$\in {}^{m}(\quad)\qquad(\quad)\ \overline{\|\qquad\|_1}$,

УДК 681.3.06:519

Н.С. КОВАЛЕНКО, П.А. ПАВЛОВ, М.И. ОВСЕЕЦ

ЗАДАЧИ ОПТИМИЗАЦИИ ЧИСЛА ПРОЦЕССОРОВ И ПОСТРОЕНИЯ ОПТИМАЛЬНОЙ КОМПОНОВКИ РАСПРЕДЕЛЕННЫХ СИСТЕМ

The algorithm of construction of optimum configuration of set of the distributed processes competing for use of the linearly-structured program resource, and method of optimization of number of processors are offered at the distributed processing.

Предлагаются алгоритм построения оптимальной компоновки множества одинаково распределенных конкурирующих процессов и метод нахождения оптимального числа процессоров, обеспечивающих директивное время выполнения заданных объемов вычислений.

1. Основные понятия и определения. Математическая модель системы распределенной обработки конкурирующих процессов включает в себя: ,

s $s \ge 2,$ – число блоков линейно-структурированного программного ресурса $PR = (Q_1, Q_2, ..., Q_s)$; $n, n \ge 2,$ – число распределенных относительно *PR* конкурирующих процессов; $p, p \ge 2,$ – число процессоров многопроцессорной системы (МС); матрицу $T = [t_{ij}]$ времен выполнения *j*-х блоков *i*-ми конкурирующими процессами $i = \overline{1,n},\ \ j = \overline{1,s};$ ε – время, характеризующее дополнительные системные расходы по организации структурирования и параллельного использования блоков *PR* [1].

Так же как и в [1–5], будем считать, что взаимодействие процессов, процессоров и блоков линейно-структурированного программного ресурса подчинено следующим условиям: 1) ни один из блоков *PR* не может обрабатываться одновременно более чем одним процессором; 2) ни один из процессоров не может обрабатывать одновременно более одного блока; 3) обработка каждого блока осуществляется без прерываний; 4) распределение блоков программного ресурса по процессорам МС для каждого из процессов осуществляется циклически по правилу: блок с номером $j = kp + i,\ \ j = \overline{1,s},$ $i = \overline{1,p},\ \ k \ge 0,$ распределяется на процессор с номером *i*. Кроме того, введем дополнительные условия, которые определяют режимы взаимодействия процессов, процессоров и блоков *PR*: 5) отсутствуют простои процессоров при условии готовности блоков, а также невыполнение блоков при наличии процессоров; 6) для каждого из *n* процессов момент завершения выполнения *j*-го блока

на i-м процессоре совпадает с моментом начала выполнения следующего (j+1)-го блока на (i+1)-м процессоре, $i=\overline{1,p-1}$, $j=\overline{1,s-1}$; 7) для каждого из блоков структурированного программного ресурса момент завершения его выполнения l-м процессом совпадает с моментом начала его выполнения (l+1)-м процессом на том же процессоре, $l=\overline{1,n-1}$.

Условия 1) – 5) определяют *асинхронный режим* взаимодействия процессоров, процессов и блоков, который предполагает отсутствие простоев процессоров МС при условии готовности блоков, а также невыполнение блоков при наличии процессоров.

Если к условиям 1) – 4) добавить условие 6), то получим *первый синхронный режим*, обеспечивающий непрерывное выполнение блоков программного ресурса внутри каждого из вычислительных процессов.

Второй синхронный режим, определяемый условиями 1) – 4), 7), обеспечивает непрерывное выполнение каждого блока всеми процессами.

2. Задача оптимизации числа процессоров при распределенной обработке. Задача состоит в том, чтобы при заданных n, s, ε, $[t_{ij}]$, $i=\overline{1,n}$, $j=\overline{1,s}$, и заданном директивном времени выполнения всех распределенных конкурирующих процессов d найти оптимальное число процессоров p^*, обеспечивающих директивное время выполнения. Решение данной задачи рассмотрим для асинхронного режима, когда процессы являются *неоднородными*, т. е. когда времена выполнения блоков программного ресурса $Q_1, Q_2, ..., Q_s$ зависят от объемов обрабатываемых данных и (или) их структуры (разные для разных процессов).

