基于蚁群算法的随机 Petri 网最优路径序列寻找

黄光球，何 星，苏海洋
(西安建筑科技大学管理学院，西安 710055)

摘 要：根据蚁群算法对 SPN 进行了一定的扩展，**为 SPN 网络中的变迁增加了过滤和保留信息功能，为库所增加了过滤信息的功能，得出了一种带有记忆性的连续时间随机 Petri 网(MESPN)**。当 MESPN 运行时，**利用充足量的托肯在网络中行走并且在行走过程中留下信息素来调整托肯路径的选择，使大量蚂蚁的行走路线不断逼近 SPN 网中时间延迟更短的变迁序列，最终在最短变迁序列上形成清晰的蚁路，从而在一定程度上解决了复杂 SPN 网的最优路径寻找问题。该算法充分考虑了每个变迁真正实施时间的概率特性，可以计算任意网型的变迁延迟时间概率分布**。仿真结果表明，托肯可以有效地在最短延时路径上形成蚁路并且能够求得从初始库所到网络中任意库所的最短路径。

关键词：蚁群算法；随机 Petri 网；路径序列；优化
中图分类号：TP301.5 **文献标识码：**A **文章编号：**1004-731X (2008) 17-4555-05

Optimum Route Sequence Search in SPN Based on Ant Colony Algorithm

HUANG Guang-qiu, HE Xing, SU Hai-yang
(School of Management, Xi'an University of Architecture and Technology, Xi'an 710055, China)

Abstract: Based on the ant colony optimization algorithm and expansion of SPN, *a memory extended and continuously timed SPN (MESPN), whose transitions could filter and save information and places could filter information, was proposed. When MESPN is running, enough ants (tokens) walk and leave odor in MESPN so that route selections of tokens can be adjusted, in this way it makes lots of ant walk routes approach to transition sequences with less delay. At last a clear ant walk route can be found on the transition sequence with least delay, and the route search problem of complex SPN is solved to a certain extend. The algorithm gives full consideration on the probability characteristics of each transition's real firing time, can determine the probability distribution that each transition's delay time obeys for any type of SPN.* The result of simulation shows that the ant walk route is found along the least delay route effectively by tokens, and the shortest route from initial places to every place of MESPN can be gotten.

Key words: ant colony algorithm; stochastic Petri net; route sequence; optimization

引 言

在 Petri 网中最初引入的时间概念是将一个变迁与一个固定的时间延迟相关联，它表示系统中某个变迁的发生需要一定的时间。后来的赋时 Petri 网(TPN)发展为将每个变迁与一个时间范围相关联，即每个变迁实施的延迟时间有一个最小值和一个最大值，变迁可实施后只能在这个时间段内实施。随后，研究方向开始向随机延迟方面展开，将一个变迁的可实施到真正实施这段时间与一个符合一定分布的随机延迟相关联，这种类型的 Petri 网叫做随机 Petri 网(SPN)[1]。把变迁的延时与随机指数分布相联系起来的思想是由 Molloy，Florin 和 Natkin 等人独立提出来的。已经证明指数分布是满足马尔可夫特性的连续随机变量的唯一分布函数，本文用到的 SPN 指的是随机时间延迟服从指数分布的连续时间 SPN。

文[2]提出了根据每个选择点的择路策略以及变迁更新实施度的方法来控制托肯搜索固定时间延迟 Petri 网中的最短路径，但是该方法在 SPN 的使用中往往表现出较低的搜索效率。为此，本文提出了一种更有效率的搜索方法：放大局部寻优过程中信息素浓度低、但有搜索潜力的路线上的信息素。当一个托肯找到了另一条路径，它的总延迟与当前信息素较浓的蚁路上总的延迟相同或更小，那么就主动在该路径的所有变迁上一次性添加更浓的信息素引导托肯在这条路径上集中搜索。如果说文[2]使用的是过程控制的话，本文将同时使用过程控制和结果反馈控制，用两种控制方式来引导托肯的搜索路线，使搜索逐步向全局最优逼近，最后到达全局最优。

1 模型的定义

根据具体的应用需求和算法需要，本文对基本 SPN 进行了一定的扩展，为网络中的变迁增加了保留信息素的功能和过滤信息功能，为库所也增加了过滤信息的功能，所以这里定义的 Petri 网为一种带有记忆性的连续时间随机 Petri 网(Memory Extended Stochastic Petri Net，MESPN)。在我们定义 MESPN 前首先对指数分布的时间延迟进行解释。

收稿日期：2007-04-16 **修回日期：**2007-07-11
基金项目：高等学校博士点学科专项科研基金课题(20070703009)，陕西自然科学基金项目(2007E$_2$17)，陕西省教育厅自然科学专项基金项目(07JK076).
作者简介：黄光球(1964-)，男，湖南桃源人，博士，教授，博导，研究方向为复杂系统建模、分析与控制，系统工程。

