

Планирование работ с неопределенными длительностями в системах реального времени

М.Г. Фуругян

ФИЦ ИУ РАН, Москва, Россия, rtscas@yandex.ru

Аннотация. Рассматривается задача планирования работ, выполняемых на многопроцессорной системе. Работы характеризуются директивными интервалами и неопределенными длительностями, которые могут принимать значения из заданных интервалов, образующих многомерный параллелепипед. Предлагается алгоритм разбиения этого параллелепипеда на подмножества, для каждого из которых допустимое расписание имеет неизменную структуру.

Ключевые слова: многопроцессорная система, неопределенные длительности, допустимое расписание, директивный интервал, многомерный параллелепипед

1. Введение

Одно из неотъемлемых требований, предъявляемых к вычислительным системам реального времени, заключается в том, что каждая работа должна выполняться в своем директивном интервале, т.е. начаться не ранее и завершиться не позднее строго определенных моментов времени. Поэтому при разработке и сопровождении таких систем возникают различные задачи планирования работ и составления расписаний, удовлетворяющих определенным временным критериям. По данной тематике имеется большое число публикаций. Так, например, в [1, 2] исследованы различные классические задачи планирования работ и составления расписаний (обслуживание с заданными директивными сроками, задачи на быстродействие, на минимизацию суммарного и максимального временного смещения). В [3–5] задачи составления расписаний в вычислительных системах реального времени исследованы с помощью конечных автоматов с остановкой таймера и временных диаграмм. В [6] предлагаются точные и приближенные алгоритмы решения некоторых NP-трудных задач на быстродействие и минимизацию максимального временного смещения для одного и нескольких приборов. В [7, 8] рассмотрены задачи составления многопроцессорных расписаний без прерываний и переключений с нефиксированными длительностями. Положив в основу исследований метод “ветвей и границ”, авторы разработали алгоритм построения многогранников устойчивости расписаний, в каждом из которых расписание имеет неизменную структуру. Это позволяет существенно сократить вычисления, выполняемые в реальном масштабе времени. В [9, 10] некоторые задачи планирования работ

сведены к минимаксным задачам. В [11] предложен псевдополиномиальный алгоритм решения задачи построения оптимального по быстродействию расписания исполнения заданий с логическими условиями предшествования. В этой задаче для каждого задания дан список его непосредственных предшественников, а также число завершенных непосредственных предшественников, необходимое для начала его выполнения. Задача сведена к циклической игре. В [12–14] предполагается, что для выполнения заданий используется неоднородный комплекс ресурсов – возобновляемых и невозобновляемых. Для построения допустимых расписаний с прерываниями разработана сетевая потоковая модель. Решение исходной задачи сведено к поиску потока в этой модели.

В настоящей статье рассматривается задача эффективного планирования работ (заданий) в многопроцессорной системе в реальном масштабе времени. Предполагается, что в ходе проведения некоторого эксперимента необходимо выполнить определенную совокупность заданий. Каждое задание характеризуется фиксированным директивным интервалом и нефиксированной длительностью, которая может принимать целые значения из заданного временного интервала. Совокупность таких интервалов образует многомерный параллелепипед, каждая точка которого определяет некоторый вектор длительностей выполнения работ, который становится известным только в ходе проведения эксперимента, т.е. в реальном масштабе времени. Поэтому предлагается проводить расчет расписаний заранее, т.е. до проведения эксперимента, для всевозможных значений длительностей заданий. Однако, при большом числе работ сделать это практически невозможно, поскольку

число различных векторов длительностей растет экспоненциально с ростом числа заданий. Задача заключается в том, чтобы разбить указанный выше параллелепипед на области, внутри каждой из которых расписание не изменяет своей структуры. Это позволит для каждой такой области вычислять расписание только в какой-нибудь одной точке. В то же время, число таких областей должно быть существенно меньше числа целочисленных точек параллелепипеда.