Для изложения метода решения поставленной задачи кроме параметров математической модели p, n, s, ε, d введем двумерный массив переменной длины M_q, составленный специальным образом из элементов матрицы $[t_{ij}^{\varepsilon}]$, где $t_{ij}^{\varepsilon}=t_{ij}+\varepsilon$, $i=\overline{1,n}$, $j=\overline{1,s}$, q, $q\in N$, – порядковый номер результирующей матрицы времен выполнения блоков (двумерного массива переменной длины M_q), а также приведенные далее определение и теорему.

Число процессоров МС будем называть *достаточным* и обозначать p^s при заданных n, s, если $p^s=s$.

Обозначим через $T_{\text{н}}^{\text{ac}}(p^s,n,s,\varepsilon)$ минимальное общее время выполнения множества конкурирующих процессов при достаточном числе процессоров p^s, а $T_{\text{н}}^{\text{ac}}(p,n,s,\varepsilon)$ – минимальное общее время при заданном (исходном) числе процессоров p. Имеет место следующая

Теорема 1. При заданных n, s, ε, $[t_{ij}]$, $i=\overline{1,n}$, $j=\overline{1,s}$, в случае достаточного ($p^s=s$) и ограниченного ($p<s$) числа процессоров МС имеет место соотношение $T_{\text{н}}^{\text{ac}}(p^s,n,s,\varepsilon)\le T_{\text{н}}^{\text{ac}}(p,n,s,\varepsilon)$ [2].

Данная теорема является отправной точкой для построения метода нахождения оптимального числа процессоров p^*, обеспечивающих директивное время d выполнения неоднородных конкурирующих процессов при распределенной обработке в условиях асинхронного режима их взаимодействия.

Входные данные: p, $p\ge 2$, – заданное (исходное) число процессоров; n, $n\ge 2$, – число конкурирующих неоднородных распределенных процессов; s, $s\ge 2$, – число блоков линейно-структурированного программного ресурса; M – двумерный массив, содержащий элементы исходной матрицы с учетом дополнительных системных расходов ε $[t_{ij}^{\varepsilon}]$, $i=\overline{1,n}$, $j=\overline{1,s}$; d – заданное (директивное) время выполнения конкурирующих процессов.

Выходные данные: p^* – минимальное (оптимальное) число процессоров, обеспечивающих выполнение конкурирующих процессов за директивное время d; M_q – двумерный массив, содержащий результирующую матрицу времен выполнения блоков программного ресурса [1]; q – порядковый номер результирующей матрицы времен выполнения блоков программного ресурса PR.

Алгоритм.

Если $d<T_{\text{н}}^{\text{ac}}(p^s,n,s,\varepsilon)$, то полагаем $p^*=0$, т. е. директивное время выполнения конкурирующих процессов d не может быть реализовано в заданных условиях ни для какого числа процессоров.

Пусть $d \ge T_{\text{н}}^{\text{ac}}(p^s, n, s, \varepsilon)$ и число процессоров МС является ограниченным, т. е. $s > p$. Тогда возможны следующие случаи: если $T_{\text{н}}^{\text{ac}}(p^s, n, s, \varepsilon) \le d = T_{\text{н}}^{\text{ac}}(p, n, s, \varepsilon)$ или $d > T_{\text{н}}^{\text{ac}}(p, n, s, \varepsilon)$, то нахождение p^* осуществляется методом деления пополам отрезка $[2, p]$; если $T_{\text{н}}^{\text{ac}}(p^*, n, s, \varepsilon) \le d < T_{\text{н}}^{\text{ac}}(p, n, s, \varepsilon)$, то нахождение p^* производится методом деления пополам отрезка $[p, p^s]$.

Пусть $s \le p$. Тогда p^* находим путем деления отрезка $[2, p^s]$ пополам.

Нетрудно подсчитать, что сложность алгоритма нахождения оптимального числа процессоров p^*, базирующегося на предложенном методе, составляет $O((k+1)np\log_2 p)$ операций в худшем случае.

