基于时间 Petri 网的多处理机的调度算法

韩 咚[1],陈 波[2]

(1. 山东科技大学 信息科学与工程学院,山东 青岛 266510;
2. 国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:任务调度是并行分布式计算机中最有挑战性的问题之一。如何合理有效地进行任务调度将直接影响到系统的并行效率。文中通过将任务图转换为时间 petri 网的方法,利用求时间 petri 网的可覆盖树的方法来分析网系统的状态变化和变迁的发生序列,从而求出关键路径和顺序队列。再将该队列分配到处理机上,来缩短相关任务图的调度长度。

关键词:并行算法;任务调度;时间 petri 网;可达树;关键路径;多处理机

中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2007)06-0015-03

Algorithm of Multiprocessor Scheduling Based on Time Petri Nets

HAN Dong[1], CHEN Bo[2]

(1. Sch. of Info. Sci. & Eng., Shandong Univ. of Sci. and Tech., Qingdao 266510, China;
2. Computer Sch., National Univ. of Defence Tech., Changsha 410073, China)

Abstract: Task scheduling is one of most challengable problems in parallel and distributed computing. How to schedule the parallel tasks onto the processors will greatly influence the parallel computing performance of the applications. By the method of translating tasks graph into time petri nets(TPN), analyzes the state covertion and transition fire sequence. By using the TPN coverability tree, finding out the critical path and the allocating sequence. Then allocate the sequence to the processors. It can shorten the scheduing length of the task graph.

Key words: parallel algorithm; task schedule; time petri net; reachable tree; critical path; multiprocessor

0 前 言

Petri 网是一种用于并发、异步系统建模和分析的重要工具。近年来随着 Petri 网研究的不断深入,利用 Petri 网在任务调度描述方面的良好表现,可以建立一种模型来模拟和控制时间调度。在对于 Petri 网的应用中,分析 Petri 网最广泛最基本的方法是利用它的可达性分析技术,建立相对应的可达树,利用可达树可以发现系统的相关行为特征,来求出转换为时间 Petri 网的可达图的关键路径和顺序队列。

并行计算调度一般分为静态调度、动态调度和混合调度三种。在仿真中选用动态调度或混合调度,因算法复杂,调度开销太大,无法满足仿真实时性要求。因此调度策略更适合于连续系统的并行仿真问题[1]。

收稿日期:2006-09-03

作者简介:韩 咚(1982-),男,山东泰安人,硕士研究生,研究方向为并行算法。

1 时延 Petri 网简介

1.1 时延 Petri 网的定义

定义 1 时延 Petri 网(Timed Petri Net)是一个五元组:TPN= $\{P, T, F, \tau, M\}$,其中:$P = \{p_1, p_2, \cdots, p_n\}(n \geqslant 0)$ 是一个位置的有限集合;$T = \{t_1, t_2, \cdots, t_m\}(m \geqslant 0)$ 为变迁的有限集合,且 $P \cap T = \varnothing$;$F \subseteq (P \times T) \cup (T \times P)$ 为一个有向弧集合;τ 是时间映射函数,$\tau: T \rightarrow 0 \cup Q^+$ (Q^+ 为正有理数),规定网中每个变迁的持续时间。当持续时间为 0 时,称之为瞬时变迁,不为 0 时称为时延变迁。

1.2 时延 Petri 网的性质

定义 2 对于 TPN= (P, T, F, τ, M),如果存在 $t \in T$,使得 $M[t > M'$,则称 M' 为从 M 直接可达的。从 M 可达的一切标识的集合记为 $R(M)$,约定 $M \in R(M)$。如果记变迁序列 $t_1, t_2, \cdots, t_k$ 为 σ,则也可以记为 $M[\sigma > M_K$[2]。

定义 3 变迁 t_i 在时刻 $\tau_i + \theta_i (\alpha_i \leqslant \theta_i \leqslant \beta_i)$,在标识 M 下是发生的当且仅当:$\forall p(M(p) \geqslant w(p,$

$t_i), p \in t_i)$。若变迁发生后的标识为 M'，则记为 $M[>M'$。

$$M'(P) = \begin{cases} M(p) - w(p,t_i) & \text{当 } p \in {}^{\cdot}t_i - t_i \\ M(p) + w(t_i,p) & \text{当 } p \in t_i^{\cdot} - {}^{\cdot}t_i \\ M(p) - w(p,t_i) + w(t_i,p) & \text{当 } {}^{\cdot}t_i \cap t_i^{\cdot} \\ M(p) & \text{当 } p \notin {}^{\cdot}t_i \cup t_i^{\cdot} \end{cases}$$