2. Постановка задачи

Комплекс работ (заданий) $W = \{1, 2, \dots, n\}$, занумерованных от 1 до n , выполняется на m -процессорной системе. Каждое задание $i \in W$ характеризуется директивным интервалом $[c_i, d_i]$ и длительностью t_i , которая может принимать натуральные значения из интервала $[t_i^1, t_i^2]$, $t_i^1, t_i^2 \in N, i = \overline{1, n}$, N – множество натуральных чисел. Пусть $N_i = \{t_i^1, t_i^1 + 1, \dots, t_i^2\}$ – возможные значения длительностей задания $i \in W$, $\Omega_0 = N_1 \times N_2 \times \dots \times N_n$. Предполагается, что проводится некоторый эксперимент (например, испытание самолета), в ходе которого вычисляются некоторые параметры, от которых зависят длительности выполнения заданий. Таким образом, характеристики работ становятся известными только в ходе проведения эксперимента, т.е. в режиме реального времени. Основная задача, которую требуется решить – это составить допустимое расписание выполнения заданий W (т.е. такое расписание, при котором каждая работа $i \in W$ выполняется строго в своем директивном интервале $[c_i, d_i]$) или определить, что такого расписания не существует. Есть несколько путей решения этой задачи. Первый – строить расписание в режиме реального времени, т.е. сразу после того, как станут известными длительности выполнения работ. Однако, это может привести к существенным временным задержкам при проведении эксперимента. Второй подход заключается в предварительном (до проведения эксперимента) построении расписания для каждого целочисленного временного вектора $(\tau_1, \tau_2, \dots, \tau_n) \in \Omega_0$. Однако, число таких векторов, равное $\prod_{i=1}^n T_i$, ($T_i = t_i^2 - t_i^1$), может быть очень велико, что делает невозможным и этот подход. Мы предлагаем разбить множество Ω_0 на подмножества (число которых существенно меньше числа точек в Ω_0), такие, что в пределах каждого такого подмножества структура допустимого расписания остается неизменной. В этом случае его можно будет вычислять до проведения эксперимента для одного временного вектора из каждого подмножества.

Для фиксированного временного вектора

$(\tau_1, \tau_2, \dots, \tau_n) \in \Omega_0$ допустимое расписание может быть найдено с помощью известных алгоритмов. Например, с помощью псевдополиномиального алгоритма в случае, когда работы не допускают прерываний и переключений с одного процессора на другой, а число процессоров фиксировано [15]. Для случая, когда работы допускают прерывания и переключения, может быть использован полиномиальный алгоритм, основанный на построении сетевой модели и поиске максимального потока в ней [1, 16]. Вычислительная сложность указанных алгоритмов составляет $O(nT^m)$ и $O(pn^3)$ соответственно, где $T = \max_{i=1, n} T_i$, m – число процессоров, p – число различных типов процессоров.

3. Разбиение множества Ω_0 на подмножества

Независимо от того, какая именно задача составления расписания рассматривается, для $\tau \in \Omega_0$ введем обозначения: $f(\tau) = 0$, если допустимого расписания не существует, и $f(\tau) = 1$ в противном случае, и пусть $R(\tau)$ – допустимое расписание, построенное для временного вектора τ . Иными словами, предполагается, что имеется некоторая программа $P(\tau)$, позволяющая для каждого временного вектора $\tau \in \Omega_0$ решать задачу поиска допустимого многопроцессорного расписания с директивными интервалами.

У т в е р ж д е н и е 1. Если $(\tau_1, \tau_2, \dots, \tau_n) \in \Omega_0$ и $f(\tau_1, \tau_2, \dots, \tau_n) = 0$, то $f(a_1, a_2, \dots, a_n) = 0$ для $(a_1, a_2, \dots, a_n) \in \Omega_0$, где $a_j \geq \tau_j, j = \overline{1, n}$.

Д о к а з а т е л ь с т в о. Действительно, если для некоторого временного вектора $\tau = (\tau_1, \tau_2, \dots, \tau_n) \in \Omega_0$ допустимого расписания не существует, то и для вектора, компоненты которого не меньше компонент вектора τ , его также не существует.

