УДК 519.86
ББК 22.18

АЛГОРИТМЫ УПРАВЛЕНИЯ МНОГОПРОЦЕССОРНЫМИ СИСТЕМАМИ С НЕОДНОРОДНЫМ МНОЖЕСТВОМ РАБОТ

Гончар Д. Р. [1], **Фуругян М. Г.**[2]
(*Учреждение Российской академии наук Вычислительный центр им. А. А. Дородницына РАН, Москва*)

Разработан приближенный алгоритм составления оптимального по быстродействию расписания для множества работ, часть из которых допускает прерывания, а часть – не допускает. Производительности процессоров произвольные. При этом используются мультиоценочный алгоритм с калибровкой и модифицированный алгоритм упаковки.

Ключевые слова: многопроцессорная система, прерываемые и непрерываемые работы, расписания.

1. Постановка задачи

Рассматривается вычислительная система, состоящая из m процессоров, производительности которых равны s_1, s_2, …, s_m. Работая в течение времени t, процессор j выполняет работу объемом ts_j. Имеется множество работ $N = N_1 \cup N_2$, где N_1 – непрерываемые работы, N_2 – работы, допускающие прерывания и переключения с одного процессора на другой. Заданы объемы w_i и v_i работ $i \in N_1$ и $i \in N_2$ соответственно. Прерывания и переключения не связаны с временными затратами. Требуется со-

[1] *Дмитрий Русланович Гончар, кандидат технических наук* (rtsccas@yandex.ru).
[2] *Меран Габибуллаевич Фуругян, кандидат физико-математических наук, доцент* (*Москва, ул. Вавилова, д. 40, тел.* (*499*) *135-40-29*).

ставить оптимальное по быстродействию расписание выполнения множества работ N. Иными словами, необходимо так распределить работы по процессорам, чтобы длина временного интервала занятости наиболее загруженного процессора была минимальной.

Подобные задачи для случая, когда все работы являются непрерываемыми и не допускают переключений с одного процессора на другой, широко освещены в литературе. Подробный обзор литературы для случая, когда все работы являются непрерываемыми, а также для случая, когда работы допускают прерывания и переключения, содержится в [5]. Задачи со смешанным типом работ мало освещены в литературе. Так, например, в [1, 2] предполагается, что каждая работа строго закреплена за конкретным процессором, на множестве работ задан частичный порядок выполнения и, кроме того, только один из приборов допускает прерывания. В [4] рассмотрены случаи, когда директивные интервалы одинаковые, а также, когда директивные интервалы могут различаться, но с рядом дополнительных ограничений. В [5] рассмотрен случай идентичных процессоров.

2. Разбиение процессоров на две группы

Сначала все процессоры разбиваются на две группы. Все непрерываемые работы выполняются только процессорами первой группы, а прерываемые работы будут выполняться процессорами как первой, так и второй групп. Число процессоров в группах – m_1 и m_2– пропорционально суммарным объемам работ из N_1 и N_2 соответственно и обратно пропорционально суммарным производительностям процессоров в этих группах, т. е.

$$(1)\quad m_1 = \max\left\{ k : k \in Z, \frac{\sum_{i \in N_1} w_i}{\sum_{i \in N_1} w_i + \sum_{i \in N_2} v_i} \geq \frac{\sum_{j=1}^{k} s_j}{\sum_{j=1}^{m} s_j} \right\}, m_2 = m - m_1.$$

Будем предполагать, что в первую группу входят процессоры $j = 1, 2, \ldots, m_1$, а во вторую – $j = m_1 + 1, m_1 + 2, \ldots, m$.

3. Распределение непрерываемых работ

Для распределения непрерываемых работ по m_1 процессорам авторами был использован приближенный мультиоценочный алгоритм с калибровкой [3], который дает неплохую оценку погрешности и является достаточно эффективным. Рекомендации по его использованию даны в [3].

Без ограничения общности можно считать, что процессоры упорядочены по невозрастанию производительностей, а работы из N_2 – по невозрастанию объемов, т. е. $s_1 \geq s_2 \geq \ldots \geq s_m$, $v_1 \geq v_2 \geq \ldots \geq v_{n_2}$. Пусть Q_j – длина интервала загруженности процессора j $(1 \leq j \leq m_1)$ по расписанию, построенному для множества работ N_1, и пусть $Q_{\max} = \max\limits_{j=1,\ldots,m_1} Q_j$, $\tau_{max} = v_1/s_m$, $\tau_{\min} = v_1 / s_{m_1+1}$, $T \geq \max(Q_{max}, \tau_{max})$, $L_j = T - Q_j$ $(j = 1, \ldots, m_1)$, $n_1 = |N_1|$, $n_2 = |N_2|$.

