论文题目：工作流挖掘与调度算法研究
专　　业：计算机软件与理论
博 士 生：顾春琴
指导教师：常会友 教授

摘　要

企业为了在日趋激烈的市场竞争中立于不败之地，需要不断优化其生产、经营过程，因而对业务过程的高效组织和管理成为提高企业效益、增强企业竞争力的重要手段。工作流建模作为一种业务过程的管理技术，为企业业务过程的高效组织和管理提供了解决方案。

工作流挖掘作为一种重要的工作流建模方法，旨在从信息系统的事件日志中发现关于业务流程的结构化过程，从而避免从空白开始进行既费时又容易出错的工作流模型的设计，其研究对于企业实现业务流程的建模和再造具有重要的意义。目前国内外，特别是国内，对工作流建模的研究主要集中在模型设计方面，而利用事件日志进行工作流挖掘的研究所见不多。此外工作流时间性能分析和调度优化也是工作流管理方面的研究热点。

本论文从启发式工作流挖掘算法、智能化工作流挖掘算法两个方面对工作流挖掘进行了深入地研究与分析；对工作流时间性能以及工作流调度优化进行了深入地研究与分析，具体的研究内容和创新点如下：

（1）能解决多种复杂任务的工作流挖掘。为了有效挖掘含有多种复杂任务的过程模型，对含有循环任务和重复任务的事件日志进行了研究，提出发现循环、重复和同一任务的启发式规则，并且给出证明；改进了 α 算法的关联关系定义，在此基础上提出了 τ 算法。实例验证该算法是有效的，对循环、重复和同一任务的判定是正确的。对挖掘出的模型进行仿真分析，仿真结果表明使用 τ 算法挖掘出的模型所产生的日志和原始日志具有逻辑等价性。

（2）基于混合自适应遗传算法的工作流挖掘。为了解决目前工作流挖掘算法大都采用局部策略因而无法保证最优挖掘以及算法对噪声敏感的问题，提出基于混合自适应遗传算法的工作流挖掘算法。该算法与启发式算法相比具有更高的

鲁棒性和对噪声的抗干扰性；与基本遗传算法相比，该算法能显著提高解的质量和收敛速度。

（3）工作流时间性能分析。工作流性能分析是对工作流进行评价和优化的基础，时间性能则是衡量工作流性能的一个重要指标。利用概率论中关于服从指数分布的随机变量的分布函数、密度函数及数学期望的基本性质，详细地分析了组成 SPN 模型的串行、并行、选择和循环四种基本结构的平均延迟时间，设计了通用的 SPN 模型平均延迟时间公式。通过对复杂 SPN 模型的等效化简，实现对工作流时间性能的分析。最后，通过实例验证了该方法的可行性和有效性。

（4）工作流调度优化。为了解决工作流调度优化问题，以 QoS 为优化目标，提出了克隆选择离散粒子群算法，运用该算法进行工作流调度优化研究。该算法增加了种群的多样性，提高了算法精确度，加快了算法的收敛速度，克服了离散粒子群算法早熟收敛和局部极值等问题，在工作流调度应用中具有良好的性能。

关键词：工作流挖掘；工作流调度；时间性能；遗传算法；离散粒子群算法

Title: Research on Workflow Mining and Scheduling Algorithm
Major: Computer Software and Theory
Name: Chunqin Gu
Supervisor: Professor Huiyou Chang

Abstract

In order to win in the increasingly severe marketing competition, an enterprise needs to constantly optimize the production and business process. The effective organization and management of business process is an important approach to improving enterprise benefit and enhancing enterprise competitiveness. Workflow modeling is a kind of technology which is closely associated with business process. It can support the effective organization and management of business process.

Workflow mining is an important workflow modeling method, which is used to discover structural process of business from event log of information system. Workflow mining can avoid designing the workflow model from scratch, which is a complicated time–consuming process. The research on workflow mining has important significance for the modeling and reengineering of business process. Despite the importance of workflow mining, there are few research work done in this area. In addition, workflow time performance analysis and workflow scheduling optimization are also the research focus in workflow management.

This dissertation studies on workflow mining, including heuristic workflow mining algorithm, intelligent workflow mining algorithm. In addition, this dissertation studies on workflow time performance analysis and workflow scheduling optimization. The main contributions of this dissertation are listed as follows:

(1) In order to mine process from event log with cyclic tasks, duplicate tasks and same tasks, and to optimize enterprise modeling method, an improved mining algorithm called τ-algorithm is presented based on improving the α-algorithm. The ordering relations between tasks are redefined; heuristic rules are put forward for

identifying cyclic tasks, duplicate tasks and same tasks among event log. The mined WF-net is simulated for generating reversely event log. The equivalence between simulated log and raw log is analyzed. The validity of τ-algorithm is proved.

(2) Current workflow mining algorithm using local strategy can't ensure that a globally optimal process model is mined. The algorithm is also sensitive to noise. To solve the problems, a hybrid adaptive genetic algorithm (HAGA) is proposed. The simulation testing results demonstrate that the new algorithm has noise immunity and is more robust than α algorithm, and that it can find better solution and converge faster than the simple genetic algorithm (SGA) employing general genetic strategy.

(3) In order to improving the efficient organization and management, Workflow time performance analysis is realized by means of equivalent simplification to complex SPN models. Using the basic properties of probability theory about distribution function, density function and mathematical expectation of random variable agreeing with exponential distribution, average delay time is calculated to analyze the average delay time of serial, parallel, selective and cyclic structure. A general method of analyzing average delay time is proposed. Workflow time performance analysis is realized by means of equivalent simplification to complex SPN models.

(4) In order to solve the problem of QoS constrained workflow scheduling optimization, the discrete particle swarm optimization algorithm based on clonal selection is given. The algorithm can increase the swarm's diversity and precision, speed up convergence speed and overcome premature convergence and local optimum. The performance of our algorithm is very promising in workflow scheduling application.

Keywords: Workflow Mining; Workflow Scheduling; Time Performance; Genetic Algorithm; Particle Swarm Optimization

论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期：2009年 6月8日

学位论文使用授权声明

本人完全了解中山大学有关保留、使用学位论文的规定，即：学校有权保留学位论文并向国家主管部门或其指定机构送交论文的电子版和纸质版，有权将学位论文用于非赢利目的的少量复制并允许论文进入学校图书馆、院系资料室被查阅，有权将学位论文的内容编入有关数据库进行检索，可以采用复印、缩印或其他方法保存学位论文。

学位论文作者签名： 导师签名：

日期：2009年 6月8日 日期：2009年 6月8日

第1章 绪论

1.1 研究背景及意义

计算机支持的协同工作（Computer Supported Cooperative Work，CSCW）[1]是指地域分散的一个群体借助计算机及其网络技术，共同协调与协作来完成一项任务。通过建立协同工作流环境，可以改善人们的沟通方式，消除人们时间和空间的距离，节省工作人员的时间和精力，提高群体工作质量和效率。

工作流管理系统作为 CSCW 系统的一个分支，是一项发展迅速的技术，已成为各种管理信息系统以及办公自动化系统的核心，在许多企业中得到了广泛的应用。Delphi Group 的创始人和主席 Thomas Koulopoulos 曾预言工作流管理系统将最终成为覆盖各类终端与网络操作系统之上的业务操作系统（Business Operation System，BOS），将带来操作系统、信息管理软件的一次革命，乃至在将来将应用从企业延伸到家庭中，成为新时代的家庭信息平台（Family Information Platform，FIP）[2]。

工作流技术又是工作流管理系统的核心技术。工作流技术是实现企业业务过程建模、业务过程仿真分析、业务过程优化、业务过程管理与集成，从而最终实现业务过程自动化的核心技术[3]。对企业利用工作流方法进行业务过程的建模和深入分析不仅可以规范化企业的业务过程，发现业务流程中不合理的环节，进而对企业的业务过程进行优化重组，而且所建立的业务过程模型本身就是企业非常重要的知识库和规则库，可以成为指导企业实施计算机管理信息系统的模型。

工作流建模作为一种业务过程的管理技术，为企业业务过程的高效组织和管理提供了解决方案。工作流挖掘作为一种重要的工作流建模方法，旨在从信息系统的事件日志中发现关于业务流程的结构化过程，从而避免了从空白开始进行既费时又容易出错的工作流模型的设计，其研究对于企业实现业务流程的建模和再造具有重要的意义。目前国内外，特别是国内，对工作流建模的研究主要集中在模型设计方面，而利用事件日志进行工作流挖掘的研究所见不多。

工作流模型的性能分析一般指通过仿真或严格的理论分析获得系统性能的量化指标，如业务平均处理时间[4]，在工作流管理系统中发挥着重要的作用，可

定量评价工作流的时间性能。工作流时间性能分析是工作流管理方面的研究热点。

工作流调度优化也是目前研究的热点，合理地调度服务资源不但可以提高工作流的运行效率，而且能降低资源的使用成本，提高系统的可靠性。

本论文从启发式工作流挖掘算法、智能化工作流挖掘算法两个方面对工作流挖掘进行了深入地研究与分析；对工作流时间性能以及工作流调度优化进行了深入地研究与分析。将本论文研究的理论成果应用到企业中，有助于企业在更短时间内生产出成本低、质量高的产品，或在更短的时间内提供更好的服务，有助于企业进行业务过程改进和优化，以提高企业的市场竞争力，在全球市场竞争中立于不败之地。因此，本论文所做的研究具有一定的理论意义和重要的应用价值。

1.2 主要研究工作

本论文从工作流挖掘、工作流时间性能分析和工作流调度三个方面对工作流相关技术进行了深入地研究。

（1）基于启发式算法的工作流挖掘

研究与分析现有的工作流挖掘理论、技术和方法。了解到目前对于工作流挖掘的研究不是很多，并且现有的研究都是只能解决简单的挖掘问题。

在对工作流挖掘问题进行深入研究的基础上，提出了新的工作流挖掘算法——τ算法。该算法解决了事件日志中含有循环任务、重复任务以及同一任务的工作流挖掘问题。

（2）基于进化算法的工作流挖掘

研究与分析了遗传算法的基本理论以及应用现状。了解到遗传算法目前应用的领域非常广泛，包括工业、交通业、经济管理、图像处理等。而在工作流挖掘领域的应用，却所见不多。虽有文献使用了遗传算法，但是由于基本遗传算法具有早熟或后期收敛缓慢等缺点，不能很好地解决活动多、复杂性高和有噪声的问题。

本文深入分析了工作流挖掘问题的模型表示，建立了工作流挖掘问题的数学模型，提出使用关联矩阵进行染色体编码，并且根据挖掘模型与原有日志的符合

性定义了染色体的适应值评价函数。结合进化过程的分阶段性，提出混合自适应遗传算法，有效地解决了活动规模大、含噪声的工作流挖掘问题。

（3）工作流时间性能分析

研究与分析了工作流时间性能分析方面的相关研究。了解到当前关于工作流时间性能分析往往使用复杂难以理解的定理的形式给出，并且不具有可读性。本论文从另外一种角度出发，使用基本的数学期望的公式详细定量分析了工作流四种基本结构的时间性能。

（4）工作流调度

研究并分析了工作流调度的相关文献以及其它领域的调度问题，分析和总结了目前工作流调度的主要问题和不足。在学习研究相关进化算法理论后，确定改进适用于连续问题空间的粒子群算法，并运用改进后的算法进行工作流调度的研究。将改进后的算法和经典的二进制离散粒子群算法进行了实验比较。实验结果表明新的算法收敛速度快，求解精度高、稳定性好、能够有效抑制早熟收敛，在工作流调度应用中具有良好的性能。

1.3 课题来源

本论文的研究是依托粤港关键重点突破项目“面向制造业信息化和电子政务领域的软件构件库平台”（编号：2006Z1-D6021）、国家自然科学基金项目“基于人工生命计算的计算协同关键问题研究”（编号：60573159）和广东省自然科学基金项目“协同软件关键技术及其应用研究”（编号：05200302）进行的。本论文的研究内容是这三个项目研究的重要组成部分。

1.4 论文组织

本论文共分为 7 章，内容安排如图 1-1 所示。

第 1 章（本章）作为本论文的绪论部分，分析了工作流挖掘以及工作流调度的重要地位，并阐述了本论文研究的背景和意义。另外，简要介绍了本论文的主要研究工作、课题来源和论文的组织结构。

第 2 章主要介绍了本论文涉及的相关理论基础及主要研究问题的现状。首先介绍了 Petri 网和工作流网的基本概念，然后介绍了遗传算法和粒子群算法这两个和本论文相关的进化算法，最后从工作流挖掘、工作流时间性能分析和工作流

调度三个方面综述了目前国内外的研究现状。

第 3、4、5、6 章围绕本论文的主题进行深入地讨论与研究，是本论文的主要内容。

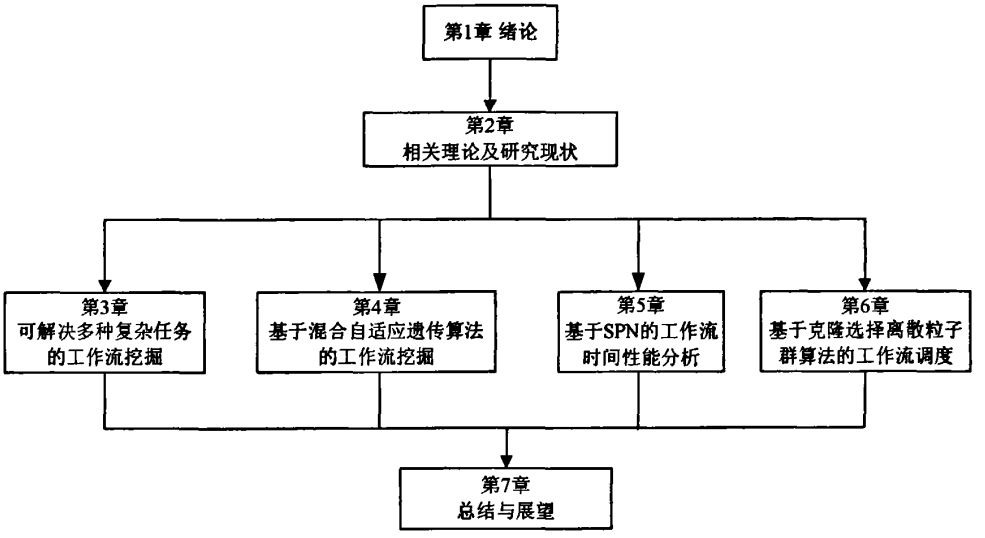


图 1-1 论文组织结构

第 3 章在分析前人工作的基础上，针对目前工作流挖掘算法存在的不足，提出了新的工作流挖掘算法—— τ 算法。首先提出启发式判定规则，用来识别事件日志中所包含的循环任务、重复任务和同一任务，然后运用 τ 算法进行挖掘提取出工作流网，并且通过 CPN Tools 仿真工具对挖掘出的工作流模型进行仿真，并分析仿真日志与原始日志的等价性，从而验证 τ 算法的可行性和正确性。

第 4 章运用混合自适应遗传算法对工作流挖掘进行了深入地研究。针对目前工作流挖掘算法采用局部策略因而无法保证最优挖掘以及算法对噪声敏感的情况，提出基于混合自适应遗传算法的工作流挖掘算法。首先定义了基本工作流网以及变迁的使能和点火规则，给出了过程模型的定义；然后提出了过程模型转换成基本工作流网的算法，设计了衡量事件日志与过程模型符合性的适应值评价函数；最后设计了根据进化阶段以及个体相似度而混合自适应的交叉率和变异率。仿真实验结果表明，该算法与 α 算法相比具有更高的鲁棒性和对噪声的抗干扰性；与基本遗传算法相比，该算法能显著提高解的质量和收敛速度。

第 5 章基于 SPN 对工作流时间性能进行了深入地分析。工作流性能分析是对工作流进行评价和优化的基础，时间性能则是衡量工作流性能的一个重要指

标。利用概率论中关于服从指数分布的随机变量的分布函数、密度函数及数学期望的基本性质，详细地研究了组成 SPN 模型的串行、并行、选择和循环四种基本结构的平均延迟时间，设计了通用的 SPN 模型平均延迟时间公式。通过对复杂 SPN 模型的等效化简，实现对工作流时间性能的分析。最后，通过实例验证了该方法的可行性和有效性。

第 6 章研究了离散粒子群算法，以及其在工作流调度方面的应用。首先提出旋转击发规则，并使用该规则对粒子的位置进行离散化处理，改进了传统的适用于连续优化领域的粒子群算法，提出了克隆选择离散粒子群算法。为了验证本章所提出的算法的优越性，与二进制离散粒子群算法进行了实验比较。实验结果表明，本章提出的克隆选择离散粒子群算法收敛速度快，求解精度高、稳定性好、能够有效抑制早熟收敛，在工作流调度应用中具有良好的性能。

第 7 章是总结与展望。主要对本论文的研究工作做了进一步的归纳和总结，并展望了下一步的研究工作。

第2章 相关理论及研究现状

本章概述了工作流挖掘和调度涉及的 Petri 网、工作流网的基本概念和基本原理；应用于工作流挖掘和调度的相关进化算法，如遗传算法、粒子群算法；工作流挖掘、工作流时间性能分析和工作流调度的基本概念以及国内外研究现状。

2.1 Petri 网

Petri 网最早在 1962 年由德国的 Petri[5]提出。Petri 网的主要优点是描述异步并发能力和它的图形表示。和其它系统模型一样，Petri 网由两类元素构成：表示状态的元素以及表示变化的元素[6]。Petri 网是一种可用图形表示的组合模型，同时又是严格定义的数学对象，既可用于静态结构分析，又可用于动态行为分析。有关 Petri 网的更详尽的阐述，可参见文献[7]。

定义 2-1 （有向网，简称网[6]）. 三元组 $N=(P, T, F)$称为有向网（简称网）的充分必要条件是：

（1）$P \cap T = \emptyset$；

（2）$P \cup T \neq \emptyset$；

（3）$F \subseteq P \times T \cup T \times P$，其中“×”表示笛卡尔乘积；

（4）$dom(F) \cup cod(F) = P \cup T$，其中

$$dom(F) = \{x \mid \exists y : (x, y) \in F\}$$

$$cod(F) = \{y \mid \exists x : (x, y) \in F\}$$

它们分别是 F 的定义域和值域，P 和 T 分别称为 N 的库所（place）集和变迁（transition）集。F 称为流关系（flow relation）。

网由两类节点组成：库所和变迁。库所用圆形“○”表示，变迁用方框“□”表示。节点间通过有向弧连接，同类节点不能直接相连，用从 x 到 y 的带箭头的弧线表示流关系中的（x，y）。

定义 2-2 （输入库所和输出库所）. 如果存在一条有向弧线从库所 p 指向变迁 t，则库所 p 称为变迁 t 的输入库所，变迁 t 的输入库所集合记为 $\cdot t$；如果存在一条有向弧线从变迁 t 指向库所 p，则库所 p 称为变迁 t 的输出库所，变迁 t 的输出库所集合记为 $t\cdot$。

若网 $N_1=(P_1, T_1, F_1)$，$P_1=\{p_1, p_2, p_3, p_4, p_5, p_6\}$，$T_1=\{t_1, t_2, t_3, t_4, t_5\}$，$F_1=\{(p_1, t_1), (t_1, p_2), (t_1, p_4), (p_2, t_2), (p_2, t_3), (p_4, t_4), (t_2, p_3), (t_3, p_3), (t_4, p_5), (p_3, t_5), (p_5, t_5), (t_5, p_6)\}$，则 N_1 网的图形表示如图 2-1 所示。

