Коррекция многопроцессорных расписаний в режиме реального времени

М.Г. Фуругян

ФИЦ ИУ РАН, Москва, Российская Федерация;

rtsccas@yandex.ru

Аннотация. Рассматривается задача планирования вычислений в многопроцессорных системах при периодически поступающих запросах на выполнение комплекса работ с нефиксированными длительностями. Используется неоднородный набор ресурсов – возобновляемых и невозобновляемых. Рассмотрены случаи, когда работы допускают прерывания и переключения с одного процессора на другой, а также когда работы непрерываемые. Разработаны алгоритмы, в которых при обработке каждого запроса корректируется расписание, построенное при обработке предыдущего запроса. Алгоритмы основаны на сетевом моделировании и поиске максимального потока и потока минимальной стоимости.

Ключевые слова: многопроцессорная система, директивный интервал, допустимое расписание, потоковая сеть, максимальный поток, поток минимальной стоимости.

1. Введение

Одно из основных требований, предъявляемых к вычислительным системам реального времени, заключается в том, что все операции должны выполняться в строго отведенных для них временных интервалах. Например, при испытании и функционировании некоторых сложных технических объектов (самолеты, электростанции, ядерные реакторы) запросы на выполнение вычислительных заданий могут поступать с очень высокой частотой и должны успевать обрабатываться в заданном темпе. В случае, когда длительности выполнения всех программных модулей фиксированы, оптимальное расписание их выполнения можно составить заранее, т.е. до проведения эксперимента.

В случае, когда длительности становятся известны в момент поступления запроса на выполнение комплекса вычислительных заданий, расчет расписания производится в режиме реального времени. Это требует эффективных алгоритмов построения оптимальных расписаний. Этим вопросам посвящено большое количество публикаций.

Так, в [1, 2] рассматриваются задачи построения прерываемых расписаний в многопроцессорных системах при заданных директивных интервалах для каждого задания. При выборе подходящего математического аппарата для построения оптимальных расписаний и временных диаграмм выполнения программных модулей в системах реального времени авторы провели сравнительный анализ обобщенных сетей петри и конечных автоматов с остановкой таймера. Предпочтение было отдано последним.

Публикации [3, 4] посвящены построению расписаний выполнения комплекса непрерываемых работ с нефиксированными параметрами (длительностями и ресурсами). Авторами разработан алгоритм, позволяющий разбивать область изменения параметров на многогранники устойчивости, внутри каждого из которых структура оптимального по быстродействию расписания остается неизменной. Способ построения расписаний основан на использовании метода ветвей и границ.

В [5. 6] авторы описывают и проводят сравнительный анализ эвристических и генетических алгоритмов, а также методов с самообучением для построения многопроцессорных расписаний без прерываний.

В [7] исследованы задачи планирования прерываемых работ с директивными интервалами в системах с неоднородным набором ресурсов – возобновляемыми (которые могут использоваться повторно, например, процессоры, приборы, различные механизмы) и невозобновляемыми (которые повторно использоваться не могу, например, электроэнергия, финансы, горюче-смазочные материалы). Разработаны алгоритмы построения оптимальных расписаний, основанные на сведении исходной задачи к нахождению потоков с заданными характеристиками в специальных сетях.

В настоящей статье рассматривается задача построения допустимого многопроцессорного расписания для комплекса работ с директивными интервалами. Рассмотрены случаи, когда работы допускают прерывания и переключения с одного процессора на другой, а также когда работы непрерываемые. Система включает в себя ресурсы двух типов – возобновляемые и невозобновляемые. Предполагается, что запросы на

выполнение комплекса работ поступают периодически, а длительности работ не являются фиксированными и становятся известными только в моменты поступления запросов. Разработаны алгоритмы, которые при поступлении каждого запроса используют расписание, построенное при обработке предыдущего запроса. Это сокращает время построения расписания, что имеет большое значение для обработки информации в реальном масштабе времени. Алгоритмы основаны на поиске потока минимальной стоимости и максимального потока в специальных сетях.