Пример 1. Пусть $p = 3$, $n = 3$, $s = 9$, $d = 48$, а исходная матрица времен выполнения блоков с учетом дополнительных системных расходов по организации структурирования и параллельного использования блоков $PR\ \varepsilon$ имеет вид:

$$T^{\varepsilon} = M = \begin{bmatrix} 4 & 1 & 3 & 5 & 2 & 4 & 7 & 3 & 1 \\ 2 & 6 & 4 & 1 & 5 & 3 & 4 & 2 & 8 \\ 5 & 3 & 1 & 7 & 4 & 2 & 6 & 4 & 5 \end{bmatrix}.$$

В данном случае достаточное число процессоров $p^s = 9$.

1. По исходной матрице M строим вершинно-взвешенный граф G_1^{ac} [1] (рис. 1) и находим величину $T_{\text{н}}^{\text{ac}}(p^s = 9, n, s, \varepsilon) = 45$.

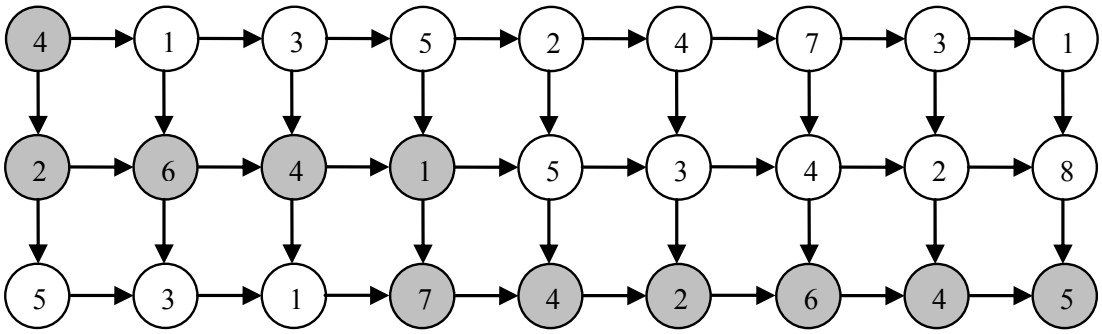

Рис. 1. Вершинно-взвешенный граф G_1^{ac}

По исходным данным p, n, s и M строим результирующую матрицу T^* вида:

$$T^* = \begin{bmatrix} \hat{4} & 1 & 3 & 5 & 2 & 4 & 7 & 3 & 1 \\ \hat{2} & \hat{6} & \hat{4} & \hat{1} & 5 & 3 & 4 & 2 & 8 \\ 5 & 3 & 1 & \hat{7} & 4 & 2 & 6 & 4 & 5 \\ 5 & 2 & 4 & \hat{7} & 3 & 1 & 0 & 0 & 0 \\ 1 & 5 & 3 & \hat{4} & 2 & 8 & 0 & 0 & 0 \\ 7 & 4 & 2 & \hat{6} & \hat{4} & \hat{5} & 0 & 0 & 0 \\ 7 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 2 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

С помощью этой матрицы находим величину $T_{\text{н}}^{\text{ac}}(p = 3, n, s, \varepsilon) = 50$, которая и будет определять минимальное общее время выполнения неоднородных распределенных конкурирующих процессов в асинхронном режиме на $p = 3$ процессорах.

Учитывая, что $T_{\text{н}}^{\text{ac}}(p^s = 9, n, s, \varepsilon) = 45 \le d = 48 < T_{\text{н}}^{\text{ac}}(p = 3, n, s, \varepsilon) = 50$, рассмотрим отрезок $[3, 9]$.