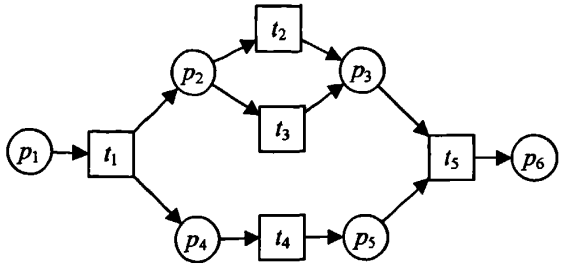


图 2-1 有向网 N_1 的图形表示

有向网是系统的结构框架，在框架上的活动是系统中流动的资源。在有向网的基础上指明资源的初始分布，并规定框架上的活动规则（即库所的容量和变迁与资源之间的数量关系），则称为网系统。

记 $N^0=\{0, 1, 2, \cdots\}$，$N^+=\{1, 2, 3, \cdots\}$，ω 表示无穷：$\omega=\omega+1=\omega-1=\omega+\omega$。

定义 2-3 [6]

（1）K：$P\rightarrow N^+\cup\{\omega\}$称为有向网 $N=(P, T, F)$的容量函数。

（2）对给定的容量函数 K，M ：$P\rightarrow N^0$ 称为 N 的一个标识的条件是：$\forall p\in P$: $M(p)\le K(p)$。

（3）W：$P\rightarrow N^+$称为 N 上的权函数，对$(x, y)\in F, W(x, y)$称为(x, y)上的权。权函数规定了每个变迁发生一次所引起的资源数量的变化。$\forall(x, y)\in F$，$0<W(x, y)<\omega$。

定义 2-4 （网系统[6]）. 六元组$\sum=(P, T, F, K, W, M_0)$构成网系统的条件是：

（1）$N=(P, T, F)$构成有向网，称为 $\sum$ 的基网。

（2）K，W，M 分别为 N 上的容量函数、权函数和标识。M_0 称为 $\sum$ 的初始标识。

以上给出的是网系统的静态特征。完整的网系统不仅包括静态特征还包括动态规律，即变迁规则。变迁规则包括变迁使能条件和变迁点火后果，定义 2-5 和定义 2-6 分别给出了变迁使能条件和变迁点火后果的定义。

设 M 为网系统 $\sum=(P, T, F, K, W, M_0)$的基网$(P, T, F)$上的任一标识，$t\in T$，为任一变迁。

定义 2-5 （变迁使能条件[6]）.t 在 M 标识下使能的条件是：

$$\forall p\in {}^{\cdot}t: M(p)\geq W(p,t) \wedge \forall p\in t^{\cdot}: M(p)+W(t,p)\leq K(p)$$

t 在 M 标识下是使能的，记作 $M[t>$，也称 M 授权 t 点火或 t 在 M 授权下点火。

定义 2-6 （变迁点火后果[6]）.若 $M[t>$，则 t 在 M 标识下是使能的，变迁点火后得到新标识 M'，对$\forall p\in P$，

$$M'(p)=\begin{cases} M(p)-W(p,t), & p\in {}^{\cdot}t-t^{\cdot} \\ M(p)+W(p,t), & p\in t^{\cdot}-{}^{\cdot}t \\ M(p)-W(s,t)+W(t,s), & p\in t^{\cdot}\cap {}^{\cdot}t \\ M(p), & p\notin {}^{\cdot}t\cup t^{\cdot} \end{cases}$$

标识由 M 变为 M 的后继 M'，记作 $M[t> M'$。

在网系统的图形表示中，每个库所 p 都有一个对应的圆，标识 M 的图形表示为：在 p 对应的圆中画上 $M(p)$个黑点（称为托肯，token）。在 $K(p)=\omega$ 和 $W(x, y)=1$ 时均采用默认法，即不标明的容量均为无穷，不标明的权均为 1。

Petri 在文献[5]中使用的系统模型就是 $K\equiv\omega$ 和 $W\equiv 1$ 的网系统，这就是传统上称为 Petri 网的网系统，又称为库所/变迁网（Place/Transition net，简称 P/T 网）。

定义 2-7 （发生序列，变迁序列[6]）. $s=M_0t_1M_1t_2M_2\cdots M_nt_n$ 为 $\sum$ 的一个有限出现序列的充分必要条件是：对所有的 i，$i=1$，2，…，$M_{i-1}[t_i>M_i$。称 $\tau=t_1t_2\cdots t_n$ 为变迁序列。

2.2 工作流网

Aalst[8]等人把工作流的概念映射到 Petri 网上，采用 Petri 网对工作流的控制流维度进行建模，并称之为工作流网（Workflow net, WF-net）。

定义 2-8（**WF-net**[8]）. Petri 网 $\sum=(P, T, F)$是 WF-net，当且仅当：

（1）$\sum$有一个源库所 $i\in P$，且$\cdot i=\emptyset$；

（2）$\sum$有一个汇库所 $o\in P$，且 $o\cdot=\emptyset$；

（3）如果在$\sum$中添加一个变迁 t 且$\cdot t=\{o\}$，$t\cdot=\{i\}$，则新的 Petri 网 $\sum'=(P, T\cup t, F\cup\{\cup, i\}\cup\{o, \cup\})$是强连通的。

文献[8]采用 WF-net 描述了工作流过程模型中的顺序、并行、选择和循环这四种基本控制结构。

（1）顺序路由

顺序路由用来描述一个活动执行完成后，另一个活动才能接着执行的情况。图 2-2 给出了一个顺序路由的例子，活动 *A* 完成后活动 *B* 才能开始执行；活动 *B* 完成后，活动 *C* 才能开始执行。

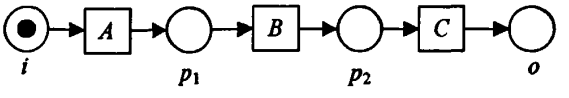


图 2-2 顺序路由

（2）并行路由

并行路由用来描述多个活动可以同时执行或以任意次序执行的情况。为了建模活动之间的并行路由关系，引入两个控制任务 AND-split 和 AND-join。图 2-3 给出了并行路由的例子，活动 *B* 和活动 *C* 的执行顺序有三种可能：*B*、*C* 两个活动同时执行；*B* 先执行，*C* 后执行；*C* 先执行，*B* 后执行。t_1、t_2 分别表示 AND-split 和 AND-join 任务。

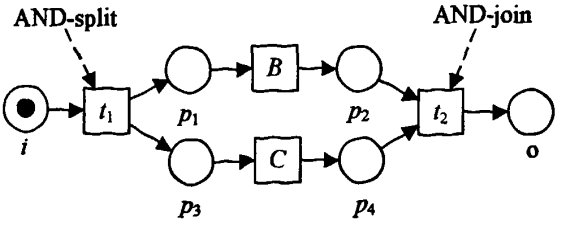


图 2-3 并行路由

（3）选择路由

选择路由用来描述只能从多个活动中选择其中一个活动来执行的情况。为了建模活动之间的选择路由关系，引入两个控制任务 OR-split 和 OR-join。图 2-4 给出了选择路由的例子，活动 *B*、*C* 中只能有一个活动被选择执行，可能是 *B* 被执行，也可能是 *C* 被执行。p_1、p_2 分别表示 OR-split 和 OR-join 任务。

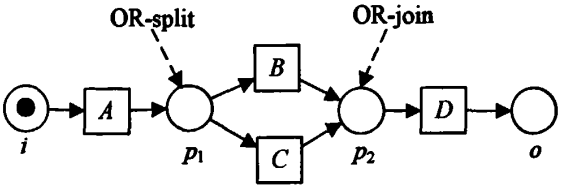


图 2-4 选择路由

（4）循环路由

循环路由用来描述某一个或多个活动需要反复执行的情况。循环路由是一种特殊的选择路由，同样引入了两个控制任务 OR-split 和 OR-join。图 2-5 给出了循环路由的例子。活动 *B* 可能被多次执行。p_1、p_2 分别表示 OR-join 和 OR-split 任务。

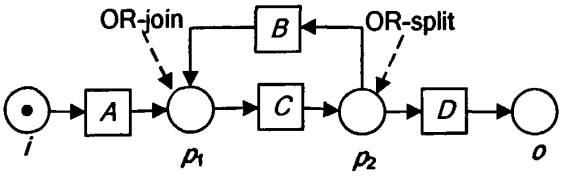


图 2-5 循环路由

2.3 相关进化算法

2.3.1 遗传算法

求解最优解或近似最优解的传统方法主要有枚举法、启发式方法和搜索算法[9]。随着问题规模的扩大、约束条件的增多，这些传统的方法在可以接受的时间内不能得到问题的最优解或近似最优解。

遗传算法是近年来迅速发展起来的一种高度并行的随机搜索与优化方法，它

为解决大规模优化问题提供了一种有效的途径。遗传算法（Genetic Algorithms，GA）[10]是美国 Michigan 大学的 J. Holand 于 1975 年受生物进化论的启发提出的，是模拟生物在自然环境中的遗传和进化过程而形成的一种自适应全局优化概率搜索算法。

（1）基本遗传算法

通过模仿自然界中生物遗传与进化机制，许多学者设计了各种不同的遗传算子来模仿不同环境下的生物遗传特性，构成了各种不同的遗传算法。但这些遗传算法都有一个共同的特点，即在模仿生物遗传与进化过程中都使用了选择、交叉、变异机制。Goldberg[11]总结了一种统一的最基本的遗传算法，称为基本遗传算法（Simple Genetic Algorithm, SGA）。

基本遗传算法从一组随机产生的初始解（称为初始种群）开始进化过程。种群中每个个体对应为问题空间的一个解，称为染色体。基本遗传算法中以个体适应度的大小来评价各个个体的优良程度，从而决定其遗传到下一代的机会大小。评价个体适应度的函数称为适应值函数。基本遗传算法主要通过选择、变异、交叉运算不断迭代产生下一代种群。选择运算把当前种群中适应值高的个体遗传到下一代种群中。交叉运算是遗传算法中产生新个体的主要操作，它以某一概率（称为交叉概率，P_m）将个体部分染色体相互交换产生两个新个体（称为后代个体），并用来替代原来的两个“旧”个体（称为父个体）。变异运算是对个体的某一个或某一些基因座上的基因值按较小的概率（称为变异概率，P_c）进行变异。种群不断地进行选择、交叉和变异运算，经过若干代，算法收敛于一个优良个体，它就是最优解或次优解。基本遗传算法的运算过程如图 2-6 所示。

该算法使用了上述三种遗传算子（选择算子、交叉算子、变异算子），主要运算步骤如下：

（1）初始化。设置进化代计数器 t=0；设置最大进化代 T；随机生成规模为 N 的初始种群 $P(0)$；

（2）选择运算。将选择算子作用于种群；

（3）交叉运算。将交叉算子作用于种群，交叉概率一般取[0.4，0.99]；

（4）变异运算。将变异算子作用于种群，变异概率一般取[0.05，0.25]，种群 $P(t)$经过选择、交叉、变异运算后得到下一代种群 $P(t+1)$；

（5）判断终止条件。若 $t<T$，则 $t=t+1$，转到（2）；若 $t=T$，则将种群中具有最大适应值的个体作为最优解输出，迭代终止。

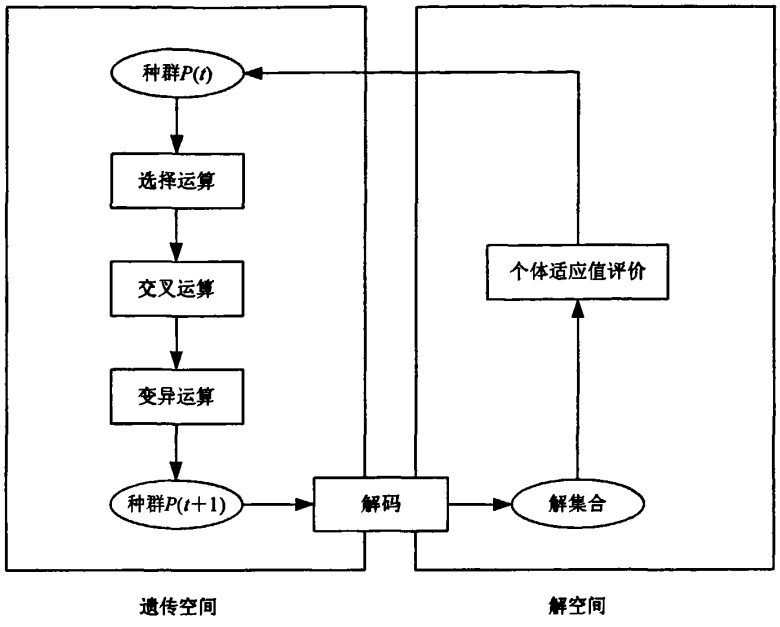


图 2-6 基本遗传算法的运算过程

基本遗传算法是一类可用于复杂系统优化计算的鲁棒搜索算法，与其他一些优化算法相比，具有如下特点[11]：

1、基本遗传算法以决策变量的编码作为运算对象。与传统的优化算法直接利用决策变量的实际值进行优化计算不同，基本遗传算法不是直接以决策变量的实际值来进行运算的，而是通过对决策变量进行编码处理后，再进行遗传算子操作。

2、基本遗传算法直接以目标函数值为搜索信息。与传统的优化算法需要利用目标函数的导数值等辅助信息确定搜索方向不同，基本遗传算法仅使用由目标函数值变换后的适应值来确定搜索方向和搜索范围。这个特性给很多目标函数无法求导或导数不存在的优化问题提供了便利。另外，利用个体适应值可以将搜索范围集中到适应值较高的搜索空间，提高了搜索的效率。

3、基本遗传算法具有很高的并行性。与传统的优化算法往往从解空间的一个初始点开始搜索不同，基本遗传算法从一定规模的初始种群开始同时在多个区域搜索最优解，从而降低了算法陷入局部最优解的可能性。

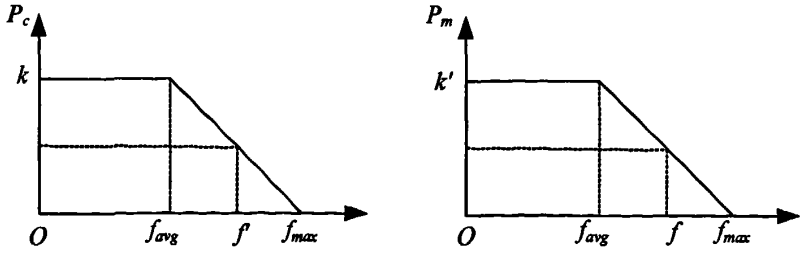
4、基本遗传算法使用概率搜索技术。与传统的优化算法使用确定性的搜索

方法不同，基本遗传算法使用概率搜索技术，其选择、交叉、变异等运算都是基于概率的方式进行的，从而增加了其搜索过程的灵活性。

（2）自适应遗传算法

基本遗传算法中交叉算子和变异算子的交叉率和变异率都是固定的。算法参数交叉率 P_c 和变异率 P_m 的取值是影响基本遗传算法行为和性能的关键所在，直接影响算法的收敛性，P_c 越大，新个体产生的速度就越快[12]。交叉率 P_c 取值过大导致遗传方式被破坏的可能性也增大，易破坏适应度高的个体结构；P_c 过小，则导致搜索过程缓慢，甚至停滞不前。变异率 P_m 取值 P_m 过小导致不易产生新的个体结构；P_m 过大，则导致遗传算法退化为纯粹的随机搜索算法。针对不同的优化问题，需要反复实验来确定 P_c 和 P_m 的取值，这是一件繁琐的工作。

Srinvivas 等人[13]提出一种自适应遗传算法（Adaptive Genetic Algorithm, AGA），该算法中 P_c 和 P_m 能够随适应度自动改变，自适应交叉和变异率的调整曲线如图 2-7 所示。当种群个体适应度趋于一致或者趋于局部最优时，增加 P_c 和 P_m，而当种群适应度比较分散时，则减少 P_c 和 P_m。同时，对于适应值越高于种群平均适应值的个体，P_c 和 P_m 的取值越低，使得该个体得以遗传到下一代；而越低于平均适应值的个体，P_c 和 P_m 的取值越高，使得该个体被淘汰的概率越高。因此，自适应的 P_c 和 P_m 能根据个体适应度的高低取到最佳 P_c 和 P_m。自适应遗传算法在保持种群多样性的同时，保证遗传算法的收敛性。



(a)自适应交叉率($k=k_1=k_2$)　　(b)自适应变异率($k'=k_3=k_4$)

图 2-7　自适应交叉率和变异率

在自适应遗传算法中，P_c 和 P_m 自适应调整公式如式(2-1)和式(2-2)所示，式中，f_{max} 表示种群中最大的适应度值；f_{avg} 表示每代种群的平均适应值；f' 表示要交叉的两个个体中较大的适应值；f 表示要变异个体的适应值；k_1，k_2，k_3，k_4 在(0，1)区间取值。

$$P_c = \begin{cases} \dfrac{k_1(f_{max} - f')}{f_{max} - f_{avg}}, & f \geq f_{avg} \\ k_2, & f < f_{avg} \end{cases} \tag{2-1}$$

$$P_m = \begin{cases} \dfrac{k_3(f_{max} - f)}{f_{max} - f_{avg}}, & f \geq f_{avg} \\ k_4, & f < f_{avg} \end{cases} \tag{2-2}$$

2.3.2 粒子群算法

粒子群算法（Particle Swarm Optimization, PSO）[14]，是由美国社会心理学家 Kenedy 和电气工程师 Eberhart 通过对鸟类群体行为的观察研究，于 1995 年共同提出的一种智能优化算法。粒子群算法自提出后，由于算法计算速度快，简单易于实现，引起了国际上相关领域学者的关注，并对其进行改进研究。由于基本粒子群算法主要适用于连续优化问题，因此为了使其能应用于离散空间优化问题，需要对基本粒子群算法进行离散化，形成离散粒子群算法。

（1）基本粒子群算法

PSO 的提出来源于对鸟群捕食行为的研究。一群鸟在随机寻找食物，在这个区域里只有一块食物，所有的鸟都不知道食物在哪里，那么鸟找到食物的最优策略就是搜寻目前离食物最近的鸟的周围区域。PSO 算法就是从这种鸟群捕食的模型中得到启示而产生的。

PSO 中称这些捕食的鸟为“粒子”（particle），粒子的位置就是解空间的一个解。“食物”就是优化问题的最优解。PSO 中每个粒子根据自己的飞行经验和同伴的飞行经验来调整自己的飞行速度和位置。每个粒子在飞行过程所经历过的最好位置，就是粒子本身找到的最优解。整个种群所经历过的最好位置，就是整个种群目前找到的最优解。前者称为个体极值（pBest），后者称之为全局极值（gBest）。每个粒子根据 pBest 和 gBest 不断更新自己的位置和速度，从而产生新一代种群。PSO 中通过适应值来评价粒子的优劣。

PSO 与其它进化类算法相似，也是采用“种群”与“进化”的概念，也是根据个体（粒子）的适应值大小进行操作[15]。不同的是，PSO 没有象其它进化算法那样对每个个体使用进化算子，而是将个体看作是在 n 维空间中没有重量和体