2. Постановка задачи

Имеется комплекс работ (заданий) $W = \{w_1, w_2, \dots, w_n\}$, запросы на выполнение которого поступают в моменты времени $\tau_1, \tau_2, \dots, \tau_k, \dots$ При поступлении запроса τ_k для каждой работы w_i устанавливается директивный интервал ее выполнения $[\tau_k + b_i;\ \tau_k + f_i]$, $f_i > b_i$, $\tau_k + f_i \leq \tau_{k+1}$, $i = \overline{1,n}$, $k = 1, 2, \dots$ Для выполнения работ имеются m идентичных процессоров (возобновляемые ресурсы) и невозобновляемый ресурс. В момент времени τ_k его объем составляет R_k и он распределяется между работами W, а при последующих запросах он использоваться не может. При обработке k-го запроса длительность выполнения работы w_i, если ей не выделен невозобновляемый ресурс, составляет t_i^k, $t_i^k \geq f_i - b_i$, $i = \overline{1,n}$, $k = 1, 2, \dots$. Эта величина становится известной только в момент времени τ_k. Если же невозобновляемый ресурс в объеме r_i^k выделен, то длительность выполнения работы w_i будет составлять $t_i^k - a_i r_i^k$. Здесь $a_i \geq 0$ – известная величина, $t_i^k - a_i r_i^k > 0$,

$$\sum_{i=1}^{n} r_i^k \leq R_k, k = 1, 2, \dots \quad (1)$$

Распределение невозобновляемого ресурса, удовлетворяющее ограничениям (1), называется допустимым.

Рассматриваются случаи, когда (1) при выполнении работ допускаются прерывания и переключения с одного процессора на другой, а соответствующие временные издержки при этом не учитываются; (2) работы являются непрерываемыми. Допустимым расписанием для комплекса работ W называется расписание, при котором каждая работа выполняется в своем директивном интервале. Не допускается параллельное выполнение нескольких работ одним процессором и одновременное выполнение одной работы несколькими процессорами. Допустимым решением будем называть пару “допустимое распределение невозобновляемого ресурса – допустимое расписание комплекса работ W”.

Требуется разработать алгоритм, который для каждого τ_k строит допустимое решение. При этом следует учесть, что длительности работ становятся известными только в момент поступления очередного запроса на их выполнение. Поэтому искомый алгоритм будет работать в режиме реального времени, что накладывает жесткие требования на его быстродействие. Предлагаемый алгоритм основан на том, что при построении решения для очередного запроса используется решение, полученное для предыдущего запроса.

3. Алгоритм решения задачи для случая прерываемых работ

Для решения поставленной задачи в случае прерываемых работ воспользуемся алгоритмом, разработанным в [8]. Для поиска решения S_1 при обработке запроса, поступившего в момент времени τ_1, как описано в [8], построим потоковую сеть G_1, в которой определим поток g_1 минимальной стоимости $c(g_1)$. Для его поиска может быть использован, например, алгоритм дефекта [9]. Как следует из [8], решение S_1 существует в том и только том случае, когда

$$c(g_1) \leq R_1 - \sum_{i=1}^{n} t_i^1 / a_i.$$

В [8] также указано, какой объем невозобновляемого ресурса следует выделить каждой работе w_i и как по найденному потоку построить допустимое расписание. Вычислительная сложность построения решения S_1 составляет

$$O\left(n^4 \sum_{i=1}^{n} t_i^1\right).$$

Для поиска решения S_kпри обработке запроса, поступившего в момент $\tau_k, k \geq 2$, можно воспользоваться решением, построенном при обработке запроса, поступившего в момент τ_{k-1}. А именно, при поиске потока минимальной стоимости g_k в сети G_k с помощью алгоритма дефекта в качестве начального потока можно взять не нулевой поток, а поток минимальной стоимости g_{k-1}, построенный ранее в сети G_{k-1}. В этом случае вычислительная сложность алгоритма будет составлять

$$O\left(n^4 \sum_{i=1}^{n} \left|t_i^k - t_i^{k-1}\right|\right),$$

а не

$$O\left(n^4 \sum_{i=1}^{n} t_i^k\right),$$

если алгоритм стартовал бы с нулевого потока. Такой выигрыш во времени работы алгоритма может иметь существенное значение при

проведении расчетов в реальном времени.

Рассмотрим теперь частный случай, когда для выполнения работ используются только процессоры (невозобновляемый ресурс отсутствует, т.е. $R_k = 0, k = 1, 2, \dots$) и, кроме того, $t_i^k \geq t_i^{k-1}$ при всех $k = 1, 2, \dots, i = \overline{1, n}$. В этом случае для построения решения эффективнее воспользоваться алгоритмом, описанным в [10], который основан на поиске максимального потока в специальной сети Н. А для поиска максимального потока можно воспользоваться алгоритмом кубической сложности, описанным в [11].

