文章编号:1001 - 9081(2008)08 - 2177 - 04

分布式嵌入式系统实时调度的建模

张海涛[1],邱联奎[1],艾云峰[2]
(1. 河南科技大学 电子信息工程学院,河南 洛阳 471003; 2. 中国科学院研究生院 计算与通信工程学院,北京 100049)
(zhang_haitao@163.com)

摘 要:针对 RBTPN 模型在建模分布式嵌入式系统实时调度时的不足,提出了一种新的扩展时间 Petri 网模型。该模型通过在需要处理器资源的变迁上引入变迁速率因子,得到具有相同优先级变迁的运行速率函数,从而在分布式嵌入式系统的调度建模中,在单个处理器上结合了固定优先级可抢先调度和轮转调度。随后给出了该模型可达图的构造方法,以便可以得到调度序列的各种性质。

关键词:Petri 网; 嵌入式系统; 调度; 建模
中图分类号: TP311 **文献标志码**:A

Modeling of real-time scheduling for distributed embedded systems

ZHANG Hai-tao[1], QIU Lian-kui[1], AI Yun-feng[2]
(1. *Electronic and Information Engineering College, Henan University of Science and Technology, Luoyang Henan* 471003, *China*;
2. *College of Computing and Communication Engineering, Graduate University of the Chinese Academy of Sciences, Beijing* 100049, *China*)

Abstract: Aiming at the deficiencies of RBTPN in modeling real-time scheduling of distributed embedded system, we put forward a new extended time Petri Net model on its base. The model introduces transition rate factor on transitions that need processor resources, and gets working rate function of transitions that have the same priority, so that the model combine preemptive scheduling based on fixed priority and round-robin scheduling on single processor in the scheduling modeling of distributed embedded system Then we gave construction method of the model's reachable graph so as to get the characteristics of scheduling sequences.

Key words: Petri net; embedded system; scheduling; modeling

0 引言

由于嵌入式系统变得越来越复杂,为了提高产品开发速度,降低开发成本,在嵌入式系统设计中,使用形式语言建模变得越来越重要。Petri 网作为一种形式语言,具有极强的建模能力,可以描述并行、同步、冲突、异步等各种事件[1]。目前,各种时间 Petri 网已经被应用于嵌入式系统的建模、评估和分析[2-3]。

调度在嵌入式系统中起着关键的作用,不同的调度算法可能产生截然不同的结果。基于资源的时间 Petri 网(Resources Based Time Petri Nets)[4-5]通过在表示计算和通信任务的变迁上附着所占用的处理器或通信资源,以及相应的优先级,从而可以直观地建模分布式嵌入式系统的实时调度。对于在同一处理器上具有相同优先级的变迁,RBTPN 模型采用随机的方法进行调度,且不可抢先。然而,在许多嵌入式系统中,常采用时间片轮转调度方法去调度相同优先级的任务。为此,需要改进 RBTPN 模型,使其可以建模时间片轮转调度。

1 基于资源的时间 Petri 网

基于资源的时间 Petri 网是由多个元素描述的有向图 $RBTPN = \{P,T,I,O,M_0,SR\}$,式中 $P = \{p_1,p_2,\cdots,p_n\}$ 是位置的有限集合,$n > 0$ 为位置的个数。$T = \{t_1,t_2,\cdots,t_m\}$ 是变迁的有限集合,$m > 0$ 为变迁的个数,且 $P \cap T = \varnothing$。$I: P \times T \rightarrow N$ 是输入函数,定义了从 P 到 T 的有向弧的权的集合。$O: T \times P \rightarrow N$ 是输出函数,定义了从 T 到 P 的有向弧的权的集合。$M_0: P \rightarrow N$ 是最初的标识函数,定义了初始时刻,每个位置中托肯的数目。调度资源 SR 主要包括以下参数:$Proc = \{proc_1, proc_2,\cdots proc_l\}$ 是有限的处理器集合,$l > 0$ 是微处理器的个数。该资源可以抢先。$Com = \{com_1,com_2,\cdots,com_s\}$ 是有限的通信资源集合,$s > 0$ 是通信资源的个数。该资源不可抢先。$\omega: T \rightarrow N$,$\gamma: T \rightarrow Proc$,$\varphi: T \rightarrow Com$ 分别为优先级、处理器和通信资源分配函数。