2. Методом деления отрезка $[3,9]$ пополам находим $p_1 = 6$ и строим по заданным n, s и полученному $p_1 = 6$ результирующую матрицу M_1 вида:

$$M_1 = \begin{bmatrix} \hat{4} & 1 & 3 & 5 & 2 & 4 & 7 & 3 & 1 & 0 & 0 & 0 \\ \hat{2} & \hat{6} & \hat{4} & \hat{1} & 5 & 3 & 4 & 2 & 8 & 0 & 0 & 0 \\ 5 & 3 & 1 & \hat{7} & \hat{4} & \hat{2} & \hat{6} & \hat{4} & \hat{5} & 0 & 0 & 0 \\ 7 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 2 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

С помощью матрицы M_1 вычисляем величину $T_{\text{н}}^{\text{ac}}(p_1 = 6, n, s, \varepsilon) = 45$. Так как $T_{\text{н}}^{\text{ac}}(p_1 = 6, n, s, \varepsilon) = 45 \le d = 48,$ то рассматриваем отрезок $[3, 6]$.

3. Методом деления отрезка $[3, 6]$ пополам находим $p_2 = 4,$ причем в качестве p_2 берем величину, которая является наименьшим целым полусуммы чисел 3 и 6. Далее по заданным *n*, *s* и полученному значению $p_2 = 4$ строим результирующую матрицу M_2:

$$M_2 = \begin{bmatrix} \hat{4} & 1 & 3 & 5 & 2 & 4 & 7 & 3 & 1 & 0 & 0 & 0 \\ \hat{2} & \hat{6} & \hat{4} & \hat{1} & 5 & 3 & 4 & 2 & 8 & 0 & 0 & 0 \\ 5 & 3 & 1 & \hat{7} & \hat{4} & \hat{2} & \hat{6} & \hat{4} & \hat{5} & 0 & 0 & 0 \\ 5 & 2 & 4 & 7 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 5 & 3 & 4 & 2 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 7 & 4 & 2 & 6 & 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 7 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 2 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 4 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

С помощью матрицы M_2 вычисляем величину $T_{\text{н}}^{\text{ac}}(p_2 = 4, n, s, \varepsilon) = 45.$ Таким образом, директивное время $d = 48$ выполнения $n = 3$ процессов реализуется при $p_2 = 4,$ так как $d = 48 > T_{\text{н}}^{\text{ac}}(p_2 = 4, n, s, \varepsilon) = 45,$ и не реализуется при $p = 3,$ так как $d = 48 < T_{\text{н}}^{\text{ac}}(p = 3, n, s, \varepsilon) = 50.$ Следовательно, $p^* = 4.$

3. Свойства оптимальных компоновок и вспомогательные результаты. Система взаимодействующих конкурирующих процессов называется *одинаково распределенной*, если времена t_{ij} выполнения блоков $Q_j,$ $j = \overline{1,s},$ программного ресурса *PR* каждым из *i*-х процессов совпадают и равны t_i для всех $i = \overline{1,n},$ т. е. справедлива цепочка равенств $t_{i1} = t_{i2} = \ldots = t_{is} = t_i$ для всех $i = \overline{1,n}.$

В [3, 4] доказано, что для одинаково распределенных систем конкурирующих процессов минимальное общее время для всех трех базовых режимов в случае *неограниченного параллелизма* $(s \le p)$ и в случае *ограниченного параллелизма* $(s > p),$ но если $T_\varepsilon^n \le pt_{\max}^\varepsilon,$ определяется по формуле

$$T(p, n, s, \varepsilon) = T_\varepsilon^n + (s-1)t_{\max}^\varepsilon, \tag{1}$$

а в остальных случаях общее время выполнения *n* одинаково распределенных процессов, использующих структурированный на *s* блоков программный ресурс *PR*, в вычислительной среде с *p* процессорами при асинхронном режиме и в режиме непрерывного выполнения каждого блока всеми процессами составляет величину

$$T(p, n, s, \varepsilon) = \begin{cases} kT_\varepsilon^n + (p-1)t_{\max}^\varepsilon & \text{при } s = kp,\ k > 1,\ T_\varepsilon^n > pt_{\max}^\varepsilon, \\ (k+1)T_\varepsilon^n + (r-1)t_{\max}^\varepsilon & \text{при } s = kp + r,\ k \ge 1, 1 \le r < p,\ T_\varepsilon^n > pt_{\max}^\varepsilon, \end{cases} \tag{2}$$

где $T_\varepsilon^n = \sum_{i=1}^{n} t_i^\varepsilon$ – суммарное время выполнения каждого из блоков Q_j всеми *n* процессами с учетом накладных расходов $\varepsilon,$ $t_{\max}^\varepsilon = \max_{1 \le i \le n} t_i^\varepsilon,$ $t_i^\varepsilon = t_i + \varepsilon,$ $i = \overline{1,n}.$

С понятием множества одинаково распределенных конкурирующих процессов свяжем понятие линейной упаковки.

