РОССИЙСКАЯ АКАДЕМИЯ НАУК

СИБИРСКОЕ ОТДЕЛЕНИЕ

АВТОМЕТРИЯ

2006, том 42, № 1

УДК 681.324

М. С. Седельников

(Новосибирск)

АЛГОРИТМЫ РАСПРЕДЕЛЕНИЯ НАБОРА ЗАДАЧ С ПЕРЕМЕННЫМИ ПАРАМЕТРАМИ ПО МАШИНАМ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

Рассматривается функционирование распределенных вычислительных систем (ВС) в режиме обработки набора параллельных задач с переменными параметрами. Предлагаются эвристические алгоритмы назначения задач набора на элементарные машины ВС, минимизирующие время или штраф за задержку их решения. Приводятся результаты моделирования, подтверждающие, что полученные алгоритмы обеспечивают, по крайней мере, субминимальные значения целевых функций.

Введение. Архитектура распределенных вычислительных систем (ВС) представляется в виде композиции множества элементарных машин (ЭМ) или процессоров и сети связей между ними [1, 2]. Число процессоров в таких ВС может быть достаточно большим, например в системе IBM Blue Gene оно достигает 10^6.

Одним из режимов функционирования распределенных ВС является режим обслуживания потоков задач, представленных параллельными программами со случайными параметрами (числом ветвей и временем решения). При достаточно большой интенсивности потоков на каждой ЭМ образуется конечная очередь задач, следовательно, возникает необходимость в распределении их набора. Вычислительная сложность такой задачи не позволяет использовать точные алгоритмы [3, 4]. Для ее решения могут быть применены эвристические и стохастические методы [2, 5]. На сегодняшний день большинство задач допускает изменение одного или нескольких параметров в некоторых диапазонах, например изменение числа параллельных ветвей в программе. В представленной работе предлагаются алгоритмы, позволяющие эффективно распределять по ЭМ вычислительной системы такие задачи.

Постановка задачи. Имеется ВС $J = \{J_1, J_2, \dots, J_n\}$, состоящая из n элементарных машин, а также конечный набор $I = \{I_1, I_2, \dots, I_L\}$ задач, каждая из которых характеризуется переменным рангом r_i (числом параллельных ветвей в программе), $r_i^- \le r_i \le r_i^+$, временем решения $t_i = t_i(r_i)$, штрафом в единицу времени за задержку решения $c_i \ge 0$, $i = 1, \dots, L$. В большинстве случаев при увеличении числа параллельных ветвей в программе время ее выполнения уменьшается, поэтому справедливо предположить, что $t_i(r_i)$ – невозрастающая функция. Требуется распределить задачи по ЭМ вычислительной системы так, чтобы минимизировать либо время решения задач набора, либо штраф за задержку их решения.

Рассмотрим формальную математическую модель распределения задач набора с постоянными параметрами. Каждую задачу I_i будем представлять в виде совокупности параллельных ветвей $I_i = \bigcup_{k=1}^{r_i} I_i^k.$ Примем следующие обозначения: $\hat{t}_i$ – время начала решения задачи I_i, J_i^k – элементарная машина для выполнения ветви I_i^k задачи I_i.

Для задачи минимизации времени решения необходимо найти вектор

$$\tau = (\hat{t}_1, \ldots, \hat{t}_L, J_1^1, \ldots, J_1^{r_1}, J_2^1, \ldots, J_2^{r_2}, \ldots, J_L^1, \ldots, J_L^{r_L}),$$

минимизирующий

$$T = \max_{i=1,\ldots,L} (\hat{t}_i + t_i) \to \min_{\tau}$$

при ограничениях:

1) $\forall i \neq j,\ i, j = 1, \ldots, L$, верно, по крайней мере, одно из следующих двух условий:

$$\text{а) } [\hat{t}_i, \hat{t}_i + t_i] \cap [\hat{t}_j, \hat{t}_j + t_j] = \varnothing,$$

$$\text{б) } \forall k_1 = 1, \ldots, r_i, \quad k_2 = 1, \ldots, r_j, \quad J_i^{k_1} \neq J_j^{k_2};$$

2) $\forall i = 1, \ldots, L, \quad \forall k_1 \neq k_2, \quad k_1, k_2 = 1, \ldots, r_i, \quad J_i^{k_1} \neq J_i^{k_2}$;

3) $\forall i = 1, \ldots, L, \quad k = 1, \ldots, r_i, \quad J_i^k \in J$;

4) $\forall i = 1, \ldots, L, \quad \hat{t}_i \geq 0,\ \hat{t} \in \mathbb{R}$.

