

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА ОПТИМИЗАЦИИ ДЛИТЕЛЬНОСТЕЙ ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНОГО ВЫПОЛНЕНИЯ ПЕРЕСЕКАЮЩИХСЯ МНОЖЕСТВ ОПЕРАЦИЙ

А. Б. Долгий¹, Г. М. Левин², Б. М. Розин², И. В. Кособуцкий²

¹LIMOS UMR CNRS 6158, Ecole des Mines de Saint-Etienne, France

E-mail: dolgui@emse.fr

²Объединенный институт проблем информатики НАН Беларуси

Минск, Беларусь

E-mail: {levin, rozin, kasabutski}@newman.bas-net.by

Рассматривается задача оптимизации длительностей последовательно-параллельного выполнения пересекающихся множеств операций. Сформулирована математическая модель задачи и описан метод ее решения, основанный на идеях декомпозиции и динамического программирования. Предложен параллельный алгоритм и реализующая его программа поиска решения с использованием многопроцессорной вычислительной системы. Приведены результаты численных экспериментов.

Ключевые слова: комплекс операций, оптимизация длительностей, параллельные вычисления.

Постановка задачи

Значительное внимание в последние десятилетия в литературе уделялось различным аспектам планирования выполнения комплексов операций в организационных и производственных системах различного назначения [1–3]. Данная статья посвящена обобщению предложенного в [4] метода оптимизации длительностей последовательно-параллельного выполнения пересекающихся множеств операций на более общий случай, а также параллельной реализации этого метода, ориентированной на суперкомпьютерные конфигурации СКИФ [5].

Рассматривается циклически повторяющийся процесс выполнения комплекса I из n различных составных операций (далее – с-операций). Каждая с-операция $i \in I = \{1, \dots, n\}$ образована некоторым подмножеством J_i множества J элементарных операций (далее – э-операций). Комплекс может включать n_i идентичных с-операций $i \in I$. В свою очередь, в состав с-операции i может входить m_{ij} идентичных э-операций $j \in J_i$, причем подмножества семейства $\{J_1, \dots, J_i, \dots, J_n\}$, образующие с-операции комплекса, могут пересекаться. Все с-операции комплекса выполняются последовательно, в то время как все э-операции, входящие в состав каждой с-операции, выполняются одновременно. Таким образом, длительность каждой с-операции равна наибольшей из длительностей входящих в нее э-операций.

Для каждой э-операции $j \in J$ задан диапазон $[t_{1j}, t_{2j}]$ возможных длительностей ее выполнения и определенная на нем функция $f_j(t_j)$ зависимости затрат на выполнение этой операции от искомой длительности t_j ее выполнения. Стоимость выполнения каждой из с-операций i в целом помимо суммарной стоимости выполнения составляющих ее э-операций включает и дополнительные затраты $\psi_i(T_i)$, возрастающие с ростом длительности T_i ее выполнения, $i \in I$.

Требуется найти значения длительностей $t_j \in [t_{1j}, t_{2j}]$ выполнения всех э-операций $j \in J$, минимизирующие суммарные затраты на выполнение всех с-операций комплекса при ограничении на суммарное время их выполнения. Таким образом, исходная задача (назовем ее задачей **A**) заключается в нахождении такого вектора $t = (t_j \mid j \in J) \in \mathbf{t} = \prod_{j \in J} [t_{1j}, t_{2j}]$, которому соответствует наименьшее значение функции общих затрат

$$F(t) = \sum_{i \in I} n_i (\psi_i (\max_{j \in J_i} t_j) + \sum_{j \in J_i} m_{ij} f_j(t_j)) = \sum_{i \in I} n_i \psi_i (\max_{j \in J_i} t_j) + \sum_{j \in J_i} p_j f_j(t_j),$$

при условии

$$\sum_{i \in I} n_i \max_{j \in J_i} t_j \leq T_0,$$

где T_0 – заданная максимально допустимая суммарная длительность выполнения всех с-операций, $p_j = \sum_{i \in I_j} n_i m_{ij}$ и $I_j = \{i \in I \mid j \in J_i\}$.