假设 $\lambda=\{\lambda_1, \lambda_2, \ldots, \lambda_m\}$是变迁平均实施速率集合。在连续时间随机 Petri 网中，一个变迁从可实施到实施需要延时，即一个变迁 $t(1 \le t \le m)$从可实施的时刻到它实施时刻之间的时间可看成是一个连续随机变量 $x_t(x_t>0)$，且服从一个分布函数 $F_t=P\{x_t \le x\}$。在 Molloy 提出的连续时间 SPN 中，相关于每个变迁延迟时间分布函数定义为一个指数分布函数：

$$\forall t \in T, F_t = 1-e^{-\lambda_t x} \tag{1}$$

其中实参数 λ_t 是变迁 t 的平均实施速率，变量 $x>0$。平均实施速率的倒数 $1/\lambda_t$ 称为变迁 t 的时间延迟的数学期望，即平均时间延迟。

定义 1 $Token=(r_1, r_2, \ldots, r_v; d_1, d_2, \ldots, d_v; F)$为托肯的数据结构，其中 r_j 的值为一个变迁索引号，$j=1,2,...,v$，v 为托肯走过的变迁数；$r_1, r_2, \ldots, r_v$ 序列标识出通过变迁的前后顺序；d_j 是托肯通过变迁 r_j 时所用的实际时间延迟，j 的意义与 r_j 中 j 的意义相同；F 为一个时间值，该值表示本托肯从 r_1 标识的初始库所到达当前库所 r_v 所用的时间，即

$$F=\sum_{i=1}^{v} d_i \tag{2}$$

定义 2 设 $MESPN=\{P, T, Fl, W, K\}$，其中，$P=\{p_1, p_2, \ldots, p_n\}$是库所的有限集合，$n$ 为库所的个数，且 $n>0$；$p_i = Token_i=(r_1, r_2, \ldots, r_v, d_1, d_2, d_3, \ldots, d_v, F_i)(i=1,2,\ldots,n)$，$v$ 为当前发现由初始库所到达本库最短路径的变迁数；d_j 是当前通过它的所有托肯中 F 函数值最小的那个托肯的时间序列表；F_i 为一个时间值，该值来自于某个托肯 j 的 F 函数值，表示该托肯从初始库所到达本库所用的时间；$T=\{t_1, t_2, \ldots, t_m\}$是变迁的有限集合，$m$ 表示变迁的个数，且 $m>0$；$P \bigcap T=\Phi$；$t_i=(\tau_i, \lambda_i)(i=1,2, \ldots, m)$，$\tau_i \in R$ 为变迁 t_i 上的信息素，是变迁实施时留下的，也是托肯选择路径的主要依据；$\lambda=\{\lambda_1, \lambda_2, \ldots, \lambda_m\}$是变迁平均实施速率集合，$\lambda_i \in \lambda$，为变迁 t_i 的平均实施速率，$1/\lambda_i$ 为变迁 t_i 的平均时间延迟；$Fl \subseteq P*T \bigcup T*P$(*为笛卡儿积)；$\mathrm{dom}(Fl) \cup \mathrm{cod}(Fl)=P \cup T$，其中 $\mathrm{dom}(Fl)=\{x \mid \exists y:(x,y) \in Fl\}$，$\mathrm{cod}(Fl)=\{y \mid \exists x:(x,y) \in Fl\}$即它们分别是 Fl 的定义域和值域；W：$Fl \to N$ 称为 N 上的权函数，对 $(x,y) \in Fl$，$W(x,y)=W((x,y))$称为 (x,y) 上的权；$K=\{k_1,k_2,\ldots,k_n\}$定义为 $P \to N \cup \{\infty\}$，即 P 上的容量函数。

2 模型与算法描述

2.1 MESPN 模型

本文依据蚁群算法的路径寻找原理[3]，使用自定义的托肯在网络上行走来寻找最优解。在托肯的行走过程中，托肯一边记录所激发过的变迁序列和实施时间，一边给变迁上留下信息素，以引导后来的托肯选择本变迁。同一个库所的输出变迁之间互相竞争库所中的托肯，那些延时少的变迁在竞争中体现出了较强的优势，所以能够引导蚁路在延时少的路径上形成。托肯是用来记录问题的解集合，托肯中记录的变迁序列便是问题的解集合，而算法并非是寻找问题解而是寻找问题的最优解，所以托肯中记录的内容只提供本算法实现的前提数据。托肯实施时在变迁上留下了信息素。本算法用的选路策略是：托肯选择信息素更浓的路径。这与文献[4]所用方法相反，文献[4]是在企图避开已经走过的路径，在没有涉及的路径上寻找最优解，所以它是以遍历全网为主要搜索思想。但这样的搜索使搜索没有引导性，呈现盲目的遍历特点，从而在许多无用的路径上耗费了托肯资源，降低了搜索效率。而本算法以引导为主要的搜索思想，这与基本蚁群算法思想相同，蚁群算法的基本理论已经证明这是一种更有效率的搜索方式[5,6]。