Ограничения 1 запрещают одновременное решение двух и более задач на одной элементарной машине, ограничения 2 обеспечивают реализацию разных ветвей каждой задачи на разных ЭМ.

Для задачи минимизации штрафа за задержку решения набора задач необходимо в приведенной выше модели заменить критерий оптимизации функцией

$$F = \sum_{i=1}^{L} \hat{t}_i c_i \to \min_{\tau}.$$

Рассмотрим пример. Пусть имеется вычислительная система из пяти ЭМ и набор из восьми задач с рангами $r_i \in \{2,1,2,2,1,2,2,3\}$ и временем решения $t_i \in \{10,15,20,10,7,2,5,3\}$, $i = 1, \ldots, 8.$ В качестве вектора решения, удовлетворяющего ограничениям 1–4 и обеспечивающего минимум T, можно рассмотреть вектор

$$\tau = (0,\ 0,\ 0,\ 10,\ 15,\ 20,\ 20,\ 22,\ 1,\ 2,\ 3,\ 4,\ 5,\ 1,\ 2,\ 3,\ 4,\ 5,\ 1,\ 2,\ 3,\ 4,\ 5).$$

Оптимальным временем решения будет $T = 25$.

Далее модернизируем приведенную модель для распределения задач набора с переменными параметрами. Введем дополнительные переменные r_i, $i = 1, \ldots, L$, – ранг распределяемых задач.

Необходимо найти вектор

$$\tau = (\hat{t}_1, \dots, \hat{t}_L, r_1, \dots, r_L, J_1^1, \dots, J_1^{r_1^+}, J_2^1, \dots, J_2^{r_2^+}, \dots, J_L^1, \dots, J_L^{r_L^+}),$$

минимизирующий

$$T = \max_{i=1,\dots,L} (\hat{t}_i + t_i(r_i)) \to \min_{\tau}$$

при ограничениях:

5) $\forall i \neq j, \quad i, j = 1, \dots, L,$ верно, по крайней мере, одно из следующих двух условий:

$$\text{а) } [\hat{t}_i, \hat{t}_i + t_i(r_i)] \cap [\hat{t}_j, \hat{t}_j + t_j(r_j)] = \varnothing,$$

$$\text{б) } \forall k_1 = 1, \dots, r_i^+, \; k_2 = 1, \dots, r_j^+, \; (J_i^{k_1}, J_j^{k_2} \neq 0) \Rightarrow (J_i^{k_1} \neq J_j^{k_2});$$

6) $\forall i = 1, \dots, L, \; \forall k_1 \neq k_2, \quad k_1, k_2 = 1, \dots, r_i^+, \; (J_i^{k_1}, J_i^{k_2} \neq 0) \Rightarrow (J_i^{k_1} \neq J_i^{k_2});$

7) $\forall i = 1, \dots, L, \quad \sum_{k=1}^{r_i^+} \operatorname{sign}(J_i^k) = r_i;$

8) $\forall i = 1, \dots, L, \quad r_i^- \leq r_i \leq r_i^+, \quad r_i^-, r_i, r_i^+ \in \mathbb{Z}^+;$

9) $\forall i = 1, \dots, L, \quad k = 1, \dots, n, \quad J_i^k \in J \cup \{0\};$

10) $\forall i = 1, \dots, L, \; \hat{t}_i \geq 0, \; \hat{t}_i \in \mathbb{R}.$

В данной модели ограничения 7 и 8 обеспечивают решение задачи на необходимом количестве элементарных машин.