积的粒子。

设 $X_i=(x_{i1}, x_{i2}, \ldots, x_{in})$为粒子 i 的当前位置，$V_i=(v_{i1}, v_{i2}, \ldots, v_{in})$为粒子 i 的当前飞行速度；$P_i=(p_{i1}, p_{i2}, \ldots, p_{in})$为粒子 i 所经历的具有最优适应值的位置。对于最小化问题，适应值越小，位置越优。基本粒子群算法的速度和位置更新公式分别是式(2-3)和式(2-4)。

$$v_{ij}(t+1)=v_{ij}(t)+c_1r_{j1}(t)(p_{ij}(t)-x_{ij}(t))+c_2r_{j2}(t)(p_{gj}(t)-x_{ij}(t)) \tag{2-3}$$

$$x_{ij}(t+1)=x_{ij}(t)+v_{ij}(t+1) \tag{2-4}$$

其中 i 表示第 i 个粒子，j 表示粒子的第 j 维，t 表示第 t 代，c_1、c_2 表示加速常数，在 0～2 间取值，c_1 调整粒子飞向自身最好位置方向的步长，c_2 调整粒子向全局最好位置飞行的步长，r_{j1}、$r_{j2}\in[0,1]$，为均匀分布的随机数。式(2-3)由三部分构成，第一部分表示粒子先前的速度，第二部分是粒子的认知属性，第三部分是粒子的社会属性。基本粒子群算法流程如图 2-8 所示。

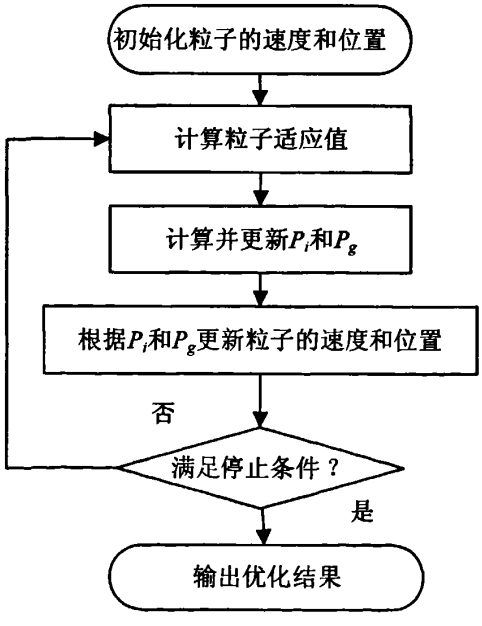


图 2-8 基本粒子群算法流程

（2）离散粒子群算法

离散粒子群算法（Discrete Particle Swarm Optimization，DPSO）首次由 Kennedy 等人[16]提出。在 DPSO 中，粒子位置 $x_{ij}(t)$采用二进制的 0 和 1 表示，但由式(2-3)可以得知，$v_{ij}(t)$的计算结果可能不是整数，并且由式(2-4)也得知，迭代

后的$x_{ij}(t+1)$可能是0、1以外的其它数值。为此，Kenndy引入模糊函数$Sig(x)$，定义如式(2-5)所示。

$$Sig(x)=\frac{1}{1+\exp(-x)} \tag{2-5}$$

这样，位置迭代公式(2-4)就变为

$$x_{ij}(t+1)=\begin{cases}0, & r_3 \geq Sig(v_{ij}(t+1)) \\ 1, & otherwise\end{cases} \tag{2-6}$$

其中，$r_3 \in [0,1]$为均匀分布的随机数。

在基本粒子群算法中，$v_{ij}(t)$表示粒子的飞行速度，能对$x_{ij}(t)$的方向和位置产生影响，而在离散粒子群中，$v_{ij}(t)$则表示粒子的每维分量取1的概率为$Sig(v_{ij}(t))$，取0的概率为$1-Sig(v_{ij}(t))$。

2.4 工作流挖掘

2.4.1 工作流技术

1993年，工作流技术的标准化组织——工作流管理联盟WfMC（Workflow Management Coalition）成立。WfMC对工作流的定义[17]：工作流是一类能够完全或者部分自动执行的经营过程，根据一系列过程规则，文档、信息或任务能够在不同的执行者之间传递、执行。因此，它通过建模企业业务过程，依据预先定义的业务逻辑关系来组织和协调企业各种活动的执行。

工作流的概念是在信息系统的建设中逐步形成的，它有一个从局部到整体、从初级到高级、从简单到复杂的发展过程[18]。

工作流的概念起源于生产组织和办公自动化领域。目的是通过将一个具体的工作分解成多个任务、角色，通过预定义的规则和过程，约束这些任务的执行和监控，以提高企业生产经营管理水平。目前，工作流的热点研究方向主要有：

（1）工作流建模及分析[19-21]。工作流模型是业务逻辑表示的基础，是业务过程的计算机化的表示。随着工作流过程的复杂化，工作流建模也越来越容易出

错，因此需要对建模后的模型进行分析：确认、验证和性能分析。确认是测试模型是否如预期一样执行。验证是模型正确性检验，发现模型中不合理之处，以减少出现逻辑错误的概率。性能分析主要通过时间、服务和资源利用率来评价模型的性能。

（2）工作流资源管理[22, 23]。工作流资源管理是工作流系统资源优化的主要手段。资源的合理建模、配置和调度能有效的改善工作流系统的性能。

（3）工作流安全性[24, 25]。工作流安全性研究主要集中在对访问控制模型的研究。一个安全的访问控制模型是工作流管理系统中重要的组成部分，也是当前工作流研究的重要内容。

（4）工作流过程优化[26, 27]。用户是根据自己的需求定义的工作流模型，可能会存在不优化的地方，因此需要采用优化方法对模型进行结构优化，从而减少工作流执行时间，提高整个工作流的执行效率。

（5）工作流挖掘[28]。工作流挖掘的目标是从信息系统的事件日志中自动发现过程模型。工作流挖掘的研究在国内起步较晚。

（6）工作流调度[29, 30]。通过工作流调度算法实现工作流任务执行过程中满足用户逻辑约束的最优资源调度策略，从而实现服务资源的最优利用。在网格环境下，服务资源分布在异构的网络中，这些分布着的服务资源的选择和调度对实现工作流调度高效性下显得尤为重要。

2.4.2 工作流挖掘技术

广义上，工作流挖掘是指信息系统的工作流知识发现，也称为过程挖掘或过程发现。工作流挖掘的最终目标是：通过对信息系统事件日志的挖掘，发现有关于工作流的知识，如模型知识、时间知识等。工作流挖掘的过程如图 2-9 所示。狭义上，工作流挖掘是指工作流模型挖掘。

工作流的生命周期包括 4 个阶段：（1）工作流建模；（2）工作流配置；（3）工作流执行；（4）工作流诊断。以往对工作流技术的研究主要集中在工作流建模和工作流配置，很少关注到工作流执行阶段的研究，更少有对工作流诊断的研究。工作流挖掘技术研究的重点放在工作流执行阶段和诊断阶段，通过分析工作流执

行阶段产生的事件日志，从而逆向支持工作流的建模与分析。

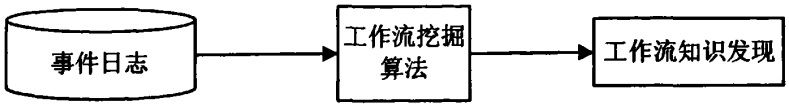


图 2-9 工作流挖掘过程

传统的工作流建模过程存在如下一些缺点：

（1）需要建模者拥有丰富的经验；

（2）建模过程需要与客户进行反复的沟通，这个过程花费的时间较多；

（3）建模的成本高；

（4）设计出的模型往往不能反映实际的业务过程；

（5）设计出的模型随着用户实际业务过程的变动也经常需要随之更改。

因此研究人员提出对模型进行挖掘而不是首先进行设计，并围绕工作流模型挖掘这个主题进行了大量的研究工作。工作流模型挖掘过程如图 2-10 所示。本文所提及的工作流挖掘除特殊说明外，均指狭义上的工作流挖掘即工作流模型挖掘。

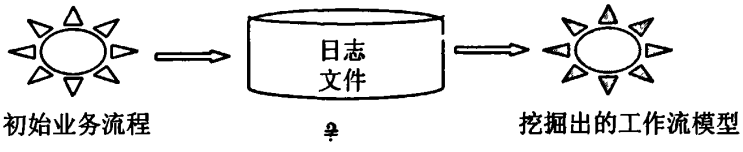


图 2-10 工作流模型挖掘过程

2.4.3 工作流挖掘国内外研究现状

文献[31]首次对软件工程中的软件过程进行挖掘，介绍了三种软件过程挖掘方法：神经网络法、纯算法、Markov 方法；将过程挖掘的概念引入到工作流管理系统中，并将研究成果在 IBM 工作流管理平台 MQSeries 上进行应用；文中利用有限状态机进行建模，但有限状态机不能建模并行活动。自此以工作流挖掘为主题的研究得到广泛地开展。

文献[32]是对文献[33]工作的延续和优化，提出活动的执行是有生命期的，将活动从准备到结束的时间段定义为活动的生命期。由于活动生命期的引入，并

发活动的判断从两个方面来考虑：（1）两个活动曾以两种先后顺序出现；（2）两个活动处于激活状态时的生命期出现重叠。只要以上两个条件满足其中一条，就可以判断这两个活动是并发活动。在日志具有不完整性的情况下，如果只是用条件（1）来判断并发活动就容易导致建模错误。而引入生命期的概念后，并发的判断就更加全面，也更合理地反映工作流的实际情况。

文献[34]采用基于块结构的建模语言进行建模，定义每个工作流模型由一系列的嵌套块构成。这种建模方法的优点在于建模语言与过程几何有关，过程几何语义定义准确，具有模块化和可扩展性。

文献[35]提出了 *α* 算法，基于符合完整性定义的事件日志，输出基于 Petri 网表示的库所/变迁（P/T）网。*α* 算法能挖掘所有结构化工作流网（Structured Workflow Net，SWF-net）。但是，*α* 算法有其局限性，主要表现在不能正确挖掘工作流中的重复任务和非自由选择结构，另外 *α* 算法不能处理短循环（单循环和双循环）以及有不完整数据和有噪声数据的情况。

文献[36]通过对日志进行预处理，识别出日志中的单循环和双循环日志，并删除循环日志。找出单双循环后，再对处理后的日志运用 *α* 算法进行挖掘，从而解决了直接运用 *α* 算法不能处理短循环的问题。

文献[37，38]在消除噪声的基础上进行了建模研究，提出了处理噪声并进行建模的三步：（1）构造依赖/频繁表（D/F-table）；（2）通过依赖/频繁表挖掘出活动关系表（R-table）；（3）最后通过关系表重构出 WF-net。消除噪声的过程中使用了阈值，但是阈值的设定是凭借经验数据设定的。这显然对于没有经验的人而言不太适合。

文献[39]提出基于机器学习的方法得出最优的阈值，从而改进了噪声消除方法。文献[40]完全采用贝叶斯方法来挖掘带噪声的日志。除了对结构化工作流模型进行挖掘，现实中有些系统的过程是非定制的，而是在系统运行过程中由用户的行为动态确定的，如 Caramba[41] 就是允许过程非定制的协同系统。文献[42]利用 TeamLog 对此类非定制过程进行了挖掘研究。

对于工作流挖掘的研究前期主要集中在过程的行为方面，但缺乏对事务工作流的性能挖掘。文献[43]提出了挖掘算法，该算法可以挖掘工作流模型，而且通过日志挖掘可以改进事务行为。文献[44]通过日志挖掘工作流的事务属性，找出

系统失败的原因，并使用一系列的规则帮助系统进行恢复。该文首次从工作流挖掘的角度，对系统进行事务恢复研究。

文献[45]将工作流视为队列系统，分析流程到达时间对流量进行建模；分析流程服务时间，挖掘影响服务时间的因素；分析等待时间，对客户等待容忍限度进行建模。文献[46]引入流程经理概念，实现基于流程分解的内部角色识别，解决角色的层次性划分问题；对角色间的活动依赖进行了刻画，从而实现对角色交互行为的挖掘。文献[47]对日志中的决策点进行挖掘，修改挖掘出的工作流模型，使得决策点尽可能定位到前期，这样可以减少不确定的活动并识别出多余的活动，从而对原模型进行了有效的改进，提高模型的平均执行时间。文献[48]利用机器学习的方法对日志进行决策挖掘，发现数据属性如何影响案例的路由选择，并设计实现了决策挖掘器，从而改进路由选择，提高模型的质量。

表 2-1[49]列举了几种经典的工作流模型挖掘工具，从工作流挖掘的数据结构、时间性能、并行性等方面进行对比，几种工具各有优劣。

表 2-1 工作流模型挖掘工具性能比较

功能	EmiT[50]	Little Thumb[51]	InWoLvE[52]	Process Miner[53]
数据结构	Graph	Graph	Graph	Block
时间分析	√	×	×	×
简单并行	√	√	√	√
非自由选择	×	×	×	×
简单循环	√	√	√	√
任意循环	√	√	×	×
隐藏任务	×	×	×	×
重复任务	×	×	√	×
噪声	×	√	√	×

2.5 工作流时间性能分析

Jind 等人[54]认为关键路径的处理时间决定了整个工作流的处理时间，因此可以用关键路径的时间性能来评价整个工作流的时间性能。林闯等人[55]提出了由任意多个变迁组成的串联、并联、选择和循环结构的性能等价公式，给出了定量分析可由这四种结构构成的工作流的时间性能的通用方法；李建国等人[56]则假定并行结构之间活动的等待时间为零，从而对自由选择 Petri 网表示的工作流模型的

时间性能进行了计算。姜浩等人[57]提出了一种基于扩展时间 Petri 网的工作流时间计算和分析方法，使用保持响应时间和分布概率不变的网变换方法，对工作流模型进行简化，从而得到工作流模型的时间性能指标。Pan 等人[58]在模糊时态工作流网的基础上提出了扩展的模糊时态工作流网，并分析了该网的时间性能。Jiang[59]提出了离散随机 Petri 网模型以及时间性能计算方法。

2.6 工作流调度

2.6.1 多目标优化

多目标优化问题的本质在于，在很多情况下，各个子目标可能是相互冲突的，即一个子目标性能的改善有可能会引起另一个子目标性能的下降，要使得多个目标同时达到最优值是不可能的，因而只能在各个子目标中进行协调和折衷处理，使各个子目标都尽可能达到最优。

权重法是求解多目标优化的经典算法。对于一个多目标优化问题，若给其各个子目标函数 $f_i(x)$，$(i=1, 2, \cdots, n)$赋予不同的权重 $\omega_i(i=1, 2, \cdots, n)$，其中 ω_i 的大小代表对应子目标 $f_i(x)$在多目标优化问题中的重要程度。各个子目标函数的线性加权和可表示为：

$$u(f(x))=\sum_{i=1}^{n}\omega_i f_i(x)，\text{其中}\sum_{i=1}^{n}\omega_i=1 \tag{2-7}$$

这样多目标优化问题便转化为单目标优化问题，求出的解则是问题的 Pareto 最优解（即不存在比其更优的解）。

2.6.2 工作流调度国内外研究现状

调度是工作流管理系统的核心问题，优化的调度有利于改善整个工作流系统的性能，如时间、费用等。调度问题是经典的多目标优化问题之一，一直是优化领域的研究热点。很多学者针对不同问题领域提出了各种启发式优化算法和智能

优化算法。

Buyya 等人[60, 61]提出了时间、费用（预算）约束的启发式调度算法和网格资源管理的经济模型；文献[62，63]分别采用扩展 Sufferage 启发式方法和 Min-Min 启发式算法对网格中资源的调度进行优化；文献[64]针对工作流调度中存在信任机制与调度机制分离的缺陷，提出了基于信任关系的工作流 QoS 调度方法；文献[65]提出一种新的分布式工作流负载平衡调度算法，解决单点引擎负载过重的问题。

文献[66]对 T-RAG 描述的网格任务调度进行优化，提出网格任务调度的粒子群算法；文献[67]对基于时间和费用约束的网格中任务调度采用粒子群算法进行了优化研究；文献[68，69]分别采用离散粒子群算法和模糊粒子群算法对网格中工作调度进行了优化；文献[70]针对网格环境下服务工作流调度提出了极值扰动的混合粒子群算法。文献[71-74]采用遗传算法对网格工作流调度和网格资源调度进行了研究。文献[75，76]采用蚁群算法分别对网格环境下作业调度和网格工作流调度进行了优化。

第3章 可解决多种复杂任务的工作流挖掘

3.1 引言

目前，工作流技术在 CIMS 中得到了广泛的应用，其中企业建模[77]是工作流技术应用的重要领域，许多学者对其进行了深入地研究[78-80]。工作流挖掘作为一种重要的企业建模方法，能从信息系统的事件日志中发现关于业务流程的结构化过程。工作流挖掘技术[81]能帮助企业实现业务流程的建模和再造，极大地提高企业的市场竞争力。

Aalst 等人提出了挖掘工作流模型的 α 算法[35]并证明了 α 算法能准确挖掘的模型种类。α 算法分析了任务之间的关联关系，分别是后继（>w）、因果（→w）、并行（‖w）和不相关（#w）。由于 α 算法采用集合来定义任务之间的关联关系，因此在判定重复任务时，算法将这些重复任务视为同一个任务，往往会导致挖掘出错误的过程模型。文献[82]提出了判定重复任务的启发式规则，扩展了 α 算法，使其能发现不含有循环任务的日志中的重复任务，然而没有考虑到在含有循环任务的日志中循环任务和重复任务的发现。

当业务流程含有循环任务、重复任务时，产生的事件日志中会出现一个任务多次出现的情况。为了判定多次出现的任务来自循环任务、重复任务还是同一任务，本章改进了 α 算法，提出了 τ 算法。首先通过启发式规则判定出事件日志中来自同一个轨迹的循环任务和重复任务以及来自不同轨迹的同一任务，然后运用 τ 算法从事件日志中提取出 WF－net。该算法同样适用于不含有循环任务的事件日志。

3.2 事件日志完备性及任务定义

3.2.1 工作流日志和工作流轨迹

工作流挖掘的目的是为了从事务日志抽取出工作流模型。在这些事务日志中包含很多信息，文中为了工作流挖掘的需要，我们只关心工作流实例编号（ID）和任务名称。表 3-1 给出了一个事件日志的例子。

表 3-2 中给出了从表 3-1 的事件日志中整理出来的 2 个工作流实例的任务轨迹。定义 3-1 给出了事件日志和任务轨迹的定义，其中事件日志的定义参考了文献[35]。

定义 3-1 （事件日志和任务轨迹）. 设 T 是任务集合，T^*是由 T 中 0 个或多个任务组成的所有序列的集合。$W \subseteq T^*$ 称为工作流日志；$s \in T^*$称为任务轨迹。

表 3-1 事件日志

实例 ID	任务名称
1	*A*
2	*A*
1	*B*
1	*C*
2	*C*
2	*B*
2	*D*
1	*D*

表 3-2 任务轨迹

轨迹 ID	任务轨迹
s_1	*ABCD*
s_2	*ACBD*

3.2.2 事件日志完备性

对于一个实际的业务流程，不完整的事件日志会导致挖掘出的过程模型有误，如循环结构的丢失等。文中所提及的事件日志若无特殊说明，都是指具有完备性的事件日志。为此下面给出事件日志完备性的定义。

定义 3-2 （事件日志完备性）. 设 W 是由任务集 T 构成的业务流程产生的事件日志，s 是业务流程一次执行过程产生的任务轨迹，$s \in W$。W 具有日志完备性，当且仅当：

（1）对于由任务集 T 构成的业务流程产生的任意事件日志 W'，有 $W' \subseteq W$；

（2）对于任意 $t \in T$，都存在 $s \in W$ 使得 $t \in s$；

（3）对于事件日志 W'，若 W'中有 $s'_1=\cdots xy\cdots$，$s'_2=\cdots xay\cdots$，$s'_3=\cdots xaay\cdots$，且 $a, x, y \in T$，则一定存在 $s_1, s_2, s_3 \in W$ 且 $s_1=\cdots xy\cdots$，$s_2=\cdots xay\cdots$，$s_3=\cdots xaay\cdots$；