Предположим, что при обработке запроса, поступившего в момент τ_{k-1}, была построена сеть H_{k-1} и найдем в ней максимальный поток h_{k-1}. Пусть, далее, при поиске максимального потока было выполнено $p < n$ этапов [11]. Тогда при обработке запроса, поступившего в момент τ_k и построении максимального потока h_k в сети H_k в качестве начального потока следует взять поток h_{k-1}, а не нулевой поток. В этом случае вычислительная сложность алгоритма поиска максимального потока h_k будет составлять $O((n-p)n^2)$, а не $O(n^3)$. Такой выигрыш во времени также может иметь большое значение при проведении вычислений в реальном времени.

4. Случай непрерываемых работ

В этом разделе рассматривается задача на быстродействие в случае, когда работы не допускают прерывания и переключения с одного процессора на другой и, кроме того, невозобновляемый ресурс отсутствует ($R_k = 0, k = 1, 2, \dots$). Как известно [12], в этом случае задача является NP-трудной в сильном смысле.

В [13] описан точный алгоритм решения этой задачи, основанный на методе ветвей и границ. При этом нижние и верхние оценки длины оптимального по быстродействию расписания на различных подмножествах множества всех расписаний, возникающих при ветвлении, вычисляются с помощью рекуррентных соотношений и использования структуры данных в виде двоичной кучи (пирамиды). Это позволило сократить трудоемкость соответствующих процедур с $O(mn)$ и $O(m+n)$ (в случае, когда при вычислении оценок рекуррентные соотношения не используются) до $O(n \log_2 m)$ и $O(1)$ соответственно.

Рассмотрим теперь приближенный “жадный” алгоритм, суть которого заключается в следующем [12]. Фиксируем некоторое k и пусть T_j^k – временная загруженность j-го процессора (т.е. сумма длительностей работ, назначенных на него).

Шаг 1. Положить $T_j^k = 0, j = \overline{1, m}$.

Для каждого $i = \overline{1, n}$ выполнять шаги 2, 3.

Шаг 2. Найти $\min\limits_{j=\overline{1,m}} T_j^k = T_{j_0}^k$.

Шаг 3. Работу w_i назначить на j_0 процессор.

Если величины T_j^k хранить в виде двоичной кучи с минимальным элементом в вершине, то вычислительная сложность алгоритма составит $O(n \log_2 m)$. Если после завершения алгоритма были получены величины $T_j^k, j = \overline{1, m}$, то длина полученного расписания равна $T^k = \max\limits_{j=\overline{1,m}} T_j^k$. Пусть T_{min}^k – это длина оптимального по быстродействию расписания. Известно, что

$$T^k / T_{min}^k \leq 1{,}5.$$

Предположим, что известна некоторая оценка снизу L^k величины T_{min}^k: $L^k \leq T_{min}^k$. Тогда можно предложить следующий приближенный алгоритм, основанный на использовании “жадного” алгоритма.

Шаг 1. Для каждого $j = \overline{1, m}$ назначить на j-й процессор работы таким образом, чтобы выполнялось неравенство $T_j^k \leq L^k$.

Шаг 2. Оставшиеся работы (если таковые имеются) назначить на процессоры с помощью “жадного” алгоритма.

Оценим вычислительную сложность этого алгоритма. Если на шаге 1 были назначены p работ, то вычислительная сложность шага 1 составляет $O(p)$, а шага 2 – $O((n-p)\log_2 m)$. Таким образом, вычислительная сложность предложенного алгоритма составляет $O(p + (n-p)\log_2 m)$. При $m > 2$ эта оценка предпочтительнее, чем $O(n \log_2 m)$.

В качестве L^k можно взять, например, величину $(\sum_{i=1}^n t_i^k)/n$. Тогда вычислительная сложность алгоритма составит $O(n + p + (n-p)\log_2 m)$, а неравенство $n + p + (n-p)\log_2 m < n \log_2 m$ выполняется при $m > 2^{\frac{n}{p}+1}$. Например, \если $p = n/2$, то при $m > 8$.