Для задачи минимизации штрафа за задержку решения нужно в приведенной выше модели заменить критерий оптимизации функцией

$$F = \sum_{i=1}^{L} \hat{t}_i c_i \to \min_{\tau}.$$

Рассмотрим пример. Пусть имеется вычислительная система из пяти ЭМ и набор из трех задач с рангами $r_i^- \in \{1, 2, 3\}, \; r_i^+ \in \{3, 4, 3\}$ и временем решения $t_i(r_i) \in \left\{ \frac{12}{r_i}, \frac{8}{r_i}, \frac{6}{r_i} \right\}, \; i = 1, 2, 3.$ В качестве вектора решения, удовлетворяющего ограничениям 5–10 и обеспечивающего минимум T, можно рассмотреть вектор

$$\tau = (0, 0, 4, 3, 2, 3, 1, 2, 3, 4, 5, 1, 2, 3).$$

Оптимальным временем решения будет $T = 6$.

Предлагаемые алгоритмы представлены в виде операторных схем, содержащих последовательность операторов, каждый из которых изображает достаточно крупную группу элементарных операций. При этом используются в основном обозначения операторов, принятые в [2]: Π, P, ϑ, S, Н и Я.

П-операторы осуществляют передачу управления. Запись $\Pi_j :\to F_i$ (или Π_j^i) означает, что при реализации оператора П осуществляется переход к оператору F_i.

P-операторы относятся к логическим. Например, запись P_i: $P\{x\}$, $P=1\to F_j$, $P=0\to P_k$ означает: проверить выполнение условия x; если окажется, что условие выполнено $(P=1)$, то осуществить переход к оператору F_j, а если же условие не выполнено $(P=0)$, то перейти к логическому оператору P_k. Это же может быть записано более коротко: $P_{i\downarrow k}^{\uparrow j}$.

Обозначение передачи управления от одного оператора к другому, непосредственно за ним следующему, опускается. Слева вверху от символа данного оператора ставится номер тех операторов, от которых передается управление. Например, ${}^{i,j}F_k$ означает, что оператору F_k управление передается от операторов с номерами i или j.

ϑ-операторы относятся к классу операторов присваивания значения выражения переменной. Например, запись ϑ_i: $T:=T+t$ означает, что при выполнении оператора ϑ_i переменной T присваивается значение суммы $T+t$.

S-операторы предназначены для сортировки последовательностей задач по определенному признаку.

Н-операторы принадлежат классу операторов назначения. Запись H_i: $I_q\Rightarrow J_j$, $j=s+1, s+2, \ldots, s+r$, означает, что при выполнении оператора с номером i (т. е. H_i) задача I_q направляется для решения на машины J_j, имеющие номера $j=s+1, s+2, \ldots, s+r$.

Я-операторы означают конец алгоритма.

Алгоритм минимизации времени решения задач набора. Рассмотрим частный случай. Пусть время решения задачи I_i составляет

$$t_i(r_i)=\frac{s_i}{r_i},$$

где $s_i>0$, $s_i\in\mathbb{R}$. Другими словами, время решения задачи I_i сокращается пропорционально увеличению ее ранга. Предлагаемый алгоритм (далее алгоритм I) базируется на следующих принципах:

1) задачи с большим временем решения назначаются на ЭМ раньше;

2) задачи распределяются по максимально возможному числу ЭМ.

Введем операторы:

ϑ_1: $T:=0,\ h:=0,\ J_{i_q}^k:=0,\ q=1,\ldots,L,\ k=1,\ldots,r_{i_q}^+$;

S_2: $I:=\{I_{i_1}, I_{i_2}, \ldots, I_{i_L}\}$ сортировать I по убыванию величины $\dfrac{s_{i_q}}{r_{i_q}^+}$, $q=1,\ldots,L$;

ϑ_3: $q:=\min\limits_{I_{il}\in I} l,\ r:=0,\ T:=T+h$;

ϑ_4: $r_{i_q}:=r_{i_q}^+,\ h:=\dfrac{s_{i_q}}{r_{i_q}},\ \hat{t}_{i_q}:=T,\ I:=I\setminus\{I_{i_q}\},\ L:=L-1$;