4. Достаточное условие существования расписания заданной длины

Определим достаточное условие существования расписания, длина которого не превосходит величины $T \geq \max(Q_{max}, \tau_{\max})$. Для этого сначала опишем алгоритм упаковки множества работ N_2 на $(m_2 + 1)$ процессорах. Пусть $\max\limits_{j=1,\ldots,m_1} L_j s_j = L_{j_0} s_{j_0}$ ($L_j s_j$ – максимальный объем работы, который процессор j может выполнить до момента времени T после выполнения работ из N_1). Алгоритм распределяет работы из N_2 по процессорам j_0, m_1+1, …, m.

Алгоритм упаковки

Шаг 1. Если $\sum_{i \in N_2} v_i \le L_{j_0} s_{j_0}$, то все работы из N_2 выполнять на процессоре j_0 последовательно, начиная с момента Q_{j_0}.

В противном случае перейти на шаг 2.

Шаг 2. Пусть номер k $(0 \le k \le n_2)$ такой, что $\sum_{i=1}^{k} v_i \le L_{j_0} s_{j_0}$, а $\sum_{i=1}^{k+1} v_i > L_{j_0} s_{j_0}$. Положить $\Theta = Q_{j_0} + (\sum_{i=1}^{k} v_i) / s_{j_0}$. Работы $1, 2, \ldots, k \in N_2$ выполнять последовательно без прерываний на процессоре j_0 в интервале $[Q_{j_0}, \Theta]$. Работу $k + 1 \in N_2$ выполнять сначала на процессоре $m_1 + 1$ в интервале $[0, (v_{k+1} - (T - \Theta) s_{j_0}) / s_{m_1+1}]$, а затем на процессоре j_0 в интервале $[\Theta, T]$. Такое переключение корректно, так как $(v_{k+1} - (T - \Theta) s_{j_0}) / s_{m_1+1} \le \Theta$, что, в свою очередь, следует из соотношений

$$(v_{k+1} - (T - \Theta) s_{j_0}) / s_{m_1+1} = v_{k+1} / s_{m_1+1} - (T - \Theta) s_{j_0} / s_{m_1+1} \le$$
$$\le v_1 / s_m - (T - \Theta) = \tau_{\max} - T + \Theta \le T - T + \Theta = \Theta .$$

Положить $\Theta = (v_{k+1} - (T - \Theta) s_{j_0}) / s_{m_1+1}$.

*Шаг 3***.** Пусть работы $i = 1, 2, \ldots, p$ из N_2 $(k + 1 \le p < n_2)$ уже распределены по процессорам $m_1 + 1, \ldots, l$ $(m_1 < l \le m)$, причем процессоры $m_1 + 1, \ldots, l - 1$ загружены в интервале времени $[0, T]$, а процессор l – в интервале $[0, \Theta]$.

Если $\Theta + v_{p+1}/s_1 \le T$, то работу $p + 1$ выполнять без прерываний на процессоре l в интервале $[\Theta, \Theta + v_{p+1}/s_1]$; положить $\Theta = \Theta + v_{p+1}/s_1$. Если $\Theta + v_{p+1}/s_1 > T$, то работу $p + 1$ выполнять сначала процессором $l + 1$ в интервале $[0, (v_{p+1} - (T - \Theta)s_l)/s_{l+1}] \le \Theta$, а затем процессором l в интервале $[\Theta, T]$;

(такое переключение корректно, так как $(v_{p+1} - (T - \Theta)s_l)/s_{l+1} \le \Theta$, что, в свою очередь, следует из соотношений

$$(v_{p+1} - (T - \Theta)\, s_l)/s_{l+1} = (v_{p+1}/s_{l+1} - (T - \Theta)\, s_l)/s_{l+1} \le$$
$$\le v_1/s_m - (T - \Theta) = \tau_{max} - T + \Theta \le T - T + \Theta = \Theta;$$

положить $\Theta = (v_{p+1} - (T - \Theta)s_l)/s_{l+1}$. Далее шаг 3 выполнять для работ $p + 2, \ldots, n_2$ из N_2.