Пусть $M = \{m_1, m_2, \ldots, m_n\}$ – конечное упорядоченное множество предметов. *Линейной упаковкой* множества *M* ранга *l* будем называть разбиение множества *M* на *l* непересекающихся подмножеств $M_1, M_2, \ldots, M_l$, причем каждое подмножество является объединением последовательных элементов множества *M*.

Так как времена выполнения блоков программного ресурса каждым из процессов совпадают $t_{i1} = t_{i2} = \ldots = t_{is} = t_i, \ i = \overline{1,n},$ то в качестве элементов множества *M* будем рассматривать последовательность, например, первых блоков $(Q_{11}, Q_{21}, \ldots, Q_{n1})$ структурированного программного ресурса одинаково распределенных процессов, которую обозначим $(q_1, q_2, \ldots, q_n)$. В этом случае линейная упаковка множества *M* получается объединением блоков $q_i, \ i = \overline{1,n},$ последовательных процессов в один программный блок. Линейную упаковку блоков $q_i, \ i = \overline{1,n},$ которая приведет к уменьшению количества процессов в МС, будем называть *линейной компоновкой* и обозначать LC_l.

Обозначим через K множество всевозможных компоновок блоков системы одинаково распределенных процессов, конкурирующих за использование программного ресурса *PR*, а через K_l – множество компоновок ранга *l*, $l = \overline{1,n}$. Отметим, что компоновкой ранга *n* является исходная одинаково распределенная система $LC_n = (q_1, q_2, \ldots, q_n)$, а ранга 1 – компоновка блоков в один программный блок $LC_1 = (q_1 \cup q_2 \cup \ldots \cup q_n)$. Нетрудно подсчитать, что $|K| = 2^{n-1}, \ |K_l| = C_{n-1}^{l-1} = \dfrac{(n-1)!}{(l-1)!(n-l)!}$.

Пусть $LC_l = (q_1', q_2', \ldots, q_l')$ – линейная компоновка блоков.

Обозначим: $t(q_i') = \sum_{q \in q_i'} t(q)$ – время выполнения *i*-го элемента компоновки $LC_l, \ i = \overline{1,l}$; $t(LC_l) = (t(q_1'), t(q_2'), \ldots, t(q_l'))$ – последовательность времен выполнения блоков $q_{i,}', \ i = \overline{1,l}$; $t_{\max}(LC_l) = \max_{1 \le i \le l} \{t(q_i')\}$ – время выполнения максимального блока компоновки LC_l; $t_{\min} = \min \{t_{\max}(LC_l) \mid LC_l \in K_l\}$.

Задача оптимальной компоновки блоков $(q_1, q_2, \ldots, q_n)$ множества одинаково распределенных конкурирующих процессов состоит в том, чтобы при заданных $p \ge 2, \ n \ge 2, \ s \ge 2, \ \varepsilon > 0$ найти такую линейную компоновку LC_l исходной одинаково распределенной системы, при которой достигается минимум функционалов (1) и (2). Такую компоновку будем называть *оптимальной*.

Для решения поставленной задачи потребуются следующие результаты [5].

Теорема 2. Если LC_l – оптимальная линейная компоновка одинаково распределенной системы, то компоновка LC_l', такая, что $t_{\max}(LC_l') = t_{\min}$, также является оптимальной.

Теорема 3. Если для компоновок LC_l и $LC_{l-1}, \ l > 2, \ t_{\max}(LC_l) = t_{\max}(LC_{l-1})$, то $T(p, LC_l, s, \varepsilon) > T(p, LC_{l-1}, s, \varepsilon)$.