Предполагается, что множество I всех с-операций неразделимо на подмножества так, чтобы любые с-операции из разных подмножеств не пересекались по э-операциям, поскольку иначе исходная задача может быть разбита на аналогичные автономные подзадачи меньшей размерности относительно каждого из таких подмножеств с-операций. Очевидно, что в рамках данной задачи для каждого $j \in J$ достаточно рассматривать только такие $t_j \in [t_{1j}, t_{2j}]$, для которых не существует таких $\dot{t}_j \in [t_{1j}, t_{2j}]$, что $t_j < \dot{t}_j$ и $f_j(\dot{t}_j) \leq f_j(t_j)$. Поэтому в дальнейшем для простоты изложения предполагается также, что функции $f_j(t_j)$ являются убывающими на $[t_{1j}, t_{2j}]$ для всех $j \in J$.

Метод решения

Для решения задачи **A** предлагается метод, основанный на сочетании идей параметризации, декомпозиции и динамического программирования.

Введем вектор $T = (T_1, \dots, T_i, \dots, T_n)$, компонента T_i которого определяет длительность выполнения с-операции $i \in I$, т. е. $T_i \in \mathbf{T}_i = [\max_{j \in J_i} t_{1j}, \max_{j \in J_i} t_{2j}]$. Положим

$$\prod_{i \in I} \mathbf{T}_i = \mathbf{T} \text{ и } t'_j(T) = \min\{t_{2j}, \min\{T_i \mid i \in I_j\}\}, \text{ где } I_j = \{i \in I \mid j \in J_i\}.$$

Поскольку функции $f_j(t_j)$ являются убывающими на отрезках $[t_{1j}, t_{2j}]$ для всех $j \in J$, то вместо задачи **A** можно рассматривать следующую параметризованную задачу **B**:

$$\Phi(T) = \sum_{i \in I} n_i \psi_i(T_i) + \sum_{j \in J} p_j f_j(t'_j(T)) \rightarrow \min,$$

$$\sum_{i \in I} n_i T_i \leq T_0, \quad T \in \mathbf{T}.$$

Задачи **A** и **B** эквивалентны в том смысле, что если t^* – решение задачи **A**, то $T(t^*) = (T_1(t^*), \dots, T_n(t^*))$ – решение задачи **B**, где $T_i(t^*) = \max_{j \in J_i} t_j^*$. В свою очередь, если T^* – решение задачи **B**, то $t'_j(T^*)$ – решение задачи **A**.

Для решения задачи **B** можно использовать следующую декомпозиционную схему:

• на нижнем уровне при фиксированном $\lambda \in [0,1]$ отыскивается значение $T^*(\lambda) = (T_i^*(\lambda) | i \in I)$ вектора $T \in \mathbf{T}$ с наименьшим значением функции Лагранжа (задача **B1**(λ)):

$$L(\lambda, T) = \sum_{i \in I} n_i (\lambda T_i + (1-\lambda) \psi_i(T_i)) + (1-\lambda) \sum_{j \in J} p_j f_j(t'_j(T)),$$

а на верхнем – наименьшее значение λ^* множителя $\lambda \in [0, 1]$ с $O(\lambda) = \sum_{i \in I} n_i T_i^*(\lambda) \leq T_0$.

При наличии процедуры решения задачи **B1**(λ) для нахождения $\lambda^* \in [0, 1]$ можно воспользоваться известными методами поиска корня уравнения с монотонной левой частью. При этом если $O(0) \leq T_0$, то $T^*(0)$ – решение задачи **B**, если $O(1) > T_0$, то задача **B** (а следовательно и исходная задача **A**) не имеют решения. В остальных случаях $T^*(\lambda^*)$ можно принять в качестве решения задачи **B**.