（4）对于事件日志 W'，若 W'中有 $s'_1=\cdots xay\cdots$，$s'_2=\cdots xabay\cdots$，且 a，b，x，$y \in T$，$a \neq b$，则一定存在 s_1，$s_2 \in W$ 且 $s_1=\cdots xay\cdots$，$s_2=\cdots xabay\cdots$。

3.2.3 任务定义

在任务轨迹 s 中任务 t 的个数记为 $\#t(s)$。轨迹 s 中的第 1 个 t 记为 $t(s, 1)$，第 2 个 t 记为 $t(s, 2)$，第 i 个 t 记为 $t(s, i)$，$i=1, 2, \cdots, \#t(s)$。

定义 3-3 （单循环任务和双循环任务）. 设 W 是由任务集 T 构成的业务流程产生的事件日志，

（1）若存在 $t \in T$，当且仅当存在 $s_1=\cdots xy\cdots$，$s_2=\cdots xty\cdots$，$s_3=\cdots xtty\cdots$ 且 s_1，s_2，$s_3 \in W$，则称 t 为单循环任务。$Loop1(t)$表示 t 的单循环任务集合。

（2）任务轨迹 s 的子串记为 $sub(s)$。$len(x)$表示字符串 x 的长度。设 $s_1, s_2 \in W$，当且仅当存在 $s=sub(s_1)$，$s'=sub(s_2)$，$len(s)>=1$，$len(s')>=1$，$s \neq s'$，$s_1=\cdots xsy\cdots$ 且 $s_2=\cdots xss'sy\cdots$，则称 s 为双循环任务。$Loop2(s)$表示任务串 s 的双循环任务集合，这里的双循环是广义的双循环，包括循环任务大于 2 的所有循环。

定义 3-4 （重复任务）. 设 W 是由任务集 T 构成的业务流程产生的事件日志。当且仅当存在 $s \in W$，$s=t_1\cdots t_i\cdots t_j\cdots t_m$，$t_i=t_j$，$t_i \notin Loop1(t_i)$且 $t_i \notin Loop2(t_i)$，则称 t_i 为重复任务。重复任务 t 的集合记为 $D(t)$。

定义 3-5 （同一任务）. 设 W 是由任务集 T 构成的业务流程产生的事件日志。当且仅当存在 s_1，$s_2 \in W$，$s_1=t_1t_2\cdots t_i\cdots t_m$，$s_2=t'_1t'_2\cdots t'_j\cdots t'_n$，$i=1, 2, \cdots, m$ 且 $j=1, 2, \cdots, n$，若 t_i、t'_j 由业务流程中同一位置的同一活动产生，则称 t_i、t'_j 为同一任务。单循环任务和双循环任务是特殊的同一任务。同一任务 t 的集合记

为 $S(t)$。

定义 3-6 （被包围任务）. 设 W 是由任务集 T 构成的业务流程产生的事件日志。$s\in W$，x，$y\in T$，$s\in T\vee T^*$。若 s 中含有子串 “$x\cdots s\cdots y$”，则称 s 被 x 和 y 所包围，记为 $s\Theta(s,[x,y])$。被 x 和 y 所包围着的 s 记为 $s[x,y]$；字符串 c 中的任务 s 记为 $s[c]$。

定义 3-7 （前驱任务和后继任务）. 设 W 是由任务集 T 构成的业务流程产生的事件日志。$s\in W$，t，$t'\in T\vee T^*$。若存在 $s=\cdots t't\cdots$，则称 t' 为轨迹 s 中 t 的前驱任务，记为 $\Box t(s,n)$；称 t 为轨迹 s 中 t' 的后继任务，记为 $t'\Box(s,n)$，n 为任务在轨迹中出现的次序号。

以图 3-1 中的工作流网 $N_{3.1}$ 为例，说明文中关于任务的概念。M 为工作流网 $N_{3.1}$ 产生的事件日志，s_1，s_2，$s_3\in M$，$s_1=XCXY$，$s_2=XACXY$，$s_3=XAACXBXY$。根据任务的定义，可得出多种任务如下：

（1）$A(s_2,1)$，$A(s_3,1)$，$A(s_3,2)\in Loop1(A)$；

（2）$X(s_3,2)$，$X(s_3,3)\in Loop2(X)$；

（3）$X(s_1,1)$，$X(s_1,2)\in D(X)$；

（4）$X(s_1,2)$，$X(s_2,2)$，$X(s_3,2)\in S(X)$；

（5）$C(s_1,1)\Theta(s_1,[X(s_1,1),X(s_1,2)])$；

（6）$\Box A(s_2,1)=X(s_2,1)$；

$A\Box(s_2,1)=C(s_2,1)$。

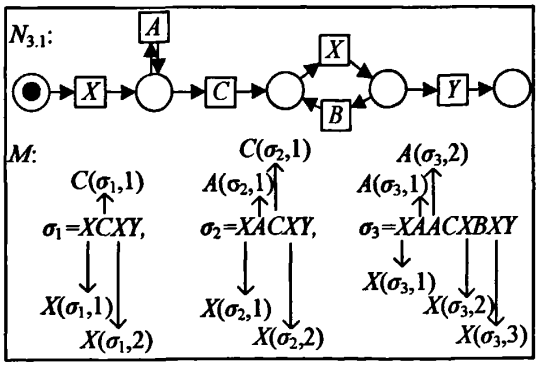


图 3-1 工作流网 $N_{3.1}$、事件日志 M 及任务轨迹中的复杂任务

3.3 τ挖掘算法

3.3.1 关联关系

在进行基于事件日志的工作流挖掘前，首先应分析任务之间的关联关系。文献[35] 关于关联关系的定义没有考虑到单循环任务及双循环任务，导致不能正确挖掘含有单循环任务或者双循环任务的工作流模型。因此文中对文献[35]的关联关系的定义进行改进，重新给出了如下的关联关系的定义。

定义 3-8 （关联关系）. 设 W 是由任务集 T 构成的业务流程产生的事件日志，a，$b \in T$：

（1）$a>b$，当且仅当存在一个任务轨迹 $s \in W$，$s=t_1t_2t_3\cdots t_n$，使得 $t_i=a$，$t_{i+1}=b$，$i \in [1, n-1]$；

（2）$a \propto b$，当且仅当存在任务轨迹 s，$s' \in W$，$s=\cdots xay\cdots$，$s'=\cdots xabay\cdots$，$a \neq b$；称 a，b 为双循环任务，$Loop2(a)$，$Loop2(b)$分别表示 a，b 的双循环任务集合；

（3）$a \rightarrow b$，当且仅当 $a>b$ 且 $b \not> a$ 或 $a \propto b$；

（4）$a \| b$，当且仅当任务轨迹 s，$s' \in W$，在 s 中 $a>b$，而在 s' 中 $b>a$，且 $s \neq s'$；

（5）$a \# b$，当且仅当 $a \not> b$ 且 $b \not> a$；

（6）$a8$，当且仅当存在 s'_1，s'_2，$s'_3 \in W$，$s'_1=\cdots xy\cdots$，$s'_2=\cdots xay\cdots$，$s'_3=\cdots xaay\cdots$；称 a 为单循环任务，$Loop1(a)$表示 a 的单循环任务集合；

（7）$a \oplus b$，当且仅当存在 s，$s' \in W$，$s=t_1t_2t_3\cdots t_n$，$s'=t'_1t'_2t'_3\cdots t'_{n'}$，使得 $t_i=a$，$t'_j=b$，$a \neq b$，$t_{i-1}=t'_{j-1}$ 且 $t_{i+1}=t'_{j+1}$。

工作流模型基本结构包括串行、并行、选择、循环。重新定义的关联关系涵盖了这四种基本结构，并且细分了循环结构，将循环结构分为单循环和双循环结构，从而能更准确的挖掘实际工作流模型。

3.3.2 判定同一任务及重复任务

属性 3-1. 设 W 是由任务集 T 构成的业务流程产生的事件日志。对于 s，$s' \in W$，t，$t' \in T$，i，$j \in [1, n]$，若 $\square t(s, i)=\square t(s', j)$ 且 $t\square(s, i)=t\square(s', j)$，则{$t(s$，

$i)$，$t(s', j)\}\in S(t)$。

该属性容易得证。

定理 3-1. 设 W 是由任务集 T 构成的业务流程产生的事件日志。$s\in W$，$t\in T$，$t\notin Loop2(t)$且 $len(s)=n(n>1)$。若存在 i，$j\in[1, n]$，$i\neq j$，$\square t(s, i)\neq\square t(s, j)$且 $t\square(s, i)\neq t\square(s, j)$，则$\{t(s, i), t(s, j)\}\in D(t)$。

证明. 用反证法证明。假设$\{t(s,i), t(s,j)\}\notin D(t)$。由于 $t(s,i), t(s,j)\notin Loop2(t)$，容易得知 $t(s, i)$，$t(s, j)\in S(t)$，因而有$\square t(s, i)=\square t(s, j)$，$t\square(s, i)=t\square(s, j)$且 $i=j$，这与已知条件 $i\neq j$，$\square t(s, i)\neq\square t(s, j)$且 $t\square(s, i)\neq t\square(s, j)$相矛盾，所以$\{t(s, i), t(s, j)\}\in D(t)$。

证毕。

定理 3-1 保证了在同一个任务轨迹中重复任务的发现。

定理 3-2. 设 W 是由任务集 T 构成的业务流程产生的事件日志。$s, s', s''\in W$，A，$A'\in sub(s)\vee sub(s'')$，$X$，$Y\in T$，那么

（1）若存在 $A\|A'$，即 $AA'\Theta(s, [X, Y])$且 $A'A\Theta(s', [X, Y])$，则

- $\{A[XAA'Y])$，$A[XA'AY]\}\in S(A)$；
- $\{A'[XAA'Y]$，$A'[XA'AY]\}\in S(A')$；
- $\{X[XAA'Y]$，$X[XA'AY]\}\in S(X)$；
- $\{Y[XAA'Y]$，$Y[XA'AY]\}\in S(Y)$。

（2）若存在 $A\oplus A'$，即 $A\Theta(s, [X, Y])$且 $A'\Theta(s', [X, Y])$，则

- $\{X[XAY]$，$X[XA'Y]\}\in S(X)$；
- $\{Y[XAY]$，$Y[XA'Y]\}\in S(Y)$。

（3）若存在 $A\propto A'$，即 $A\Theta(s, [X, Y])$且 $AA'A\Theta(s', [X, Y])$，则

- $\{A[XAY]$，$A([XAA'AY]$，$1)$，$A([XAA'AY]$，$2)\}\in S(A)$；
- $\{X[XAY]$，$X[XAA'AY]\}\in S(X)$；
- $\{Y[XAY]$，$Y[XAA'AY]\}\in S(Y)$。

（4）若存在 $A8$，即 $\emptyset\Theta(s, [X, Y])$，$A\Theta(s', [X, Y])$且 $AA\Theta(s'', [X, Y])$，则

- $\{A[XAY]$，$A([XAAY]$，$1)$ ，$A([XAAY]$，$2)\}\in S(A)$；
- $\{X[XY]$，$X[XAY]$，$X[XAAY]\}\in S(X)$；

- $\{Y[XY], Y[XAY], Y[XAAY]\} \in S(Y)$。

证明.（1）由 $s=\cdots XAA'Y\cdots$，$s'=\cdots XA'AY\cdots$，得知 $X>A$，$A>A'$，$A'>Y$，$X>A'$，$A'>A$，$A>Y$。由于 $A \| A'$，所以有 $X\rightarrow A$，$A'\rightarrow Y$，$X\rightarrow A'$，$A\rightarrow Y$。在重新定义关联关系的基础上运用 a 算法，可得图 3-2 中的工作流网 $N_{3.2.1}$，从而可得如下结论：

- $\{A[XAA'Y]), A[XA'AY]\} \in S(A)$;
- $\{A'[XAA'Y], A'[XA'AY]\} \in S(A')$;
- $\{X[XAA'Y], X[XA'AY]\} \in S(X)$;
- $\{Y[XAA'Y], Y[XA'AY]\} \in S(Y)$。

（2）已知 $A \oplus A'$，即 $A\Theta(s, [X, Y])$，$A'\Theta(s', [X, Y])$，类似地运用 a 算法，我们可得到图 3-2 中的工作流网 $N_{3.2.2}$。由图可得出如下结论：

- $\{X[XAY], X[XA'Y]\} \in S(X)$;
- $\{Y[XAY], Y[XA'Y]\} \in S(Y)$。

（3）已知 $A \propto A'$，即 $A\Theta(s, [X, Y])$，$AA'A\Theta(s', [X, Y])$，类似地运用 a 算法，我们可得到图 3-2 中的工作流网 $N_{3.2.3}$。由图可得出如下结论：

- $\{A[XAY], A([XAA'AY], 1), A([XAA'AY], 2)\} \in S(A)$;
- $A'[XAA'AY] \in S(A')$; $\{X[XAY], X[XAA'AY]\} \in S(X)$;
- $\{Y[XAY], Y[XAA'AY]\} \in S(Y)$。

（4）已知 $A8$，即 $\emptyset\Theta(s, [X, Y])$，$A\Theta(s', [X, Y])$且 $AA\Theta(s'', [X, Y])$ 类似地运用 a 算法，我们可得到图 3-2 中的工作流网 $N_{3.2.4}$。由图可得出如下结论：

- $\{A[XAY], A([XAAY], 1), A([XAAY], 2)\} \in S(A)$;
- $\{X[XY], X[XAY], X[XAAY]\} \in S(X)$;
- $\{Y[XY], Y[XAY], Y[XAAY]\} \in S(X)$。

证毕。

定理 3-3. 设 W 是由任务集 T 构成的业务流程产生的事件日志。$t \in T$，s，$s' \in W$。若在任务轨迹 s 中任务 t、t'存在关系 $tRt'(R \in \{>, \|, \oplus, \propto, 8\})$，在任务轨迹 s'中任务 t、t'存在关系 $tR't'(R' \in \{>, \|, \oplus, \propto, 8\})$且 $R \neq R'$，则

$\{t(s, [tRt']), t(s', [tR't'])\} \in D(t)$;

$\{t'(s, [tRt']), t'(s', [tR't'])\} \in D(t)$。

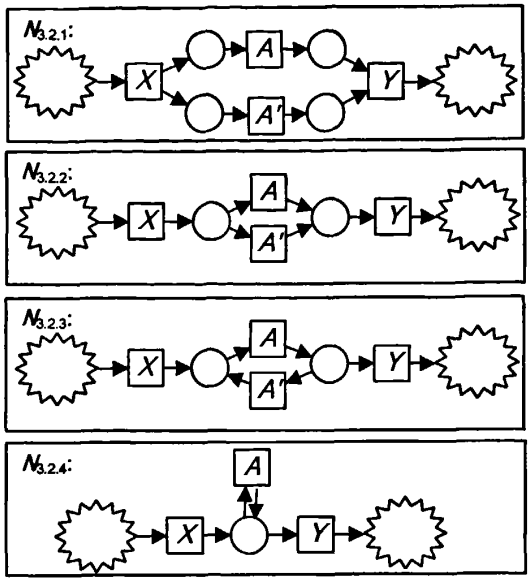


图 3-2 并行、选择以及循环结构工作流网

证明. 用反证法证明。假设$\{t(s, [tRt']), t(s', [tR't'])\}\notin D(t)(R\neq R')$且$\{t'(s, [tRt']), t'(s', [tR't'])\}\notin D(t)(R\neq R')$，可得$\{t[(s, [tRt']), t(s', [tR't'])\}\in S(t)$，$\{t'(s, [tRt']), t'(s', [tR't'])\}\in S(t)$，运用定理 3-2，可得 $R=R'$，这与题设 $R\neq R'$相矛盾，因此可得出如下结论：

- $\{t(s, [tRt']), t(s', [tR't'])\}\in D(t)$;
- $\{t'(s, [tRt']), t'(s', [tR't'])\}\in D(t)$。

证毕。

3.3.3 τ挖掘算法步骤

工作流挖掘过程主要有三步，分别是预处理、处理和后期处理。在预处理阶段，在事件日志完备的前提下，识别出重复任务，循环任务和同一任务。处理阶段则是挖掘算法的主要组成部分，在该阶段重命名重复任务，删除单循环任务并且在重新定义任务之间的关联规则的基础上，挖掘出工作流模型，这是 τ 算法的主要部分。在后期处理阶段，主要完成工作流模型的微调工作：添加预处理阶段删除的单循环任务以及与库所之间的关联，并将重命名后的重复任务名还原为原任务名。 τ 算法的过程描述如下：

设 W 是由任务集 T 构成的业务流程产生的事件日志。基于 W 进行挖掘后得到的工作流网记为 $\tau(W)$，具体的挖掘过程如下：

步骤 1：根据定义 3-3 识别单循环任务及双循环任务，在此基础上再运用属性 3-1 和定理 3-1，3-2，3-3 识别出同一任务和重复任务，并将重复任务 t 加入 $D(t)$；

步骤 2：对 $D(t)$中的任务通过添加下标重命名，更新重命名后的日志，得到新的事件日志 W^{+DT}；

步骤 3：将步骤 1 发现的单循环任务 t，$t\Theta[m, n]$，加入 $Loop1(t)$；

步骤 4：删除 W^{+DT} 中发现的重复任务 $Loop1(t)$后，得到新的日志 $W^{+DT\text{-}Loop1}$；

步骤 5：运用改进后的 a 算法，得到初步的工作流网 $a(W^{+DT\text{-}Loop1}) = N^{+DT\text{-}Loop1} = (P^{+DT\text{-}Loop1}, T^{+DT\text{-}Loop1}, F^{+DT\text{-}Loop1})$；

步骤 6：将单循环任务加入 $T^{+DT\text{-}Loop1}$，得到新的任务集合 T^{+DT}，$P^{+DT}=P^{+DT\text{-}Loop1}$，添加任务变迁和库所之间的关联得，$F^{+DT}=F^{+DT\text{-}Loop1}\cup\{(t, p(m, n), (p(m, n), t))\}$，$p(m, n)$表示连接变迁 m 和 n 的库所，从而得到了新的工作流网 $N^{+DT}=(P^{+DT}, T^{+DT}, F^{+DT})$；

步骤 7：最后删除给重复任务添加的下标，得出挖掘后的工作流网 $\tau(W)=N=(P, T, F)$。

τ 算法的执行过程如下：首先检查事件日志并识别出单循环任务、双循环任务、重复任务和同一任务（步骤 1）。在步骤 2 中，将识别出的重复任务添加下标进行重新命名。单循环任务加入单循环任务集（步骤 3）并在步骤 4 中被删除。在步骤 5 中，运用 a 算法对预处理后的事件日志进行挖掘得到工作流网 $N^{+DT\text{-}Loop1}$。步骤 6 在工作流网 $N^{+DT\text{-}Loop1}$ 上添加步骤 4 中删除的单循环任务以及任务对应的变迁和库所之间的关联，得到新的工作流网 N^{+DT}。最后，删除在步骤 2 中给重复任务添加的下标，将任务名恢复为原来的任务名，从而得到最终的工作流模型。

