工作流模型死锁的 Petri 网分析*)

谭 玲 郑 栋 顾 庆 陈道蓄
(南京大学软件新技术国家重点实验室 南京 210093)

摘 要 Petri 网具有坚实的理论基础和易于使用的图形表示,是一种理想的建模和分析工具,因此在工作流的建模和分析方面具有广泛的应用。本文应用 Petri 网的理论对于工作流模型中的死锁进行分析,给出了解决死锁的基本算法,并分析了这些算法的优缺点。最后给出了一个实例:软件测试过程模型。

关键词 工作流,Petri 网,WF-net,死锁,可达树,可达图

The Analysis on Workflow Model Deadlock by Petri Nets

TAN Ling ZHENG Dong GU Qing CHEN Dao-Xu
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093)

Abstract There are a lot of modeling and analysis tools available for workflow, but Petri nets is more preferable because of its solid mathematical foundation and graphical nature. This paper applies Petri nets to analyze the workflow model, set forth several solutions and compare the merits and limitations among them, point out the common method on analysis and solution of deadlock. At last, we will introduce an example for deadlock analysis——the Software Test Workflow System.

Keywords Workflow, Petri-net, WF-net, Deadlock, Reachability-tree, Reachablility-graphic

1 引言

软件过程是开发有质量保证的软件系统的关键因素,因为其目标是管理用户需求并把用户需求转换为符合满足需求的软件产品。因此,软件过程意味着生产一个软件系统的活动集合,这些活动由有组织的人员依赖一系列工具执行。

工作流的概念起源于生产组织和办公自动化领域。它是对日常工作中具有固定程序的活动而提出的一个概念。从概念上讲,软件过程是工作流在软件领域应用的一个特例,具备工作流的特征。因此,我们借用工作流的理论和技术来研究软件过程,如建模和分析。

随着计算机网络技术和分布式数据库技术迅速发展,多机协同工作技术的日趋成熟,并且将向具有灵活性、自适应性、规模可扩充性以及互操作性等先进特性的方向发展。因此,对这些过程进行建模的复杂性也就相应提高,进行正确的过程定义也就越来越难。其中,死锁是过程定义中经常出现的问题。

一般来说,解决死锁问题有 3 种基本方法:

① 死锁的预防。通过设置某些限制条件来破坏产生死锁的必要条件,以防止死锁的发生。此方法易于实现,但限制条件严格,降低了资源的利用率和系统的并行性。

② 死锁的避免。该方法不需要事先采取措施,破坏产生死锁的必要条件,而是在系统动态执行过程中,根据当前的系统的具体情况确定是否执行某些活动,避免死锁的发生。该方法与方法①相比,可以获得较高的资源利用率和并行性,但其实现复杂,而且决策上往往要花费较大的代价,该方法所花的代价要大于它所得到的好处。

③ 死锁的检测与解除。这种方法预先不采取任何措施,允许系统运行过程中发生死锁。当系统发生死锁时,及时检测出死锁的发生,并精确地确定出与死锁有关的活动;然后,采取一定措施,解除死锁。该方法有可能使系统获得较好的资源利用率和并行性,但其实现难度较大,而且有可能检测到假死锁。

因此,近年来,对工作流模型分析方法的研究得到了普遍重视。用于工作流分析的工具也越来越多,使用 Petri 网来分析工作流模型也被越来越多的人接受。选择 Petri 网分析工作流模型有以下原因[1]:

① 形式化的语义。Petri 网描述的工作流过程有着清楚、精确的含义。

② 直观的图形表示。Petri 网是一个图形化的语言,易于学习,也易于与最终用户交流。

③ 表达能力强。Petri 网支持工作流建模所需的所有原语。

④ Petri 网有着坚实的数学基础。

本文将重点介绍工作流模型死锁的 Petri 网分析技术,并结合一个实例(软件测试过程模型)进行说明。

2 WF-net 的相关概念

2.1 Petri 网的概念

定义 1 Petri 网是一个六元组,$\Sigma = \{P, T, F, K, W, M0\}$,其中:

① $P \cap T = \emptyset$

② $P \cup T \neq \emptyset$

③ $F \subseteq P \times T \cup T \times P$

*)本文得到国家 863 项目支持,项目编号:2004AA112090。谭 玲 硕士研究生,研究方向:工作流、软件过程管理;郑 栋 硕士研究生,研究方向:过程建模、软件过程管理;顾 庆 博士,副教授,研究方向:分布式计算;陈道蓄 博士生导师,研究方向:分布式计算与并行处理。