因为 SPN 中各个变迁的延时不确定，这就增加了搜索难度。在搜索技术方面，本算法尝试一种新技术来搜索全局最优。当搜索进行到一定阶段，发现搜索的值没有太大改进，那么我们就从搜索阶段进入解的调整阶段，这时我们认为搜索已经进入了某个局部极小值但并不确定是否是全局最优，即寻到某条路径是当前的一个分支最小而不能够确定是否是由初始某库所出发到终止某库所或目标某库所的全局最小。此时，我们使托肯以更大些的概率跳出极小分支，搜索其他分支并且如果某个托肯寻找到了到达某个库所更短的路径或同样长短的路径，那么它立即以较大幅度增加整条路线上的信息素浓度，使该条路径各个变迁上的信息素浓度有更大优势来竞争托肯来对此路径进行重点搜索。

这里我们设库所 p_i 的目标函数为

$$F_i^{new}=\min\{F_{token}^1, F_{token}^2, ..., F_{token}^a, F_i^{old}\}$$

下次计算时，令

$$F_i^{old}=F_i^{new},\ i=1,2,\ldots,n$$

式中，F_{token}^b，$b=1,2,\ldots,a$ 表示某时刻同时存在于 p_i 中的托肯的 F 函数值，a 为托肯数；F_i^{old} 为 p_i 接受托肯前的 F 函数值，初始为 0；F_i^{new} 记录当前发现的，由初始某库所到达本库所 p_i 最短的时间值。

这里用图 1 来帮助说明 MESPN 的特殊性。

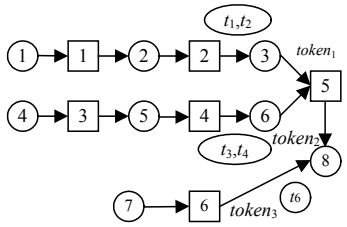

图 1 MESPN 示例图

我们设 $M_0=(1,0,0,1,0,0,1,0)$，设 p_1 处的托肯为 $token_1$，p_4 处的托肯为 $token_2$，p_7 中的托肯为 $token_3$。如果 t_1 实施，在 t_1 延时后，$token_1$ 将 t_1 的索引号和实际延时添入自己的序列表并且用式(2)计算 F 函数。当 t_2 被激发，$token_1$ 再将 t_2 的索引号和实际延时添入自己的序列表，再用式(2)计算 F 函数值。t_3，t_4 的变化与 t_1，t_2 相同。当 p_3 和 p_6 出现托肯时，t_5 变成了使能变迁。当 t_5 被激发，它比较来自于 t_2 和 t_4 的 F

函数值，将 F 函数值大的托肯作为当前托肯，这是因为 t_5 必须等到两个库所的托肯都到达，才能比较。假如 $token_1$ 的 F 函数值大于 $token_2$ 的 F 函数值，那么 t_5 将 $token_1$ 作为当前托肯。当 t_5 的延时完毕后当前托肯再将 t_5 的索引和实际耗时添入自己的序列表，同时计算 F 函数值，输出当前托肯。当 $token_1$ 和 $token_3$ 以任意次序到达 p_8 时，p_8 保留 F 函数值较小的托肯作为当前托肯，输出时输出当前托肯。

在这个过程中有几个地方需要讨论。t_i 对应的延时 $1/\lambda_i(i=1,2,\ldots,6)$都是服从如式(1)所示的指数分布。当大量托肯由 p_1、p_4 和 p_7 进入 MESPN 时，由于变迁延迟，t_1 的输出间隔服从的分布密度函数为下式分布。

$$f_{t_1}(x)=\lambda_1 e^{-\lambda_1 x}$$

当 t_1 走出的托肯进入了 t_2，又因为 t_2 的时间延迟服从参数为 λ_2 的指数分布，所以 t_2 的输出服从的分布为：

$$f(x)=\begin{cases}\dfrac{\lambda_1\lambda_2}{\lambda_2-\lambda_1}(e^{-\lambda_1 x}-e^{-\lambda_2 x}) & \lambda_1\neq\lambda_2, x>0\\ \lambda_1\lambda_2 x e^{-\lambda_2 x} & \lambda_1=\lambda_2, x>0\\ 0 & x\le 0\end{cases} \tag{3}$$

如果变迁序列存在多个变迁时，那么公式(3)变为式(4)：

$$f(x)=\begin{cases}(-1)^{n-1}\lambda_1,\ldots,\lambda_n\displaystyle\sum_{i=1}^{n}\frac{e^{-\lambda_i x}}{\prod_{j=1\ j\neq i}^{n}(\lambda_i-\lambda_j)},\\ \qquad \lambda_k(k=1,2,\ldots,n)\text{两两不相等, 且}x>0\\ \dfrac{\lambda^r}{(r-1)!}x^{r-1}e^{-\lambda x},\\ \qquad \lambda_k(k=1,2,\ldots,n)\ \text{相等且为}\lambda, x>0\\ 0, \qquad x\le 0\end{cases} \tag{4}$$

