из которых величина $R_{ij}(t)$ положительна, отрицательна или равна нулю.

Интервал $(a,\, b)$ называется интервалом очередности типа $i \to j$, если $R_{ij}(t) < 0$ для всех $t \in (a,\, b)$. Если $R_{ij}(t) > 0$ для всех $t \in (a,\, b)$, то интервал $(a,\, b)$ называется интервалом очередности типа $j \to i$. При $R_{ij}(t) = 0$ для всех $t \in (a,\, b)$ порядок выполнения заданий безразличен.

Величина $R_{ij}(t)$ по определению зависит от значений t, t_i, t_j и функций штрафа $\Phi_i(x)$ и $\Phi_j(x)$. В каждом конкретном случае в результате несложных преобразований могут быть выделены интервалы очередности. Найдем такие пары функций $\Phi_i(x)$ и $\Phi_j(x)$, для которых знак $R_{ij}(t)$ не зависит от t при любых сочетаниях t_i и t_j.

Допустим, что необходимо выполнять n различных заданий на вычислительной машине. Известны длительность выполнения каждого задания t_k и срок D, за который необходимо выполнить эти задания.

Теорема. *Пусть $\Phi_i(x)$ и $\Phi_j(x)$ – строго возрастающие на $(0,\, T)$ функции. Для того чтобы величины $R_{ij}(t)$ не зависели от t, $0 < t < T$, при любых $t_i \le t_j$ достаточно, чтобы $\Phi_k(x) = \max(x - D,\ 0)$, $k = \overline{i,\, j}$; и интервал $(0,\, T)$ является интервалом очередности типа $i \to j$.*

Доказательство. Найдем интервал очередности при различных сочетаниях t_i, t_j. С помощью (1)-(3) получим:

$$R_{ij}(t) = \Phi_i(t + t_i) + \Phi_j(t + t_i + t_j) - \Phi_j(t + t_j) - \Phi_i(t + t_i + t_j) =$$

$$= \max(t + t_i - D,\ 0) + \max(t + t_i + t_j -$$

$$-D,\ 0) - \max(t + t_j - D,\ 0) - \max(t + +t_i + t_j - D,\ 0) =$$

$$= \max(t + t_i - D, 0) - \max(t + t_j - D,\ 0) = \begin{cases} t_i - t_j, \ t + t_i \ne D, \ \ t + t_j \ne D, \\ 0\,, \quad \text{в противном случае.} \end{cases}$$

Отсюда получим, что $R_{ij}(t) \le 0$, $t \in (0,\, T)$. В этом случае $R_{ij}(t)$ не зависит от t и при последовательном выполнении заданий i и j, задание i всегда выполняется первым. В этом случае интервал $(0,\, T)$ является интервалом очередности типа $i \to j$.
Теорема доказана.

Если функцией штрафа возьмем $\Phi_k(x) = \max(x - D,\ 0)$, то средний штраф будет

$$\frac{1}{n}\sum_{k=1}^{n} \max\left(\overline{t_k} - D,\, 0\right). \qquad (4)$$

Здесь $\overline{t_k}$ – фактическое время завершения выполнения k – го задания. Последовательность выполнения заданий обозначим $\Omega = (i_1,\, i_2,\, ..,\, i_n)$. Допустим, что эта последовательность удовлетворяет условию: $t_{i_\nu} \le t_{i_{\nu+1}}$, $\nu = \overline{1,\, n-1}$.
Тогда последовательность выполнения заданий $\Omega = (i_1,\, i_2,\, ..,\, i_n)$ – является оптимальной последовательностью и средний штраф (4) принимает наименьшее значение.

Решение уравнения $R_{ij}(t) = 0$ обозначим через t_0. Интервал $(-\infty, \infty)$ можно разбить на три интервала очередности $(-\infty, t_0)$, (t_0), (t_0, ∞). Тип интервалов $(-\infty, t_0)$ и (t_0, ∞) определяется знаком $R_{ij}(t)$ в этих интервалах. При $t = t_0$ порядок выполнения заданий i и j безразличен. По предположению, выполнение всех заданий осуществляется во временном интервале $(0, T)$ или во всяком случае, в интервале $(0, \infty)$. Следовательно в практическом отношении интерес представляет выделение интервалов очередности в интервале планирования $(0, T)$ или $(0, \infty)$. Если $t_0 < 0$ или $t_0 < T - t_i - t_j$, то $(0, T)$, является интервалом очередности, тип которого определяется знаком $R_{ij}(t)$. Если $0 < t_0 < T - t_i - t_j$, то $(0, T)$ разбивается на три интервала очередности. Если $t = 0$ или $t_0 = T - t_i - t_j$, то имеем два интервала очередности, один из которых (t_0).

