АЛГОРИТМ ПОСТРОЕНИЯ ОПТИМАЛЬНОЙ КОМПОНОВКИ РАСПРЕДЕЛЕННЫХ СИСТЕМ

П. А. Павлов

Полесский государственный университет, Пинск, Республика Беларусь
E-mail: pavlov.p@polessu.by

Предлагается алгоритм оптимальной компоновки блоков структурированного программного ресурса в системах распределенных конкурирующих процессов.

ALGORITHM FOR CONSTRUCTING OPTIMAL LAYOUT DISTRIBUTED SYSTEMS

P. A. Pavlov

The algorithm of optimal arrangement of structured program resource blocks in systems of distributed competing processes is offered.

Введение. Решение “больших” задач неразрывно связано с параллельными многопроцессорными системами (МС) и вычислительными комплексами (ВК) [1]. В связи с этим необходимо постоянно работать над созданием принципиально новых математических методов и алгоритмов решения больших задач из различных предметных областей, использующих принципы структурирования и конвейеризации [2].

1. Основные понятия и постановка задачи. Как и в [2–7] *процесс* – последовательность блоков $Q_1, Q_2, ..., Q_s$, для выполнения которых используется множество процессоров (процессорных узлов, интеллектуальных клиентов). Процесс будем называть *распределённым*, если все блоки или их часть обрабатываются разными процессорами. *Ресурсами* будем считать любые объекты МС, которые используются процессами для своего выполнения. *Реентерабельные (многократно используемые)* ресурсы характеризуются возможностью одновременного использования несколькими вычислительными процессами. Последовательность блоков будем называть *программным ресурсом*, если ее часть необходимо процессорам выполнять многократно. Множество соответствующих процессов будем называть *конкурирующими***.**

Математическая модель системы распределенной обработки конкурирующих процессов включает в себя: $p \geq 2$ – число процессоров многопроцессорной системы; $s \geq 2$ – число блоков линейно структурированного программного ресурса $PR = (Q_1, Q_2, ..., Q_s)$; $n \geq 2$ – число распределенных относительно *PR* конкурирующих процессов; матрицу $T_p = [t_{ij}]$ 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 программного ресурса *PR*.

В [2] определены режимы взаимодействия процессов, процессоров и блоков линейно структурированного программного ресурса.

Первый синхронный режим обеспечивает непрерывное выполнение блоков программного ресурса внутри каждого из вычислительных процессов.

Второй синхронный режим обеспечивает непрерывное выполнение каждого блока всеми процессами.

Асинхронный режим взаимодействия процессоров, процессов и блоков, предполагает отсутствие простоев процессоров МС при условии готовности блоков, а также невыполнение блоков при наличии процессоров.

Система взаимодействующих конкурирующих процессов называется *одинаково распределенной*, если времена выполнения блоков программного ресурса каждым из процессов совпадают, т.е. справедлива цепочка равенств $t_{i1} = t_{i2} = ... = t_{is} = t_i$ для всех $i = \overline{1,n}$.

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

В [2] доказано, что для всех трех базовых режимов для одинаково распределенных систем конкурирующих процессов минимальное общее время в случае *неограниченного параллелизма* ($s \le p$) и в случае *ограниченного параллелизма* ($s > p$) при $T_\varepsilon^n \le pt_{max}^\varepsilon$, определяется по формуле:

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

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

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

Понятие линейной упаковки свяжем с понятием множества одинаково распределенных конкурирующих процессов.

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 $M_1, M_2, \ldots, M_l$ такое, что каждое подмножество есть объединение последовательных элементов множества *M*.

С учетом того, что для системы одинаково распределенных процессов времена выполнения блоков программного ресурса каждым из процессов совпадают $t_{i1} = t_{i2} = ... = t_{is} = t_i$, $i = \overline{1,n}$, то в качестве элементов множества *M* будем рассматривать последовательность первых блоков $(Q_{11}, Q_{21}, \ldots, Q_{n1})$ структурированного программного ресурса, которую обозначим $(q_1, q_2, \ldots, q_n)$. В этом случае линейная упаковка множества *M* получается объединением блоков q_i, $i = \overline{1,n}$, принадлежащих подряд идущим процессам, 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.

Обозначим через K множество всевозможных компоновок блоков одинаково распределенной МС, 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: $t(q_i')=\sum\limits_{q\in q_i'}t(q)$ – 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', $i=\overline{1,l}$; $t_{\max}(LC_l)=\max\limits_{1\le i\le l}\{t(q_i')\}$ – 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*.

2. Свойства оптимальных компоновок и вспомогательные результаты.

Теорема 1. Если LC_l – оптимальная линейная компоновка одинаково распределенной системы, то компоновка LC_l', такая, что $t_{\max}(LC_l')=t_{\min}$, также является оптимальной [7].

Теорема 2. 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 1 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 $(n-1)$ компоновки.

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 является естественным 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, $B \ge \max\limits_{1 \le i \le n} \{v(m_i)\}$, требуется найти такую линейную упаковку множества *M*, чтобы размер каждого элемента упаковки $v(M)$ 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: 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.

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

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

$$x_{nj} = t_j,\ j = \overline{1,n},\quad x_{n-1,j} = x_{nj} + t_{j+1},\ j = \overline{1,n-1},\quad \ldots,$$

$$x_{n-k,j} = x_{n-k+1,j} + t_{j+k},\ j = \overline{1,n-k},\quad \ldots,\quad x_{2\text{j}} = x_{3\text{j}} + t_{j+n-2},\ j = 1,2.$$

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 $x_{ij} < \max\limits_{1\le j\le n}\{t_j\}$. В результате получим возрастающую последовательность чисел $v_1 < v_2 < \ldots < v_k$, для которой $v_1 \le \max\limits_{1\le j\le n}\{t_j\}$, $n-1 \le k < \dfrac{n(n+1)}{2} - 1$.

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 $(q_1, q_2, \ldots, q_n)$.

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.

Трудоемкость предложенного алгоритма не превосходит $O(n^3)$ 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 $O(n^3)$.

В заключении следует отметить, что правильность работы предложенного алгоритма подтверждают результаты, полученные в работах [3,5], где доказано, что оптимальную одинаково распределенную систему следует искать среди стационарных одинаково распределенных систем.

СПИСОК ЛИТЕРАТУРЫ

1. *Воеводин В. В., Воеводин В. В.* Параллельные вычисления / СПб. : БХВ-Петербург, 2002. 608 с.

2. *Павлов П. А., Коваленко Н. С.* Математическое моделирование параллельных процессов. // Germany: Lambert Academic Publishing. 2011. 246 с.

3. *Павлов П. А.* Эффективность распределенных вычислений в масштабируемых системах // Научно-технические ведомости СПбГПУ. 2010. № 1. С. 83–89.

4. *Павлов П. А.* Масштабируемые распределенные системы конкурирующих взаимодействующих процессов и их оптимальность // Вестник Самарского аэрокосмического университета имени академика С. П. Королева. 2010. № 1. С. 234–242.

5. *Коваленко Н. С., Павлов П. А.* Алгоритм построения оптимальной компоновки одинаково распределенных систем // Программирование. 2012. № 3. С. 3–10.