Для описания возможного подхода к решению задачи **B1**(λ) введем следующие вспомогательные функции, где $\Omega \subseteq J$, $Z' \subseteq Z \subseteq I$ и $\tau, \tau' \in \mathbb{R}^+$:

$$t''_j(\tau) = \min\{t_{2j}, \tau\}; \quad t_1(\Omega) = \max_{j \in \Omega} t_{1j}; \quad t_2(\Omega) = \max_{j \in \Omega} t_{2j};$$

$$J(Z) = \bigcup_{i \in Z} J_i; \quad \tilde{J}(Z, Z') = J(Z) \setminus J(Z');$$

$$\varphi(\lambda, Z, Z', \tau) = \sum_{i \in Z \setminus Z'} n_i (\lambda \tau + (1-\lambda) \psi_i(\tau)) + (1-\lambda) \sum_{j \in J(Z, Z')} p_j f_j(t''_j(\tau));$$

$$\underline{\tau}(\lambda, Z, Z', \tau') = \operatorname{argmin}\{\varphi(\lambda, Z, Z', \tau) | \tau \in [\tau', \max\{\tau', t_2(\tilde{J}(Z, Z'))\}]\}.$$

Предлагаемый подход к решению задачи **B1**(λ) базируется на ее сведении к отысканию кратчайшего пути из начальной вершины некоторого бесконтурного орграфа $(\mathfrak{R}, \mathbf{D})$ в одну из его концевых вершин. Начальной вершине орграфа сопоставляется пара $(\emptyset, 0)$, а концевым вершинам – пары (I, τ) с различными значениями $\tau \in [t_1(J), t_2(J)]$. Остальным вершинам сопоставляются пары (Z, τ) , где Z – некоторое собственное подмножество с-операций из I , а τ – их принимаемая максимальная длительность, причем каждая из пар (Z, τ) такова, что $\tau \in [t_1(J(Z)), t_2(J(Z))]$ и Z – максимальное по включению множество в том смысле, что не существует другого $Z' \subseteq I$, такого, что $Z \subset Z'$ и $J(Z) = J(Z')$. В свою очередь, любая дуга $((Z', \tau'), (Z, \tau)) \in \mathbf{D}$ такова, что $Z' \subset Z$ и $\tau \in [\tau', \underline{\tau}(\lambda, Z, Z', \tau')]$, причем не существует другого $Z'' \subset I$, такого что $Z' \subset Z'' \subset Z$ и $J(Z'') \subset J(Z)$. Дуге $((Z', \tau'), (Z, \tau))$ сопоставляется ее длина $\varphi(\lambda, Z, Z', \tau)$.

Пусть для некоторого $\lambda \in [0, 1]$ $P = ((Z_0 = \emptyset, \tau_0 = 0), (Z_1, \tau_1), \dots, (Z_k, \tau_k), \dots, (Z_{q(P)}, \tau_{q(P)}))$ – кратчайший путь в орграфе $(\mathfrak{R}, \mathbf{D})$ из начальной вершины в одну из его концевых вершин, т. е. путь с наименьшим значением $G^*(\lambda)$ функции $G(\lambda, P) = \sum_{k=1}^{q(P)} \varphi(\lambda, Z_k, Z_{k-1}, \tau_k)$. Положим $T_i^*(\lambda) = \tau_k$ для всех $i \in Z_k \setminus Z_{k-1}$ и $k = 1, \dots, q(P)$.

Можно показать, что полученный таким образом вектор $T^*(\lambda) = (T_i^*(\lambda) | i \in I)$ является решением задачи **B1**(λ), причем $\Phi(T^*(\lambda)) = G^*(\lambda)$.