RBTPN 模型主要用于建模以下的分布式嵌入式系统的实时调度:在处理器上采用基于固定优先级的抢先式调度,而处理器之间的通信采用基于固定优先级的不可抢先式调度。

如图 1 所示,使用 RBTPN 模型建模了一个简单的系统。处理器 $proc_1$ 有三个相关的任务,通过通信通道 com_2 传递到另一个位置。其中,$P = \{p_1,p_2,p_3,p_4,p_5,p_6,p_7,p_8,p_9\}$,$T = \{t_1,t_2,t_3,t_4,t_5,t_6\}$,$Proc = \{\phi,proc_1\}$,$Com = \{\phi,com_2\}$,$t_1^* = p_2$,$^*t_1 = \{p_1\}$,$M_0 = (1,0,0,1,0,0,1,0,0)$。

在图 1 中,由于变迁 t_1、t_3 和 t_5 具有不同的优先级,因而可以实现基于固定优先级的不可抢先式调度。在处理器 $proc_1$ 上,变迁 t_1 具有较高的优先级,因而可以确定变迁 t_5 首先执行,然后变迁 t_3 执行,最后变迁 t_5 执行。然而,在图 1 中,如果使变迁 t_3 和 t_5 具有相同的优先级 $w = 2$,那么在变迁 t_1 执行完

收稿日期:2007 - 12 - 17;**修回日期**:2008 - 02 - 18。 **基金项目**:国家自然科学基金资助项目(60573078)。

作者简介:张海涛(1972 -),男,河南洛阳人,副教授,博士,主要研究方向:智能控制、嵌入式系统; 邱联奎(1974 -),男,河南洛阳人,副教授,博士,主要研究方向:机器人视觉伺服控制、模式识别、自主移动机器人、计算机视觉、嵌入式实时系统、数据可视化; 艾云峰(1979 -),男,山东济南人,博士,主要研究方向:实时嵌入式系统、汽车电子、智能交通。

毕后，变迁 t_3 和 t_5 将被随机调度；如果变迁 t_1、t_3 和 t_5 都具有相同的优先级，那么这三个变迁都将被随机调度。

然而，在嵌入式系统中，对于同一个处理器上具有相同优先极的任务，经常采用时间片轮转调度。为此，可以改进 RBTPN 模型，引入一种扩展的基于资源的时间 Petri 网（ERBTPN）。

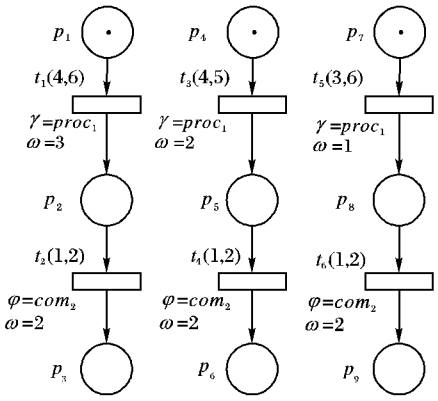

图1 示例1

2 ERBTPN 模型

在 RBTPN 模型的基础上，可以引入新的参数，定义 ERBTPN 模型。

2.1 模型的定义

定义1 扩展的基于资源的时间 Petri 网，是由多个元素描述的有向图：

$$ERBTPN = \{N, g\}$$

其中，N 是一个 RBTPN 模型；g 是每个变迁的运行速率因子。在 ERBTPN 模型中，在变迁内的数字就是速率因子 g。g 的取值范围为 $[0, +\infty)$。

在时间轮转调度中，每个进程被分配一个时间片，时间片的长度决定了该进程运行的速率。在 ERBTPN 模型中，g 就是进程运行的速率因子。当然，如果在某个状态下，仅有一个变迁处于运行状态，那么参数 g 也就没有意义了。

定义2 每个变迁 $t_i \in T$，具有相关联的静态时间间隔 $STI(t_i) = (\alpha^s(t_i), \beta^s(t_i))$ 和动态的时间间隔 $(\alpha(t_i), \beta(t_i))$。