3.4 实例验证及结果分析

3.4.1 实例验证

为了验证 3.3.2 节提出的启发式判定规则及 τ 挖掘算法的可行性，本节以三个来自某企业业务流程的事件日志为例，进行实例验证。

某企业的三个业务流程分别产生如表 3-3 至表 3-5 的第一、二列所示的事件日志。(为了日志挖掘的需要，仅提取了日志中的轨迹 ID 和不重复的任务轨迹信息)。

表 3-3 事件日志 $L_{3.1}$

轨迹 ID	任务轨迹	重名后的任务轨迹
s_1	*XADBY*	*XADBY*
s_2	*XAEADBY*	*XAEADBY*
s_3	*XADBEBY*	*XADBEBY*
s_4	*XAEADBEBY*	XAE_1ADBE_2BY

对表 3-3 中的任务轨迹，运用定理 3-1 可判定出$\{E(s_4, 1), E(s_4, 2)\}\in D(E)$。

从任务轨迹中识别出重复任务后，对重复任务按顺序添加 1、2…下标进行重命名。表 3-3 列出了轨迹 ID、原始任务轨迹以及重命名后的任务轨迹。

表 3-4 事件日志 $L_{3.2}$

轨迹 ID	任务轨迹	重名后的任务轨迹
s_1	*XABCE*	XA_1B_1CE
s_2	*XACBE*	XA_1CB_1E
s_3	*XDAF*	XDA_2F
s_4	*XDBAF*	XDB_2A_2F
s_5	*XDBBAF*	$XDB_2B_2A_2F$

由表 3-4 中的任务轨迹，可得出如下结论：

(1) $\{B(s_1, 1), B(s_2, 1)\}\in S(B)$；(定理 3-2)

(2) $\{B(s_4, 1), B(s_5, 1), B(s_5, 2)\}\in S(B)$；(定理 3-2)

(3) $\{D(s_3, 1), D(s_4, 1), D(s_5, 1)\}\in S(D)$；(定理 3-2)

(4) $\{A(s_3, 1), A(s_4, 1), A(s_5, 1)\}\in S(A)$；(定理 3-2)

(5) $\{A(s_1, 1), A(s_3, 1)\}\in D(A)$；(定理 3-3)

(6)$\{B(s_1, 1), B(s_4, 1)\}\in D(B)$；(定理 3-3)

类似地，可发现表 3-5 中的重复任务，并进行重命名，得出重命名后的任务

轨迹，如表 3-5 所示。

表 3-5 事件日志 $L_{3.3}$

轨迹 ID	任务轨迹	重名后的任务轨迹
s_1	*XAXY*	X_1AX_2Y
s_2	*XABY*	X_1AB_1Y
s_3	*XDBF*	X_1DB_2F
s_4	*XDBXBF*	$X_1DB_2X_3B_2F$

运用 τ 算法对事件日志 $L_{3.1}$-$L_{3.2}$ 进行挖掘后分别得出工作流网 $N_{3.3.1}$、$N_{3.3.2}$ 和 $N_{3.3.3}$，如图 3-3 所示。

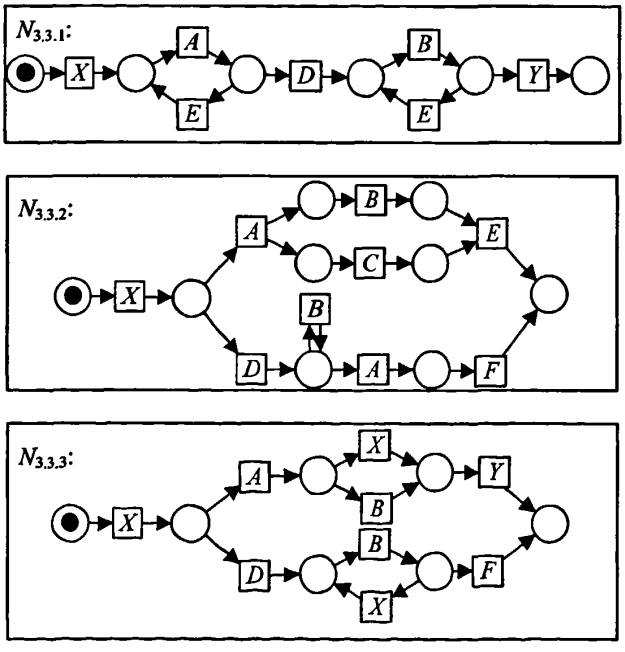


图 3-3 运用 τ 算法挖掘后的工作流网

3.4.2 结果分析

为验证挖掘算法的正确性，首先使用 CPN Tools 进行建模、仿真，然后使用 $ProM_{import}$ 将仿真后生成的任务轨迹进行合并，得到业务流程的事件日志。通过对比仿真后生成的日志以及用于工作流挖掘的原始日志，评价挖掘算法的有效

性。

以工作流网 $N_{3.3.1}$ 以及某制造业业务流程为例，通过 CPN Tools 对其进行建模分别如图 3-4 和图 3-5 所示。为了扩展 CP-net 使其仿真生成含有 1000 个任务轨迹的 MXML 格式的事件日志，在建模时增加了填充色为黑色的产生器变迁并使用 ML 脚本给每个变迁增加了输入、输出和动作标注。

使用 $\text{ProM}_{\text{import}}$ 合并仿真后生成的 1000 个任务轨迹，分别产生如图 3-6 和图 3-7 所示的 MXML 格式的事件日志。图 3-6 和图 3-7 分别显示了工作流网 $N_{3.3.1}$ 产生的仿真日志中的前 3 个过程实例轨迹：*XADBY*，*XAEADBY*，*XADBEBY* 和某制造业业务流程产生的仿真日志中的前 4 个过程实例轨迹，分别为 *XABICHY*，*XEDBBGFY*，*XACBIHY*，*XEDGBFY*。对比仿真日志和用于工作流挖掘的原始日志，得出两者具有等价性，从而验证了挖掘结果的正确性。

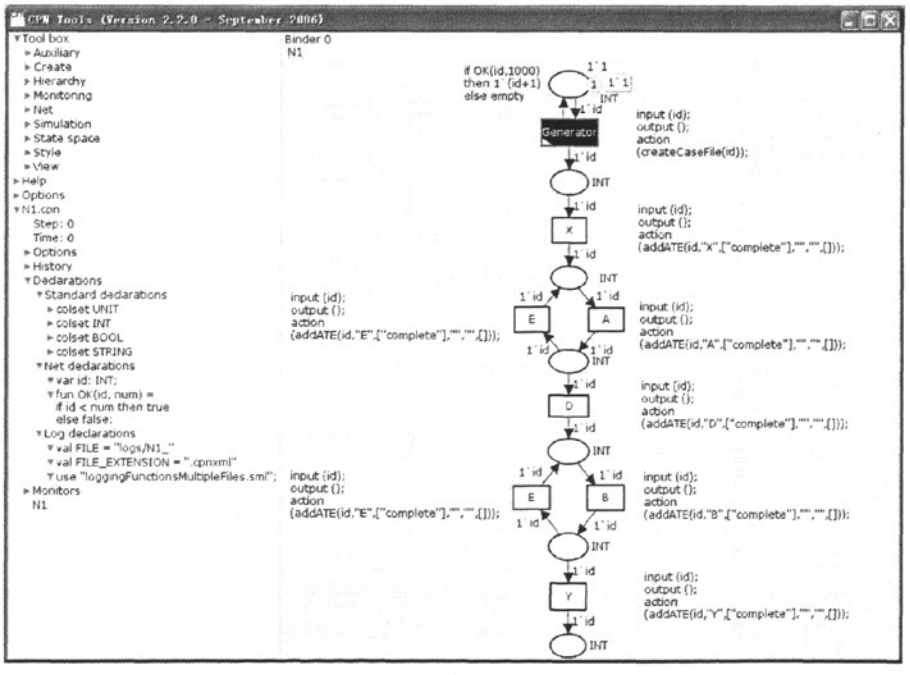


图 3-4　CPN Tools 建模工作流模型 $N_{3.3.1}$

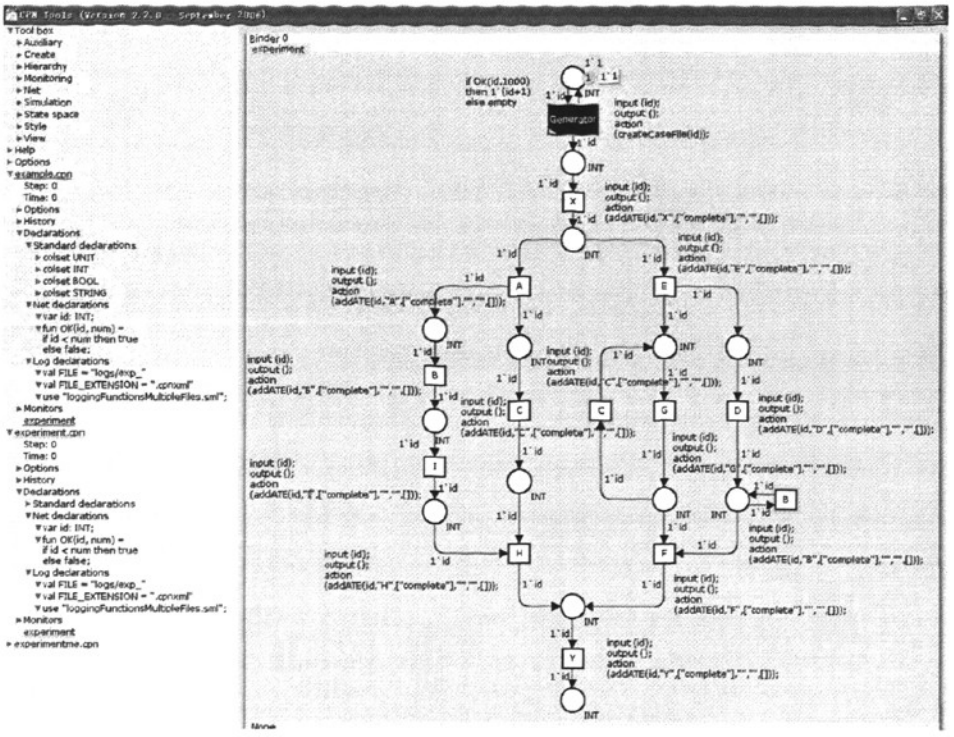


图 3-5 某制造业业务流程

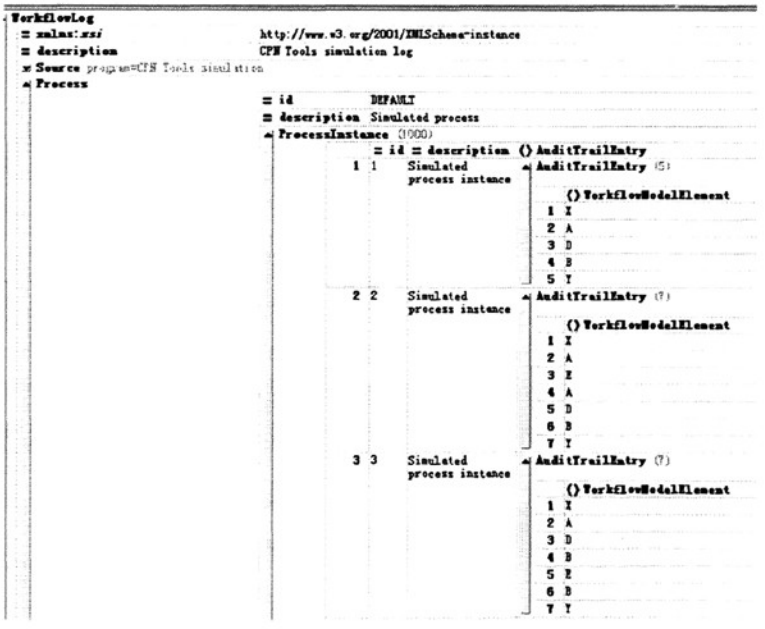


图 3-6 $N_{3.3.1}$ 使用 $\text{ProM}_{\text{import}}$ 合并后的 MXML 格式的部分日志

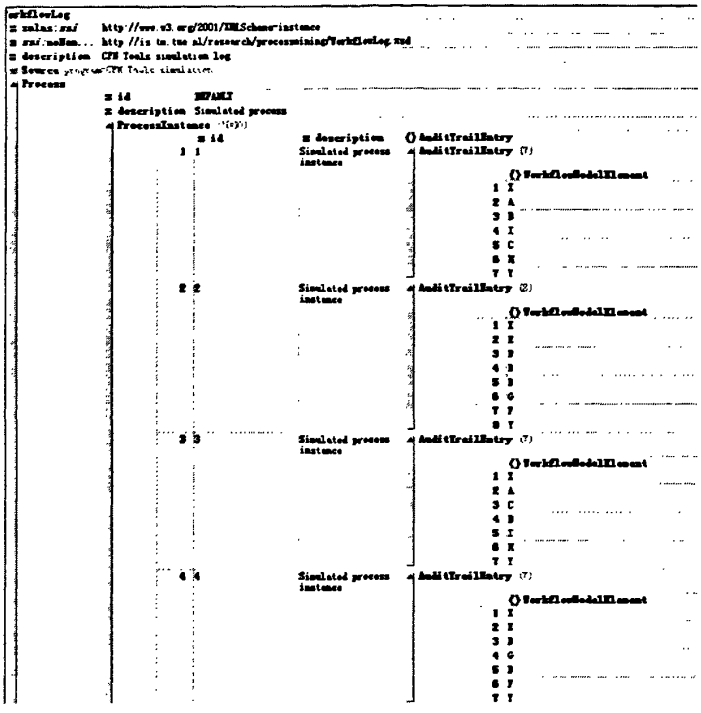


图 3-7 图 3-5 的模型使用 $ProM_{import}$ 合并后的 MXML 格式的部分日志

表 3-6 列举了针对不同事件日志的挖掘结果。结果表明 α 算法不能发现单循环，双循环任务和重复任务，α^{**}算法不能发现同时含有循环任务和重复任务的工作流模型，而 t 算法能发现日志中所有的活动。实验结果进一步说明 τ 算法能正确的挖掘含有循环和重复任务的事件日志。

表 3-6 三种工作流挖掘算法的实验比较

事件日志	不能正确挖掘的活动		
	α 算法[35]	α^{**}算法[82]	τ 算法
$L_{3.1}$	A、B、E	A、B	$\emptyset$
$L_{3.2}$	B	B	$\emptyset$
$L_{3.3}$	B、X	B、X	$\emptyset$
图 3-5 产生的日志	B、C	B、C	$\emptyset$

3.5 本章小结

本章对含有循环任务和重复任务的事件日志进行了研究，提出发现循环、重

复和同一任务的启发式规则，并且给出证明；改进了 α 算法的关联关系定义，在此基础上提出了 τ 算法。实例验证表明该方法是有效的，对循环、重复和同一任务的判定是正确的。挖掘出的模型的仿真分析表明其产生的日志和原始日志具有逻辑等价性，并通过实验对比分析证实了 τ 算法的优越性。

实际业务流程产生的事件日志可能含噪声或不完整，在下一章，我们将考虑在这种情况下使用遗传算法等智能计算方法来优化 τ 挖掘算法使其能够挖掘带噪声或不完整的事件日志。

第4章 基于混合自适应遗传算法的工作流挖掘

4.1 引言

当前，e-Business、e-Government、e-Science 等网络应用的发展对协同工作环境提出更高的要求，需要支持跨平台、跨组织、跨地域甚至跨行业的协同工作、支持无处不在的商务、政务、科研等各类事务活动。面向上述网络应用领域的综合协同工作系统多数是多平台、多系统的复杂系统，并且正在朝向虚实融合的方向发展。在综合协同工作系统中，如何科学、快捷、精确地构造出系统中的过程模型就成为当前工作流建模领域的一个热点和难点问题。工作流挖掘的最终目的就是从综合协同工作系统的流程执行日志中发现关于过程模型的知识[83]。工作流挖掘能发现工作流模型，从而避免了从空白开始进行既费时又容易出错的工作流模型的设计。基于流程日志的工作流挖掘引起了学术界和工业界的重视。通过工作流挖掘帮助企业实现业务流程的建模和再造，极大地提高企业的市场竞争力。

Cook 和 Wolf [31]将过程挖掘技术运用到软件工程过程中，提出了过程发现的三种方法：神经网络、纯算法和 Markov 方法，并且提出后两种方法更有前景。在文献[84]中 Cook 和 Wolf 在其前期研究的基础上进行基于概率的并发过程的研究。文献[85]提出了量化的指标，用来衡量过程模型和以事件日志形式存在的实际行为之间的关联程度。Herbst 等人[86]基于马尔可夫模型提出了自底向上和自顶向下的挖掘算法。Aalst 等人提出了能挖掘出工作流网形式过程模型的 α 算法[35]。Medeiros 等人分析了 α 算法的局限并给出能处理短循环的算法[87]，Wen 等人改进了 α 算法，提出能处理非自由选择结构也能发现任务间的隐式依赖的方法[88]。

鉴于上述文献，普遍采用概率统计和归纳推理方法进行工作流挖掘算法研究。这种根据局部事件日志分析任务间的关联关系的研究策略称之为局部策略，其缺点是不能保证挖掘出的模型是全局最优的。这类算法对事件日志的完备性和无噪声要求也限制了算法的实际应用。此外随事件日志中涉及的活动数目的增加，工作流挖掘问题也是一个 NP-Hard 计算问题[89]。遗传算法是模拟生物在自然环境中的遗传和进化过程而形成的一种自适应全局优化概率搜索算法。作为一

个重要的进化算法， 其目前在数据挖掘领域仍有广泛的应用[90-92]。文献[93]运用基本遗传算法(Simple Genetic Algorithm, SGA)进行工作流挖掘研究，但其关于活动逻辑的定义不够全面，适应值函数定义不够准确，不能很好地解决活动多、复杂性高和有噪音的情况，且 SGA 易出现早熟或后期收敛缓慢等缺点。

针对上述问题，面向活动数目多，事件日志不完备并含有噪声的复杂应用，本章提出了一种混合自适应遗传算法（Hybrid Adaptive Genetic Algorithm, HAGA)。利用该算法的全局策略进行工作流挖掘研究。该算法不要求事件日志具有完备性和无噪声，具有较好的健壮性和对噪声的抗干扰性；与基本遗传算法相比，该算法能显著提高解的质量和收敛速度。

4.2 基本工作流网

定义 4-1 （**P/T 系统**）. P/T 系统是一个六元组 S=(P, T, F, K, W, M)，其中：

（1）$N=(P, T, F)$是一个网，称为 S 的基网，P 为库所(place)的有限集，T 为变迁(transition)的有限集，$F \subseteq (P \times T) \cup (T \times P)$，是有向弧的集合，称为流关系；

（2）K: $P \rightarrow \{1, 2, 3, \cdots\}$是库所集上的容量函数，若 $K(p)=w$，表示库所 P 的容量为无穷；

（3）W: $F \rightarrow \{1, 2, 3, \cdots\}$是弧集上的权函数；

（4）M: $P \rightarrow \{0, 1, 2, 3, \cdots\}$，并满足$\forall p \in P$: $M(p)=K(p)$; M 称为 S 的标识。

定义 4-2 （**EN 系统**）. EN 系统是一个四元组(P, T, F, M)，在 P/T 系统中，若对于$\forall p \in P: K(p)=1, \forall (x,y) \in F: W(x,y)=1$，且$\forall p \in$p: $M(p) \leq 1$，则称该网系统为基本网系统(Elementary Net System，EN 系统)。