У т в е р ж д е н и е 2. Если $(\tau_1, \tau_2, \dots, \tau_n) \in \Omega_0$ и $f(\tau_1, \tau_2, \dots, \tau_n) = 1$, то $f(b_1, b_2, \dots, b_n) = 1$ для $(b_1, b_2, \dots, b_n) \in \Omega_0$, где $b_j \leq \tau_j, j = \overline{1, n}$.

Д о к а з а т е л ь с т в о. Действительно, если для некоторого временного вектора $\tau = (\tau_1, \tau_2, \dots, \tau_n) \in \Omega_0$ допустимое расписание существует, то и для вектора, компоненты которого не превосходят компонент вектора τ , оно также существует. Отметим также, что если в расписании для каждого задания указывать процессоры, которые его выполняют, и соответствующие временные интервалы, то для вектора $(b_1, b_2, \dots, b_n)$ можно использовать то же расписание, которое построено для вектора $(\tau_1, \tau_2, \dots, \tau_n)$, т.е. расписание с той же структурой. Отличие заключается только в том, что в расписании $R(b_1, b_2, \dots, b_n)$

некоторые работы могут завершаться раньше, чем в расписании $R(\tau_1, \tau_2, \dots, \tau_n)$.

Пусть τ_i^0 – ближайшее целое к величине $(t_i^2 - t_i^1)/2$, $i = \overline{1, n}$. Фиксируем произвольное k , $1 \leq k \leq n$, и рассмотрим функцию $f(\tau_1^0, \dots, \tau_{k-1}^0, \tau_k, \tau_{k+1}^0, \dots, \tau_n^0)$, $\tau_k \in [t_k^1, t_k^2] \cap N$, как функцию одной переменной τ_k . С помощью дихотомической процедуры деления отрезка $[t_k^1, t_k^2]$ пополам найдем $\tau_k^0 \in [t_k^1, t_k^2] \cap N$, такое, что $f(\tau_1^0, \dots, \tau_{k-1}^0, \tau_k^0, \tau_{k+1}^0, \dots, \tau_n^0) = 1$, $f(\tau_1^0, \dots, \tau_{k-1}^0, \tau_k^0 + 1, \tau_{k+1}^0, \dots, \tau_n^0) = 0$. Отметим, что при этом число обращений к программе $P(\tau)$ составляет $O(\log T)$.

В результате, в силу утверждения 2 получим, что

$$f(\tau_1, \tau_2, \dots, \tau_n) = 1 \quad (1)$$

при $\tau_j \leq \tau_j^0$, $j = \overline{1, n}$, и для временного вектора $(\tau_1, \tau_2, \dots, \tau_n)$ можно использовать расписание, построенное для вектора $(\tau_1^0, \tau_2^0, \dots, \tau_n^0)$. Кроме того, в силу утверждения 1

$$f(\bar{\tau}_1, \bar{\tau}_2, \dots, \bar{\tau}_n) = 0 \quad (2)$$

при $\bar{\tau}_j \geq \tau_j^0$, $j = \overline{1, n}$, $j \neq k$; $\bar{\tau}_k \geq \tau_k^0 + 1$.

Из (1), (2) следует, что множество временных векторов Ω_0 разбилось на четыре подмножества: $\Omega_1^1, \Omega_1^2, \bar{\Omega}_1^1, \bar{\Omega}_1^2$ со следующими характеристиками.

1) Для множеств Ω_1^1 и Ω_1^2 вопрос о существовании допустимого расписания остается открытым.

2) $f(\tau) = 1$ при $\tau \in \bar{\Omega}_1^1$, т.е. для $\tau \in \bar{\Omega}_1^1$ допустимое расписание существует и имеет такую же структуру, как и для вектора $(\tau_1^0, \dots, \tau_k^0, \dots, \tau_n^0)$.

3) $f(\tau) = 0$ при $\tau \in \bar{\Omega}_1^2$, т.е. для $\tau \in \bar{\Omega}_1^2$ допустимого расписания не существует.