P 是库所(Place)集合，T 是变迁(Transition)集合，F 是一有向弧集。(P,T,F)构成有向图。$K:F\to N^+\cup\{\infty\}$是位置容量函数。$W:F\to N^+$，是弧(arc)的权值函数，用 $W(x,y)$ 表示，默认值为 1。$M0$ 为Σ的初始标识，系统处于某种标识 M，也可以称为处于某种状态 M。并且用 Γ^- 表示前集，Γ^+ 表示后集。

定义 2 一个变迁 $t\in T$ 在 M 下是可执行的当且仅当

$\forall p\in P, W(p,t)\leqslant M(p)\leqslant K(p)-W(p,s)$

当 M 下有变迁 t 可以触发时，我们称 M 是可执行的。

2.2 WF-net 的概念

定义 3 一个 Petri 网是工作流网(WF-net)$PN=\{P,T,F,K,W,M_0\}$当且仅当[1]：

① 该 Petri 网有两个特殊的库所(Place)：i 和 o。i 为起始库所且 $\Gamma^-(i)=\phi$。o 为终止库所且 $\Gamma^+(o)=\varnothing$。

② 如果给这个 Petri 网加上一个变迁 t_0，使其连接 o 和 i (即 $\Gamma^-(t_0)=\{o\}$，$\Gamma^+(t_0)=\{i\}$)，那么所得到的 Petri 网是强连通的。

定义 4 工作流管理联盟定义了 4 种基本的工作流路由结构[4]：

① 顺序路由。表示两个任务之间存在时序依赖关系，在前一个任务完成之前，后一个任务不能执行。

② 并行路由。表示并行的两个或多个任务之间可以任意顺序执行。

③ 选择路由。表示在流程的某一点，依执行结果选择一条路径继续执行。

④ 循环路由。表示某一个任务可能需要执行多次。

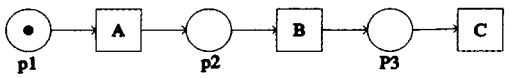

图1-1 顺序路由

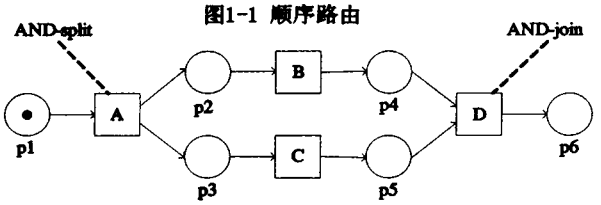

图1-2 并行路由

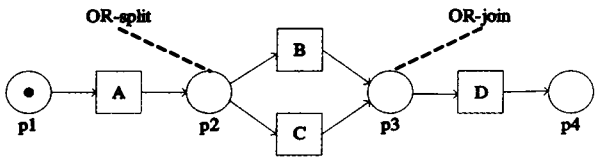

图1-3 选择路由

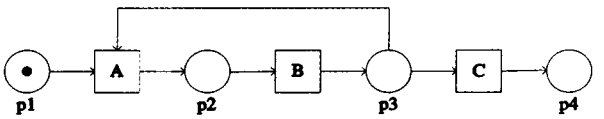

图1-4 循环路由

图 1　4 种基本路由结构的 Petri 网表示

这 4 种结构是工作流执行的基本结构，所有工作流的执行结构可以由这 4 种基本结构组合而成。以上 4 种基本结构可以用图 1-1 到图 1-4 所示的 WF-Net 描述。

在本文中，如果没有特殊说明，提到的 Petri 网都是指工作流网。

3　WF-net 的死锁检测算法

3.1　基于 Petri 网死锁定义的最小结构死锁检测算法

定义 5 对于 Petri 网$\Sigma=\{P,T,F,K,W,M_0\}$，死锁 D (Deadlock)是库所的非空子集，即 $D\neq\phi$，且 $D\subseteq P$，并且满足：$\Gamma^-(D)\subseteq\Gamma^+(D)$。

由死锁的定义可知，如果 $\exists p1\in D$，$\exists t\in T$，使得$(t,p1)\in F$，那么必然 $\exists p2\in D$，使得$(p2,t)\in F$，即如果 $\exists t\in\Gamma^-(D)$，那么必然 $t\in\Gamma^+(D)$。