3. Об одном алгоритме построения оптимальной последовательности

Обозначим через $D_k \geq 0$ желательное время завершения выполнения заданий $k = \overline{1, n}$. Не нарушая общности, пронумеруем все задания в порядке не убывания D_k. Ясно что если $\overline{t_k} = \sum_{i=1}^{k} t_i \leq D_k$, $k = \overline{1, n}$, то выполнение всех заданий может быть завершено в заданные сроки и оптимальная последовательность выполнения заданий $\Omega^* = (1, 2, .., n)$. Если хотя бы при одном значении k значение $\overline{t_k} > D_k$, то не существуют последовательности выполнения заданий, при которой эти задания выполнялись бы в заданные сроки. В этом случае $F_1(\Omega^*) > 0$ и построение оптимальной последовательности сопряжено с определенными вычислительными затруднениями. В общем случае не удается построить такого расписания, при котором выполнение каждого задания k завершается не позднее заданного директивного срока D_k. Иными словами, каждому заданию $k = \overline{1, n}$ отнесена неубывающая кусочно-непрерывная функция штрафа

$$\Phi_k(x) = \begin{cases} 0, & \text{если } x \leq D_k, \\ \text{положительно}, & \text{если } x > D_k. \end{cases}$$

Пусть $\Phi_k(x)$ есть ступенчатые функции, т.е. функции вида
$\Phi_k(x) = \begin{cases} 0, & \text{если } x \leq D_k, \\ a_k > 0, & \text{если } x > D_k, \end{cases}$ где $k = \overline{1, n}$.

Предположим, что имеется способ, посредством которого из множества $N = \{1, 2, \ldots, n\}$ всех заданий может быть выделено подмножество B^*, элементы которого обладают характеристическим свойством: существует оптимальная последовательность Ω^*, при которой задание k в срок не выполняется, тогда и только тогда, когда $k \in B^*$. Тогда очевидно, $F_1(\omega^*) = \sum_{k \in B^*} a_k$.

Покажем, что в этом случае и построение самой оптимальной последовательности не вызывает затруднений. Действительно, пусть Ω_{B^*}—произвольная последовательность элементов множества B^*, а $\overline{\omega}_{A^*}$—упорядоченная по возрастанию последовательность элементов множества $A^* = N \backslash B^*$. Будем выполнять задания в последовательности $\omega = (\overline{\omega}_{A^*}, \omega_{B^*})$. Учитывая тот факт, что все задания пронумерованы в порядке не убывания значений D_k. Поэтому задание множества

A^* будет выполняться в срок. Что касается заданий множества B^*, то ни одно из них при этой последовательности не будет выполнено в срок, так как в противном случае исходная последовательность не была бы оптимальной. Следовательно, $F_1(\overline{\omega}_{A^*}, \omega_{B^*}) = F_1(\omega^*)$ и $\omega = (\overline{\omega}_{A^*}, \omega_{B^*})-$ оптимальная последовательность.

Если каждому заданию $k = \overline{1, n}$ отнесем переменную x_k, принимающую значение 1, если $k \in B^*$, и значение 0, если $k \in A^*$, то искомое разбиение множества N может быть получено в результате решения задачи целочисленного линейного программирования с n ограничениями и n булевыми переменными:

$$\sum_{k=1}^{n} a_k x_k \to \min, \tag{5}$$

$$\sum_{k=1}^{i} t_k (1 - x_k) \le D_i, i = \overline{1, n}, \tag{6}$$

$$x_k \in \{0, 1\}, k = \overline{1, n}. \tag{7}$$

Положим $Z_k = \max\big(0, \overline{t_k} - D_k\big)$, где $\overline{t_k}$ есть фактическое время завершения выполнения задания k, а D_k-соответствующий директивный срок. Если задания выполняются в последовательности $\omega = \{i_1, i_2, \ldots, i_n\}$, то $\overline{t_{ij}} = \sum_{k=1}^{j} t_{i_k}$. Пусть $B \in B^*$, $A = N \backslash B$ и $\overline{\omega_A}$ есть последовательность заданий множества A, расположенных в порядке их возрастания.

На основании предыдущих рассуждений можно заключить, что если $\bar{\omega} = \{i_1, \ldots, i_p, \ldots, i_q\}$, $Z_{i_p} > 0$ и $Z_{i_j} = 0$ при $j < p$, то в B^* содержится по крайней мере, одно из заданий множества $C = \{i_1, i_2, \ldots, i_p\}$.