Из теоремы 2 следует, что, если для каждого ранга $l = 2, \ldots, n,$ можно эффективно строить линейную компоновку LC_l блоков систем одинаково распределенных процессов с наименьшим максимальным элементом среди компоновок этого ранга $(t_{\max}(LC_l) = t_{\min})$, то «эффективно» будет решена исходная задача, поскольку в этом случае оптимальную компоновку необходимо будет выбирать из $(n-1)$ компоновок.

Очевидно также, что наименьший максимальный элемент среди компоновок ранга *l* с убыванием *l* не убывает, т. е. $t_{\min}(LC_{l_1}) \ge t_{\min}(LC_{l_2}), \ 1 < l_1 < l_2 \le n,$ что позволяет при решении задачи оптимальной компоновки исключить из рассмотрения компоновку в один программный блок.

С практической точки зрения естественным является предположение $\varepsilon \le t_i, \ i = \overline{1,n},$ что позволяет при решении задачи оптимальной компоновки исключить из рассмотрения компоновку в один программный блок.

Наряду с исходной задачей рассмотрим следующую оптимизационную задачу «линейной упаковки в контейнеры».

Для заданных предметов конечного упорядоченного множества $M = \{m_1, m_2, \ldots, m_n\}$ и соответствующей последовательности их размеров $v(m_1), v(m_2), \ldots, v(m_n), v(m_i) > 0, i = \overline{1,n},$ числа $B > 0$ – вместимости контейнера, $B \geq \max\limits_{1 \leq i \leq n}\{v(m_i)\},$ требуется найти такую линейную упаковку множества M, чтобы размер каждого элемента упаковки $v(M_l)$ не превосходил B и l было наименьшим.

В общем случае, т. е. когда отсутствует условие линейности упаковки, эта задача является *NP*-трудной в сильном смысле, поскольку при $v(m_i) \in (0,1), i = \overline{1,n}, B = 1,$ дает классическую оптимизационную задачу упаковки в контейнеры. Условие линейности упаковки, связанное с задачей оптимальной компоновки блоков одинаково распределенных систем, существенно упрощает ее решение.

Задача линейной упаковки в контейнеры эффективно решается с помощью следующего *LF*-алгоритма (last-fit):

1) первый предмет m_1 загружается в первый контейнер, а остальные предметы – в порядке возрастания их номеров;

2) предмет $m_i, i = \overline{2,n},$ загружается в последний контейнер из числа частично упакованных, если сумма помещенных в него предметов не превосходит $B - m_i,$ в противном случае он загружается в следующий пустой контейнер.

Оптимальность линейной упаковки, которую строит *LF*-алгоритм, легко доказывается методом от противного. *LF*-алгоритм требует не более $3n$ элементарных операций и является составной частью алгоритма решения исходной задачи оптимальной компоновки.

4. Алгоритм построения оптимальной компоновки. Пусть $P_n = (t_1, t_2, ..., t_n)$ – последовательность времен выполнения каждого из блоков $q_i, i = \overline{1,n},$ всеми n процессами, $n \geq 3,$ $p \geq 2$ – число процессоров, ε – время, характеризующее дополнительные системные расходы, $\varepsilon \leq t_i, i = \overline{1,n}.$

Алгоритм построения оптимальной линейной компоновки блоков состоит из следующих этапов.

1) Строим массив из $\frac{n(n+1)}{2} - 1$ чисел $x_{ij}, i = \overline{2,n}, j = \overline{1,i},$ по правилу:

$$\begin{aligned} x_{nj} &= t_j, \quad j = \overline{1,n}, \\ x_{n-1,j} &= x_{nj} + t_{j+1}, \quad j = \overline{1,n-1}, \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ x_{n-k,j} &= x_{n-k+1,j} + t_{j+k}, \quad j = \overline{1,n-k}, \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ x_{2j} &= x_{3j} + t_{j+n-2}, \quad j = 1,2. \end{aligned} \tag{3}$$

Здесь числа $x_{ij} = t_j + t_{j+1} + \ldots + t_{j+n-i}$ представляют собой длительности всевозможных линейных компоновок блоков.