H_5: $I_{i_q}\Rightarrow J_j,\ j=r+1, r+2, \ldots, r+r_{i_q}$;

ϑ_6: $J_{i_q}^j:=J_{r+j},\ j=1,\ldots,r_{i_q},\ r:=r+r_{i_q}$;

ϑ_7: $q:=q+1$;

P_8: $P\{I=\varnothing\},\ P=1\to Я_{19}$;

P_9: $P\left\{q > \max\limits_{I_{i_l} \in I} l\right\},\ P = 1 \to \vartheta_3$;

ϑ_{10}: $r_{i_q} := r_{i_q}^{+}$;

P_{11}: $P\left\{r_{i_q}^{-} \le \left]\dfrac{s_{i_q}}{h}\right[\right\},\ P = 0 \to P_{13}$; ϑ_{12}: $r_{i_q} := \left]\dfrac{s_{i_q}}{h}\right[$;

P_{13}: $P\{r + r_{i_q} > n\},\ P = 1 \to \vartheta_{17}$;

ϑ_{14}: $\hat{t}_{i_q} := T,\ I := I \setminus \{I_{i_q}\},\ L := L - 1$;

Н_{15}: $I_{i_q} \Rightarrow J_j,\ j = r+1, r+2, \ldots, r + r_{i_q}$;

ϑ_{16}: $J_{i_q}^{j} := J_{r+j},\ j = 1, \ldots, r_{i_q},\ r := r + r_{i_q}$;

ϑ_{17}: $q := q + 1$; $\Pi_{18} :\to P_8$;

Я_{19}: конец алгоритма.

Операторная схема алгоритма I имеет вид

$$\vartheta_1 S_2{}^{2,9} \vartheta_3 \vartheta_4 \text{Н}_5 \vartheta_6 \vartheta_7{}^{7,18} P_8^{\uparrow 19} P_9^{\uparrow 3} \vartheta_{10} P_{11\downarrow 13} \vartheta_{12}{}^{11,12} P_{13}^{\uparrow 17} \vartheta_{14} \text{Н}_{15} \vartheta_{16}{}^{13,16} \vartheta_{17} \Pi_{18}^{8}\, {}^{8}\text{Я}_{19}.$$

Утверждение 1. Вектор решения

$$\tau = (\hat{t}_1, \ldots, \hat{t}_L, r_1, \ldots, r_L, J_1^1, \ldots, J_1^{r_1^+}, J_2^1, \ldots, J_2^{r_2^+}, \ldots, J_L^1, \ldots, J_L^{r_L^+}),$$

полученный алгоритмом I, удовлетворяет условиям 5–10.

Доказательство. Условиям 9, 10 τ, очевидно, удовлетворяет. Выполнение условия 8 в операторе ϑ_{12} достигается неравенством $r_{i_q}^{-} \le \left]\dfrac{s_{i_q}}{h}\right[$ в P_{11}, а также операторами S_2 и ϑ_4: $\dfrac{s_{i_q}}{r_{i_q}^{+}} \le h$, следовательно, $\dfrac{s_{i_q}}{h} \le r_{i_q}^{+}$; учитывая, что $r_{i_q}^{+}$ – целое, получим $\left]\dfrac{s_{i_q}}{h}\right[\le r_{i_q}^{+}$.

После выполнения ϑ_6 или ϑ_{16} $\forall i_q$ в точности r_{i_q} переменных $J_{i_q}^{j}$ будут положительны, поэтому 7 выполнено. Истинность 6 также обеспечивается ϑ_6 или ϑ_{16}.

Докажем выполнение условия 5.