定义 4　变迁的化减规则：若 $(m_0(p) = 0)$，并且有 $(t_1 = \{p\} = {}^{\cdot}t_2) \wedge (p \notin {}^{\cdot}t_1, p \notin t_2)(({}^{\cdot}t_1) = \{t_1\}) \vee (SI(t_2) = [0,0]))$，可用变迁 t_f 替换变迁 $t_1, t_2, SI(t_f) = SI(t_1) + SI(t_2); {}^{\cdot}t_f = {}^{\cdot}t_1, t_f = t_2^{\cdot}$。

2　任务图与时延 Petri 网(TPN)之间的转换过程

定义 5　网 $N = (P, T; F)$ 称 S－图当且仅当 $\forall t(|{}^{\cdot}t| = |t^{\cdot}|) = 1, (t \in T)$；如果 N 是一个 S－图，M 为 N 的一个标识，则称 (M, N) 为标识 S－图。

p－结点和一个 t－结点串连在一起标识一个任务，其中 P_i 表示任务 j 的开始，P_{i+1} 表示任务 j 的结束，t_j 中有一个托肯表示任务 j 在执行。每个 j 的持续时间为 t_j，当持续时间为 0 时，称之为瞬时变迁，不为 0 时称为时延变迁。其时间即为任务的执行时间，在图 1 中为 d 个时间单位。因篇幅所限，详细的构造与化简过程见参考文献[3]。

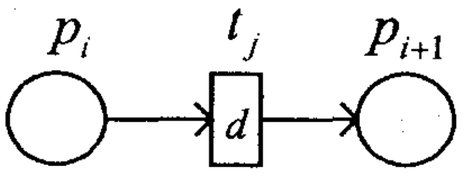

图 1　单个任务表达

整个任务图(如图 2 所示)的时延 Petri 网可以构造如图 3 所示。

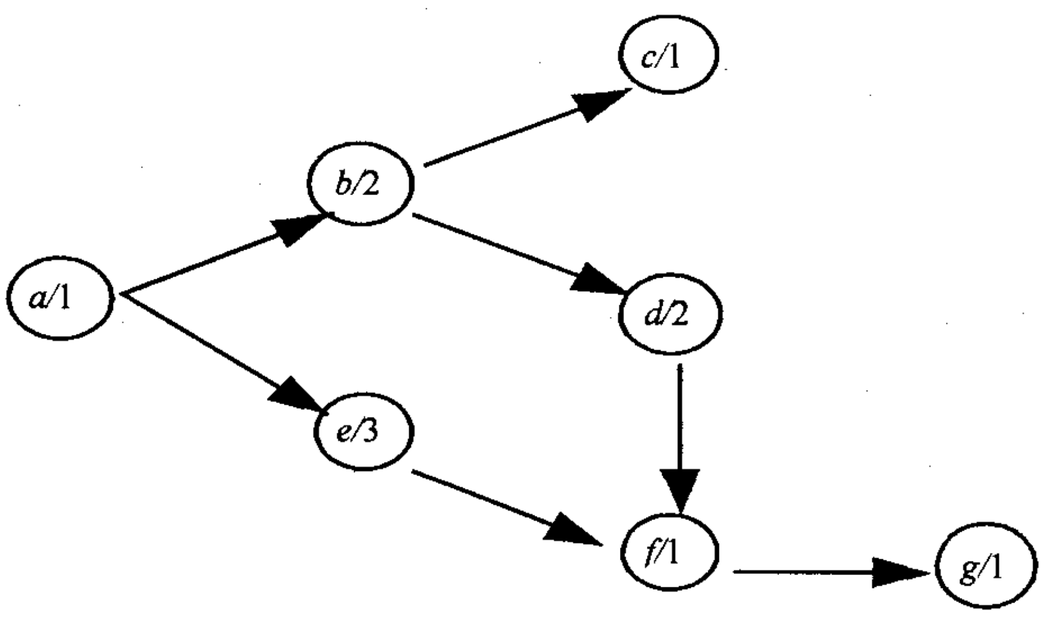

图 2　任务图

将时延 Petri 网简化后，每个变迁都可以对应一个任务图中的子任务，如 t_a 延迟时间值等于子任务 a，τ_i 表示 t_i 的延迟时间。在初始库所中加上标识，使得 $M_0(p_1) = 1$ 且 $\forall p(p \in P, p \neq p_1, M_0(p) = 0)$。若任务图由多个初始子任务构成，则建立时延 Petri 网时要建立辅助变迁 t_0 和辅助库所 P_0，使得 $(P_0, t_0) \in F$，$M_0(p_0) = 1, \forall p(p \in P, p \neq p_0, M_0(p) = 0)$。

