УДК 519.85

РЕШЕНИЕ ЗАДАЧИ МИНИМИЗАЦИИ ВРЕМЕНИ ВЫПОЛНЕНИЯ ЗАКАЗА ДЛЯ КОНВЕЙЕРА С ОТНОШЕНИМИ ПРЕДШЕСТВОВАНИЯ ЗАДАННЫМИ НАБОРОМ РЕКУРСИВНЫХ ФУНКЦИЙ

А.А. Лазарев

Институт проблем управления им. В.А. Трапезникова РАН

Россия, 117997, Москва, Профсоюзная ул., 65

E-mail: jobmath@mail.ru

Б.В. Куприянов

Институт проблем управления им. В.А. Трапезникова РАН

Россия, 117997, Москва, Профсоюзная ул., 65

E-mail: kuprianovb@mail.ru

Ключевые слова: теория расписаний, задачи RCPSP, конвейеры.

Аннотация: В статье вводится определение конвейера, описываемого связным ациклическим графом, каждая вершина которого, представляет собой операцию или функцию управления, ассоциированную с соответствующей рекурсивной функцией из некоторого конечного набора. Каждая рекурсивная функция определяет отношение предшествования операции конвейера. Рассматривается решение задачи минимизации времени выполнения заказа конвейером на конечном множестве возобновляемых ресурсов. Решение осуществляется методом Удовлетворения Ограничений, являющимся составной частью Constraint Programming.

1. Введение

Одним из основных классов задач теории расписаний являются RCPSP задачи. Суть их состоит в том, что дано множество требований и множество возобновляемых ресурсов. Необходимо таким образом распределить ресурсы, чтобы минимизировать время выполнения заказа. Такого рода задачи имеют место для различных систем [1], в том числе и для конвейерных систем. Как правило такие задачи являются полиномиально или NP трудными. В последнее время одним из направлений решения данной задачи является применение методов Constraint Programming и в частности сведению к Задаче Удовлетворения Ограничений (ЗУО) [2–4]. В данной работе рассматривается решение задачи RCPSP применительно к конвейерам, описываемым рекурсивными функциями [5] в постановке, когда время выполнения операции зависит от используемого ресурса, т.е. время выполнения операции i при использовании

ресурса j равно p_{ij}. Решение данной задачи сводится к ЗУО.

2. Определение конвейера

Модель конвейера представляет собой связный ациклический ориентированный граф с единственной конечной вершиной. Вершины графа помечены номерами из множества $I = \{1, 2, \ldots, n\}$ таким образом, что первые n_0 $(1 \leqslant n_0 < n)$ вершин являются начальными, а вершина n конечной. Множество дуг графа - упорядоченные пары вида (i, j), где $i, j \in I$. Тот факт, что в графе существует дуга (i, j) будем определять логической функцией $pred : I \times I \to \{true, false\}$. $pred(i, j)$ равна $true$, если в графе существует дуга (i, j) и $false$ в противном случае. . Каждая вершина