2) Упорядочиваем числа x_{ij} по возрастанию с одновременным удалением избыточных одинаковых элементов и элементов $x_{ij} < \max\limits_{1 \leq j \leq n}\{t_j\}.$ В результате получим возрастающую последовательность чисел $v_1 < v_2 < \ldots < v_k,$ для которой $v_1 \leq \max\limits_{1 \leq j \leq n}\{t_j\},$ $n - 1 \leq k < \frac{n(n+1)}{2} - 1.$

3) Полагаем $T_0 = T(p, n, s, \varepsilon),$ $P_0 = P_n,$ $l_0 = n,$ $i = 1.$

4) Принимая вместимость B равной $v_i, i = \overline{1,k},$ к исходному множеству одинаково распределенных конкурирующих процессов применяем *LF*-алгоритм линейной упаковки. Пусть l_i – ранг полученной компоновки блоков $(q_1, q_2, ..., q_n).$

5) Если $l_i = l_{i-1},$ то полученную компоновку LC_i не принимаем в рассмотрение, вычисляем $i = i + 1$ и переходим к п. 4.

6) Вычисляем значение $T_i = T(p, LC_{l_i}, s, \varepsilon)$. Если $T_i < T_0$, то полагаем $T_0 = T_i$, $P_0 = P_t(LC_{l_i})$, иначе T_0 и P_0 оставляем без изменений.

7) Если $l_i > 2$, то вычисляем $i = i + 1$ и переходим к п. 4, иначе $l_i = 2$. Алгоритм заканчивает работу.

После окончания работы алгоритма T_0 будет давать минимальное значение функционалов (1), (2), P_0 – оптимальную компоновку.

Приведенный алгоритм требует не более $O(n^3)$ элементарных операций, поскольку на первом этапе для построения массива чисел x_{ij}, $i = \overline{2,n}$, $j = \overline{1,i}$, требуется $O(n^2)$ элементарных операций, на втором, используя быстрые алгоритмы сортировки, – $O(n^2 \log_2 n)$, на этапе 4 в цикле по v_i – не более $O(n^3)$.

Пример 2. Пусть $P_9 = (4, 1, 9, 2, 4, 4, 1, 4, 2)$ – последовательность времен выполнения блоков q_i, $i = \overline{1,9}$, $p = 3$ – число процессоров, $s = 5$ – число блоков линейно-структурированного программного ресурса, $\varepsilon = 0{,}1$ – дополнительные системные расходы на каждый блок, связанные со структурированием и конвейеризацией.

Поскольку в этом случае $5 = s > p = 3$ и суммарное время выполнения каждого из блоков Q_j всеми n процессами с учетом накладных расходов ε равно

$$T_\varepsilon^n = \sum_{i=1}^{n} t_i^\varepsilon = \sum_{i=1}^{9} t_i + n\varepsilon = 31 + 9 \cdot 0{,}1 = 31{,}9 > 27{,}3 = 3(9 + 0{,}1) = pt_{\max}^\varepsilon,$$

то общее время выполнения $n = 9$ одинаково распределенных процессов, использующих структурированный на $s = 5$ блоков программный ресурс *PR*, в вычислительной среде с $p = 3$ процессорами при асинхронном режиме и в режиме непрерывного выполнения каждого блока всеми процессами согласно второй формуле в (2) составит величину

$$T(3; 9; 5; 0{,}1) = (k+1)T_\varepsilon^n + (r-1)t_{\max}^\varepsilon = 2 \cdot 31{,}9 + 1 \cdot 9{,}1 = 72{,}9 \text{ единиц времени.}$$

Найдем линейную компоновку LC_l исходной одинаково распределенной системы, при которой достигается минимум функционалов (1) и (2) с помощью приведенного ранее алгоритма.