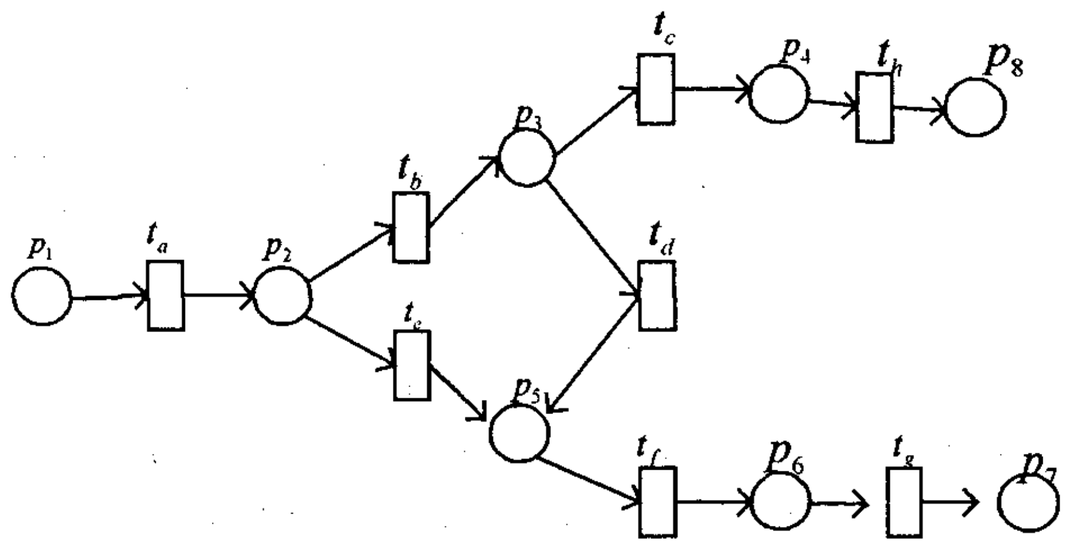

图 3　化减后的时延 Petri 网图

3　构造可达树与求其关键路径

定义 6　关键路径(CP)：在可达树中，初始标识为 M_0, M_K 为终端结点标，存在变迁序列 σ，使得 $M_0[\sigma > M_k$，变迁延迟时间之和最长的变迁序列 σ 称为可达图的关键路径。需要注意的是：在求关键路径的时候，若时延 Petri 网中某个变迁发生多次，可达树上仅计算其发生一次的时间延迟。关键路径上的变迁结点记为 CPN。

定义 7　构造可达树的方法如下：

输入 $\sum = (S, T; F, M_0)$

输出 $CT(\sum)$

算法步骤：

0 以 M_0 作为 $CT(\sum)$ 的根结点，并标以“新”

1 while 标注为新的结点存在 do 任选一个标注为新的结点，设为 M；

2 if 从 M_0 到 M 的有向路上有一个结点的标识等于 M then 把 M 的标注改为“旧” 返回 Step1；

3 if $\forall t \in T: \neg M[t >$ then 把 M 的标注改为“端点” 返回 Step1；

4 For 每个满足 $M[t >$ 的 $t \in T$ do；

4.1　计算 $M[T > M'$ 中的 M'；

4.2　if 从 M_0 到 M' 的有向路上存在 M''，使得 $M'' < M'$，then 找出使 $M''(s_j) < M'(s_j)$ 中的分量 j，把 M' 的第 j 个分量改为 ω；

5 新增加一项如下：if $\forall p \in P, {}^{\cdot}p = t, p^{\cdot} = \varnothing$，$M_i$ 为网中任意一个标识，且 $M_i[t > M$，把 M 的标注改为端点，返回 Step1。