Для удобства реализации данного подхода вершины графа разбиваются на группы с одинаковым множеством Z . Разработана процедура построения орграфа $(\mathfrak{R}, \mathbf{D})$, вершины которого организуются в уровни в порядке возрастания $|Z|$ соответствующих групп вершин, а также параллельный алгоритм динамического программирования, который осуществляет вычисление кратчайшего пути в орграфе $(\mathfrak{R}, \mathbf{D})$ из вершины

$(\emptyset, 0) \in \mathfrak{R}$ в одну из вершин вида $(I, \tau) \in \mathfrak{R}$, по порядку проходя все уровни. Выявленные в ходе вычислений заведомо «бесперспективные» вершины удаляются из дальнейшего рассмотрения. В частности, к таковым относятся вершины $(Z, \tau) \in \mathfrak{R}$, для которых существуют такие $(Z', \tau') \in \mathfrak{R}$, что $Z \subseteq Z'$, $\tau \geq \tau'$ и $H(\lambda, Z, \tau) \geq H(\lambda, Z', \tau')$, где $H(\lambda, Z, \tau)$ – длина кратчайшего пути из начальной вершины орграфа в вершину (Z, τ) .

Ниже приводится краткое описание разработанной параллельной программы, реализующей предложенные алгоритмы для решения исходной задачи **A** на базе описанного подхода.

Параллельная реализация

Модель параллелизма базируется на специальной форме параллелизма по данным: «одна программа – множество потоков данных» (SPMD). В этой модели на каждом процессоре выполняется одна и та же последовательность действий, применяемая к собственному подмножеству элементов структуры данных. Все переменные и массивы размножены на каждый процессор, но вычисления над разными секциями массивов распределены между разными процессорами. Взаимодействие между параллельно выполняющимися процессами организовано посредством обмена сообщениями средствами библиотеки MPI [6], обеспечивающими достаточно эффективную реализацию для кластеров гомогенных SMP-узлов. Программное обеспечение реализовано для исполнения на ВМВС семейства СКИФ [5]. При этом используются коллективные коммуникации, операции редукции, операции группировки данных.

Параллелизм по данным при определении условно-оптимальной вершины в орграфе по схеме динамического программирования реализован распределением витков тесно-гнездового цикла между процессами в коммуникаторе. При этом каждый виток такого параллельного цикла полностью выполняется на одном процессоре. В ходе алгоритма для очередной группы вершин орграфа с одинаковым Z запускается параллельный цикл, в котором одномерный массив длины N допустимых значений длительности τ этой группы вершин распределяется последовательно между P процессорами блоками по $[N/P]$ элементов. После вычисления соответствующей секции массива значений функции $H(\lambda, Z, \tau)$ текущей группы вершин каждый процесс выполняет упаковку данных посредством *MPI_Pack()*. Затем каждый процесс вызывает процедуру *MPI_Allgatherv()* пересылки этой секции всем остальным процессам коммуникатора и получения всех вычисленных на других процессорах секций массива. Завершающим этапом каждой итерации перебора групп вершин с одинаковым Z является распаковка полученных данных посредством *MPI_Unpack()* и заполнение вычисленными значениями всего массива значений функции $H(\lambda, Z, \tau)$ в памяти каждого процесса.

Вычислительные эксперименты

Основная цель проведенных численных экспериментов – анализ зависимости среднего процессорного времени от характеристик задачи **A**. Ниже приводятся некоторые результаты этих экспериментов на кластере «СКИФ-Триада» для задач с 10 и 20 с-операциями и с 20 и 40 э-операциями, при $t_j \in [1, 5]$, $j \in J$. Шаг сетки значений длительности при $n = 20$ равен 0,01 и $T_0 = 60$; при $n = 10$ шаг 0,002 и $T_0 = 30$. Функции $f_j(t_j)$ – степенные функции либо экспоненциальные, а функции $\psi_i(T) = E_i T_i$, где $E_i > 0$, $i \in I$.

На рис. 1, а представлены полученные графики зависимости среднего процессорного времени при решении задач на одном процессоре от средней степени пересечения

составных операций, а на рис. 1, б – зависимости процессорного времени от числа процессоров (в пределах от 1 до 9) при средней степени пересечения с-операций 30 %.