在 EN 系统中，每个库所至多含一个托肯，即 $M(p)=0$ 或 $M(p)=1$，库所称为条件，变迁则称为事件或活动。

定义 4-3（**基本工作流网, Elementary Workflow net, EW-net**） 令 $N=(P,$

T，F，M)为 EN 系统，$t' \notin P \cup T$，N 是基本工作流网当且仅当：

（1）N 有两个特殊的库所：i 和 o，库所 i 是开始库所，即 $\cdot i = \emptyset$ 且 $M(i) = 1$，库所 o 是结束库所，即 $o\cdot = \emptyset$；

（2）$N' = (P, T \cup t', F \cup \{(o,t'),(t',i)\})$ 是强连通的。

图 4-1 显示了由 9 个库所和 8 个变迁构成的 EW-net，该模型中初始标识 $M_0(i) = 1$，由变迁 A 的前驱库所中的小黑点表示。

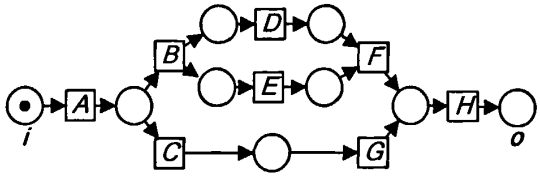
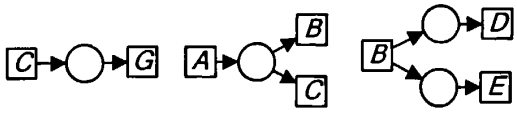


图 4-1 EW-net 实例



(a)顺序路由　　(b)选择路由　　(c)并行路由

图 4-2 三种基本路由结构图

规则 4-1 设 EW-net = (P，T，F，M)，则其变迁使能规则(transition enabling rule)如下：

对于变迁 $t \in T$，如果 $\forall p \in \cdot t$：$M(p)=1 \wedge \forall p \in t\cdot$：$M(p)=0$，则称变迁 t 在 M 标识下是使能的，记作 $M[t\rangle$，也称 M 授权 t 点火或 t 在 M 授权下点火。

规则 4-2 设 EW-net = (P，T，F，M)，则其变迁点火规则(transition firing rule)如下：

若 $M[t\rangle$，则 t 在 M 标识下是使能的，变迁点火后得到新标识 M'，对 $\forall p \in P$，

$$M'(p) = \begin{cases} M(p)-1, & p \in \cdot t - t\cdot \\ M(p)+1, & p \in t\cdot - \cdot t \\ M(p), & otherwise \end{cases} \tag{4-1}$$

图 4-1 所示的 EW-net，由规则 4-1 可知 A 是使能的；根据规则 4-2，变迁 A 点火后，则 $\cdot A$ 中每个库所的托肯数减 1，$A\cdot$ 中每个库所的托肯数加 1。图 4-2 是三种基本路由结构图(循环结构视为特殊的选择结构)，在顺序路由结构中变迁 C 点火后，其输出库所产生一个托肯，此时变迁 G 是使能的；在选择路由结构

中变迁 A 点火后，其输出库所产生一个托肯，此时变迁 B，C 是使能的，两者将争夺公共输入库所中的唯一的托肯，因此最终只有一个变迁实施点火。在并行路由中结构中变迁 B 点火后，其两个输出库所各自产生一个托肯，此时变迁 D，E 均是使能的，并并发实施点火。

定义 4-4. 设 $N = (P, T, F, M)$为 EW-net，M_0: $M(i) = 1$，如果存在变迁序列 t_1，t_2，…，t_k(记为 s)，使得 $M[s>M_k$，且 M_k: $M(o) = 1$，则称 s 为 EW-net 的点火序列。

图 4-1 所示的 EW-net 具有三个点火序列，分别为 $s_1 = ABDEFH$，$s_2 = ABEDFH$，$s_3 = ACGH$。

4.3 混合自适应遗传算法的设计

4.3.1 染色体与种群的生成

由于挖掘对象(即事件日志)是以活动序列表现的，因此考虑用活动的关联矩阵，活动出逻辑和活动入逻辑来描述过程模型。关联矩阵描述了活动之间的前驱和后继关联关系，该矩阵是一个 $n \times n$ (n 为活动数目)方阵。出逻辑描述了当前活动与后继活动之间的逻辑关系，入逻辑描述了当前活动与前驱活动之间的逻辑关系。过程模型(*PM*)具体可描述为：

$PM = (ASet, CMatrix, OSet, ISet)$，其中，

ASet 是活动集合，$ASet = \{A_1, A_2, \cdots, A_n\}$，$n$ 为活动数，A_i 为过程中的第 i 个活动；

CMatrix 是活动关联矩阵。$CMatrix = [c_{ij}]_{n \times n}$

$$c_{ij} = \begin{cases} 1, & A_i, A_j\text{关联，}A_i\text{是起点，}A_j\text{是终点} \\ 0, & \text{没有关联} \end{cases} \tag{4-2}$$

若 $c_{ij}=1$，则 A_i 是 A_j 的前驱活动，记为 $A_i = PreActivity(A_j)$。同时 A_j 是 A_i 的后继活动，记为 $A_j = PostActivity(A_i)$；若 $c_{ij} = 0$，则 A_i 和 A_j 没有关联关系。

OSet 是活动的输出逻辑，$OSet = \{O_1, O_2, \cdots, O_n\}$，$n$ 为活动数，O_i 为活动 A_i 的输出逻辑。O_i 描述为 $\{A_j|A_k\}^*$，$j \neq k$，A_j 和 A_k 为 A_i 的后继活动。

ISet 是活动的输入逻辑，*ISet* = $\{I_1, I_2, \cdots, I_n\}$，$n$ 为活动数，I_i 为活动 A_i 的输入逻辑。I_i 描述为 $\{A_j|A_k\}^*$，$j \neq k$，A_j 和 A_k 为 A_i 的前驱活动。

如图 4-1 所示的 EW-net 的过程模型定义如下：

ASet = {*A*，*B*，*C*，*D*，*F*，*E*，*G*，*H*}；

$$CMatrix = \begin{bmatrix} 0&1&1&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0 \end{bmatrix};$$

OSet = {{*BC*}，{*D*，*E*}，{*G*}，{*F*}，{*F*}，{*H*}，{*H*}，{ ϕ }}；

ISet = {{ϕ}，{*A*}，{*A*}，{*B*}，{*B*}，{*D*，*E*}，{*C*}，{*FG*}}。

对 *CMatrix* 采用二进制编码方式进行编码，矩阵中元素按行组合形成了一个染色体，染色体长度为 n^2(n 为活动数目)。*CMatrix* 染色体的生成步骤如下：

步骤 1：将染色体每个基因位随机取值为 0 或 1；

步骤 2：判定生成的染色体是否有效(有开始活动和结束活动)，符合条件转步骤 3，否则转步骤 1。

步骤 3：将生成的染色体加入初始种群 *X*；

步骤 4：重复上述步骤，直到 *X* 中包含 *PopSize* 个可行解。

对 *OSet* 采用符号编码方式进行编码。基因位上的符号为活动名 A_i。*OSet* 染色体的生成方法如下：

算法 *CreateOSet*

输入：*ASet* = $\{A_1, A_2, \cdots, A_n\}$,*CMatrix*

输出：*OSet* = $\{O_1, O_2, \cdots, O_n\}$

①计算每个活动 A_i 的出度(ODegree)记为 *ODegree*(A_i)；

②若 *ODegree*(A_i) = 0，则 O_i = { ϕ }；

③若 *ODegree*(A_i) = 1，则 A_i 仅有一个后继活动 H = *PostActivity*(A_i)，O_i = {H}；

④若 *ODegree*(A_i)>1，则 *OSet* = *CreatBoolExpr*(*ODegree*(A_i)，*PostActivity*(A_i))，O_i = {*OSet*}。

算法 *CreateBoolExpr* 根据输入活动 A_i 的出度，以及活动 A_i 的后继活动集，生成由 A_i 的后继活动组成的布尔逻辑表达式。算法 *CreateBoolExpr* 如下：

算法 *CreateBoolExpr*

输入：$ODegree(A_i)$，$PostActivity(A_i)$

输出：O_i

①$k = ODegree(A_i)$，$PostActivity(A_i) = B_1, B_2, \cdots, B_k, B_j \neq A_i$；

②随机生成元素为 0 或 1 的 $k \times k$ 方阵 $Marix = [m_{ij}]_{k \times k}$，且方阵中每列只有一个元素值为 1；

③for(i=1; i<k; i++){

$b = \phi$;

for(j=1; j<k; j++)

若 m_{ij}=1，则 b +=B_j;

若 O_it 中不包含 b，则将 b 加入; }

ISet 生成算法和 *OSet* 类似，在此省略。

4.3.2 适应值函数

在工作流挖掘中，符合性测试的基本思想是直接将事件日志同染色体所表示的过程模型比较，即判断事件日志同过程模型是否吻合。目前对工作流挖掘的符合性测试方法有 Medeiros[93]等人提出的适应值函数，其思想只从局部模型来考虑事件日志活动的发生情况，并且没有考虑事件日志可能含有噪声的情况，因此需要新的适应值函数来评价日志与模型的符合性。为了从全局考虑我们给出适应值函数的定义：

定义 4-5. 假设事件日志为 L，过程模型为 M，k 是事件日志中不同类型的轨迹数，$n_i(1 \leqslant i \leqslant k)$是 i 类轨迹的数目，$IsFiredTrace_i$ 表示 i 类轨迹能否正常点火，取值为 0(正常点火)或 1(不能正常点火)，$NumFiredActivity_i$ 表示 i 类轨迹中能点火的活动数目，$NumActivity_i$ 表示 i 类轨迹中的活动总数，*Punishment* 是罚函数，$NumArtificalToken_i$ 是 i 类轨迹重放时人工添加的托肯数，则事件日志与模型的符合性适应值函数为式(4-3)，其中罚函数为式(4-4)。

$$Fitness(M,\ L) = \alpha \times \frac{\sum_{i=1}^{k} n_i \times isFiredTrace_i}{\sum_{i=1}^{k} n_i} + \beta \times \frac{\sum_{i=1}^{k} n_i \times NumFiredActivity_i}{\sum_{i=1}^{k} n_i \times NumActivity_i} - \lambda \times Punishment, \tag{4-3}$$

$$Punishment = \frac{\sum_{i=1}^{k} n_i \times NumArtificalToken_i}{\sum_{i=1}^{k} n_i \times NumActivity_i} \tag{4-4}$$

为计算出适应值函数公式中的各个参数的值，需要首先将用关联矩阵和出入逻辑描述的过程模型转换成基本工作流网。前者以活动为描述对象，而基本工作流网由库所、变迁和有向弧构成，活动可以直接转换为基本工作流网中的变迁，因此转换算法的关键在于：如何根据活动的出、入逻辑给变迁添加相应的库所，并且添加库所和变迁之间的有向弧。转换的主要算法如下：

算法 *ActivityConvertToWorkflow*

输入：$ASet = \{A_1, A_2, \cdots, A_n\}$, *CMatrix*, $OSet = \{O_1, O_2, \cdots, O_n\}$, $ISet = \{I_1, I_2, \cdots, I_n\}$

输出：EW-net = (P, T, F)

①将 *ASet* 直接转换为变迁集 $TSet = \{T_1, T_2, \cdots, T_n\}$；

②计算每个变迁 T_i 的出度(*ODegree*)和入度(*IDegree*),分别记为 $ODegree(T_i)$ 和 $IDegree(T_i)$；

③若 $IDegree(T_i) = 0$，则给变迁 T_i 添加开始库所 P_0，设置其托肯为 1，并添加 P_0 和 T_i 之间的有向弧，使得 $P_0 \rightarrow T_i$；

④若 $ODegree(T_i) = IDegree(T_j) = 1$，则添加库所 P_k(k 为当前 EW-net 中库所的个数)，使得 $T_i \rightarrow P_k$，$P_k \rightarrow T_j$；

⑤若 $ODegree(T_i)>1$，且 $PostTransitionSet(T_i)=\{T_{ij}, \cdots, T_{ik}\}$，$OSet(T_i)=O_i = \wedge(A_i)$，$A_i = \vee(T_{ir})$，$T_{ir} \in \{T_{ij}, \cdots, T_{ik}\}$，则对每一个元素 A_i 添加一个库所 P_k，使得 $T_i \rightarrow P_k$，$P_k \rightarrow T_{ir}$；

⑥若 $IDegree(T_i)>1$，且 $PreTransitionSet(T_i)=\{T_{ij}, \cdots, T_{ik}\}$，$ISet(T_i)= I_i = \wedge(A_i)$，$A_i = \vee(T_{ir})$，$T_{ir} \in \{T_{ij}, \cdots, T_{ik}\}$，则对每一个元素 A_i 添加一个库所 P_k，使得 $T_{ir} \rightarrow P_k$，$P_k \rightarrow T_i$；

⑦若 $ODegree(T_i)=0$，则给变迁 T_i 添加结束库所 P_{k+1}，使得 $T_i \rightarrow P_{k+1}$；

4.4 混合自适应遗传操作

4.4.1 选择操作

选择操作使用轮盘赌选择结合精英保持策略。若个体 a_i 的适应度为 $Fitness(a_i)$，种群规模为 $PopSize$，则选中 a_i 为下一代个体的概率如式(4-5)所示。

$$P(a_i) = Fitness(a_i) \Big/ \sum_{j=1}^{PopSize} Fitness(a_j) \tag{4-5}$$

为了避免适应度高(精英)的个体被淘汰，算法中保留 2%的精英个体，并且这些被保留下来的精英不参加交叉和变异操作。

4.4.2 混合自适应交叉变异操作

文献[93]运用固定交叉率和变异率分别进行交叉和变异操作，但是生物进化论指出交叉率和变异率随进化阶段的变化而变化，即自适应于进化所处的阶段[94]。图 4-3 描述了在进化初始期、半成熟期、成熟期、后成熟期四个阶段交叉率和变异率的自适应调整曲线。其中$\triangle P_x \in [-(1-P_x), 1-P_x]$且$\triangle P_m \in [-(1-P_m), 1-P_m]$。

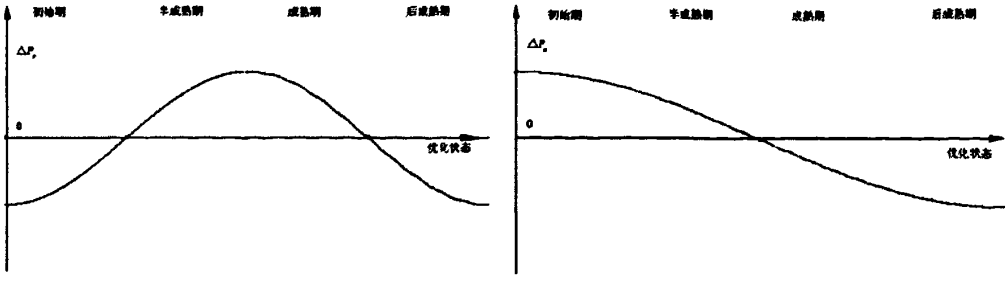


图 4-3 在不同的进化阶段$\triangle P_x$和$\triangle P_m$的调整曲线[94, 95]

此外固定交叉率的交叉操作不考虑个体之间的相似度，所有个体均以固定的概率进行交叉，从而影响算法寻优的效率。我们根据个体的相似度自适应地确定交叉个体的交叉率，对相似度大的交叉个体以相对小的概率交叉，相似度小的交叉个体以相对大的概率交叉。个体相似度指两个个体对应基因相同的基因个数与染色体长度的比值。设种群中染色体 X 与 Y，其相似度如式(4-6)所示。

$$Similarity(X,Y)=1-(\sum_{i=1}^{N^2}X_i xor Y_i)/N^2 \tag{4-6}$$

X_i是染色体 X 的第 i 个基因，Y_i是染色体 Y 的第 i 个基因，xor 表示异或运算。

Srinvivas[13]等提出的自适应遗传算法交叉率和变异率能随适应度进行自适应调整，但此算法对个体适应度接近或等于最大适应度时，P_c和 P_m 接近或等于零，这对进化初期是不利的，使得进化初期的优良个体几乎处在一种不发生变化的状态，进化走向局部最优解的可能性增加。我们通过混合进化阶段和相似度两个因素，对算法的交叉率和变异率进行自适应调整。交叉率 P_{ac}和变异率 P_{am} 的自适应调整公式如式(4-8)和式(4-10)所示。交叉算子采用单点交叉，交叉时以活动作为交叉点对活动的关联矩阵对应的染色体进行交叉，并对活动的出逻辑做相应的交叉。交叉后如入逻辑出现与活动关联矩阵不一致的情况，则对相应活动的入逻辑进行修补。

在活动的关联矩阵对应的染色体上进行基本位变异操作，以变异概率 P_{am} 在随机指定的某一位基因位上进行变异操作。变异后如果产生活动出、入逻辑和活动关联矩阵不一致的情况，则修补活动的出、入逻辑。

$$P_c=\begin{cases}\dfrac{k_1(f_{max}-f')}{f_{max}-f_{avg}}, & f\geq f_{avg}\\ k_2, & f<f_{avg}\end{cases} \tag{4-7}$$

$$P_{ac}(X,Y)=\begin{cases}0, & Similarity(X,Y)\in[0.9,1]\\ P_c+\Delta P_c, & Similarity(X,Y)\in(0.1,0.9)\\ (P_c+\Delta P_c)(1-Similarity(X,Y)), & Similarity(X,Y)\in[0,0.1]\end{cases} \tag{4-8}$$

$$P_m=\begin{cases}\dfrac{k_3(f_{max}-f)}{f_{max}-f_{avg}}, & f\geq f_{avg}\\ k_4, & f<f_{avg}\end{cases} \tag{4-9}$$

$$P_{am}(X)=P_m+\Delta P_m \tag{4-10}$$

通过混合自适应的方法，根据进化阶段自适应调整交叉和变异率，并且减少高相似度个体因交叉而导致种群多样性丢失的可能性，增加低相似度个体的搜索效率，从而提高遗传寻优的效率，加快算法的收敛速度。

4.4.3 混合自适应遗传算法流程

算法流程如图 4-4 所示。

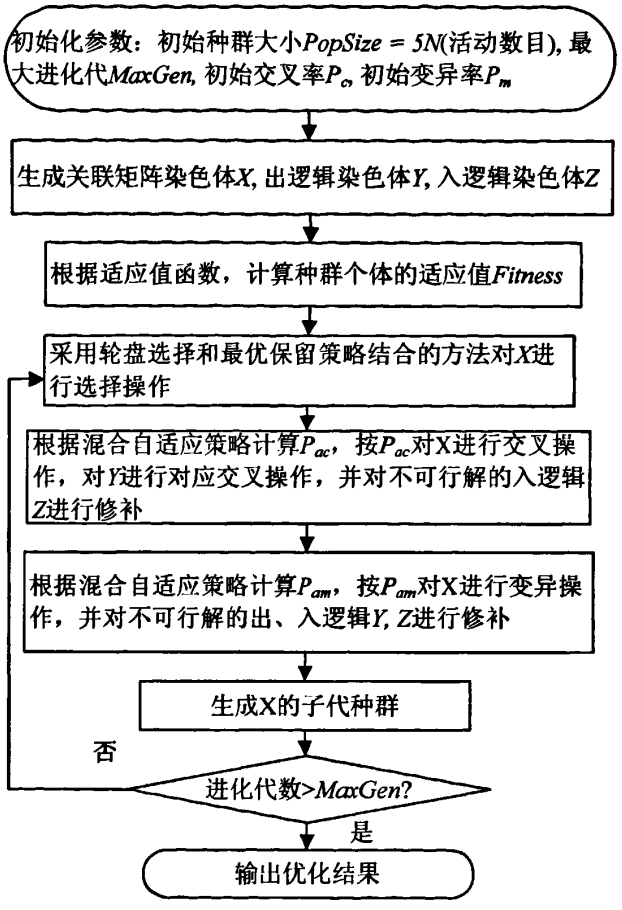


图 4-4 混合自适应遗传算法流程

4.4.4 仿真实验与算法比较

为了验证算法的效果，进行了大量的计算机仿真实验，绝大数结果都是非常令人满意的。为了测试算法的性能，选用了包含 10%的噪声数据，活动数目分别为 6，18，36 的流程日志进行测试。利用 CPN Tools[96]分别对图 4-5 至图 4-7 所示的 EW-net 进行仿真，生成测试仿真日志。在仿真日志的基础上随机产生 10%的噪声数据。测试参数为：初始种群规模 $PopSize = 5N$(N 为活动数目)，适应值函数系数 α=0.3，β=0.7，γ=0.2，交叉操作的固定交叉率 P_c=0.85，变异操作的