定义 6 如果 D 是最小死锁，当且仅当除了 D 本身和空集 ϕ 以外，D 中不再包含其他的任何死锁，$|D|\geqslant 2$。

Barkaui，Bermond，Murata 等人对死锁做了深入、细致的分析和描述，得出了最小死锁的 2 个定理。

定理 1 设 D 是 Petri 网$\Sigma=\{P,T,F,K,W,M_0\}$中的死锁，G_D 是由 $D\cup\Gamma^-(D)$给出的图，$\Gamma^-D(t)$为 G_D 中变迁 t 的输入库所。如果死锁 D 满足$|D|\geqslant 2$是一个最小死锁，当且仅当在 G_D 中存在一个通过 D 中所有库所的环 λ 满足，对所有的 $t\in\lambda$，要么$|\Gamma^-D(t)|=1$，要么$|\Gamma^+D(t)|\geqslant 2$，并且存在一个交替环(或由压缩后得到)通过 $\Gamma^-D(t)$的所有库所。

定理 2 D 为 Petri 网$\Sigma=\{P,T,F,K,W,M_0\}$的库所子集，若 D 为最小死锁，当且仅当：由$(D,\Gamma^-(D))$导出的子图可以压缩为一个单一的库所，该库所不与任何转移相连接。

算法 1 小死锁的检测算法

根据定义 5、定义 6 和定理 1、定理 2，很容易构造出最小死锁的检测算法。

步骤 1：$\forall D\subseteq P$，如果 $\Gamma^-(D)\subseteq\Gamma^+(D)$，则 D 为死锁，执行步骤 2，

否则 D 不是死锁，结束。

步骤 2：如果对于 $\forall p\in D$，$\exists t\in\Gamma^+(D)$使得 $\Gamma^+(p)=\{t\}$，则 D 不是最小死锁，结束。

否则执行步骤 3。

步骤 3：对于 G_D 进行压缩，直到$|D|<2$，此时 D 为最小死锁。

重复执行步骤 1～3。

上述算法基于网结构的死锁定义，并不包含资源竞争的语义。另外，基于网结构的死锁检测，很可能只是检测出潜在的死锁，而这些死锁在实际运行过程中也许根本不会出现。事实上，死锁的出现很大程度上和 Petri 网的初始标识以及实际运行过程有关。在实际应用中，其实更加关心运行是否会发生死锁(动态死锁)，以及死锁产生的过程。下面将利用可达图这个工具对动态死锁进行分析。

3.2　基于可达图的死锁检测算法

定义 7 对于 Petri 网$\Sigma=\{P,T,F,K,W,M_0\}$，如果存在可达标识 $M_d\in(\Sigma,M_0)$，并且对于 $\forall t\in T$，t 在 M_d 都是不可执行的，则 M_d 称为死锁标识。

定义 8 对于 Petri 网$\Sigma=\{P,T,F,K,W,M_0\}$，如果可达集 $R(\Sigma,M_0)$中存在死锁标识，则Σ中有死锁。

显然，定义 8 定义的死锁为动态死锁，它反映网的动态运行特性。如果 Petri 网有死锁，则其可达集中必然包含相应的死锁标识。因此，动态死锁的问题可以归约为 Petri 网的可达集中是否存在死锁标识的问题。如果可达集中存在死锁标识，则 Petri 网有死锁，否则无死锁。

因为可达图覆盖整个可达集，根据死锁定义，只要对可达图中的叶子节点进行分析，是否为死锁标识，就可知道该 Pe-

tri 网中是否存在死锁。

算法 2 构造可达图

步骤 1:初始节点 s 用 M_0 来标识。

步骤 2:对于可达图中一个标识为 M 的节点 x,如果 t 在标识 M 下是可执行的,且生成 M'。若 M' 所标识的节点 y 已经存在于可达图中,则增加一条由 x 指向 y 的有向弧,并在弧上标注 t。若 M' 不在可达图中,则在可达图中添加一个以 M' 标识的节点 z,并画一条有 x 指向 z 的有向弧,在弧上标注 t。

步骤 3:找出在标识 M 下可以执行的所有变迁 t 及相应的后继标识,执行步骤 2。

步骤 4:对可达图中所有标识执行步骤 2 和步骤 3,便构造出该 Petri 网的可达图。

在可达图中出度为 0 的节点为叶节点。假设叶子节点的集合为 L,如果 $\exists M\in L$,若 $M(o)=0$,则 M_0 为死锁标识,根据定义 7 可知本系统会出现死锁,死锁的位置为 $M(p)$,其中 $p\neq o$。