如果 $\lambda_k(k=1,2,\ldots,n)$中有相等的值，也有不相等的值，那么可以用卷积公式[7]计算。计算时可以将参数相同的随机变量移到后边。即在 λ 集合中，$\lambda_k(k=1,2,\ldots,q)(0<q<n))$两两不相等，$\lambda_k(k=q+1,2,\ldots,n)$都相等，那么式(4)变为式(5).

t_4 输出托肯的时间间隔密度函数与 t_2 相同，类似于式(3)。因 $x\le 0$ 无意义，以下我们只考虑 $x>0$ 情况。

对于 t_5 来说，因为我们知道来自于两边路径托肯的 F 函数值的分布，所以当大量托肯进入 t_5 时，它们的延时长短是由其数学期望来决定的。

$$f(z)=\begin{cases}\displaystyle\int_{-\infty}^{+\infty}((-1)^{q-1}\lambda_1,\ldots,\lambda_n\sum_{i=1}^{q}\frac{e^{-\lambda_i x}}{\prod_{j=1\ j\neq i}^{q}(\lambda_i-\lambda_j)})*\\ \qquad (\dfrac{\lambda^{n-q}}{(n-q-1)!}(z-x)^{n-q-1}e^{-\lambda(z-x)})dx \qquad x>0\\ 0 \quad x\le 0\end{cases} \tag{5}$$

来自于 t_2 的 F 函数值的均值为：

$$E(x)=\int_0^{+\infty}x\frac{\lambda_1\lambda_2}{\lambda_2-\lambda_1}(e^{-\lambda_1 x}-e^{-\lambda_2 x})dx=\frac{\lambda_1+\lambda_2}{\lambda_1\lambda_2} \tag{6}$$

如果变迁序列存在多个变迁的话，那么式(6)变为：

$$E(x)=\frac{1}{\lambda_1}+\frac{1}{\lambda_2}+\ldots+\frac{1}{\lambda_n} \tag{7}$$

对于 t_4 也用同样方法得到 F 函数值的平均值。t_5 将留下该值大的托肯作为当前托肯并添入 t_5 的索引号和实际延时，计算 F 函数值，最后输出给 p_8。

在本例中 p_8 同时收到两个托肯，一个是来自于 t_2 或 t_4，一个来自于 t_7。它比较两个托肯的 F 函数值，将较小值托肯作为当前托肯，把它的变迁索引序列和 F 函数值记录在自己的数据结构中。当大量托肯运行且比较时，可以用式(7)计算出的数学期望值来比较所得到的结果。

2.2 路径选择算法

当 Petri 网简单时可以用计算的方法来得到 MESPN 的主干路径，但对于复杂 Petri 网，网络节点前后关系模糊，使计算效率不能令人满意[8,9]。

本文将蚁群算法引入，试图通过局部的协作来在全局中产生最优解。在系统的初始化阶段，各个变迁上的信息素浓度相等且为一自然数 z，即 $\tau_i=z$，$i=1,2,\ldots,m$。托肯的个数为 M_0 中所有不为“0”的数字之和 c 的 x 倍，即 $c*x$，$x\in N$。我们视 c 为一个分组中蚂蚁的个数，x 为蚂蚁群中蚂蚁的分组数。然后使蚂蚁由 M_0 所表示的库所，按照 M_0 所标识的各个库所中托肯的比例进入 MESPN。进入 MESPN 后在确定的路径上托肯以 Petri 网的激发规则移动。在有选择路径上托肯以本算法进行路径选择。

2.2.1 变迁 t_h 实施时库所的变化

对于 $\forall p_i\in {}^*t_h$，$i=1,2,\ldots,x_{in}$，x_{in} 为 ${}^*t_h$ 中元素个数。像基本 Petri 网一样提取 $W(p_i,t_h)$个托肯送入变迁，即在 t_h 中生成 $W(p_i,t_h)$个同样数据内容的托肯后，删除 p_i 中的 $W(p_i,t_h)$个托肯。

对于 $\forall p_i\in t_h^*$，$i=1,2,\ldots,x_{out}$，x_{out} 为 t_h^* 中元素个数。初始状态 p_j^{old} $(j=1,2,\ldots,n)$的序列表为空，它的 F 函数值 $F_{p_j^{old}}=0$。p_i 可能同时收到来自于多个变迁的托肯。设某一时刻 $\forall token_x\in p_i$，$x=1,2,\ldots,x_{end}$，x_{end} 为 p_x 中元素个数，F_{Token}^x 为该托肯的 F 函数值，则有

$$p_i^{new}=\begin{cases}p_i^{old} & \text{如果}F_{Token}^x>F_{p_i^{old}}\text{且}F_{p_i^{old}}\neq 0\\ Token_x & \text{如果}F_{Token}^x<=F_{p_i^{old}}\\ Token_x & \text{如果}F_{s_i^{old}}=0\end{cases}$$