固定变异率 P_m=0.25。

图 4-8 至图 4-10 是对不同规模的事件日志进行挖掘的最优适应值走势曲线。由图中可以看出，HAGA 算法具有很好的进化特性，提高了收敛速度和最优适应值。

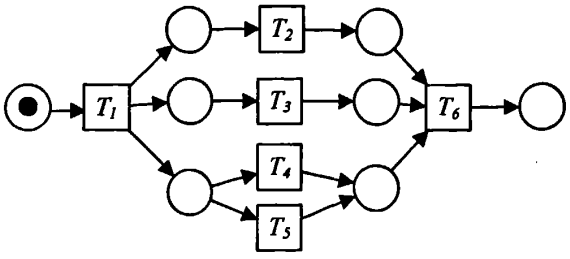


图 4-5 含 6 个变迁的 EW-net

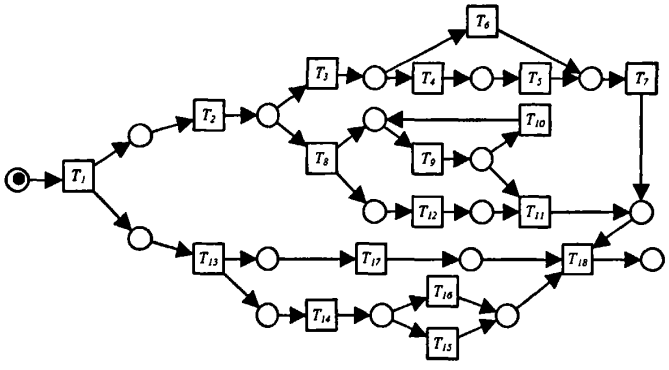


图 4-6 含 18 个变迁的 EW-net

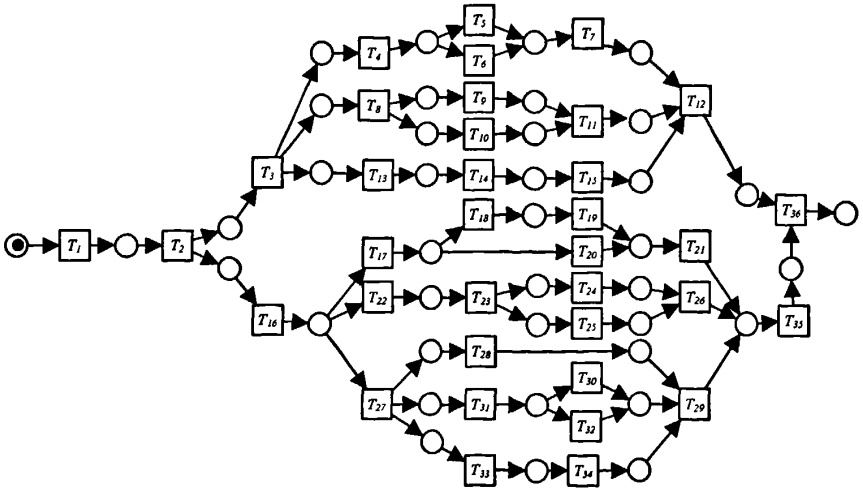


图 4-7 含 36 个变迁的 EW-net

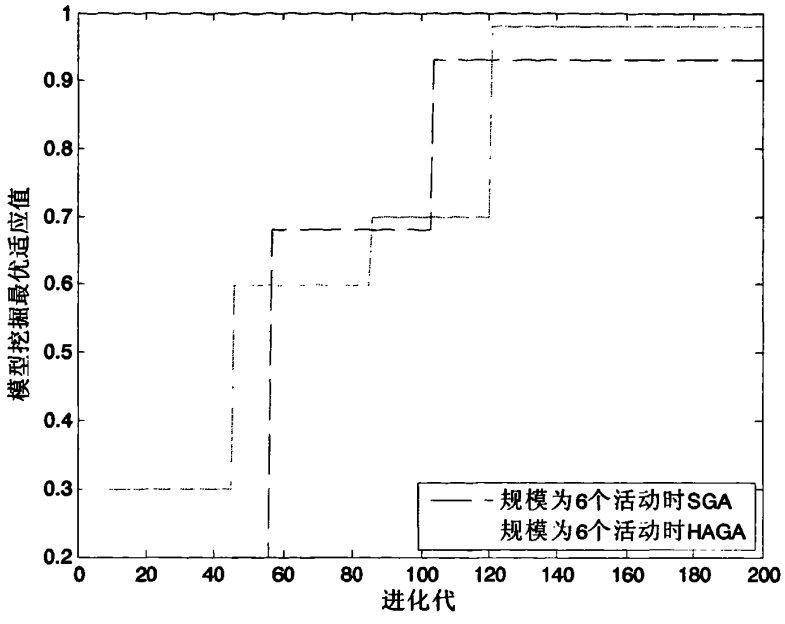


图 4-8 含 6 个活动的模型挖掘最优适应值走势图

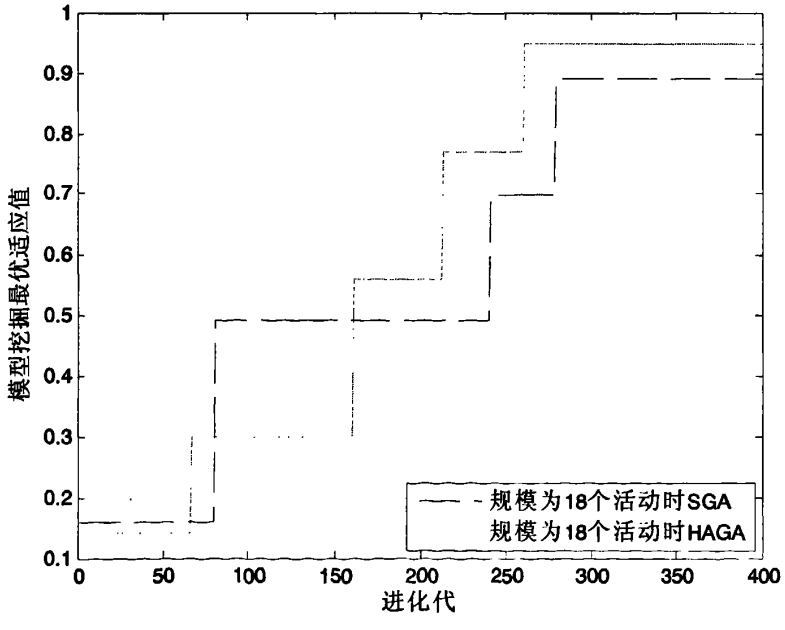


图 4-9 含 18 个活动的模型挖掘最优适应值走势图

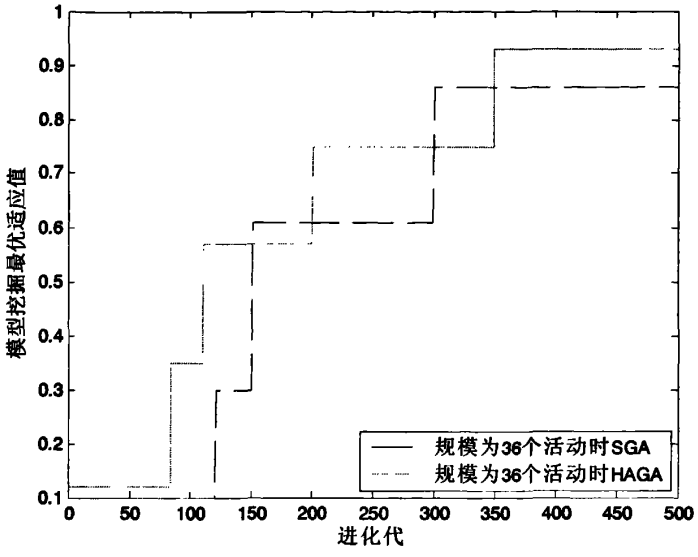


图 4-10 含 36 个活动的模型挖掘最优适应值走势图

表 4-1 对比了运用 *a* 算法[7]、Medeiros[13]提出的 SGA 算法以及本章提出的混合自适应遗传算法（HAGA）三种算法进行工作流挖掘所得到的模型最优适应值、达优率和执行时间。从表 4-1 中可以看出，HAGA 比 *a* 算法、SGA 获得了更好的最优值；对比 SGA，HAGA 具有更高的达优率，并且其执行时间也有所降低。

表 4-1 工作流挖掘算法比较

规模	最优适应值			达优率	
	a	SGA	HAGA	SGA	HAGA
6	0.5	0.93	0.98	0.95	0.95
18	0.4	0.89	0.95	0.90	0.90
36	0.3	0.86	0.93	0.87	0.88

4.5 本章小结

目前工作流挖掘算法大都采用局部策略并且无法处理噪声的情况；生物进化论指出在进化初始期、半成熟期、成熟期、后成熟期四个阶段交叉率和变异率是

随着进化阶段的变化而变化的；当交叉个体非常相似时，交叉操作很难产生新的个体，影响算法对新的解空间进行搜索，从而导致种群多样性的丢失。针对以上情况，设计了根据进化阶段以及交叉个体相似度的大小而自适应的交叉率和变异率，提出混合自适应遗传算法进行工作流挖掘。仿真实验表明，该算法与 α 算法相比具有更高的强壮性和对噪声的抗干扰性；与基本遗传算法相比，该算法在提升解的收敛速度和质量方面有明显的优势。

第5章 基于 SPN 的工作流时间性能分析

5.1 引言

近年来，工作流管理技术已引起了越来越多的研究人员的关注。性能分析对于工作流管理的成功起到了举足轻重的作用。时间性能是衡量工作流性能的一个重要指标。

本章利用概率论中关于服从指数分布的随机变量的分布函数、密度函数及数学期望的基本性质，详细地研究了组成 SPN 模型的串行、并行、选择和循环四种基本结构的平均延迟时间，得出了通用的 SPN 模型平均延迟时间公式。通过对复杂 SPN 模型的等效化简，实现对工作流时间性能的分析。最后，通过实例说明了该方法的可行性和有效性。

5.2 随机 Petri 网

定义 5-1 （随机 Petri 网－Stochastic Petri Net，SPN）[97]. SPN 是一个四元组，$SPN=(P, T, F, \lambda)$，其中：

1) (P, T, F)是一个 Petri 网；

2) $\lambda=\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$，是与网中的各个变迁相关联的实施速率的集合。

在 SPN 中，一个变迁从可实施时刻到实施时刻之间需要延迟，这个延迟时间是一个连续的随机变量 X_i，此随机变量服从参数为 λ_i 的指数分布。

变迁的实施速率表示变迁在可实施情况下单位时间内平均实施次数。变迁 t_i 的平均实施速率的倒数 $1/\lambda_i$ 称为 t_i 的平均实施延迟或平均延迟时间。

定义 5-2. 设 X 为概率分布服从指数分布的随机变量，其分布函数、密度函数和数学期望定义公式如下：

分布函数：$F_X(t)=1-e^{-\lambda t}, \quad t \geq 0$

密度函数：$f_X(t)=\lambda e^{-\lambda t}, \quad t \geq 0$

数学期望：$E[X]=\int_0^{\infty} tf_X(t)dt=\frac{1}{\lambda}$

5.3 平均延迟时间分析

5.3.1 串行 SPN 模型的平均延迟时间

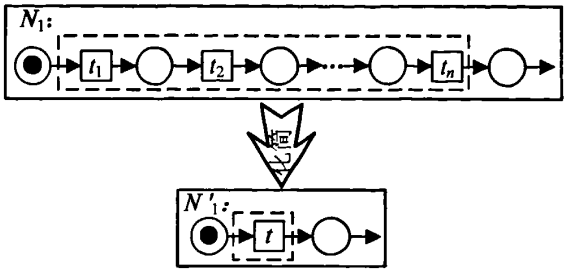


图 5-1 串行 SPN 模型及等效化简模型

串行 SPN 模型定义了工作流中一系列按一定顺序执行的活动。由 n 个变迁串行组成的 SPN 模型，如图 5-1 上图所示。为了简化模型的复杂性，可将模型 N_1 进行性能等效化简，化简后的模型如图 5-1 下图所示，化简的部分为虚线框部分。

设模型 N_1 的 n 个串行变迁的延迟时间为 n 个相互独立的随机变量，且分别服从参数为 λ_1，λ_2，…，λ_n 的指数分布，则该模型中的 n 个变迁的等效变迁 t 的平均延迟时间 T 的计算过程如下：

设 n 个相互独立的随机变量为 X_1，X_2，…，X_n，已知这 n 个随机变量分别服从参数为 λ_1，λ_2，…，λ_n 的指数分布，则变迁 t 的平均延迟时间 T 为：

$$\begin{aligned} T &= E[X_1+X_2+\cdots+X_n] \\ &= E[X_1]+E[X_2]+\cdots+E[X_n] \\ &= \frac{1}{\lambda_1}+\frac{1}{\lambda_2}+\cdots+\frac{1}{\lambda_n} \\ &= \sum_{i=1}^{n}\frac{1}{\lambda_i} \end{aligned} \tag{5-1}$$

5.3.2 并行 SPN 模型的平均延迟时间

并行 SPN 模型中多个变迁可以同时实施或以任意次序实施。模型如图 5-2 上图所示。化简后的性能等效模型如图 5-2 下图所示。

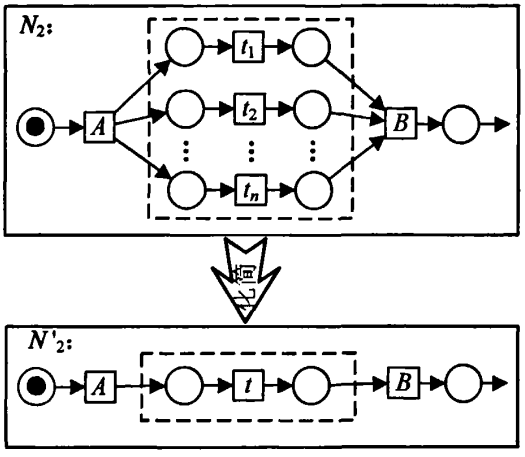


图 5-2 并行 SPN 模型及等效化简模型

设模型 N_2 的 n 个并行变迁的延迟时间为 n 个相互独立的随机变量 $X_1, X_2, \cdots, X_n$，且分别服从参数为 λ_1，λ_2，…，λ_n 的指数分布，则该模型中虚线框内的 n 个变迁的等效变迁 t 的平均延迟时间 T 的计算过程如下：

设随机变量 X_1，X_2，…，X_n 对应的分布函数分别为：

$$F_1(t) = 1 - e^{-\lambda_1 t},$$
$$F_2(t) = 1 - e^{-\lambda_2 t},$$

$$F_n(t) = 1 - e^{-\lambda_n t}。$$

并行 SPN 模型的平均延迟时间分布函数为 $F_T(t)$，$T=\max(X_1, X_2, \cdots, X_n)$。由于并行 SPN 模型的平均延迟时间等于并行结构各分支中最大变迁延迟时间，所以得出：

$$
\begin{aligned}
F_T(t) &= P(T \le t) \\
&= P(\max(X_1, X_2, \cdots, X_n) \le t) \\
&= P(X_1 \le t)P(X_2 \le t)\cdots P(X_n \le t) \\
&= F_1(t)F_2(t)\cdots F_n(t) \\
&= (1-e^{-\lambda_1 t})(1-e^{-\lambda_2 t})\cdots(1-e^{-\lambda_n t}) \\
&= 1-\sum_{i=1}^{n} e^{-\lambda_i t} + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} e^{-(\lambda_i+\lambda_j)t} - \\
&\quad \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} e^{-(\lambda_i+\lambda_j+\lambda_k)t} + \cdots + (-1)^n e^{-\sum_{i=1}^{n}\lambda_i t}
\end{aligned}
$$

并行 SPN 模型的平均延迟时间密度函数为 $f_T(t)$：

$$
\begin{aligned}
f_T(t) = F'_T(t) &= \sum_{i=1}^{n}\lambda_i e^{-\lambda_i t} - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\lambda_i+\lambda_j)e^{-(\lambda_i+\lambda_j)t} + \\
&\quad \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n}(\lambda_i+\lambda_j+\lambda_k)e^{-(\lambda_i+\lambda_j+\lambda_k)t} + \cdots + \\
&\quad ((-1)^{n+1}\sum_{i=1}^{n}\lambda_i)e^{-\sum_{i=1}^{n}\lambda_i t}
\end{aligned}
$$

从而变迁 t 的平均延迟时间 T 为：

$$
\begin{aligned}
T = E(t) &= \int_0^\infty t f_T(t)dt = \int_0^\infty t\Big(\sum_{i=1}^{n}\lambda_i e^{-\lambda_i t} - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\lambda_i+\lambda_j)e^{-(\lambda_i+\lambda_j)t} + \\
&\quad \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n}(\lambda_i+\lambda_j+\lambda_k)e^{-(\lambda_i+\lambda_j+\lambda_k)t} + \cdots + ((-1)^{n+1}\sum_{i=1}^{n}\lambda_i)e^{-\sum_{i=1}^{n}\lambda_i t}\Big)dt \\
&= \sum_{i=1}^{n}\frac{1}{\lambda_i} - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{1}{\lambda_i+\lambda_j} + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n}\frac{1}{\lambda_i+\lambda_j+\lambda_k} + \cdots + \\
&\quad ((-1)^{n+1}\frac{1}{\sum_{i=1}^{n}\lambda_i}
\end{aligned}
\tag{5-2}
$$

5.3.3 选择 SPN 模型的平均延迟时间

选择 SPN 模型定义了分支变迁之间的选择执行关系，从多个分支中选择其中一个分支执行，选择依赖于库所相关属性值所体现出的特定性质。如图 5-3 上图所示，化简后的性能等效模型如图 5-3 下图所示。

设模型N_3的n个选择变迁的延迟时间为n个相互独立的随机变量$X_1, X_2, \cdots, X_n$，且分别服从参数为$\lambda_1, \lambda_2, \cdots, \lambda_n$的指数分布，变迁$t_1, t_2, \cdots, t_n$的实施概率分别为$q_1, q_2, \cdots, q_n$，且$q_1+q_2+\cdots+q_n=1$。则该模型中虚线框内的$n$个变迁的等效变迁$t$的平均延迟时间$T$的计算过程如下：