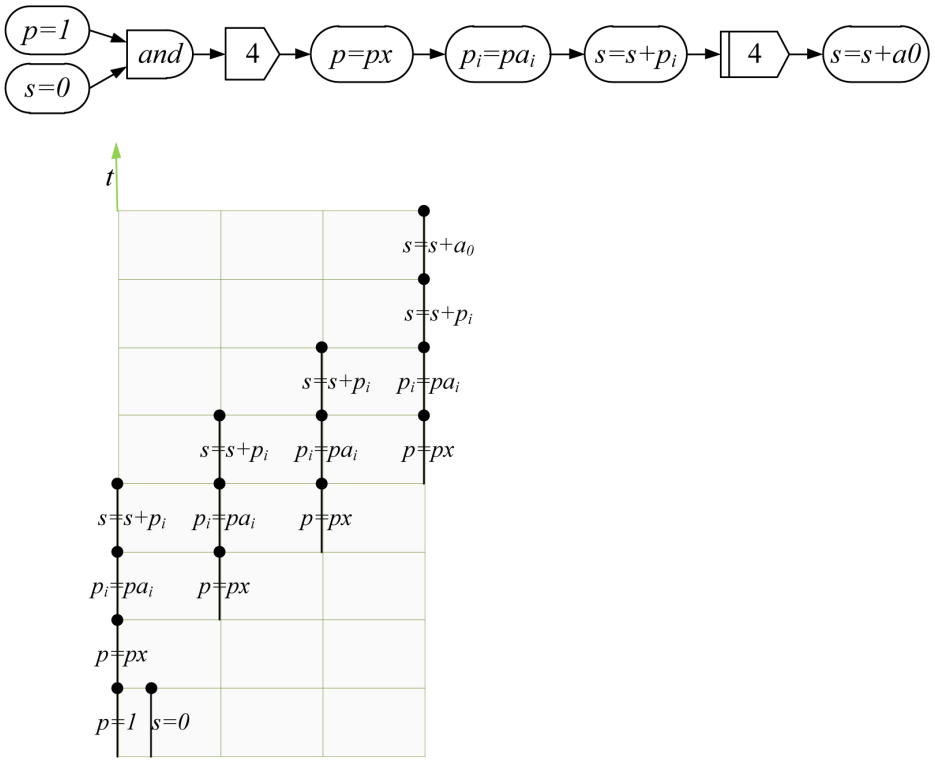

Рис. 1. Примеры графа конвейера и соответствующей временной диаграммы

графа по смыслу может быть либо операцией, обрабатывающей заказ, либо спусковой функцией, определяющей отношение временного предшествования операций. Под операциями подразумеваются производственные операции. На рис.1 приведен пример компьютерного конвейера, использующего функции мультиплицирования и редуцирования, который для потока входных данных $\{x\}$ вычисляет последовательность значений:

$P_1(x) = a_0 + a_1x_1 + a_2x^2 + a_3x^3 + a_4x^4$

$P_2(x) = a_0 + a_1x_1 + a_2x^2 + a_3x^3 + a_4x^4$

....

Пример временной диаграммы предполагает, что каждая операция выполняется собственным процессором.

Вершины графа помечены с помощью отображения $type : I \to E$, где $E = \{\mathbf{bop}, \mathbf{op}, \mathbf{and}, \mathbf{mul}, \mathbf{red}, \mathbf{get1}, \mathbf{get2}, \mathbf{put}\}$ – конечное перечислимое множество. Каждая константа множества E обозначает тип вершины графа, а сама вершина имеет соответствующую графическую нотацию (см. на рис. 1 верхний граф).

Расписание выполнения операций строится с помощью рекурсивных функций, определенных на множестве $I \times K$, где $K \subset N_0$ конечное множество и вычисляющих время завершения обработки i-й операцией k-го заказа при $k \in K$. Для каждого элемента множества E своя рекурсивная функция. Каждая вершина i графа в соответствии с ее типом ассоциируется с одной из рекурсивных функций.

$$t(n,k) = \begin{cases} bop(i,k),\ if\ type(i) = \mathbf{bop}; \\ op(i,k),\ if\ type(i) = \mathbf{op}; \\ and(i,k),\ if\ type(i) = \mathbf{and}; \\ mul(i,k),\ if\ type(i) = \mathbf{mul}; \\ red(i,k),\ if\ type(i) = \mathbf{red}; \\ get1(i,k),\ if\ type(i) = \mathbf{get1}; \\ get2(i,k),\ if\ type(i) = \mathbf{get2}; \\ put(i,k),\ if\ type(i) = \mathbf{put}; \end{cases} \tag{1}$$

Рекурсивная функция $t(i,k)$, обеспечивает суперпозицию рекурсивных функций, соответствующих вершинам графа и является обращением к одной из них в соответствии с типом вершины (см. на рис. 1 нижний граф). Вычисление времени завершения выполнения k-го заказа конвейером осуществляется с помощью обращения к рекурсивной функции $t(n,k)$. Эта же формула вычисляет время завершения обработки k-го заказа любой i-ой операцией конвейера с помощью обращения $t(i,k)$.