Лемма 1. Достаточным условием существования расписания длины, не превышающей $T \ge \max(Q_{\max}, \tau_{\max})$, является выполнение неравенства

$$(2) \quad \sum_{i \in N_2} v_i \le L_{j_0} s_{j_0} + T \sum_{j=m_1+1}^{m} s_j .$$

Доказательство леммы следует из описанного выше алгоритма упаковки. Переписав соотношение (2) в виде

$$\sum_{i \in N_2} v_i \le (T - Q_{j_0}) s_{j_0} + T \sum_{j=m_1+1}^{m} s_j ,$$

получаем, что при $T \ge \left(\sum_{i \in N_2} v_i + Q_{j_0} s_{j_0} \right) \Big/ \left(\sum_{j=m_1+1}^{m} s_j + s_{j_0} \right)$

существует расписание, длина которого не превосходит T. Отсюда следует, что достаточным условием существования расписания длины T является выполнение неравенства

$$(3) \quad T \ge T_{\max} = \max\left(\frac{\sum_{i \in N_2} v_i + Q_{\max} s_1}{\sum_{j=m_1+1}^{m} s_j + s_{m_1}}, Q_{\max}, \tau_{\max} \right).$$

5. Необходимое условие существования расписания заданной длины

Лемма 2. Необходимым условием существования расписания длины, не превосходящей T, является выполнение неравенства

(4) $$\sum_{i\in N_1} w_i + \sum_{i\in N_2} v_i \le T\sum_{j=1}^{m} s_j$$

(при условии, что работы из N_1 уже распределены).

Доказательство леммы следует из того, что невыполнение условия (4) означает превышение величины требуемого суммарного объема работы процессоров над величиной максимально возможного объема в интервале длиной T.

Из леммы 2 следует, что не существует расписания, длина которого меньше величины

(5) $$T_{\min} = \max\left(\frac{\sum_{i\in N_1} w_i + \sum_{i\in N_2} v_i}{\sum_{j=1}^{m} s_j},\ Q_{\max},\ \tau_{\min} \right).$$

6. Модифицированный алгоритм упаковки

Алгоритм распределяет работы множества N_2 сначала по процессорам первой группы (если это возможно), а затем – по процессорам второй группы. При этом предполагается, что работы из N_1 были уже распределены по процессорам первой группы. Длина построенного расписания не превосходит T. Если такого расписания не существует, алгоритм сообщает об этом.

Шаг 1. Расположить процессоры первой группы в порядке не убывания величин $L_j s_j$, а работы из N_2 – в порядке не убывания объемов. Будем считать, что

$$L_1 s_1 \le L_2 s_2 \le \ldots \le L_{m_1} s_{m_1};\ \ v_1 \le v_2 \le \ldots \le v_{n_2}.$$

Шаг 2. Положить $p = 1, j = 1$.

Шаг 3. Пусть p, $p + 1$, …, $n_2 \in N_2$ – работы, ранее не назначенные на процессоры. Если существует номер $k \ge p$ такой, что $\sum_{i=p}^{k} v_i \le L_j s_j$ и $\sum_{i=p}^{k+1} v_i > L_j s_j$, то назначить работы $p, \ldots, k \in N_2$ на процессор j, на котором они должны выполняться последовательно в интервале $\left[Q_j, Q_j + (\sum_{i=p}^{k} v_i)/s_j\right]$. Положить $L_j = L_j - (\sum_{i=p}^{k} v_i)/s_j$; $Q_j = Q_j + (\sum_{i=p}^{k} v_i)/s_j$; $p = k + 1$. Если $p > n$, то остановиться, все работы из N_2 назначены. Если $p \le n$, перейти на шаг 4.

Шаг 4. Положить $j = j + 1$. Если $j \le m_1$, то перейти на шаг 3; если $j > m_1$ – на шаг 5.

Шаг 5. Расположить процессоры первой группы в порядке невозрастания величин $L_j s_j$. Будем считать, что $L_1 s_1 \ge L_2 s_2 \ge \ldots \ge L_{m_1} s_{m_1} > 0$ (если $L_j = 0$ при некоторых j, то соответствующие процессоры первой группы исключаем из дальнейшего рассмотрения). Положить $\Theta = 0$; $j_1 = 1$; $j_2 = m_1 + 1$.