Таким образом, в результате указанных выше действий для половины временных векторов из Ω_0 (т.е. для $\tau \in \bar{\Omega}_1^1 \cup \bar{\Omega}_1^2$) вопрос о существовании допустимого расписания и его построении в случае положительного ответа будет решен. Вопрос остается открытым для двух множеств – Ω_1^1 и Ω_1^2 , суммарное количество целочисленных векторов в которых вдвое меньше числа векторов в Ω_0 . Далее, каждое из множеств Ω_1^1 и Ω_1^2 делим на два подмножества аналогично тому, как это делалось для Ω_0 . В результате, Ω_1^1 разобьется на четыре подмножества $\Omega_2^1, \Omega_2^2, \bar{\Omega}_2^1, \bar{\Omega}_2^2$, такие, что $f(\tau) = 1$ при $\tau \in \bar{\Omega}_2^1$, $f(\tau) = 0$ при $\tau \in \bar{\Omega}_2^2$, т.е. вопрос о существовании и построении допустимого расписания для подмножеств $\bar{\Omega}_2^1$ и $\bar{\Omega}_2^2$ будет решен, а для Ω_2^1 и Ω_2^2 останется открытым.

Аналогично, Ω_1^2 также разобьется на четыре подмножества $\Omega_2^3, \Omega_2^4, \bar{\Omega}_2^3, \bar{\Omega}_2^4$, такие, что для

$\bar{\Omega}_2^3$ и $\bar{\Omega}_2^4$ вопрос о существовании и построении допустимого расписания будет решен, а для Ω_2^3 и Ω_2^4 останется открытым. При этом $|\Omega_2^1| + |\Omega_2^2| + |\Omega_2^3| + |\Omega_2^4| = 0.5(|\Omega_1^1| + |\Omega_1^2|)$, т.е. число векторов из Ω_0 , для которых вопрос о существовании допустимого расписания остается открытым, после каждой операции деления на подмножества сокращается вдвое. Продолжим этот процесс для подмножеств $\Omega_2^1, \Omega_2^2, \Omega_2^3, \Omega_2^4$ и далее для вновь образовавшихся подмножеств, для которых вопрос о существовании допустимого расписания остался открытым. Указанный процесс будет завершен, когда для каждого $\tau \in \Omega_0$ вопрос о существовании и построении допустимого расписания будет решен.

Таким образом, на каждом шаге указанного выше процесса деления множества Ω_0 на подмножества число векторов $\tau \in \Omega_0$, для которых вопрос о существовании допустимого расписания остается открытым, сокращается вдвое, а число подмножеств, содержащих эти вектора, увеличивается вдвое. Для каждого такого подмножества вычислительная сложность дихотомической процедуры составляет $O(\log T)$.

Будем предполагать, что все вектора $\tau \in \Omega_0$ равновероятны. Пусть S – это вычислительная сложность программы $P(\tau)$ построения допустимого расписания $R(\tau)$. Если фиксировать некоторое $r \in N$ и остановиться после выполнения r шагов процедуры деления множества Ω_0 на подмножества (включая расчет допустимого расписания, когда оно существует), то вычислительная сложность такой процедуры составит $O(2^r S \log T)$. При этом вероятность того, что до проведения эксперимента расписание не было построено и его необходимо будет находить в реальном масштабе времени, т.е. в ходе проведения эксперимента, составляет $1/2^r$. В то же время, вычислительная сложность процедуры построения допустимого расписания для всех векторов $\tau \in \Omega_0$ составляет $O(ST^n)$. Учитывая, что T и n могут достигать больших значений, можно подобрать такое r_0 , при котором $2^{r_0} S \log T \ll ST^n$, а величина $1/2^{r_0}$ достаточно мала. Иными словами, при $r = r_0$ вычислительная сложность предложенной процедуры много меньше вычислительной сложности полного перебора векторов $\tau \in \Omega_0$. В то же время, вероятность того, что при проведении эксперимента возникнет необходимость в расчете допустимого расписания $R(\tau)$, достаточно мала.