3.3 死锁算法分析

可达图算法时间与可能出现的状态成正比。原则上可以对任意 WF-net 进行检查,但是对任意的 WF-net 而言,可能状态数与库所成指数级增长。下面我们来考察一类特殊的 WF-net:基本 WF-net。

定义 9(基本 WF-net) 基本 WF-net 由顺序、选择、并行、循环结构构成,选择与并行可以相互包含,但不能交叉。

定理 3 可达图算法对于基本 WF-net PN 可以在多项式时间内完成死锁检测。

证明:对结构做归纳

顺序结构。假设 PN 是顺序结构的,有库所 n 个,则可能标识为 n 种,算法在库所个数的多项式时间内结束。

选择结构。假设 PN 是选择结构的,有 2 个分支子网(多个分支的情况可以化归到两个分支)。分支 1 有 n_1 个库所、p_1 种标识,p_1 是 n_1 的多项式;分支 2 有 n_2 个库所、p_2 种标识,p_2 是 n_2 的多项式。则 PN 一共有 $n=n_1+n_2$ 个库所、$p=p_1+p_2$ 种标识。显然 p 是 $n(n_1+n_2)$ 的多项式,引入选择结构后,算法仍可以在多项式时间内结束。

并行结构。假设 PN 是并行结构的,有 2 个分支子网(多个分支的情况可以化归到两个分支)。分支 1 有 n_1 个库所、p_1 种标识,p_1 是 n_1 的多项式;分支 2 有 n_2 个库所、p_2 种标识,p_2 是 n_2 的多项式。则 PN 一共有 $n=n_1+n_2$ 个库所、$p=p_1\times p_2$ 种标识。显然 p 是 $n(n_1+n_2)$ 的多项式,引入并行结构后,算法仍可以在多项式时间内结束。