定义3 RBTPN 的状态类记作 $C = (M, D)$，其中 M 是状态类的标识，表示每个位置所含的托肯数。D 是所有使能的变迁的时间触发域。

定义4 一个变迁 $t_i \in T$ 在状态类 C 下称作使能的，当且仅当 $\forall p \in {}^*t_i: M(p) \geqslant I(p,t)$ 和 $\forall p \in t_i^*: M(p) + O(t,p) \leqslant C$。一个使能的变迁 $t_i \in T$ 称作运行变迁，当且仅当 t_i 请求的资源没有被另一个具有较高优先级的使能的变迁所请求。一个使能的，但没有运行的变迁称作就绪变迁。在状态类 C 下所有使能的、运行的、就绪的变迁集合分别记作 $enabled(C)$，$working(C)$，$halting(C)$。

如图2所示，显示了一个 ERBTPN 模型，在初始状态时，由于变迁 t_1 具有较高的优先级，因而首先执行。当该变迁执行完毕，释放处理器资源 $proc_1$ 之后，变迁 t_3 和 t_5 具有相同的优先级，都可以占用处理器资源 $proc_1$。但与 RBTPN 模型不同的是，变迁 t_3 和 t_5 都具有速率因子1。

2.2 变迁运行速率的定义

假定在状态类 C 下，有多个变迁处于使能状态。那么该变迁要成为运行变迁 $working(C)$，就必须满足以下条件之一：

1）该变迁不需要任何资源，包括处理器资源和通信资源；

2）该变迁需要通信资源，并获得了该资源；

3）该变迁需要处理器资源，该资源没有被优先级更高的变迁所占用。

而对于满足第三个条件，需要处理器资源 $proc_j$ 的变迁，记为 $EWorking(C, proc_j)$。那么就可以作以下定义。

定义5 在 ERBTPN 模型中，假定在某个状态下，需要相同处理器资源的多个变迁都处于使能状态，那么每个变迁的执行速率，可以按照下面的公式计算[6]。

1）对于变迁 $t_i \in EWorking(C, proc_j)$：

$$r(t_i, C) = \frac{g(t_i)}{\sum_{t_i \in EWorking(C, proc_j)} g(t_i)}$$

2）对于变迁 $t_i \in Working(C) \cap t_i \notin EWorking(C, proc_j)$：

$$r(t_i, C) = 1$$

3）对于变迁 $t_i \in halting(C)$：

$$r(t_i, C) = 0$$

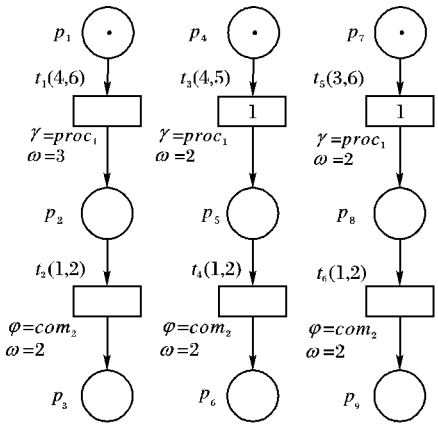

图2 示例2

2.3 变迁运行时间的计算

为每个变迁 $t_i \in T$，引入局部时钟变量 ν_i，即 $V: T \to R_+$ 是时钟函数，其中，$V = \{\nu_1, \nu_2, \cdots, \nu_m\}$ 表示时钟变量集合，每个时钟变量都是一个正实数。

变迁 t_i 的时钟变量 ν_i 表明自从变迁 t_i 成为运行变迁后，所经历的时间。当然，对于需要处理器资源的变迁，这段时间不一定连续增加，这类变迁可能由于处理器资源的丢失而成为就绪变迁。总之，ν_i 是变迁 t_i 从上次复位后到目前所执行的时间，中间可能由于处理器被抢占而成为就绪变迁。

在 ERBTPN 中，假定 ν_0 变迁成为运行变迁的时刻，而从 ν_0 时刻到 ν_i 时刻，变迁 t_i 所占用的时间可以记为 $st_{v_0}^{v_i}(t)$，其值可以按照以下公式计算。