Пусть $\exists i_q, i_p$ такие, что неверно «а»:

$$\left[\hat{t}_{i_q}, \hat{t}_{i_q} + \frac{s_{i_q}}{r_{i_q}}\right] \cap \left[\hat{t}_{i_p}, \hat{t}_{i_p} + \frac{s_{i_p}}{r_{i_p}}\right] \ne \varnothing.$$

Докажем, что в этом случае верно условие «б». Время начала решения обеих задач будет совпадать: $\hat{t}_{i_q} = \hat{t}_{i_p}$, поскольку если, например, $\hat{t}_{i_q} < \hat{t}_{i_p}$, то $\hat{t}_{i_p} < \hat{t}_{i_q} + h \left(\frac{s_{i_q}}{r_{i_q}^{+}} \le h \right)$, что невозможно согласно операторам S_2, ϑ_4 или ϑ_{14}. Из равенства $\hat{t}_{i_q} = \hat{t}_{i_p}$ вытекает, что задачи I_{i_q}, I_{i_p} назначены на ЭМ до изменения величины T (оператор ϑ_4), следовательно, после выполнения ϑ_6 или ϑ_{16} условие «б» будет выполнено.

Пусть неверно «б»: $\exists k_1, k_2, i_q, i_p$, $J_{i_q}^{k_1} = J_{i_p}^{k_2} \ne 0$, т. е. ветви $I_{i_q}^{k_1}$, $I_{i_p}^{k_2}$ задач I_{i_q}, I_{i_p} соответственно выполняются на одной ЭМ $J_{i_q}^{k_1}$. Это возможно в том случае, если между назначениями этих задач был выполнен оператор ϑ_3 (параметр r обнулен), тогда $T := T + h$. Поскольку последовательность задач I упорядочена по убыванию минимального времени решения $\frac{s_{i_k}}{r_{i_k}^{+}}$, будет выполнено условие «а».

Утверждение доказано.

Справедливо следующее утверждение о вычислительной сложности алгоритма I.

Утверждение 2. Трудоемкость алгоритма I

$$T_{\mathrm{I}} = O(L^2).$$

Доказательство. Трудоемкость оператора S_2 оценивается величиной времени сортировки массива алгоритмом QuickSort и равна $O(L \ln L)$. В случае хранения набора задач в виде списка трудоемкость остальных операторов не превосходит $\frac{L(L-1)}{2} = O(L^2)$.

Утверждение доказано.

Приведем результаты моделирования работы алгоритма I. Легко заметить, что оценка нижней границы оптимального времени решения задач набора имеет вид

$$\Theta^n = \frac{1}{n} \sum_{i=1}^{L} r_i t(r_i) = \frac{1}{n} \sum_{i=1}^{L} s_i.$$

Используем наборы задач со следующими параметрами: $L = 1000$, $n = 8$; величины r_i^-, r_i^+ и s_i равномерно распределены в интервалах $[1, 8)$, $[1, 8)$ и $[1, 10)$ соответственно. В качестве оценки эффективности работы алгоритма рассматривалась величина $\delta = \frac{\left| T - \Theta^n \right|}{\Theta^n}$, где T – время решения задач набора, распределенных алгоритмом. Моделирование показало, что для δ оценки математического ожидания и дисперсии составляют $\mathrm{M}\delta = 1{,}1 \cdot 10^{-1}$, $\mathrm{D}\delta = 4 \cdot 10^{-5}$, т. е. предложенный алгоритм дает решение поставленной задачи хуже нижней оценки оптимального решения в среднем на 11 %.

Отметим также следующее очевидное утверждение.

Утверждение 3. Если $t_i(r_i)=\frac{s_i}{r_i}$ и $\forall i$ определено $t_i(n)$, тогда если для решения каждой задачи выделяется *n* ЭМ, то любая последовательность решения задач является оптимальной и время решения задач набора равно нижней оценке.

В случае, если $t_i(r_i)$ – произвольная невозрастающая функция и существует обратная $t_i^{-1}(t)$, то для минимизации времени решения задач набора может быть использована модификация алгоритма I, в которой в операторах $S_2, \vartheta_4, P_{11}, \vartheta_{12}$ величины $\frac{s_{i_q}}{r_{i_q}^+}$, $h:=\frac{s_{i_q}}{r_{i_q}}$, $r_{i_q}^- \le \left]\frac{s_{i_q}}{h}\right[$ и $r_{i_q} := \left]\frac{s_{i_q}}{h}\right[$ заменены величинами: $t_{i_q}(r_{i_q}^+)$, $h:=t_{i_q}(r_{i_q})$, $r_{i_q}^- \le \left]t_{i_q}^{-1}(h)\right[$ и $r_{i_q} := \left]t_{i_q}^{-1}(h)\right[$ соответственно.