Строим массив из $\frac{9(9+1)}{2} - 1 = 44$ чисел x_{ij}, $i = \overline{2,9}$, $j = \overline{1,i}$, согласно правилу (3). На рис. 2 приведена схема формирования этих чисел. Они представляют собой сумму чисел, стоящих у основания соответствующего треугольника.

i	*j*								
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
2	29	27							
3	25	25	26						
4	24	21	24	17					
5	20	20	20	15	15				
6	16	16	19	11	13	11			
7	14	12	**15**	10	9	9	7		
8	5	10	11	6	8	5	5	6	
9	4	1	9	2	4	4	1	4	2

Рис. 2. Массив чисел x_{ij}

Упорядочивая x_{ij} по возрастанию с одновременным удалением избыточных одинаковых элементов и элементов $x_{ij} < 9$, получим следующую возрастающую последовательность чисел (9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 20, 21, 24, 25, 26, 27, 29).

Принимая последовательно вместимость *B* равной значениям элементов последовательности, к исходной системе одинаково распределенных конкурирующих процессов применяем *LF*-алгоритм до тех пор, пока он не даст компоновку ранга 2. Вычисления, проводящиеся на каждом шаге, представлены в таблице.

Линейные компоновки

B	$P_t(LC_i)$	l_i	$T(p, LC_{l_i}, s, \varepsilon)$
9	**5, 9, 6, 9, 2**	**5**	$T_{0,1}^5 = 31+0,5 = 31,5 > 27,3 = 3(9+0,1) = pt_9^{0,1}$, тогда $T(3;5;5;0,1) = (1+1)T_{0,1}^5 + (2-1)t_9^{0,1} = 2\cdot 31,5 + 1\cdot 9,1 = 72,1$
10	5, 9, 10, 7	4	$T_{0,1}^4 = 31+0,4 = 31,4 > 30,3 = 3(10+0,1) = pt_{10}^{0,1}$, тогда $T(3;4;5;0,1) = (1+1)T_{0,1}^4 + (2-1)t_{10}^{0,1} = 2\cdot 31,4 + 1\cdot 10,1 = 72,9$
11	5, 11, 9, 6	4	$T_{0,1}^4 = 31+0,4 = 31,4 < 33,3 = 3(11+0,1) = pt_{11}^{0,1}$, тогда $T(3;4;5;0,1) = T_{0,1}^4 + (5-1)t_{11}^{0,1} = 31,4 + 4\cdot 11,1 = 75,8$
15	14, 15, 2	3	$T_{0,1}^3 = 31+0,3 = 31,3 < 45,3 = 3(15+0,1) = pt_{15}^{0,1}$, тогда $T(3;3;5;0,1) = T_{0,1}^3 + (5-1)t_{15}^{0,1} = 31,3 + 4\cdot 15,1 = 91,7$
16	16, 15	2	тогда

Как видно из таблицы, оптимальной компоновкой будет для которой и

Таким образом, общее время выполнения исходной одинаково распределенной системы улучшено за счет оптимальной компоновки на единиц времени, или на

Правильность работы предложенного алгоритма можно проверить, используя результаты, полученные в работах [3, 4]. Так, по данным примера, число процессов что подтверждает ранг оптимальной компоновки блоков.

Полученные результаты и предложенные алгоритмы могут быть использованы при проектировании многопроцессорных систем параллельной распределенной обработки.

1. Коваленко Н.С., Самаль С.А. Вычислительные методы реализации интеллектуальных моделей сложных систем. Мн., 2004.
2. Коваленко Н.С., Метельский В.М., Самаль С.А. // Кибернетика и системный анализ. 2003. № 6. С. 52.
3. Капитонова Ю.В., Коваленко Н.С., Павлов П.А. // Там же. 2005. № 6. С. 3.
4. Коваленко Н.С., Павлов П.А. // Докл. НАН Беларусі. Сер. фіз.-мат. навук. 2006. № 2. С. 25.
5. Капитонова Ю.В., Овсеец М.И. // Кибернетика. 1986. № 1. С. 5.

Поступила в редакцию 02.02.11.

Николай Семенович Коваленко – доктор физико-математических наук, профессор кафедры высшей математики БГЭУ.
Павел Александрович Павлов – кандидат физико-математических наук, доцент кафедры высшей математики и информационных технологий Полесского государственного университета.
Михаил Иванович Овсеец – кандидат физико-математических наук, первый проректор Частного института предпринимательства.