1）对于变迁 $t_i \in EWorking(C, proc_j)$：

$$st_{v_0}^{v_i}(t) = r(t_i, C) \times (v_i - v_0)$$

2）对于变迁 $t_i \in Working(C) \cap t_i \notin EWorking(C, proc_j)$：

$$st_{v_0}^{v_i}(t_i) = (v_i - v_0)$$

从上可知，某个变迁 t_i 从其成为运行变迁后，直到其触发，执行完毕，其执行速率会不断发生改变。例如，在图2中，变迁 t_1 执行完毕，释放处理器资源 $proc_1$ 之后，变迁 t_3 和 t_5 都成为运行变迁。由于变迁 t_3 和 t_5 的速率函数都是1，因而变迁 t_3 和 t_5 的运行速率都是：

$$r(t_3, C) = r(t_5, C) = \frac{1}{g(t_3) + g(t_5)} = \frac{1}{2}$$

而假定变迁 t_3 首先执行完毕,那么变迁 t_5 的运行速率 $r(t_5,C')=1$;同样,如果假定变迁 t_5 首先执行完毕,那么变迁 t_3 的运行速率 $r(t_3,C')=1$;

在图 2 中,如果也令变迁 t_1 的优先级 $w=2$,而令其速率因子等于 ∞,那么执行顺序是一样的。

3　变迁的触发

3.1　状态类

同 RBTPN 模型类似,在 ERBTPN 模型的状态类 C = (M, D)中,触发域 D 表示在标识 M 下,所有使能的变迁的触发时间间隔。触发域 D 可以表示如下[7]:

$$\begin{cases}\alpha_i \leqslant \tau(t_i) \leqslant \beta_i, & \forall t_i \in enabled(C)\\ \tau(t_i)-\tau(t_j) \leqslant \gamma_{i,j}, & \forall t_i,t_j \in enabled(C)\end{cases}$$

其中,$\tau(t_i)$ 是变迁 t_i 相对于状态类 $C=(M,D)$ 的触发时间。

ERBTPN 的初始状态类记为 $C_0=(M_0,D_0)$,M_0 是初始的标识,D_0 定义如下:

$$\begin{cases}\alpha_i^s \leqslant \tau(t_i) \leqslant \beta_i^s\\ \tau(t_i)-\tau(t_j) \leqslant \beta_i^s-\alpha_i^s\\ \forall t_i,t_j \in enabled(C_0)\end{cases}$$

仍然仿照文献[7],定义以下变量,以便得到触发域 D 的正规形式。

α_i^*:变量 $\tau(t_i)$ 最小的可能的值,β_i^*:变量 $\tau(t_i)$ 最大的可能的值,$\gamma_{i,j}$:差值 $\tau(t_i)-\tau(t_j)$ 最大的可能的值。

3.2　变迁的触发条件

在状态类 C 下,令 $TP=\min\{\beta(t_i)\mid t_i \in working(C)\}$,表示状态类 C 下所有运行变迁的最大触发时间的最小值。

定义 6　如果一个变迁 t_i 是运行变迁,其要成为可以触发的变迁,那么必须满足以下条件:

1) 变迁 t_i 的运行时间位于动态触发间隔内:

$\alpha_i \leqslant \tau(t_i) \leqslant \beta_i$

2) 变迁 t_i 的触发时间可以小于其他运行变迁的最大触发时间的最小值:

$\tau(t_i) \leqslant TP$

3) 变迁 t_i 的执行速率 $r(t_i,M)>0$。所有可以触发的变迁的集合记为 $Fire(C)$。如果属于 $Fire(C)$ 的某个变迁 t_i 触发了,那么还必须满足条件:

$\tau(t_i)-\tau(t_j) \leqslant 0\ ;\ \forall t_j \in Fire(C)$

4　可达图的建立

利用可达图,可以获得时间 Petri 网的各种性质。因此,下面着重讨论 ERBTPN 模型可达图的构造方法。

令 $UP=\min\{\beta_i^* \mid t \in Fire(C)\}$。

假定变迁 t_* 从状态类 $C=(M,D)$ 在时间间隔 (α_i^*,UP) 内触发,得到新的状态类 $C'=(M',D')$。新的状态类 C' 可以用下式计算:

$\forall p \in P, M'(p)=M(p)-I(p,t)+O(p,t_i)$

为了计算新的触发域 D',首先引入下面两个计算规则。

4.1　计算规则

由于时间基准的改变,使能变迁 $t_i \in enabled(C')$ 的触发域必然发生改变。

在变迁 t_0 触发后,在状态类 C 下使能的两类变迁具有不同的计算规则。

1) 对于任何在状态类 C 中运行,在 C' 中使能的变迁 t_i,触发时间应该满足下列限制:

$$\tau'(t_i)=[\tau(t_i)-\tau(t_0)]\times r(t_i,C)/r(t_i,C')$$

式中 $\forall t_i \in working(C) \cap enabled(C')$。

2) 对于任何在状态类 C 中就绪,而在 C' 中使能的变迁 t_k,触发时间应该保持不变:

$\tau'(t_k)=\tau(t_k)$ 式中 $\forall t_k \in halting(C) \cap enabled(C')$。

4.2　触发域的计算

类似于文献[7],D' 可以按照以下三个步骤计算

(1) 由于时间基准的改变,依然使能的变迁的触发域将被改变,即:

$D'=$

$$\begin{cases}\max(0,\alpha_k^*-\gamma_{0,k}^*+\alpha_0^*) \leqslant \dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k) \leqslant \min(\beta_k^*,\gamma_{k,0}^*+UP)\\ \max(0,\alpha_i^*-UP,-\gamma_{0,i}^*) \leqslant \dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i) \leqslant \min(\beta_i^*-\alpha_0^*,\gamma_{i,0}^*)\\ \dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i)-\dfrac{r(t_j,C')}{r(t_j,C)}\times\tau'(t_j) \leqslant \min(\beta_i^*-\alpha_i^*,\gamma_{i,j}^*)\\ \dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k)-\dfrac{r(t_l,C')}{r(t_l,C)}\times\tau'(t_l) \leqslant \min(\beta_k^*-\alpha_l^*,\gamma_{k,l}^*)\\ \dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i)-\dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k) \leqslant \min(\beta_i^*-\alpha_k^*-\alpha_0^*,\gamma_{i,k}^*-\alpha_0^*)\\ \dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k)-\dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i) \leqslant \min(\beta_k^*-\alpha_i^*+UP,\gamma_{k,i}^*+UP)\end{cases}$$

式中,$\forall t_i,t_j \in working(C) \cap enabled(C')$,$\forall t_k,t_l \in halting(C) \cap enabled(C')$。

(2) 一些变迁由于变迁 t_0 的触发成为非使能的变迁,从 D 中移走和这些变迁有关系的不等式,并对 D' 作一下改变:

$D'=$

$$\begin{cases}\max(0,\alpha_i^*-UP,\alpha_e^*-\gamma_{e,i}^*-UP) \leqslant \dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i) \leqslant \min(\beta_i-\alpha_0^*,\beta_e+\gamma_{i,e}^*-\alpha_0^*)\\ \max(0,\alpha_k^*,\alpha_e^*-\gamma_{e,k}^*) \leqslant \dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k) \leqslant \min(\beta_k,\beta_e+\gamma_{k,e}^*)\\ \dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i)-\dfrac{r(t_j,C')}{r(t_j,C)}\times\tau'(t_j) \leqslant \min(\gamma_{i,e}^*+\gamma_{e,j}^*,\gamma_{i,j}^*)\\ \dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k)-\dfrac{r(t_l,C')}{r(t_l,C)}\times\tau'(t_l) \leqslant \min(\gamma_{k,e}^*+\gamma_{e,l}^*,\gamma_{k,l}^*)\\ \dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i)-\dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k) \leqslant \min(\gamma_{i,e}^*+\gamma_{e,k}^*-\alpha_0^*,\gamma_{i,k}^*-\alpha_0^*)\\ \dfrac{r(t_k,C')}{r(t_k,C)}\times\tau'(t_k)-\dfrac{r(t_i,C')}{r(t_i,C)}\times\tau'(t_i) \leqslant \min(\gamma_{k,e}^*+\gamma_{e,i}^*+UP,\gamma_{k,i}^*+UP)\end{cases}$$