循环结构。循环结构并不引入新的标识。所以,引入循环结构后,算法还是在多项式时间内结束。

证毕。

4 实际应用

图 2 是软件的测试过程模型。例子中的方框表示测试过程中的任务,它们对应于工作流中的活动。方框之间的连接弧表示各个任务之间的关联。

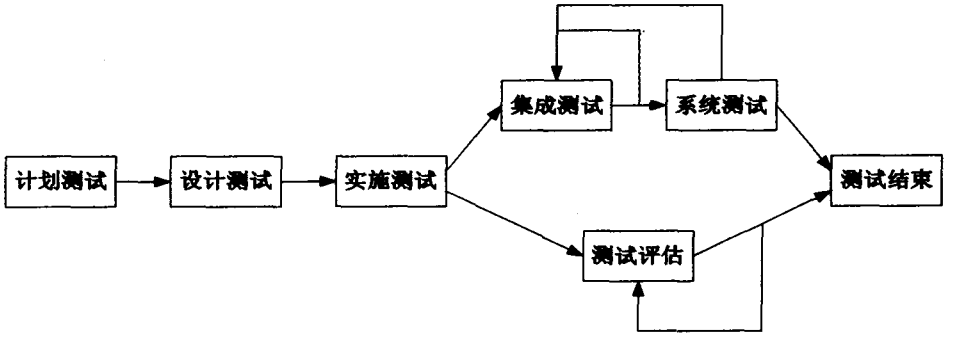

图 2 软件测试工作流

使用 WF-Net 对上述软件测试过程建模,可以得到图 3 所示的 Petri 网。

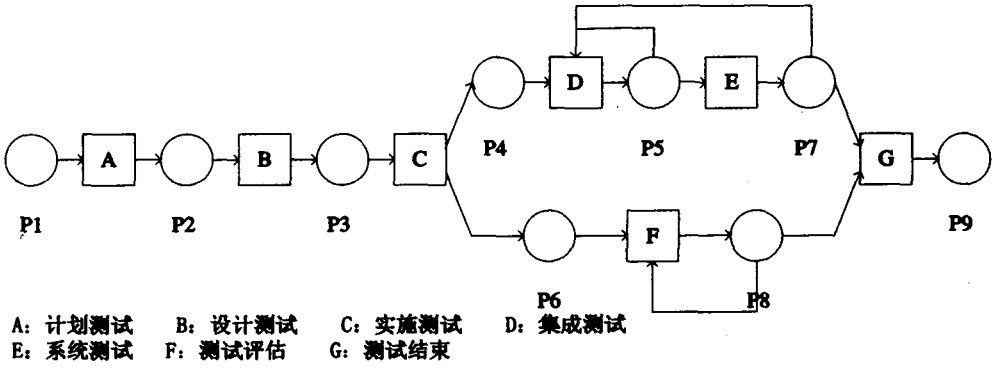

图 3 软件测试过程 Petri 网表示

1)首先,使用最小结构死锁检测算法对它进行分析。

步骤 1:对于 $\forall D\subseteq P$,如果 $\Gamma^-(D)\subseteq\Gamma^+(D)$,则 D 为死锁。例如若 $D=\{P1,P2,P3,P4,P5,P6,P7,P8,P9\}$,则 $\Gamma^-(D)=\{A,B,C,D,E,F,G\}$,$\Gamma^+(D)=\{A,B,C,D,E,F,G\}$,满足 $\Gamma^-(D)\subseteq\Gamma^+(D)$,则 D 为死锁。进行步骤 2。

步骤 2:对于 $\forall p\in D$,$\exists t\in\Gamma^+(D)$ 使得 $\Gamma^+(p)=\{t\}$,则 D 不是最小死锁。由于 $\Gamma^+(P9)=\phi$,所以进行步骤 3。

步骤 3:对于 $G_D=D\cup\Gamma^-(D)$ 中由于不存在一个通过 D

中所有库所的环 λ，所以 D 不是最小死锁。

应用步骤 1～3 的算法对于所有库所的组合进行检测，最后可以得出最小结构死锁为{P5}和{P8}。

2)下面用可达图算法对该 Petri 网进行分析。

初始状态 $M_0=\{1,0,0,0,0,0,0,0,0\}$

步骤 1：只有变迁 A 可以触发，得到状态 $M_1(0,1,0,0,0,0,0,0,0)$，其中 M_1 是新状态。在图中增加一个以 M_1 为标识的节点，并从 M_0 标识的节点到该节点做一条有向弧，用 A 标注。

步骤 2：只有变迁 B 可以触发，得到状态 $M_2(0,0,1,0,0,0,0,0,0)$，其中 M_2 是新状态。在图中增加一个以 M_2 为标识的节点，并从 M_1 标识的节点到该节点做一条有向弧，用 B 标注。

步骤 3：只有变迁 C 可以触发，得到状态 $M_3(0,0,0,1,0,1,0,0,0)$，其中 M_3 是新状态。在图中增加一个以 M_3 为标识的节点，并从 M_2 标识的节点到该节点做一条有向弧，用 C 标注。

步骤 4：有变迁 D 和 F 可以触发，得到状态 $M_4(0,0,0,0,1,1,0,0,0)$、状态 $M_5(0,0,0,1,0,0,0,1,0)$，其中 M_4，M_5 是新状态。在图中增加一个以 M_4 为标识的节点，并从 M_3 标识的节点到该节点做一条有向弧，用 D 标注。并增加一个以 M_5 为标识的节点，并从 M_3 标识的节点到该节点做一条有向弧，用 F 标注。

步骤 5：有变迁 D、E、F(在状态 M_4 下)，变迁 D 和 F(在状态 M_5 下触发)可以触发，分别得到状态 M_4、$M_6(0,0,0,0,0,1,1,0,0)$、$M_7(0,0,0,0,1,0,0,1,0)$，M_7、M_5，其中 M_6、M_7 是新状态。对于新状态 M_6 和 M_7，在图中增加一个以 M_6 为标识的节点，并从 M_4 标识的节点到该节点做一条有向弧，用 E 标注。并增加一个以 M_7 为标识的节点，并从 M_4 标识的节点到该节点做一条有向弧，用 F 标注。另外，增加 M_4 到 M_4(用 D 标注)、M_5 到 M_7(用 D 标注)、M_5 到 M_5(用 F 标注)的有向弧。

步骤 6：有变迁 D、F(在状态 M_6 下触发)，变迁 D、E、F(在状态 M_7 下)可以触发，分别得到状态 M_4、$M_8(0,0,0,0,0,0,1,1,0)$、M_7、M_8、M_7 其中 M_8 是新状态。对于新状态 M_8，在图中增加一个以 M_8 为标识的节点，并从 M_6 标识的节点到该节点做一条有向弧，用 F 标注。并增加 M_6 到 M_4(用 D 标注)，M_7 到 M_7(用 D 标注)，M_7 到 M_8(用 E 标注)，M_7 到 M_7(用 F 标注)的有向弧。