Каждая вершина графа может иметь одну или две входных дуги в зависимости от типа вершины.

Время начала функционирования конвейера полагается равным 0.

Каждая операция i конвейера использует некоторый возобновляемый ресурс $\mu_j \in m_i$ $(1 \leqslant j \leqslant |D_i|)$ и характеризуется временем выполнения p_{ij}, которое зависит от используемого ресурса и является константой. $m_i \in D_i$ множество ресурсов, которое реально используется для выполнения операции i, а D_i домен переменной m_i (множество ее значений) представляет собой множество множеств ресурсов каждое из которых потенциально может быть использовано для выполнения операции i при обработке какого либо заказа.

M – множество возобновляемых ресурсов конвейера. Обозначим через M^+ булеан множества M (т.е. множество всех его подмножеств) за исключением пустого множества. В этом случае $D_i \subseteq M^+$. В каждый момент времени каждая операция может использовать только один ресурс и каждый ресурс может использоваться только одной операцией.

Прерывание выполнения операции запрещено.

Спусковые функции ресурсы не потребляют и время их выполнения равно 0.

3. Постановка задачи

Пусть имеется конвейер с описанными выше свойствами на базе ациклического связного графа. Ставится задача - распределением возобновляемых ресурсов выпол-

нить на конвейере за минимальное время k заказов. Т.е. задача сводится к минимизации функции $t(n,k)$ для заданного k на конечном множестве возобновляемых ресурсов M.

Первым этапом решения задачи оптимизации методом УО является формулировка постановки задачи. В теории ЗУО [6] рассматривается как четверка (V,D,R,C), где $V=\{v_1,v_2,\ldots,v_n\}$, множество переменных, $D=\{D_1,\ldots,D_n\}$ - множество доменов переменных, R - множество отношений различной арности над областями D. $C=\{C_1,...,C_m\}$ - множество ограничений, связывающих множество значений переменных из V посредством отношений из R. Решение ЗУО это присвоение значений всем переменным множества V, которое удовлетворяет всем ограничениям. Целью решения ЗУО может быть нахождение одного или всех решений.

Перенесем общую постановку ЗУО в рассматриваемую нами прикладную область. Четверка ЗУО будет выглядеть следующим образом:

1) $V=\{m_i\}$, где m_i - переменная связаная с соответствующей i-й вершиной графа $(1\leqslant i\leqslant n)$ и представляет собой множество ресурсов, выделенных i-й операции для обработки k заказов.

2) D_i - домен или множество значений переменной m_i.

3) $R\subseteq(I\times K\times M)$.

4) C- множество ограничений, которое делится на несколько групп.

1- я группа это множество унарных ограничений, определенных для начальных вершин графа и имеющих вид:

$(bop(i,k)=p_{ij})\&(1\leqslant i\leqslant n_0)\&(\mu_j\in m_i)\&(k=0)\&(type(i)=\mathbf{bop})$;
$(bop(i,k)=bop(i,k-1)+p_{ij})\&(1\leqslant i\leqslant n_0)\&(\mu_j\in m_i)\&(k>0)\&(type(i)=\mathbf{bop})$.

2- я группа это множество бинарных ограничений, определенных для пары вершин графа (l,i) и имеющих вид:

$pred(l,i)\&(op(i,k)=\max(t(l,k),op(i,k-1))+p_{ij})\&(n_0<i)\&\ \&(\mu_j\in m_i)\&(k>0)\&(type(i)=\mathbf{op})$;
$pred(l,i)\&(op(i,k)=t(l,k)+p_{ij})\&(n_0<i)\&(\mu_j\in m_i)\&(k=0)\&(type(i)=\mathbf{op})$;
$pred(l,i)\&(mul(i,k)=t(l,\lfloor k/q_i\rfloor)\&(n_0<i)\&(k\geqslant 0)\&(type(i)=\mathbf{mul})$;
$pred(l,i)\&(red(i,k)=t(l,(k+1)q_i-1)\&(n_0<i)\&(k\geqslant 0)\&(type(i)=\mathbf{red})$;
$pred(l,i)\&(get1(i,k)=t(l,2k)\&(k\,mod 2=0)\&(n_0<i)\&(k\geqslant 0)\&(type(i)=\mathbf{get1})$;
$pred(l,i)\&(get2(i,k)=t(l,2k+1)\&(k\,bmod 2=1)\&(n_0<i)\&(k\geqslant 0)\&(type(i)=\mathbf{get2})$.