画出可达树图如图 4 所示，其中括号中的 τ_i 代表了变迁的延迟。

通过图 4 知道文中例子中 CPN 依次为 t_a, t_b, t_d, t_f, t_g。

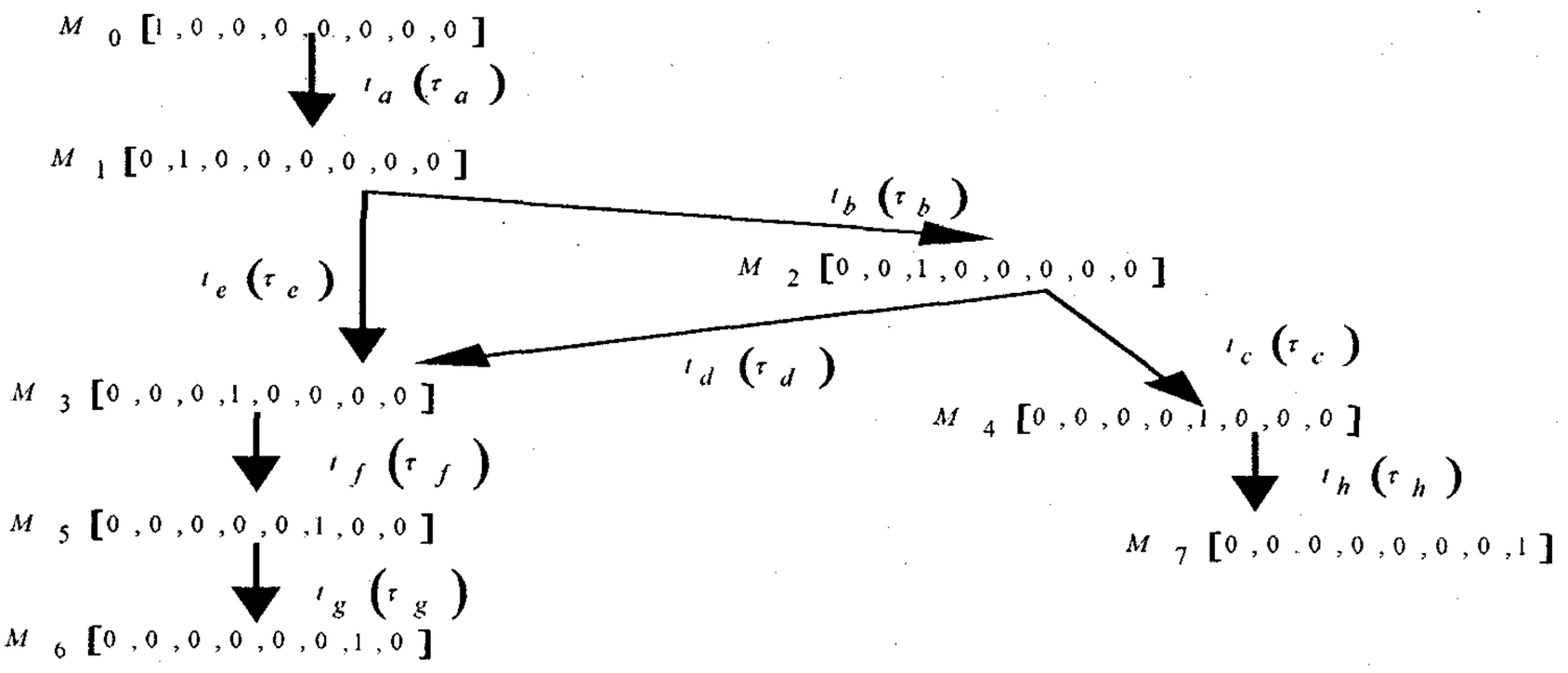

图 4　变迁延迟的可达树

4　支配序列的构造和处理机分配

定义 8　分支上结点(IBN):若 $t_1 \in CPN, t_1^{\cdot} = {}^{\cdot} t_2$ 或者 $t_2^{\cdot} = {}^{\cdot} t_1$ 则称 t_2 为 IBN。分支外结点(OBN) 指的是除去 IBN 与 CPN 所剩下的时间 Petri 网中变迁[4]。

为描述方便,给出下面几个记号:$\mathrm{ect}(t_i)$ 表示结点 t_i 的最早完成时间,显然,$\mathrm{ect}(t_i) = \mathrm{est}(t_i) + \tau_i$。$\mathrm{est}(t_i)$ 表示结点 t_i 的最早启动时间,满足 $\mathrm{est}(t_j) = \left\{ \begin{array}{l} 0, \mathrm{pred}(t_j) = \varnothing \\ \min\{\mathrm{est}(t_j), \mathrm{est}(t_k)\}, t_j \in \mathrm{pred}(t_j) \cap t_k \in \mathrm{pred}(t_j), i \neq k \end{array} \right\}$