在进行下一轮计算时，应令：

$$p_i^{old}=p_i^{new}$$

2.2.2 变迁 t_h 实施时托肯本身的变化情况

在初始状态，托肯中的列表为空，F 函数值为 0。在网络运行过程中逐渐填充列表和 F 函数值，当 t_h 激发后托肯要做一些计算。首先，每次托肯离开库所时需要按下式进行计算:

$$Token_x^{new}=p_i^{old},\ x=1,2,\ldots,x_{end}$$

然后，当托肯进入了变迁，它检查变迁 t_h 的索引号是否已经出现在已记录到的索引序列中了。如果在托肯的列表中找到了变迁 t_h 的索引号，那么就要删除序列中该变迁索引号之后的部分，在延时结束后确定本次实施的延时时间，然后

将其添入序列表中变迁 t_h 的索引号对应的延时位置，按照式(2)重新计算 F 函数值。如果变迁的索引没有在序列中出现过，那么就将该索引加入托肯的序列表，然后等到变迁实施后，将本次实施时间添入到序列表中，再用式(2)计算 F 函数的值。

当每次托肯离开变迁时需要进行如下计算：

$Token_{other}^{new} = Token_{x}^{old}$，如果 $F_{Token}^{x} > F_{Token}^{other}$，$other$= 1, 2, …, x_{end} 且 $other \neq x$

2.2.3 变迁 t_h 实施时变迁的变化情况

蚁群算法[5]中，蚂蚁是靠信息素的浓度对比来决定路径选择，在 MESPN 中，信息素确定方法是由下式确定。变迁 t_h 的信息素 τ_h，其初始值为 $\tau_h^{old} = z$ (z 为常数)，当延迟时间固定时，可用文[2]提出的公式修改信息素；当在 SPN 中延迟时间是服从一定分布的随机变量时，信息素的修改公式则变为：

$$\Delta\tau(t_h) = \lambda_h e^{-\lambda_h At_h^{-1}} * At_h^{-2} \tag{8}$$

$$\tau_h^{new} = \lambda_h e^{\left[-\lambda_h \frac{A*act_{new}(t_h)}{x-\rho^w \tau_h^{old}}\right]} * \frac{Aact_{new}(t_h)}{(x-\rho^w\tau_h^{old})^2} \tag{9}$$

在进行下一轮计算时，应令：

$$\tau_h^{old} = \tau_h^{new}$$

以上各式中，ρ 为每个时间单位(s)挥发后留下的信息素比例；w 为 Petri 网上次实施到本次实施所经过的时间长度；A 为常数，表示每只蚂蚁在做一次变迁时留给所走过变迁的信息素总量；变迁 t_h 的时间延迟服从以 λ_h 为参数的指数分布；由式(8)可以得到每次变迁实施而留下的信息素，由式(9)可以得到信息素更新后新的信息素浓度；τ_h^{old} 和 τ_h^{new} 分别是实施前、实施后的浓度。

公式(8)，(9)表明变迁 t_h 上的信息素调整后的浓度是相邻两次实施的间隔时间段内信息素挥发后剩下浓度与新加入的信息素浓度的和。这样对于那些最后一次实施的变迁来说它的信息素将不会进行调整。也就是说，网络运行时信息素以时间为单位进行调整。当网络运行结束，每个变迁上的信息素保持最后一次调整的结果。

为了避免某条固定路径因速度不一致而导致的托肯积累问题。本文引入实施度[1]概念。在 t_h 激发前我们需要决定本次激发的实施度，这里用 $act(t_h)$表示变迁 t_h 的实施度，设初始状态 $act_{old}(t_i)$＝1，i=1,2,…,m，$act_{old}(t_h)$为 t_h 上一次激发时的实施度，$act_{new}(t_h)$为本次将要使用的实施度，用 $M(p)$表示库所 p 的托肯数，p_{in} 和 p_{out} 分别表示输入，输出库所集合中的元素。在变迁 t_h 实施前需要依照文献[2]提出的方法决定本次实施度。这样对于无选择的路径来讲短延时变迁的实施度将有被提高的倾向，长延时变迁的实施度有被降低的倾向，当大量的托肯在该路径上运行一段时间后此路径上各个变迁的输入，输出速度将达到一个平衡值。因为此平衡值只由各个变迁延时决定所以它反映该无选择路径上各变迁的平均延迟信息。蚁路将以较高概率在平衡值高的路径上形成。