3- я группа это множество ограничений, определенных для тройки вершин графа (l,i) и (m,i) имеющих вид:

$pred(l,i)\&pred(m,i)\&(and(i,k)=\max(t(l,k),t(m,k))\&(n_0<i)\&(type(i)=\mathbf{and})$;
$pred(l,i)\&pred(m,i)\&(put(i,k)=t(l,k))\&(k=0)\&(type(i)=\mathbf{put})$;
$pred(l,i)\&pred(m,i)\&(put(i,k)=\max(put(i,k-1),t(l,(k-1)/2)))\&$
$\&(k\,mod 2=1)\&(k>0)\&(type(i)=\mathbf{put})$;
$pred(l,i)\&pred(m,i)\&(put(i,k)=\max(put(i,k-1),t(m,k/2)))\&$
$\&(k\,mod 2=0)\&(k>0)\&(type(i)=\mathbf{put})$.

4-я последняя группа ограничений относится к глобальным ограничениям вида **all-different**. Суть ее в том, что использование ресурса на некотором временном интервале исключает использование этого ресурса на другом интервале, который пересекается с первым. Для формализации данного ограничения введем следующие обозначения:

$\tau_{js}=(j;t'_s,t''_s)$ обозначает интервал времени от t' до t'' в течение которого ис-

пользуется ресурс μ_j и интервал имеет номер s. Далее приведем определение операции $\bigcap$, которая принимает значение истины, если временные отрезки пересекаются.
$\tau_{js} \bigcap \tau_{lf} = \begin{cases} true,\ if\ t'_{lf} < t''_{js} \leqslant t''_{lf}; \\ false,\ otherwise. \end{cases}$

L_j определяет количество использований ресурса μ_j при обработке конвейером k заказов.

В этих определениях глобальное ограничение будет выглядеть следующим образом:

$(1 \leqslant j \leqslant |M|)\&(1 \leqslant s, f \leqslant L_j)\&(s \neq f)\&\neg(\tau_{js} \bigcap \tau_{jf})$.

Решением ЗУО будет нахождение таких значений переменных m_i и троек (i, μ_j, k), при которых все перечисленные условия будут истинны.

В данном случае ЗУО будет с дискретной переменной и конечным множеством значений.

Построение расписания для конвейера вычислением суперпозиции рекурсивных функций однозначно предполагает для решения ЗУО вариант поиска с возвратами.

Работа выполнена при поддержке Российского научного фонда № 17-19-01665.

Список литературы

1. Лазарев А.А., Гафаров Е.Р. Теория расписаний. Задачи и алгоритмы. М.: МГУ, 2011. 223 с.
2. Щербина О.А. Интеллектуальные системы. 2011. Т. 15, № 1-4. С. 53-170.
3. Van Beek P. Reasoning about qualitative temporal information // Artificial Intelligence. 1992. Vol. 58. P. 297–326.
4. Нариньяни А.С., Седреева Г.О., Седреев С.В., Фролов С.А. TimeEX/Windows - новое поколение недоопределенной технологии календарного планирования // Проблемы представления и обработки не полностью определенных знаний / Под ред. И.Е. Швецова. Москва-Новосибирск, 1996. С. 101-116.
5. Куприянов Б.В. Рекурсивные конвейерные процессы – основные свойства и характеристики // Экономика, статистика и информатика. Вестн. УМО. 2015. № 1. С. 163-170.
6. Куприянов Б.В. Применение модели конвейерных процессов рекурсивного типа для решения прикладных задач // Экономика, статистика и информатика. Вестник УМО. 2014. № 5. С. 170-179.