4. Заключение

Исследована задача построения допустимого многопроцессорного расписания для совокупно-

сти работ с заданными директивными интервалами и нефиксированными длительностями. Разработан метод предварительного (до проведения эксперимента в реальном масштабе времени) расчета расписаний, вычислительная сложность которого существенно меньше вычислительной сложности полного перебора всех

возможных векторов длительностей. При этом существует небольшая вероятность того, что возникнет необходимость в расчете допустимого расписания в реальном масштабе времени.

Work Planning with Indefinite Duration in Real Time Systems

Meran Furugyan

Abstract. The problem of scheduling work performed on a multiprocessor system is considered. Jobs are characterized by directive intervals and indefinite durations, which can take values from given intervals that form a multidimensional parallelepiped. An algorithm is proposed for partitioning this parallelepiped into subsets, for each of which the admissible schedule has an invariable structure.

Keywords: multiprocessor system, indefinite durations, admissible schedule, directive interval, multidimensional parallelepiped

Литература

1. Танаев В.С., Гордон В.С., Шафранский Я.М. Теория расписаний. Одностадийные системы. М.: Наука, 1984, 383 с.
2. P. Brucker. Scheduling Algorithms. Heidelberg: Springer, 2007, 378 с.
3. А.Б. Глонина, В.В. Балашов. О корректности моделирования модульных вычислительных систем реального времени с помощью сетей временных автоматов. «Моделирование и анализ информационных систем», Т. 25(2018), № 2, 174 – 192.
4. А.Б. Глонина. Обобщенная модель функционирования модульных вычислительных систем реального времени для проверки допустимости конфигураций таких систем. «Вестник ЮУрГУ. Сер. Вычисл. математика и информатика», Т. 6(2017), № 4, 43 – 59.
5. А.Б. Глонина. Инструментальная система проверки выполнения ограничений реального времени для конфигураций модульных вычислительных систем. // Вестн. МГУ. Сер. 15. Вычисл. математика и кибернетика. (2020), № 3, 16 – 29.
6. А.А. Лазарев. Теория расписаний. Оценка абсолютной погрешности и схема приближенного решения задач теории расписаний. М.: МФТИ, 2008, 222 с.
7. 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.
8. А.В. Мищенко, П.С. Кошелев. Оптимизация управления работами логистического проекта в условиях неопределенности // Известия РАН. Теория и системы управления. (2021), № 4, 123-134.
9. А.А. Миронов, В.И. Цурков. Минимум в моделях транспортного типа с интегральными ограничениями // Известия РАН. Теория и системы управления. (2003), № 4, 69 – 81.
10. А.А. Миронов, В.И. Цурков. Минимум при нелинейных транспортных ограничениях // Доклады академии наук. (2001), т. 381, № 3 305 – 308.
11. Д.В. Алифанов, И.Н. Лебедев, В.И. Цурков. Оптимизация расписаний с логическими условиями предшествования // Известия РАН. Теория и системы управления. (2009), № 6, С.88 – 93.
12. М.Г. Фуругян. Планирование вычислений в многопроцессорных АСУ реального времени с дополнительным ресурсом // Автоматика и телемеханика. (2015), №3, 144 – 150.
13. М.Г. Фуругян. Составление расписаний в многопроцессорных системах с несколькими дополнительными ресурсами // Известия РАН. Теория и системы управления. (2017), № 2, 57 – 66.
14. М.Г. Фуругян. Планирование вычислений в многопроцессорных системах с несколькими типами дополнительных ресурсов и произвольными процессорами // Вестн. МГУ. Сер. 15. Вычисл. математика и кибернетика. (2017), № 3, 38 – 45.

15. М.Г. Фуругян Некоторые алгоритмы решения минимаксной задачи составления многопроцессорного расписания // Известия РАН. Теория и системы управления. (2014), №2, 50 - 56.
16. A. Federgruen, H. Groenevel. Preemptive Scheduling of Uniform Machines by Ordinary Network Flow Technique // Management Science. (1986). Vol. 32, № 3, 341 – 349.