2.2.4 路径选择

当 SPN 是一个复杂网络时，散漫地遍历整个 Petri 网去寻找变迁序列虽然可能找到最优路径，但是效率是相当低的。在这里我们应用蚁群算法的原理，一方面集中托肯在很有发掘潜力的路径上寻找局部最优，一方面调整整体的搜索方向通过对比多个局部最优来得到全局最优。在系统运行中，当托肯在某库所时有多个输出路径可以选择的话，托肯以一定的概率来选择所要激发的变迁，其概率用各个变迁的延迟，当前的信息素浓度来决定。但是，变迁的延迟是一个随机变量，我们不能确定某一次的延迟时间，但是当多数托肯来到时我们可以用平均时间延迟来确定路径选择。

$$Q_{kij} = \begin{cases} \dfrac{\eta_{ij}^{\beta}(t_j)\,\tau_j^{\alpha}(\lambda_j)^{\gamma}}{\sum\limits_{s\in allowed_k} \eta_{is}^{\beta}(t_s)\,\tau_s^{\alpha}(\lambda_s)^{\gamma}}, & allowed_k \neq \varnothing \\ 0, \text{ 否则} \end{cases} \tag{10}$$

式中，$allowed_k$＝{t_1,t_2,…,t_a}表示托肯 k 下一步允许选择的变迁，a 表示本库所中的托肯允许选择的变迁数，该集合中的数据由基本 Petri 网变迁实施条件而决定；τ_j 为当前变迁 t_j 上的信息素的量，$\eta_{ij}(t_j)$为由 p_i 到 t_j 的引导因数。由式(10)可知，托肯 k 在库所 p_i 向变迁 t_j 的转移概率 Q_{kij} 与 $\eta_{ij}^{\beta}(t_j)\,\tau_j^{\alpha}(\lambda_j)^{\gamma}$ 成正比；α，β 和 γ 分别反映了托肯在运动过程中所积累的信息素，引导因数和路径可见度在托肯选择路径时的相对重要性。

2.2.5 反馈型信息素调整方法

因为 SPN 中变迁延时是随机变量，增加了搜索难度，所以我们提出新的搜索技术来协助搜索。我们取托肯走出 MESPN 时所带的 F 函数值作为反馈信息。当系统运行到一定阶段，各个输出库所出来托肯的序列表和 F 函数值已经趋于稳定，即 F 函数值的方差小于一个接受值时，如果此时收到一个托肯带有不同于其他托肯的序列表，并且 F 函数值小于当前发现路径的 F 函数值，那么，我们将给该新路线上涂上当前路线上大于平均浓度且小于最大浓度的信息素，使搜索向该方向进行，进一步寻找该路段的潜力。

3 仿真实现

为了体现新算法对时间延迟的灵敏度和对路径选择的准确，本文采用路径选择更复杂些的 MESPN 网形如图 2 所示。图 2 中变迁 1~16 的延时分别为 5、5、4、3、3、8、6、4、3、6、4、6、9、9、4、2。本文的参数根据文献[5]的建议设定参数为：信息素挥发系数 ρ＝0.6。库所 1，2，3 中初始托肯数各为 20，每次涂上的信息素总量 A＝40。在“或”变迁的路径选择公式中 α=0.5, β=0.5,γ=1,$\eta_{ij}(t_j)$=1, 初值 τ_j=10[10]，i=1,2,…,16，j=1,2,...,13。在本网中的指数分布的参数值相差不大，这是用来说明本网的灵敏度的，表示本算法可以分辨出变迁延迟之间的细微差别。

图 3、4 分别表示了在时刻 0、56 秒时仿真计算情况，图 3、4 中比图 2 多出的库所是用来控制实现实施度的库所，里边的托肯数显示出对应变迁可用的实施度，可用实施度为当前还可以接受的任务数，小于零时表示不能再接受任务。图 3、4 中，库所中的点为托肯。变迁中的点表示信息素浓度。可用实施度为当前还可以接受任务的实施度，小于零时表示不能再接受实施。本网的运行过程大致描述如下。

在第 0 秒时，变迁 1~16 的信息素均为 10，可用实施度分别为 3、3、3、3、3、3、3、3、3、3、3、3、1、1、3、3。在运行阶段，变迁 1，2，3 的启动时间相同，而变迁 1，2 的延迟比变迁 3 的延迟大 1 秒钟，这导致变迁 1，2 的信息素浓度相同且稍淡些，变迁 3 的信息素稍浓些。在库所 4，5，6 中来到的托肯也是没有先后顺序，所以变迁 4，5，6 的发生也是同时的。网络运行 56 秒后的情况如图 4 所示，图中明显看到了 4，5 的信息素比 6 的信息素要浓，这说明，在同时可以得到托肯的情况下，延时短的变迁更有可能得到托肯。变迁 4，5 和变迁 6 的延迟分别是 3 和 8 相差很大，但信息素相差不多，这是由变迁的信息素浓度变化公式决定的，公式按照比例减少且按照实施次数和实施度来增加信息素浓度。当变迁 5 一次增加很多信息素时它的减少量也增加；变迁 6 一次增加的信息素不多，但信息素挥发掉的也不多。所以在最后的结果中看起来好像两者一样。根据最后信息素浓度可知，到达库所 12 的最优序列为 3，5，11，15；到达库所 13 的最优序列为 3，5，11，16。