Алгоритм минимизации штрафа за задержку решения задач набора. Рассмотрим частный случай. Пусть время решения задачи I_i составляет

$$t_i(r_i)=\frac{s_i}{r_i},$$

где $s_i>0,\ s_i\in\mathbb{R}$, т. е. время решения задачи I_i сокращается пропорционально увеличению ее ранга.

Величину $\frac{c_i}{t_i(r_i)}$ назовем удельным штрафом за задержку решения задачи I_i. Предлагаемый алгоритм решения (далее алгоритм II) основан на следующих принципах:

1) задачи с большим удельным штрафом назначаются на ЭМ раньше;

2) задачи распределяются по максимально возможному числу ЭМ.

Введем операторы:

ϑ_1: $T:=0,\ F:=0,\ h:=0,\ C:=\sum_{i=1}^{L} c_i,\ J_{i_q}^k:=0,\ q=1,\dots,L,\ k=1,\dots,r_{i_q}^+$;

S_2: $I=\{I_{i_1}, I_{i_2},\dots,I_{i_L}\}$ сортировать I по возрастанию величины $\frac{s_{i_q}}{c_{i_q} r_{i_q}^+}$, $q=1,\dots,L$;

ϑ_3: $q:=\min_{I_{i_l}\in I} l,\ r:=0,\ F:=F+Ch,\ T:=T+h$;

ϑ_4: $r_{i_q}:=r_{i_q}^+,\ h:=\frac{s_{i_q}}{r_{i_q}},\ \hat{t}_{i_q}:=T,\ C:=C-c_{i_q},\ I:=I\setminus\{I_{i_q}\},\ L:=L-1$;

H_5: $I_{i_q}\Rightarrow J_j,\ j=r+1, r+2,\dots,r+r_{i_q}$;

ϑ_6: $J_{i_q}^j:=J_{r+j},\ j=1,\dots,r_{i_q},\ r:=r+r_{i_q}$;

ϑ_7: $q:=q+1$;

P_8: $P\{I=\varnothing\},\ P=1\rightarrow Я_{19}$;

P_9: $P\left\{q > \max\limits_{I_{i_l} \in I} l\right\}$, $P = 1 \to \vartheta_3$;

ϑ_{10}: $r_{i_q} := r_{i_q}^{+}$;

P_{11}: $P\left\{r_{i_q}^{-} \le \left]\frac{s_{i_q}}{h}\right[\text{ и } \left]\frac{s_{i_q}}{h}\right[\le r_{i_q}^{+}\right\}$, $P = 0 \to P_{13}$; ϑ_{12}: $r_{i_q} := \left]\frac{s_{i_q}}{h}\right[$;

P_{13}: $P\{r + r_{i_q} > n\}$, $P = 1 - \vartheta_{17}$;

ϑ_{14}: $\hat{t}_{i_q} := T,\ C := C - c_{i_q},\ I := I \setminus \{I_{i_q}\},\ L := L - 1$;

H_{15}: $I_{i_q} \Rightarrow J_j,\ j = r+1, r+2, \ldots, r + r_{i_q}$;

ϑ_{16}: $J_{i_q}^{j} := J_{r+j},\ j = 1, \ldots, r_{i_q},\ r := r + r_{i_q},\ h := \max\left\{h, \frac{s_{i_q}}{r_{i_q}}\right\}$;

ϑ_{17}: $q := q + 1$; $\Pi_{18} :\to P_8$;

Я_{19}: конец алгоритма.

Операторная схема алгоритма II имеет вид

$$\vartheta_1 S_2{}^{2,9} \vartheta_3 \vartheta_4 \mathrm{H}_5 \vartheta_6 \vartheta_7{}^{7,18} P_8^{\uparrow 19} P_9^{\uparrow 3} \vartheta_{10} P_{11\downarrow 13} \vartheta_{12}{}^{11,12} P_{13}^{\uparrow 17} \vartheta_{14} \mathrm{H}_{15} \vartheta_{16}{}^{13,16} \vartheta_{17} \Pi_{18}^{8}{}^{8} \text{Я}_{19}.$$

Корректность алгоритма II обеспечивается утверждением 1. Справедливо также следующее утверждение о вычислительной сложности алгоритма II.