Следовательно, задачу (5) - (7) можно заменить эквивалентной ей задачей целочисленного линейного программирования, в ряде случаев меньшей размерности. Обозначим через S множество элементов последовательности $(1, 2, \ldots, n)$, которым соответствует строго возрастающая последовательность $Z_k > 0$. Имеем

$$\sum_{k=1}^{s} a_k x_k \to \min, \tag{8}$$

$$\sum_{k=1}^{i} t_k x_k \ge Z_i, i \in S, \tag{9}$$

$$x_k \in \{0, 1\}, k = \overline{1, l}. \tag{10}$$

Здесь $l-$наибольший элемент множества S.

Будем говорить, что множество $\bar{C} \subseteq C$ является критическим, если $\sum_{ij \in \bar{c}} t_{ij} \ge Z_{ip}$ и $\sum_{ij \in \breve{C}} t_{ij} < Z_{ip}$ для всех собственных подмножеств $\breve{C}$ множества $\bar{C}$. Одно из критических подмножеств множества C содержится в B^*. Этот факт позволяет организовать целенаправленный поиск оптимального разбиения множества N всех заданий на два подмножества A^* и B^*.

Алгоритм организации такого поиска может быть описан следующим образом:

1. Положим $B = B^0 = \emptyset$, $A = A^0 = N \backslash B^0 = N$ и построим, $\overline{\omega_{A^0}} = (1, 2, \ldots, n)$. Если все $Z_k = 0$, то $B^* = B^0$, $A^* = A^0$ и $\overline{\omega_{A^0}}$–искомая последовательность. Пусть $Z_p > 0$ и $Z_k = 0$ при $k < p$. Образуем все критические подмножества $\overline{C_1^0}, \overline{C_2^0}, \ldots, \overline{C_p^0}$ множество $C = \{1, 2, \ldots, p\}$. Положим $B_e^1 = B^0 \bigcup C_e^0$, $A_e^1 = N \backslash B_e^1$, $l = 1, \ldots, r_0$.

2. Переходим к последовательному рассмотрению разбиений $\left(A_e^1, B_e^1\right)$. Полагая $B = B_e^1$, $A = A_e^1$, строим $\omega_{A_e^1} = \left(i_1^1, \ldots, i_{p_1}^1, \ldots, i_{q_{1_1}}^1\right)$. Если все $Z_{i_j^1} = 0$, то строим $\omega = \left(\omega_{A_i^1}, \omega_{B_i^1}\right)$, вычисляем $F_1(\omega)$ и приписываем это значение $F_1(\omega)$ разбиению $\left(A_i^1, B_i^1\right)$. Если $Z_{i_{p_1}^1} > 0$ и $Z_{i_j^1} = 0$ при $j < p_1$, то образуем все критические подмножества $C_{e1}^1, C_{e1}^1, \ldots, C_{er_e}^1$ множества $C_e^1 = \left(i_1^1, \ldots, i_p^1, \ldots, i_{p_1}^1\right)$. Положим $B_{et}^2 = B_e^1 \bigcup C_{et}^1$, $A_{et}^2 = N \backslash B_{et}^2$, $t = 1, \ldots, r_e$.

3. Переходим затем к последовательному рассмотрению разбиений $\left(A_{11}^2, B_{11}^2\right), \ldots, \left(A_{r_0r_0}^2, B_{r_0r_0}^2\right)$ и т.д. Этот процесс очевидно конечен. В результате, будет получен набор разбиений множества N, каждому из которых будет приписано некоторое значение $F_1(\omega)$. То из разбиений, которому приписано наименьшее значение $F_1(\omega)$, является оптимальным разбиением (A^*, B^*), а последовательность $\omega = (\omega_{A^*}, \omega_{B^*})$, искомой оптимальной последовательностью выполнения заданий.

Список литературы

[1] Кязимов Д.К., Гасанов Х.А. Оптимальные расписания для выполнения заданий с заданным сроком в распределенных системах, Известия НАН Азербайджана, 2011, т.21, no.3, с. 3-8.

[2] Кязимов Д.К., Гасанов Х.А. Необходимые и достаточные условия существования оптимального расписания для планирования вычислений в распределенных системах. Проблемы информационной технологии, 2011, no.1, с. 81-86.

[3] Топорков В.В. Модели распределенных вычислений. М.: Физмат лит., 2004.

[4] Теория расписаний и вычислительных машин/Под. ред. Коффмона Э.Г. - М.: Наука, 1984.

Дж. К. Кязимов
Бакинский Государственный Университет,
Ул. З. Халилова, 23, Баку, Азербайджан
E-mail: dkazimov@mail.ru

Received 03 March 2015
Accepted 05 May 2015