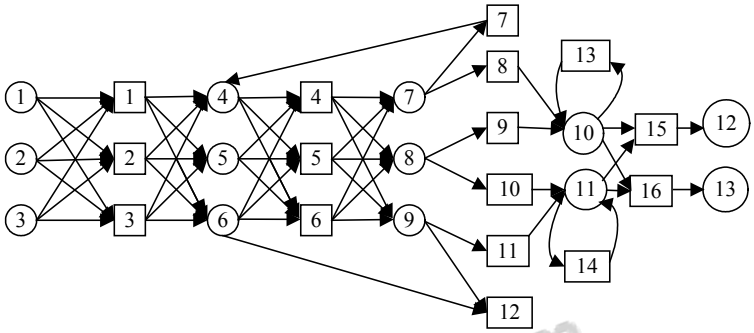

图 2　SPN 仿真用例图

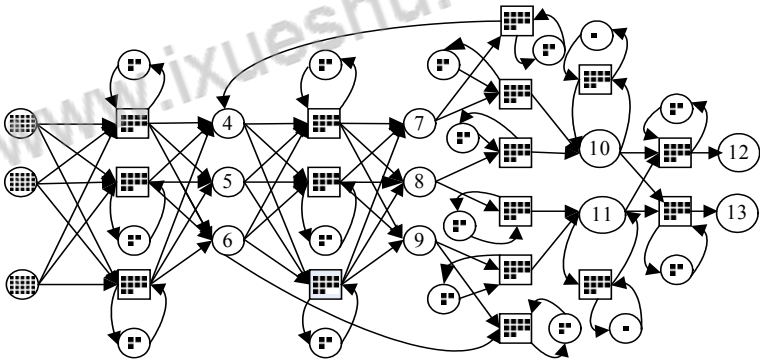

图 3　运行变迁 0 秒时库所可用实施度与变迁信息素情况

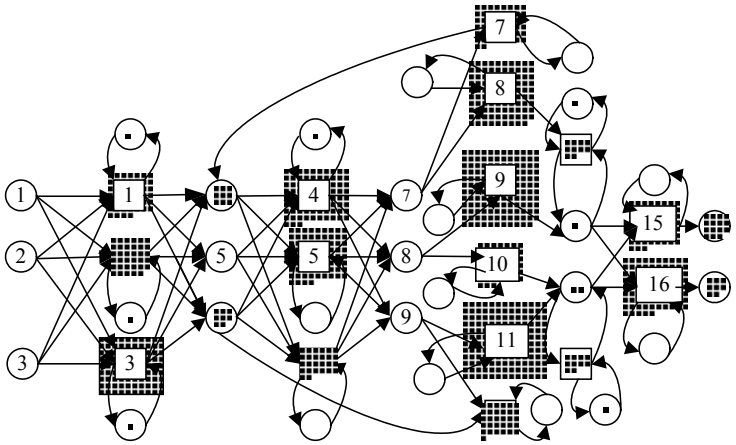

图 4　变迁 56 秒(结束)时库所可用实施度与变迁信息素情况

（下转第 4581 页）

为了验证仿真平台方案的可行性及仿真模型的正确性，对实际系统起竖工况的角位移进行采样，并与仿真结果比较。图 8 所示为起竖角度的仿真曲线和实装运行采样曲线，可以看出整个起竖过程仿真角度曲线与实际的运行过程采样曲线基本一致[10]，从而验证了仿真平台的可行性和有效性。

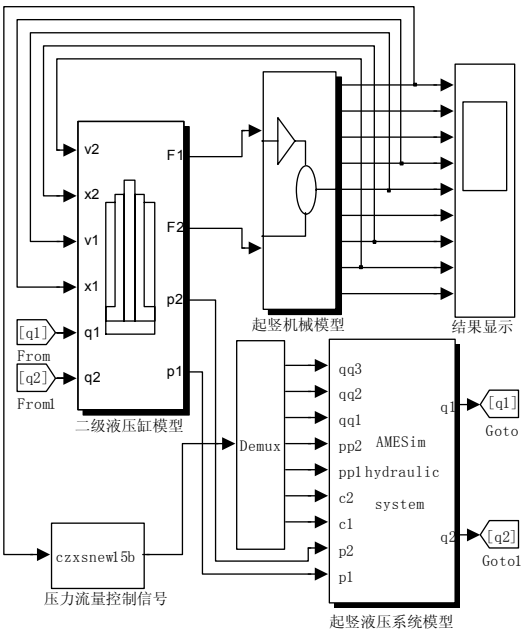

图 7　液压驱动大型起竖系统 Simulink 仿真模型

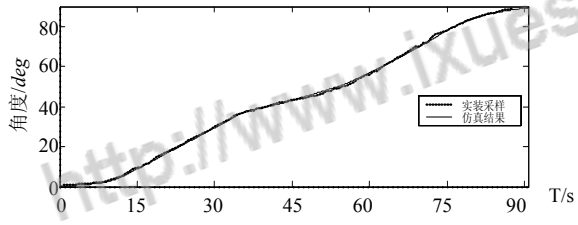

图 8　起竖角度仿真曲线与实际采样曲线比较

5　结论

本文构建了基于 Simulink 环境的机电液耦合系统的集成化建模与仿真平台，研究了软件间联合的组织内部协同仿真方法，并以仿真实例验证了该平台的有效性。仿真试验表明，基于 Simulink 集成化仿真平台能够有效支持机电液一体化系统虚拟样机开发，这种基于软件接口的多学科协同仿真建模方法可以综合液压系统专家、控制技术专家、计算与计算机技术等专家的优势，为机电液一体化系统的协同研发搭建一个支撑平台，能够高效、经济、可靠地处理机电液系统非线性建模和优化控制问题。