步骤 7：有变迁 D、F、G 可以触发，分别得到状态 M_7、M_8、$M_9(0,0,0,0,0,0,0,0,1)$，其中 M_9 是新状态。对于新状态 M_9，在图中增加一个以 M_9 为标识的节点，并从 M_8 标识的节点到该节点做一条有向弧，用 G 标注。

步骤 8：没有变迁可以发生，结束。

可达图如图 4 所示。

图中 M_9 为唯一的叶子节点，$M_9(o)=1$，且不符合 Petri 网的死锁定义，所以该 Petri 网不会发生死锁。

3)通过实例分析，可以看出结构死锁反映了 Petri 网的结构特性，对建模和分析具有一定的意义。但是作用有限，主要因为：

① 结构死锁是潜在的，有时候它并不影响 Petri 的运行。结构死锁和网的初始标识无关，而实际上初始标识对网的运行过程影响很大，因此结构死锁很难反映网的动态运行特性。

② 对于复杂的 Petri 网，满足结构死锁的库所集合较多，并且结构死锁无法反映死锁产生的原因以及死锁产生的过程。因此，根据结构死锁难以指导正确的建模。

利用可达图来进行死锁检测主要有以下 2 个特点：

① 覆盖了网的可达集，如果存在死锁，那么死锁标识必然被可达集包含。

② 根据可达图，可以了解到导致死锁的运行过程，为死锁的解决提供了必要的信息。

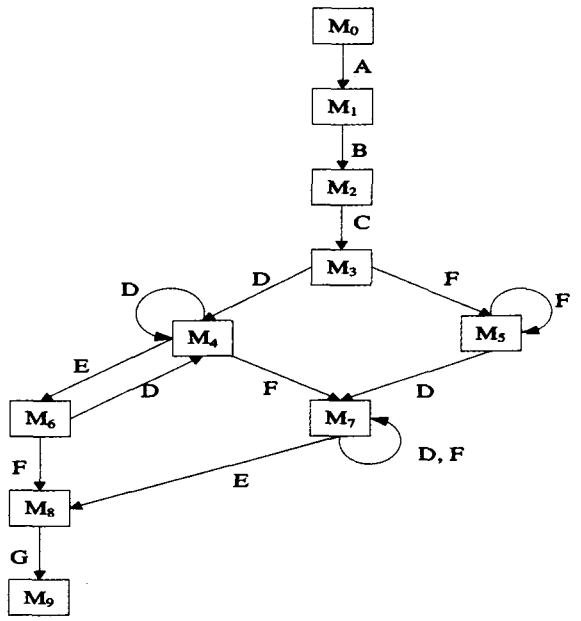

图 4　可达图

总结　Petri 网自提出以来，已经发展成为一个庞大的研究领域。它的形式化的语义、直观的图形表示使得它成为工作流建模的良好工具。本文根据死锁的定义给出最小结构死锁检测算法和基于可达图死锁检测算法。基于网结构的最小死锁检测算法并不包含资源竞争的语义，并且很可能只是检测出潜在的死锁；利用可达图进行死锁检测，动态地模拟网的运行状态，可以有效地检测出存在的死锁，并且可以了解到导致死锁的运行过程，因而实用性较大。本文最后用这两个算法验证了一个实例。

对于 WF-Net 的死锁检测仍然存在以下问题：

1)对于大型的业务过程来说，其规模极其庞大，其建模也十分复杂，验证尤其困难。

2)无死锁只是一个方面，对一个工作流模型来说，其他方面的正确性也必须同时保证，另外它的效率和可维护性也是十分重要的。

今后的工作将在这几个方面展开。

参 考 文 献

1　van der Aalst W M P. The Application of Petri nets to Workflow management. [J] The Journal of Circuits, Systems and Computers, 1998, 8(1): 21～66

2　van der Aalst W M P. A class of Petri net for modeling and analyzing business processes. Computing Science Reports 95/26: Eindhoven, Eindhoven University of Technology, 1995

3　史美林，杨光信，向勇. WFMS：工作流管理系统[J]. 计算机学报，1999(3)

4　Workflow Management Coalition. The workflow reference model. WFMC TC00-1003, 1994

5　袁崇义. Petri 网原理. 北京：电子工业出版社，1998

6　Hauschildt D. A Petri-net-based Workflow Analyzer [R]: [Computing Science Reports 97/12]. Eindhoven: Eindhoven University of Technology, 1997

7　van der Aalst W M P. Structural Characterizations of Sound Workflow Nets [R]: [Computing Science Reports 96/23]. Eindhoven: Eindhoven University of Technology, 1996