若 $t_i^{\cdot} = {}^{\cdot} t_j$,则称 t_i 为 t_j 的直接前驱变迁结点,记为 $\mathrm{pred}(t_j)$,$f\mathrm{pred}(t_i)$ 表示 t_i 的最优直接前驱变迁结点。有 $f\mathrm{pred}(t_i) = \{t_j \mid \mathrm{ect}(t_j) \geqslant \mathrm{ect}(t_k), t_j, t_k \in \mathrm{pred}(t_i), k \neq j\}$。

l_i:设 M_i 为网中的任意标识,$M_i[\sigma > M_k, M_k$ 为终端标识,$\sigma = \{\sigma_1, \sigma_2, \cdots, \sigma_i\}$ 拥有最大变迁延迟时间的 σ_i 的具体时间值称为 l_i。

$P(t_i)$ 表示变迁结点 t_i 分配到的处理机(注意:假设可应用的处理机数目没有限制)。

4.1　CPN 支配序列的构造

计算可达树关键路径和每个变迁结点的 l_i,将 CPN 中第一个结点作为 CPN 支配队列的第一个结点。

WHILE 关键路径上还有未分配的变迁结点 DO

设 t_i 为下一个需要考虑的 CPN 结点　IF t_i 的所有父亲结点都在队列中　THEN

将 t_i 加入到 CPN 支配队列中; ELSE

如果 $t_j \in \mathrm{pred}(t_i)$ 不在队列中,且有多个这样的父亲结点,则 l_i 最大者优先进入 CPN 支配队列。如果 t_j 的所有父亲结点都在队列中,就将 t_j 放置到队列中,否则,将 t_j 的所有前驱变迁结点递归地放置到队列中。将下一个 CPN 结点 t_i 放置到队列中,所有父亲结点都在队列中的 IBN,以 l_i 大者优先调度到 CPN 支配序列

END IF END WHILE 将 OBN 按 DAG 的拓扑逆序依次放入

依照图 2 的例子,根据算法得出的 CPN 支配队列为 abdefgch。

4.2　处理机的分配

对于 CPN 支配队列的不同结点,在处理机数目不限制的情况下处理机分配[5]遵循以下规则:

(1) 对于 CPN 上结点 t_i,只要成为就绪结点,就调度到 $p(f\mathrm{pred}(t_{(i)}))$。

(2) 对于 IBN 上节点 t_k,若 $t_k \in \mathrm{pred}(t_i)$,$t_i$ 是 CPN,且

$$\mathrm{est}(P_{t_k, p(f\mathrm{pred}(t_i))}) + \tau_k < \mathrm{est}(P_{k, p(f\mathrm{pred}(t_k))}) + \tau_k \quad (1)$$

则调度 t_k 到 $P(f\mathrm{pred}(t_i))$,否则,调度 t_k 到 $P(f\mathrm{pred}(t_k))$;如果 $t_k \notin \mathrm{pred}(t_i)$,则调度 t_k 到 $P(f\mathrm{pred}(t_k))$。

(3)对于 OBN 上变迁结点,调度到使其启动时间最短的处理机。

依据上述算法所构造的甘特图如图 5 所示。

a	b	d	c	
	e	f	g	

图 5　相关甘特图

5　结　论

在对静态多处理机调度问题处理时,通过建立时间 Petri 网模型对其可达树进行分析是一种比较直观有效的方法,但是对过于复杂的调度问题,构造可达树进行分析的过程太复杂,甚至有引发状态爆炸的可能,另外,文中没有考虑到子任务间存在延迟和处理机之间的通信延迟问题,需要在以后工作中加以研究。

参考文献:

[1] 陈国良.并行算法——排序和选择[M].北京:高等教育出版社,1999.

[2] 吴哲辉.Petri 网导论[M].北京:机械工业出版社,1985.

[3] 孔德华,吴哲辉.一个基于时间 Petri 网的多处理机静态调度的方法[J].系统仿真学报,2005(10):135－138.

[4] 吴哲辉,王美琴.一类含时间因素的 P 网及其在工程上的应用[J].应用数学学报,1987,10(3):290－299.

[5] 尚明生.相关任务图的一种有效并行调度算法[J].计算机工程,2005,14(7):18－21.