参考文献：

[1] 熊光楞，郭斌，陈晓波，等：协同仿真与虚拟样机技术[M]. 北京：清华大学出版社，2004.

[2] 万昌江，谭建荣，刘振宇. 基于语义的组件化样机建模技术研究[J]. 中国机械工程，2005, 16 (8): 142-144.

[3] Gianni F, Gian A M, Paolo R. Virtual Prototyping of Mechatronic System [J]. Annual Review in Control (S1367-5788), 2004, 28(2): 193-206.

[4] 潘双夏，刘静，冯培恩，高峰. 挖掘机器人虚拟样机的机电液一体化建模与仿真实现[J]. 中国工程机械学报，2003, 1(1): 49-50.

[5] 薛定宇，陈阳泉. 基于 MATLAB/Simulink 的系统仿真技术与应用[M]. 北京：清华大学出版社，2002.

[6] 郑建荣. ADAMS——虚拟样机技术入门与提高[M]. 北京：机械工业出版社，2001.

[7] 付永领,祁晓野. AMESim 系统建模和仿真——从入门到精通[M]. 北京：北京航空航天大学出版社，2006.

[8] 陈宏亮，李华聪. AMESim 与 MATLAB/Simulink 联合仿真接口技术应用研究[J]. 流体传动与控制，2006，14(1): 14-16.

[9] 黄先祥，高钦和，郭晓松. 大型装置起竖过程动力学建模研究[J]. 系统仿真学报, 2002, 14(3): 271-273.

[10] 马长林. 导弹起竖过程动力学仿真与控制策略研究[D]. 第二炮兵工程学院，2006.

(上接第 4559 页)

4　结论

本文是文[2]内容的一个延伸，它使用了文[2]的蚁群算法，但重点为对 SPN 的分析和在 SPN 中如何应用蚁群算法。与其他文献[3,4,8,9,11]的不同之处在于本文详细描述了在 SPN 中每个变迁真正实施时间的概率特性，也提出一种方法可以计算任意一种网型的变迁时间概率分布。在分析了时间特性后，我们针对 SPN 环境设计出新的数据结构来适应这样的环境最终计算出 SPN 的主干路径。

参考文献:

[1] 林闯. 随机 Petri 网和系统性能评价[M]. 第 2 版. 北京: 清华大学出版社, 2005: 22.

[2] 黄光球，苏海洋，刘冠. 基于蚁群算法的 Petri 网最优路径序列寻找[J]. 计算机应用, 2007, 27(4): 932-935.

[3] 乐晓波，李京京，唐贤瑛. 基于 Petri net 建模的资源调度的蚁群算法[J]. 计算机技术与发展, 2006, 16(1): 44-46.

[4] 邵志芳，刘重英，钱省三. 整合 Petri 网和蚁群优化算法用于柔性制造系统调度优化研究[J]. 计算机应用, 2006, 26(11): 2753-2764.

[5] 李士勇. 蚁群算法及其应用[M]. 哈尔滨: 哈尔滨工业大学出版社, 2004.

[6] Dorigo M, Maniezzo V, Colorni A. The Ant System Optimization by a Colony of Cooperation Agents [J]. IEEE Trans on Systems, Man and Cybernetics-Part B (S1083-4419), 1996, 26(1): 29-41.

[7] 赵彦晖. 概率论[M]. 陕西: 陕西科学技术出版社, 1999: 82.

[8] 李勇，曹广益，朱新坚. 一种基于单亲遗传算法的 Petri 网发射路径求解算法[J]. 系统仿真学报，2005, 17(1): 203-206. (LI Yong, CAO Guang-yi, ZHU Xin-jan. An Algorithm for Finding Firing Sequences of Petri Nets Based on Partheno-Genetic Algorithm [J]. Journal of System Simulation, 2005, 17(1): 203-206.)

[9] 郝东，蒋昌俊，林琳. 基于 Petri 网与 GA 算法的 FMS 调度优化[J]. 计算机学报, 2005, 28(2): 201-208.

[10] 黄永青，梁昌勇，张祥德. 基于均匀设计的蚁群算法参数设定[J]. 控制与决策，2006, 21(1): 93-96.

[11] 周卫东，杨加敏，贾磊，李歧强. 一种 Petri 网结合遗传算法的优化方法及应用[J]. 山东大学学报, 2005, 35(4): 59-63.