Шаг 6. Если $j_1 \le m_1$ и $\Theta + (v_p - L_{j_1} s_{j_1})/s_{j_2} \le Q_{j_1}$, перейти на шаг 7;

если $j_1 \le m_1$, $\Theta + (v_p - L_{j_1} s_{j_1})/s_{j_2} > Q_{j_1}$ и $\Theta + v_p / s_{j_2} \le T$, перейти на шаг 8;

если $j_1 \le m_1$, $\Theta + (v_p - L_{j_1} s_{j_1})/s_{j_2} > Q_{j_1}$ и $\Theta + v_p / s_{j_2} > T$, перейти на шаг 9;

если $j_1 > m_1$ и $\Theta + v_p / s_{j_2} \le T$, перейти на шаг 8;

если $j_1 > m_1$ и $\Theta + v_p / s_{j_2} > T$, перейти на шаг 9;

Шаг 7. Работу p назначить на процессор j_2 в интервале $[\Theta, \Theta + (v_p - L_{j_1} s_{j_1})/s_{j_2}]$ и на процессор j_1 в интервале $[Q_{j_1}, T]$. Положить $\Theta = \Theta + (v_p - L_{j_1} s_{j_1})/s_{j_2}$; $p = p + 1$; $j_1 = j_1 + 1$.

Перейти на шаг 11.

Шаг 8. Работу p назначить на процессор j_2 в интервале $[\Theta, \Theta + v_p / s_{j_2}]$. Положить $\Theta = \Theta + v_p / s_{j_2}$; $p = p + 1$.

Перейти на шаг 11.

Шаг 9. Работу p назначить на процессор $j_2 + 1$ в интервале $[0, (v_p - (T - \Theta)s_{j_2}) / s_{j_2+1}]$ и на процессор j_2 в интервале $[\Theta, T]$. Положить $\Theta = (v_p - (T - \Theta)s_{j_2}) / s_{j_2+1}$; $j_2 = j_2 + 1$; $p = p + 1$.

Перейти на шаг 10.

Шаг 10. Если $j_2 \le m_2$, перейти на шаг 11. Если $j_2 > m_2$, остановиться; не все работы из N_2 могут быть назначены на процессоры; расписание длины не более T не построено.

Шаг 11. Если $p > n$, остановиться; все работы из N_2 назначены на процессоры. Если $p \le n$, перейти на шаг 6.

Сделаем несколько замечаний. На шаге 3 алгоритма некоторые работы из N_2 назначаются на процессоры первой группы, на которых они выполняются без прерываний. На шаге 7 очередная работа $p \in N_2$ выполняется сначала на процессоре j_2 второй группы, а затем – на процессоре j_1 первой группы. Корректность такого переключения следует из того, что $\Theta + (v_p - L_{j_1} s_{j_1}) / s_{j_2} \le Q_{j_1}$. На шаге 9 работа p выполняется сначала на процессоре $j_2 + 1$, а затем на процессоре j_2. Такое переключение корректно, поскольку

$$(v_p - (T - \Theta)s_{j_2}) / s_{j_2+1} \le \Theta .$$

Это неравенство следует из соотношений:

$$(v_p - (T - \Theta)s_{j_2}) / s_{j_2+1} = v_p / s_{j_2+1} - (T - \Theta)s_{j_2} / s_{j_2+1} \le$$
$$\le v_1 / s_m - (T - \Theta) = \tau_{\max} - T + \Theta \le T - T + \Theta = \Theta.$$

7. Алгоритм решения исходной задачи

Поскольку решается задача на быстродействие, будем искать такое значение T^*, что расписание длины T^* существует, а расписания длины $T^* - 1$ не существует. Используя леммы 1, 2 и формулы (3), (5), этот поиск будем проводить в интервале $[T_{min}, T_{max}]$ с помощью дихотомической процедуры (деление отрезка пополам). При этом для выбранного значения T будем использовать модифицированный алгоритм упаковки.

Алгоритм решения исходной задачи

Шаг 1. По формулам (1) вычислить величины m_1, m_2.

Шаг 2. С помощью мультиоценочного алгоритма с калибровкой [3] построить расписание выполнения работ N_1 на m_1 процессорах.

Шаг 3. По формулам (3), (5) вычислить величины T_{min}, T_{max}.

Шаг 4. С помощью алгоритма деления отрезка $[T_{min}, T_{max}]$ пополам найти такое целое $\widetilde{T} \in [T_{min}, T_{max}]$, что при $T = \widetilde{T}$ модифицированный алгоритм упаковки строит расписание длины, не превосходящей T, а при $T = \widetilde{T} - \delta$ – нет, где $0 < \delta \leq 1$.

Расписание, построенное с помощью модифицированного алгоритма упаковки при $T = \widetilde{T}$, – это искомое расписание выполнения работ из N_2.