Эксперименты выявили экспоненциальную зависимость времени вычислений от степени пересечения составных операций комплекса, а также квадратичную его зависимость от мощности сеток длительностей. С учетом этого для повышения эффективности поиска решения целесообразно применять метод «блуждающей трубки» с варьированием частоты сеток и диапазонов поиска.

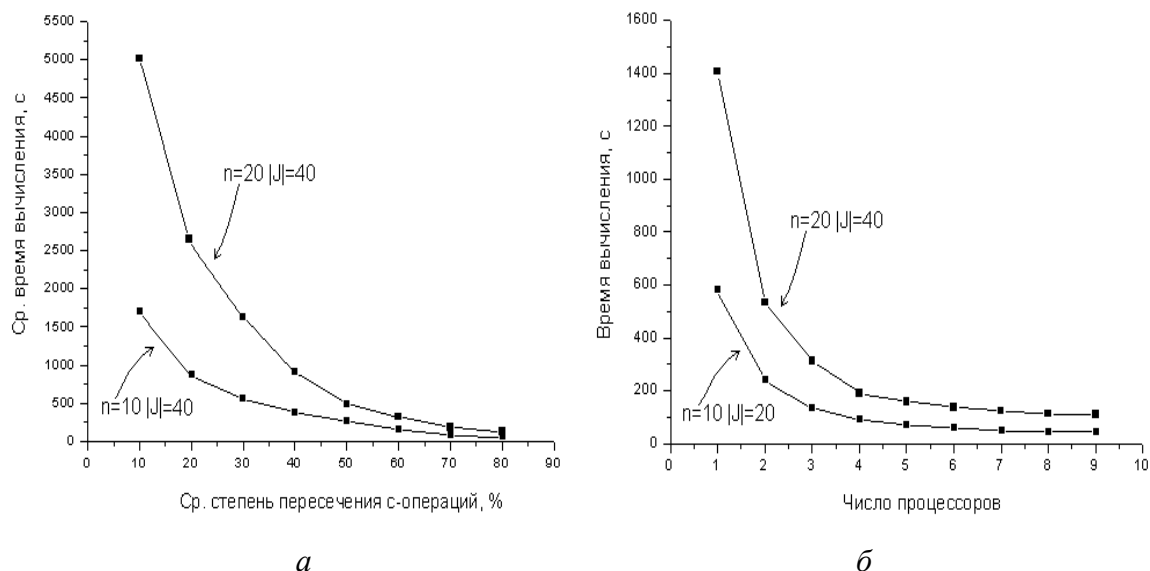


Рис. 1. Зависимость процессорного времени от средней степени пересечения с-операций (а) и числа процессоров (б)

Данная работа была выполнена при поддержке совместного проекта Ф12ФП-001 БРФФИ и PICS CNRS 6064 (Франция).

Библиографические ссылки

1. Boysen N., Flidner M., Scholl A. A classification of assembly line balancing problems // European Journal of Operational Research. 2007. V. 183. P. 674–693.
2. Bukchin J., Tzur M. Design of flexible assembly line to minimize equipment cost // IIE Transactions. 2000. V. 32. P. 585–598.
3. Burkov V. N., Novikov D. A. Models and methods of multiprojects' management // Systems Science. 1999. V. 256. № 2. P. 5–14.
4. Левин Г. М., Розин Б. М. Оптимизация длительностей последовательно-параллельного выполнения пересекающихся множеств операций // Информатика. 2012. № 4. С. 87–92.
5. Суперкомпьютерные конфигурации СКИФ / С. В. Абламейко [и др.]. Минск : ОИПИ НАН Беларуси, 2005.
6. Средства параллельного программирования в ОС Linux / Р. Х. Садыхов [и др.]. Минск : БГУИР, 2004.