式中 $\forall t_i$, $t_j \in working(C) \cap enabled(C')$, $\forall t_e \in enabled(C) \cap disabled(C')$, $\forall t_k, t_l \in readyng(C) \cap enabled(C')$。

(3) 在 D' 中插入新使能的变迁的静态时间函数。即:

$\forall t_g \in disabled(C) \cap enabled(C')$

$$\alpha^s(t_g) \leqslant \frac{r(t_g,C')}{r(t_g,C)}\tau(t_g) \leqslant \beta^s(t_g)$$

结合以上三种情况,可以得到最终的 D'。

4.3 例子

为了说明问题,采用图2的简单例子,建立其可达图。需要指出的是下面的触发时刻都是相对于进入该状态类的时刻。

初识状态时,

$C_0 = (M_0, D_0), M_0 = (1,0,0,1,0,0,1,0,0)$,

$$D_0 = \begin{cases} 4 \leqslant \tau(t_1) \leqslant 6 \\ 4 \leqslant \tau(t_3) \leqslant 5 \\ 3 \leqslant \tau(t_5) \leqslant 6 \end{cases}$$

变迁 t_1 在(4,6)时刻触发后,得:

$C_1 = (M_1, D_1), M_1 = (0,1,0,1,0,0,1,0,0)$,

$$D_1 = \begin{cases} 1 \leqslant \tau(t_2) \leqslant 2 \\ 8 \leqslant \tau(t_3) \leqslant 10 \\ 6 \leqslant \tau(t_5) \leqslant 12 \\ \tau(t_3) - \tau(t_5) \leqslant 4 \\ \tau(t_5) - \tau(t_3) \leqslant 4 \end{cases}$$

变迁 t_2 在(1,2)时刻触发后,得:

$C_2 = (M_2, D_2), M_2 = (0,0,1,1,0,0,1,0,0)$,

$$D_1 = \begin{cases} 6 \leqslant \tau(t_3) \leqslant 9 \\ 4 \leqslant \tau(t_5) \leqslant 11 \\ \tau(t_3) - \tau(t_5) \leqslant 5 \\ \tau(t_5) - \tau(t_3) \leqslant 5 \end{cases}$$

此时可以分为两种情况。

1) 变迁 t_3 触发。变迁 t_3 在(6,9)时刻触发后,得:

$C_3 = (M_3, D_3), M_3 = (0,0,1,0,1,0,1,0,0)$,

$D_3 = \begin{cases} 0 \leqslant \tau(t_5) \leqslant 2.5 \\ 1 \leqslant \tau(t_4) \leqslant 2 \end{cases}$,此时,又可以分为以下两种情况。

(1) 变迁 t_4 触发:

变迁 t_4 在(1,2)时刻触发后,得 $C_4 = (M_4, D_4), M_4 = (0,0,1,0,0,1,1,0,0), D_4 = \{0 \leqslant \tau(t_5) \leqslant 1.5$;

变迁 t_5 在(0,1.5)时刻触发后,得 $C_5 = (M_5, D_5), M_5 = (0,0,1,0,0,1,0,1,0), D_5 = \{1 \leqslant \tau(t_6) \leqslant 2$;

变迁 t_6 在(1,2)时刻触发后,得 $C_6 = (M_6, D_6), M_6 = (0,0,1,0,0,1,0,0,1), D_6 = \varnothing$。

(2) 变迁 t_5 触发:

变迁 t_5 在(0,2)时刻触发后,得 $C_7 = (M_7, D_7), M_7 = (0,0,1,0,1,0,0,1,0), D_7 = \begin{cases} 0 \leqslant \tau(t_4) \leqslant 2 \\ 1 \leqslant \tau(t_6) \leqslant 2 \end{cases}$;

此时,再分为两种情况:

① 变迁 t_4 在(0,2)时刻触发后,得 $C_8 = (M_8, D_8), M_4 = (0,0,1,0,0,1,0,1,0), D_8 = 0 \leqslant \tau(t_6) \leqslant 2$;

变迁 t_6 在(0,2)时刻触发后,得 $C_9 = (M_9, D_9), M_9 = (0,0,1,0,0,1,0,0,1), D_9 = \varnothing$;

② 变迁 t_6 在(1,2)时刻触发后,得 $C_{10} = (M_{10}, D_{10}), M_{10} = (0,0,1,0,1,0,0,0,1), D_{10} = \{0 \leqslant \tau(t_4) \leqslant 1$;

变迁 t_4 在(0,1)时刻触发后,得 $C_{11} = (M_{11}, D_{11}), M_{11} = (0,0,1,0,0,1,0,0,1), D_{11} = \varnothing$。

通过比较,可知 $C_6 = C_9 = C_{11}$。

2) 变迁 t_5 触发。变迁 t_5 在(4,9)时刻触发后,得:

$C_{12} = (M_{12}, D_{12}), M_{12} = (0,0,1,1,0,0,0,1,0)$,

$D_3 = \begin{cases} 0 \leqslant \tau(t_3) \leqslant 2.5 \\ 1 \leqslant \tau(t_6) \leqslant 2 \end{cases}$

此时,同(1),可计算变迁的触发顺序。

因此,可以得到图2的可达图,见图3。

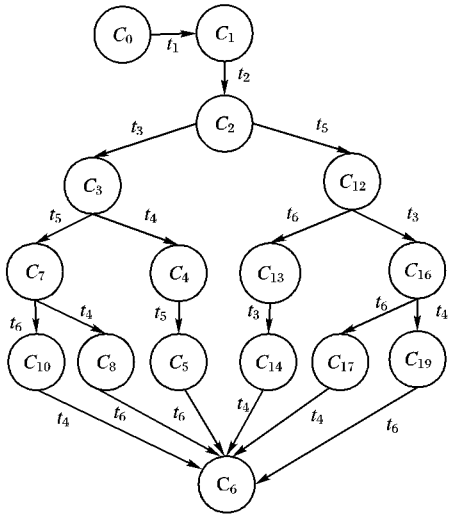

图3 图2的可达图

通过建立可达图,可以获得模型的各种性质。比如,在状态类 C_2 下,在(6,9)时刻调度变迁 t_3,可以到达状态类 C_3,而在(4,9)时刻调度变迁 t_5,可以到达状态类 C_{12}。

5 结语

将RBTPN模型进行扩展,得到eRBTPN模型后,不仅保持了原有的建模分布式嵌入式系统实时调度的特点,而且可以在单处理器上,实现固定优先级可抢先式调度和同一优先级的时间轮转调度。通过建立模型的可达图,可以进一步分析各个调度序列的性质。

参考文献:

[1] EDWARDS S, LAVAGNO L, LEE E A, *et al*. Design of embedded systems: Formal models, validation, and synthesis[J]. Proceedings of the IEEE, 1997, 85(3): 366-390.

[2] ESSER R. An object oriented Petri net approach to embedded system design[D]. Swiss: Federal Institute of Technology Zurich, 1996.

[3] CORTES L A. Modeling and formal verification of embedded systems based on a Petri net representation[J]. Journal of Systems Architecture, 2003, 49(12): 571-598.

[4] 张海涛, 艾云峰. 基于Petri网的分布式实时嵌入式系统调度的建模[J]. 计算机工程, 2006, 32(18): 6-9.

[5] 张海涛, 艾云峰. 基于Petri网的分布式实时嵌入式系统的调度分析[J]. 吉林大学学报: 工学版, 2007, 37(3): 616-620.

[6] OLIVER O H. A T-time Petri net extension for real-time task scheduling modeling[J]. JESA Modeling of Reactive Systems, 2002, 36(7): 973-986.

[7] BERTHOMIEU B, DIAZ M. Modeling and verification of time dependent systems using time Petri nets[J]. IEEE Transaction on Software Engineering, 1993, 17(3): 259-273.