Отметим, что вычислительная сложность мультиоценочного алгоритма с калибровкой (шаг 2) – $O(n_1 \log n_1)$, а модифицированного алгоритма упаковки – $O(n_2 \log n_2)$. Сложность шагов 1 и 3 – $O(n_1 + n_2)$. Поскольку

$$T_{max} \leq T'_{max} = \max\left(\frac{\sum\limits_{i \in N_1} \tau_i + Q_{\min}}{m_2 + 1},\ Q_{\max},\ \tau_{\max}\right),$$

то число обращений к модифицированному алгоритму упаковки не более $\log(T'_{\max} - T_{\min})$. Поэтому сложность предложенного алгоритма составляет $O(n_1 \log n_1 + n_2\ \log n_2 \cdot \log(T'_{\max} - T_{\min}))$.

Опишем работу алгоритма на следующем примере: $n_1 = n_2 = 4$, $m = 4$, $s_i = 1$ $(i = 1, \ldots, 4)$, $w_1 = 5$, $w_2 = 1$, $w_3 = 4$, $w_4 = 4$, $v_1 = 3$, $v_2 = 1$, $v_3 = 5$, $v_4 = 4$. Пусть $N_1 = \{a_1, a_2, a_3, a_4\}$, $N_2 = \{b_1, b_2, b_3, b_4\}$. На рис. 1 изображена диаграмма распределения работ по процессорам при значениях $m_1 = m_2 = 2$, вычисленных по формуле (1). Мультиоценочный алгоритм с калибровкой распределил работы из N_1 следующим образом: на процессоре 1 выполняется работа a_1 в интервале [0; 5] и работа a_2 в интервале [5; 6], а на процессоре 2 – работа a_3 в интервале [0; 4] и работа a_4 в интервале [4; 8]. В этом случае $T = 8$. Модифицированный алгоритм упаковки назначает работы из N_2 следующим образом. На шаге 3 работа b_2 назначается на процессор 1 в интервале [6; 7]. На шаге 7 работа b_3 назначается на процессор 3 в интервале [0; 4] и на процессор 1 в интервале [7; 8]. На шаге 8 работа b_4 назначается на процессор 3 в интервале [4; 8], а работа b_1 – на процессор 4 в интервале [0; 3].

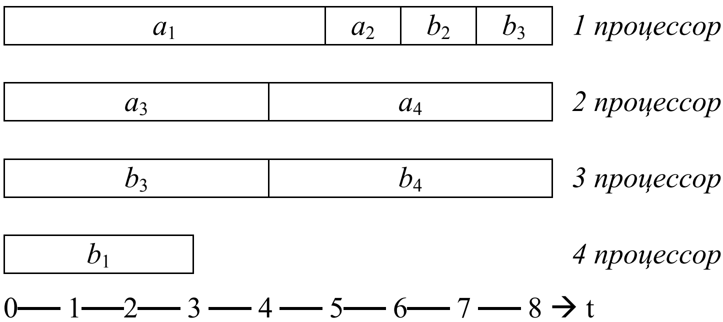

Рис. 1. Распределение работ по процессорам при $m_1 = m_2 = 2$

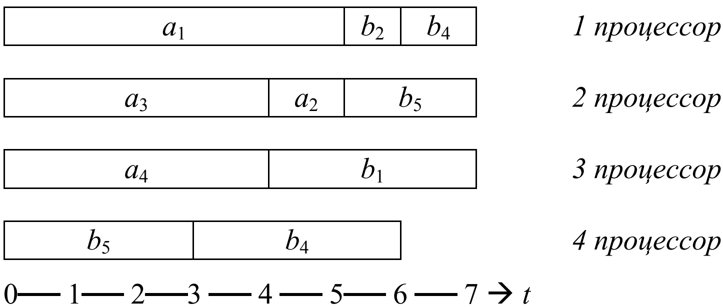

Рис. 2. Распределение работ по процессорам при $m_1' = 3$, $m_2' = 1$

На рис. 2 изображена диаграмма распределения работ по процессорам при $m_1' = 3$ и $m_2' = 1$. В этом случае $T = 7$.