Утверждение 4. Трудоемкость алгоритма II

$$T_{\mathrm{II}} = O(L^2).$$

Доказательство. Трудоемкость оператора S_2 оценивается величиной времени сортировки массива алгоритмом QuickSort и равна $O(L \ln L)$. В случае хранения набора в виде списка трудоемкость остальных операторов не превосходит $\frac{L(L-1)}{2} = O(L^2)$.

Утверждение доказано.

Приведем результаты моделирования работы алгоритма II. Используем наборы задач со следующими параметрами: $L = 1000$, $n = 8$; величины r_i^-, r_i^+, t_i и c_i равномерно распределены в интервалах $[1, 8)$, $[1, 8)$, $[1, 100)$ и $[1, 10)$ соответственно. В качестве оценки работы алгоритма рассматривалась величина $\delta = \frac{\widetilde{F} - F}{F}$, где F и $\widetilde{F}$ – штрафы за задержку решения задач набора, распределенных алгоритмом и случайным назначением соответственно. Моделирование показало, что для δ оценки математического ожидания и дисперсии составляют $\mathrm{M}\delta = 1 \cdot 10^0$, $\mathrm{D}\delta = 3 \cdot 10^{-3}$, т. е. предложенный алгоритм дает решение поставленной задачи в среднем в 2 раза лучше, чем полученное случайным назначением.

В случае, если $t_i(r_i)$ – произвольная невозрастающая функция и существует обратная $t_i^{-1}(t)$, то для минимизации времени решения задач набора может быть использована модификация алгоритма II, в которой в операторах $S_2, \vartheta_4, P_{11}, \vartheta_{12}$ и ϑ_{16} произведены замены величин $\frac{s_{i_q}}{c_{i_q} r_{i_q}^+}$, $h := \frac{s_{i_q}}{r_{i_q}}$, $\left]\frac{s_{i_q}}{h}\right[$, $r_{i_q} := \left]\frac{s_{i_q}}{h}\right[$ и $h := \max\left\{h, \frac{s_{i_q}}{r_{i_q}}\right\}$ величинами $\frac{t_{i_q}(r_{i_q}^+)}{c_{i_q}}$, $h := t_{i_q}(r_{i_q})$, $]t_{i_q}^{-1}(h)[$, $r_{i_q} :=$ $:=]t_{i_q}^{-1}(h)[$ и $h := \max\{h, t_{i_q}^{-1}(r_{i_q})\}$ соответственно.

Заключение. Как видно из результатов моделирования, предложенные алгоритмы позволяют получить достаточно близкое к оптимальному (в смысле приведенных целевых функций) распределение задач набора по ЭМ вычислительной системы. Данные алгоритмы обладают небольшой вычислительной сложностью, что дает возможность рекомендовать их для организации эффективного функционирования вычислительных систем в режиме обработки набора параллельных задач с переменными параметрами.

СПИСОК ЛИТЕРАТУРЫ

1. **Хорошевский В. Г.** Архитектура вычислительных систем. М.: МГТУ им. Н. Э. Баумана, 2005.
2. **Хорошевский В. Г.** Алгоритмы функционирования однородных универсальных вычислительных систем // Автоматика и вычисл. техн. 1972. № 3. С. 35.
3. **Гэри М., Джонсон Д.** Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.
4. **Седельников М. С.** Точный алгоритм распределения набора задач по машинам вычислительной системы // Мат. Всерос. науч. конф. молодых ученых «Наука. Техника. Инновации». Новосибирск: НГТУ, 2004. Ч. 2. С. 65.
5. **Хорошевский В. Г., Седельников М. С.** Эвристические алгоритмы распределения задач по машинам вычислительной системы // Автометрия. 2004. **40**, № 4. С. 76.

Сибирский государственный университет телекоммуникаций и информатики,
E-mail: smsprog@mail.ru

Поступила в редакцию 13 мая 2005 г.