В случае, когда $t_i^{k-1} \leq t_i^k$ при всех $i = \overline{1, n}$, в качестве L^k можно взять величину $\min\limits_{j=\overline{1,m}} T_j^{k-1}$. Тогда вычислительная сложность алгоритма составит $O(p + (n-p)\log_2 m)$ и, как было отмечено выше, такая оценка предпочтительнее той, которая получается при непосредственном применении “жадного” алгоритма.

4.Заключение

Исследована задача планирования вычислений в многопроцессорных системах при периодически поступающих запросах на выполнение

комплекса работ с нефиксированными длительностями. Используется неоднородный набор ресурсов – возобновляемых и невозобновляемых. Рассмотрены случаи, когда работы допускают прерывания и переключения с одного процессора на другой, а также когда работы непрерываемые. Разработаны алгоритмы, в которых при обработке каждого запроса корректируется расписание, построенное при обработке предыдущего запроса. Алгоритмы основаны на сетевом моделировании и поиске максимального потока и потока минимальной стоимости. Определена вычислительная сложность алгоритмов.

Correcting Multiprocessor Schedules in Real Time

M.G. Furugyan

Abstract. The task of planning computing in multiprocessor systems is considered in periodically incoming requests to perform a set of work with non -fixed durations. A heterogeneous set of resources is used - renewable and non -renewable. There are cases when work allow interruptions and switching from one processor to another, as well as when the work is uninterrupted. Algorithms have been developed in which, when processing each request, the schedule built during the processing of the previous request is adjusted. Algorithms are based on network modeling and searching for the maximum flow and flow of the minimum cost.

Keywords: multiprocessor system, directive interval, feasible schedule, flow network, maximum flow, minimum cost flow.

Литература

1. А.Б. Глонина, В.В. Балашов. О корректности моделирования модульных вычислительных систем реального времени с помощью сетей временных автоматов // Моделирование и анализ информационных систем. (2018), Т. 25, № 2, 174 – 192.

2. А.Б. Глонина. Инструментальная система проверки выполнения ограничений реального времени для конфигураций модульных вычислительных систем // Вестник МГУ. Сер. 15. Вычисл. математика и кибернетика. (2020), № 3, 16 – 29.

3. А.В. Мищенко, П.С. Кошелев. Оптимизация управления работами логистического проекта в условиях неопределенности // Известия РАН. Теория и системы управления. (2021), № 4, 123 – 134.

4. М.А. Горский, А.В. Мищенко, Л.Г. Нестерович, М.А. Халиков. Некоторые модификации целочисленных оптимизационных задач с учетом неопределенности и риска // Известия РАН. Теория и системы управления. (2022), № 5, 106 – 117.

5. В.А. Костенко, А.С. Смирнов. Потоковые алгоритмы планирования вычислений в интегрированной модульной авионике // Изв. РАН. ТиСУ. (2019), № 3, 77 – 86.

6. В.В. Балашов, В.А. Костенко, И.А. Федоренко, Ц. Гао, Ч.М. Сун, Ц. Сун. Алгоритм имитации отжига для построения списочных расписаний с ограничениями на количество межпроцессорных передач данных // Автоматика и телемеханика. (2023), № 8, 138 – 152.

7. Д.А. Кононов, М.Г. Фуругян. Распределение неоднородного комплекса ресурсов при региональном планировании в условиях неопределенности // Труды XV межд. конф. “Управление развитием крупномасштабных систем (MLSD'2022)”. – Москва, ИПУ РАН. – 26 – 28 сент. (2022) 952 – 958.

8. Е.О. Косоруков, М.Г. Фуругян. Некоторые алгоритмы распределения ресурсов в многопроцессорных системах // Вестник МГУ. Сер. 15. Вычисл. математика и кибернетика. (2009), № 4, 34 – 37.

9. Э. Майника. Алгоритмы оптимизации на сетях и графах. М.: Мир, 1981, 325 с.

10. Танаев В.С., Гордон В.С., Шафранский Я.М. Теория расписаний. Одностадийные системы. М.: Наука, 1984, 383 с.

11. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы. Построение и анализ. М.: Вильямс, 2005,1296 с.

12. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982, 416 с.

13. Фуругян М.Г. Некоторые алгоритмы решения минимаксной задачи составления многопроцессорного расписания // Изв. РАН, ТиСУ. (2014), №2, с. 50 - 56.