Результаты численных экспериментов. Были проведены численные эксперименты. Число работ n полагалось равным 100, 400 и 1000, а число процессоров m – 20, 60 и 100. Эксперименты проводились для различных значений числа n_1 непрерываемых работ и числа n_2 прерываемых работ. Для каждого набора значений n, m, n_1, n_2 проводилось по 50 экспериментов со значениями объемов работ и производительностей процессоров, полученными с помощью программного генератора случайных чисел, позволяющего получать псевдослучайные числа с равномерным распределением на заданном множестве. Для объемов работ таким множеством был отрезок [1, 2600], а для производительностей процессоров – отрезки [1, 4] и [1, 16]. Расчеты проводились для различных комбинаций упорядочения (по не убыванию и не возрастанию) процессоров первой группы (относительно величин $L_j s_j$) и работ из N_2 (относительно их объемов v_i). В таблице 1 приведены результаты для случая, когда работы из N_2 упорядочены по невозрастанию объёмов, а процессоры первой группы – по невозрастанию величин $L_j s_j$, поскольку в этом случае погрешность была наименьшей.

В каждом эксперименте вычислялись значения m_1 и m_2, задаваемые формулами (1), и среднее значение Δ оценки погрешности (по 50 расчетам) для каждого набора n, m, n_1, n_2. Оценка относительной погрешности алгоритма вычислялась по формуле $\Delta = \left(\widetilde{T} - T^*\right)/T^* \times 100\%$, где

$$T^* = \left(\sum_{i \in N_1} w_i + \sum_{i \in N_2} v_i \right) / \sum_{j=1}^{m} s_j \ .$$

(T^* не превосходит длины оптимального расписания.)

Далее аналогичные расчеты проводились для всевозможных разбиений процессоров на две группы, соответствующих значениям m_1' и m_2' $(m_1' + m_2' = m)$, и для каждого такого разбиения вычислялось среднее значение оценки погрешности Δ'.

Из результатов численных экспериментов можно сделать следующие выводы.

1. При упорядочении работ из N_2 по невозрастанию их объемов v_i, а процессоров первой группы – по невозрастанию величин $L_j s_j$ погрешность существенно меньше, чем в остальных

случаях. Если не считать вырожденных примеров (когда $n_2 = 1$), погрешность Δ' в этих случаях не превосходила 2,7 %.

2. Значения m'_1 и m'_2 не более чем на единицу отличаются от значений m_1 и m_2, задаваемых формулой (1).

3. С ростом числа непрерываемых работ, как правило, растет доля экспериментов, в которых $\Delta' < \Delta$, а также растут значения Δ, Δ' и $\Delta - \Delta'$.

Таблица 1. Результаты численных экспериментов

№	n	M	s_j	Δ'
1	100	20	[1; 4]	0 – 2
2	400	60	[1; 4]	0 – 1
3	1000	100	[1; 4]	0 – 0,2
4	100	20	[1; 16]	0 – 2
5	400	60	[1; 16]	0 – 2,7
6	1000	100	[1; 16]	0 – 0,5

Литература

1. БУЛАНЖЕ Д. Ю. *Оптимальная коррекция директивных интервалов для задачи одного прибора.* – М.: ВЦ АН СССР, 1983
2. БУЛАНЖЕ Д. Ю., СУШКОВ Б. Г. *Оптимальная коррекция директивных интервалов для задачи одного прибора* // Изв. АН СССР, Техн. кибернетика. – 1982. – №6. – С. 160–169.
3. ГОНЧАР Д. Р. *Мультиоценочный алгоритм решения минимаксной задачи составления расписания* // Системы управления и информационные технологии. – 2007. – №1.3(27). – С. 324–328.
4. СКИНДЕРЕВ С. А., ФУРУГЯН М. Г. *Алгоритмы планирования вычислений в многопроцессорных системах с неоднородным множеством работ*. – М.: ВЦ РАН, 2006.
5. ФУРУГЯН М. Г., ГОНЧАР Д. Р. *Минимаксная задача планирования вычислений в многопроцессорной системе со*

смешанным набором работ // Системы управления и информационные технологии. – 2009. – №2(36). – С. 36–39.

ALGORITHMS OF CONTROL IN MULTIPROCESSOR SYSTEM WITH MIXED JOBS SET

Dmitry R. Gonchar, Computing Center of RAS, Moscow, Cand.Sc. (rtsccas@yandex.ru).
Meran G. Fourougian, Computing Center of RAS, Moscow, Cand.Sc. assistant professor (rtsccas@yandex.ru).

Abstract: *We propose an approximate algorithm to build the speed-optimal schedule for the set of heterogeneous jobs. Some jobs admit interruption while others do not. We allow for the arbitrary processors' speed. We use the multicoasting algorithm with calibration and the modified algorithm of packing.*

Keywords: multiprocessor system, interruptible and non-interruptible jobs, scheduling.

Статья представлена к публикации членом редакционной коллегии В. Н. Лебедевым