设随机变量$X_1, X_2, \cdots, X_n$对应的分布函数分别为：

$$F_1(t)=1-e^{-\lambda_1 t},$$
$$F_2(t)=1-e^{-\lambda_2 t},$$

$$F_n(t)=1-e^{-\lambda_n t}。$$

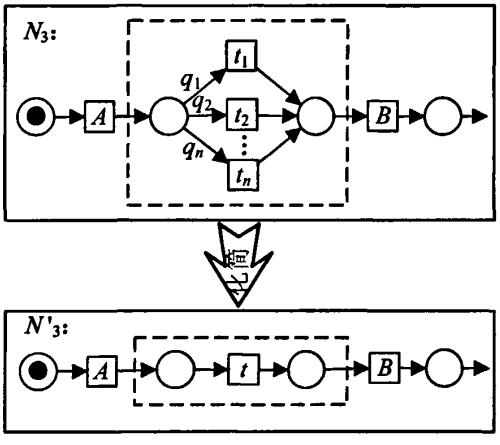


图 5-3 选择 SPN 模型及等效化简模型

由于n个变迁的延迟时间是n个相互独立的随机变量，所以选择实施的等效变迁延迟时间概率分布为：

$$\begin{aligned} F_T(t) &= q_1F_1(t)+q_2F_2(t)+\cdots+q_nF_n(t) \\ &= q_1(1-e^{-\lambda_1 t})+q_2(1-e^{-\lambda_2 t})+\cdots+q_n(1-e^{-\lambda_n t}) \end{aligned}$$

选择结构的平均延迟时间密度函数为$f_T(t)$：

$$f_T(t)=F'_T(t)=q_1\lambda_1e^{-\lambda_1 t}+q_2\lambda_2e^{-\lambda_2 t}+\cdots+q_n\lambda_ne^{-\lambda_n t}$$

从而变迁t的平均延迟时间T为：

$$
\begin{aligned}
T = E(t) &= \int_0^{\infty} t f_T(t) dt \\
&= \int_0^{\infty} t(q_1\lambda_1 e^{-\lambda_1 t} + q_2\lambda_2 e^{-\lambda_2 t} + \cdots + q_n\lambda_n e^{-\lambda_n t}) dt \\
&= \frac{q_1}{\lambda_1} + \frac{q_2}{\lambda_2} + \cdots + \frac{q_n}{\lambda_n} \\
&= \sum_{i=1}^{n} \frac{q_i}{\lambda_i}
\end{aligned}
\tag{5-3}
$$

5.3.4 循环 SPN 模型的平均延迟时间

循环随机 Petri 网模型中一个或多个变迁被多次重复实施，如图 5-4 上图所示，化简后的性能等效模型如图 5-4 下图所示。

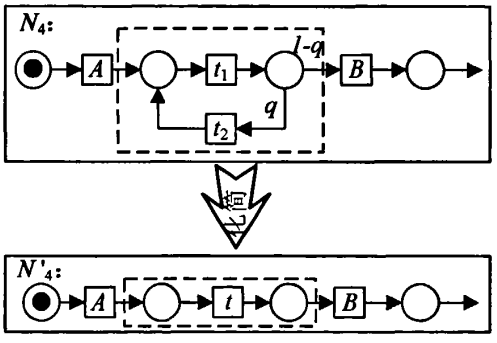


图 5-4 循环 SPN 模型及等效化简模型

设模型 N_4 中的两个循环变迁的延迟时间为相互独立的随机变量 X_1，X_2，且分别服从参数为 λ_1，λ_2 的指数分布，变迁 t_2，B 的实施概率分别为 q，$1\text{-}q$。该模型中循环发生的次数、概率及相应的平均延迟时间如表 5-1 所示。

表 5-1 循环发生的次数、概率及相应的平均延迟时间

循环的次数	0	1	2	…	n
概率	$1\text{-}q$	$q(1\text{-}q)$	$q^2(1\text{-}q)$	…	$q^n(1\text{-}q)$
平均延迟时间	$\frac{1}{\lambda_1}$	$\frac{2}{\lambda_1}+\frac{1}{\lambda_2}$	$\frac{3}{\lambda_1}+\frac{2}{\lambda_2}$	…	$\frac{n+1}{\lambda_1}+\frac{n}{\lambda_2}$

图 5-5 的上图中虚线框内的模型结构等效变换成下图所示的选择型结构模型。分支选择的概率分别为 $1\text{-}q$，$q(1\text{-}q)$，…，$q^n(1\text{-}q)$。根据选择结构的平均延迟

时间公式可得循环结构模型的等效变迁 t 的平均延迟时间 T 为：

$$\begin{aligned}
T &= \sum_{i=0}^{\infty} q^i(1-q)\left(\frac{i+1}{\lambda_1}+\frac{i}{\lambda_2}\right) \\
&= (1-q)\left(\sum_{i=0}^{\infty} q^i\left(\frac{i+1}{\lambda_1}+\frac{i}{\lambda_2}\right)\right) \\
&= (1-q)\left(\frac{1}{\lambda_1}\sum_{i=0}^{\infty} q^i\times(i+1)+\frac{1}{\lambda_2}\sum_{i=0}^{\infty} q^i\times i\right) \\
&= (1-q)\left(\frac{1}{\lambda_1}\sum_{i=0}^{\infty} q^i\times i+\frac{1}{\lambda_1}\sum_{i=0}^{\infty} q^i+\frac{1}{\lambda_2}\sum_{i=0}^{\infty} q^i\times i\right) \\
&= (1-q)\left(\left(\frac{1}{\lambda_1}+\frac{1}{\lambda_2}\right)\sum_{i=0}^{\infty} q^i\times i+\frac{1}{\lambda_1}\sum_{i=0}^{\infty} q^i\right)
\end{aligned} \tag{5-4}$$

设 $S=\sum_{i=0}^{\infty} q^i\times i = q+2q^2+3q^3+\cdots+nq^n+\cdots$，则等式两边分别乘以 q 得：

$$qS = q^2+2q^3+\cdots+nq^{n+1}+\cdots$$

$$S-qS = q+q^2+q^3+\cdots+q^{n+1}+\cdots = \frac{q}{1-q}$$

从而，$S=\frac{q}{(1-q)^2}$，代入式(5-4)得：

$$\begin{aligned}
T &= (1-q)\left(\left(\frac{1}{\lambda_1}+\frac{1}{\lambda_2}\right)S+\frac{1}{\lambda_1}\sum_{i=0}^{\infty} q^i\right) \\
&= (1-q)\left(\left(\frac{1}{\lambda_1}+\frac{1}{\lambda_2}\right)\frac{q}{(1-q)^2}+\frac{1}{\lambda_1}\times\frac{1}{1-q}\right) \\
&= \left(\frac{1}{\lambda_1}+\frac{1}{\lambda_2}\right)\frac{q}{1-q}+\frac{1}{\lambda_1} = \frac{q\lambda_1+\lambda_2}{\lambda_1\lambda_2(1-q)}
\end{aligned} \tag{5-5}$$

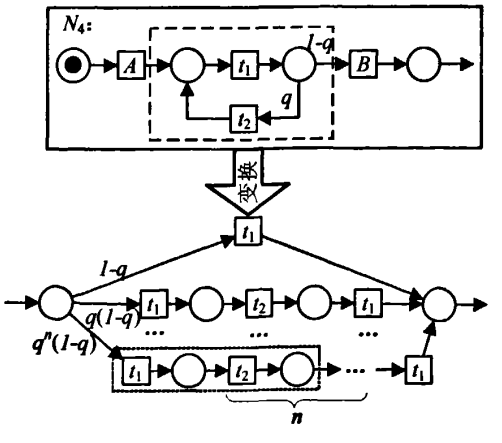


图 5-5 循环结构模型等效变换成选择结构模型

5.4 基于 SPN 的工作流时间性能分析实例

设一工作流管理系统中存在如图 5-6 所示的 SPN 模型，该 SPN 模型包含有 10 个变迁，其平均延迟时间如表 5-2 所示。SPN 模型中有由 t_6 和 t_7 构成的循环结构，该循环结构中执行 t_9 和 t_7 的概率分别为 0.8 和 0.2；该模型中还存在 t_5 和循环结构构成的并行结构，分别以 0.4 和 0.6 的概率选择执行 t_2 和 t_3 的选择结构以及由 t_2，t_4，t_8 组成的串行结构。该模型的平均延迟时间的计算如下：

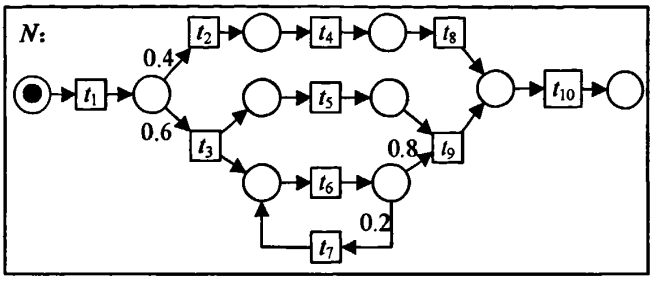


图 5-6 SPN 模型实例

表 5-2 变迁及其平均延迟时间

变迁 t_i	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
$1/\lambda_i$	9	2	2	3	2	5	3	4	7	8

设与循环结构等效的变迁为 t_{6+7}，等效平均延迟时间记为 T_{6+7}，结构中的 q=0.2，运用循环结构式(5-5)得：

$$T_{6+7}=\frac{1}{\lambda_{6+7}}=\frac{q\lambda_6+\lambda_7}{\lambda_6\lambda_7(1-q)}=\frac{0.2\times\frac{1}{5}+\frac{1}{3}}{\frac{1}{5}\times\frac{1}{3}\times0.8}=7$$

设 t_{6+7} 与 t_5 构成的并行结构的等效变迁为 $t_{5+(6+7)}$，等效平均延迟时间记为 $T_{5+(6+7)}$，运用并行结构式(5-2)得：

$$T_{5+(6+7)}=\frac{1}{\lambda_{5+(6+7)}}=\frac{1}{\lambda_5}+\frac{1}{\lambda_{6+7}}-\frac{1}{\lambda_5+\lambda_{6+7}}$$

$$=2+7-\frac{1}{\frac{1}{2}+\frac{1}{7}}=7.4$$

设 t_3，$t_{5+(6+7)}$，t_9 构成的串行结构的等效变迁为 $t_{3+(5+(6+7))+9}$，等效平均延迟时

间记为 $T_{3+(5+(6+7))+9}$，运用串行式(5-1)得：

$$T_{3+(5+(6+7))+9}=\frac{1}{\lambda_{3+(5+(6+7))+9}}=\frac{1}{\lambda_3}+\frac{1}{\lambda_{5+(6+7)}}+\frac{1}{\lambda_9}$$
$$=2+7.4+7=16.4$$

设 t_2, t_4, t_8 构成的串行结构的等效变迁为 t_{2+4+8}，等效平均延迟时间记为 T_{2+4+8}，运用串行结构式(5-1)得：

$$T_{2+4+8}=\frac{1}{\lambda_{2+4+8}}=\frac{1}{\lambda_2}+\frac{1}{\lambda_4}+\frac{1}{\lambda_8}=2+3+4=9$$

设由等效变迁 t_{2+4+8} 和 $t_{3+(5+(6+7))+9}$ 构成的选择结构的等效变迁为 $t_{(2+4+8)+(3+(5+(6+7))+9)}$，选择变迁 t_{2+4+8} 和 $t_{3+(5+(6+7))+9}$ 的概率分别为 q_1=0.4，q_2=0.6，q_1+q_2=1。等效平均延迟时间记为 $T_{(2+4+8)+(3+(5+(6+7))+9)}$，运用选择结构式(5-3)得：

$$T_{(2+4+8)+(3+(5+(6+7))+9)}=\frac{1}{\lambda_{(2+4+8)+(3+(5+(6+7))+9)}}$$
$$=\frac{q_1}{\lambda_{2+4+8}}+\frac{q_2}{\lambda_{3+(5+(6+7))+9}}$$
$$=\frac{0.4}{\lambda_{2+4+8}}+\frac{0.6}{\lambda_{3+(5+(6+7))+9}}$$
$$=\frac{0.4}{\frac{1}{9}}+\frac{0.6}{\frac{1}{16.4}}=13.4$$

5.5 本章小结

本章利用概率论中关于服从指数分布的随机变量的分布函数、密度函数及数学期望的基本性质，详细地研究了组成 SPN 模型的串行、并行、选择和循环四种基本结构的平均延迟时间，得出了通用的 SPN 模型平均延迟时间的公式。通过对复杂 SPN 模型的等效化简，实现对工作流时间性能的分析。最后，通过实例说明了该方法的可行性和有效性。

第6章 基于克隆选择离散粒子群算法的工作流调度

6.1 引言

当前，在计算机综合协同应用系统中，网格[98]集成动态、跨机构的虚拟组织的各种资源在 e-Science、e-Business 等应用领域实现了广域范围、多机构间的资源共享和协同工作。随着服务网格、数据网格等进一步地扩展和丰富，网格应用越发复杂，为保证各种网格应用自动化或半自动化高效运行，需要采用工作流来对网格应用进行构建、执行调度和管理，从而实现网格资源协同工作。由于网格系统异构和资源动态变化，工作流在网格环境下的调度问题成为计算机领域的一个研究难点和热点。

网格工作流调度一般使用有向无环图 DAG(Directed Acrylic Graph) [99]进行建模，记作 $G=\{V, E\}$，其中节点集 V ={1, 2, , N} ($|V|=N$)表示网格工作流中所有任务集合；E 是有向边集合，表示活动间的约束关系，为了实现工作流任务间协同完成整个网格应用，合理分配网格资源，需要有效调度优化相关资源。基于有向无环图的调度问题是一个 NP 问题[100]。

随着开放网格服务架构 OGSA(Open Grid Service Architecture)[101]的提出，网格技术和 Web 服务技术相融合，QoS 成为网格工作流中候选服务的选择依据。金海等[102]提出了一种合成服务的服务质量 QoS（Quality of Service）优化模型，并且对模型的求解速度和优度进行了研究。候选服务的调度和执行依照用户提出的服务参数进行，可以更好地满足用户需求，实现网格工作流调度的优化。文献[103]提出了时间、费用、可靠性的三个参数的 QoS 标准，并在此基础上对网格工作流调度进行了优化。文献[71]针对网格工作流调度问题提出了时间等五个参数的 QoS 标准。

当前，网格工作流的调度多是建立在网格资源是 100%可靠的基础上，基于时间和费用约束的调度优化，对网格调度的可靠性缺乏研究；虽有部分学者从用户角度提出了可靠性等多 QoS 的调度需求，但是多 QoS 的系统化程度不够，忽

视了调度的实际可行性，将调度模型简化。本章在时间、费用和可靠性基础上，从多目标优化角度提出了新的网格工作流调度模型，改进网格资源管理与调度策略；同时改进了传统的 PSO 算法，使用速度对候选服务位置击发旋转，实现了 PSO 离散化处理。通过克隆选择，实现种群多样性，克服早熟收敛和局部极值等问题，实现最优调度。

6.2 网格工作流体系架构

网格工作流的系统架构建立在 OGSA 的基础上，是一个多层的体系架构，如图 6-1 所示。整个系统分为四层[104]：网格工作流应用层，抽象网格工作流层，具体网格工作流层，网格资源层。

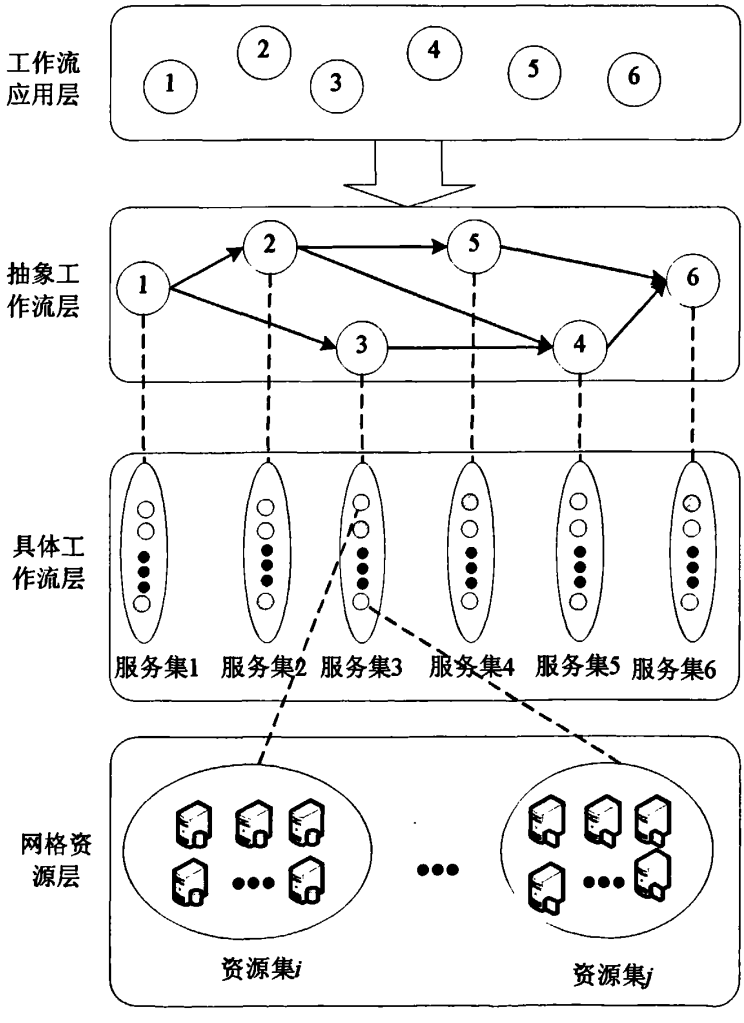


图 6-1 网格工作流体系架构

用户根据自身的需求来设定相关的 QoS 参数，网格系统依据相应的参数到候选服务集中进行候选服务的选择，完成网格资源的调度。按照这一思想，我们将网格工作流系统的四层架构划分为三种映射。

（1）网格工作流应用—抽象网格工作流层映射：通过有向无环图对具体的网格工作流应用进行构建，获得相应的网格工作流模型；

（2）抽象网格工作流层—具体网格工作流层映射：按照网格服务资源使用者提供的 QoS 要求，从候选服集中选择合适服务来完成工作流任务；

（3）具体网格工作流层—网格资源层映射：将网格资源映射为功能相同但是 QoS 各异的各种服务，构建具体的工作流层。

这个网格系统中网格工作流的调度过程包含上述的两次映射关系，即抽象工作流与具体工作流的映射和具体网格工作流与网格资源的映射，而不是仅仅定位于候选服务集中候选服务的选择，整个调度过程是候选服务选择中的可靠性考量基础上的时间、费用资源的优化，真正实现网格工作流调度的最优化目标。

网格系统中用户分为两种，服务资源的提供者和服务资源的使用者。不同的服务资源提供者提供服务功能相同但是服务质量各异的服务。服务质量使用一个三元组 QoS=(T，C，Rel)来表示，T_{ij}，C_{ij}，Rel_{ij} 分别表示第 i 个任务所对应第 j 个候选服务的时间、费用、可靠性。其中：

时间：是一个衡量服务完成请求的速度指标，时间可以表示为服务执行所需要的时间与服务通信所需要的时间之和。

费用：费用是一个描述服务执行所需要的花费指标，表示服务执行一次所需要的总费用。

可靠性：可靠性是体现一个服务执行能力的指标，可以使用服务成功执行的概率来表示，一般使用服务成功执行次数与总的执行次数之间的比例来处理。

网格工作流的调度主要是按照服务资源使用者提供的不同服务质量参数来完成相关服务资源的调度。

6.3 工作流调度问题的数学模型

对于分布异构的网格环境，同一个任务可由多个不同资源提供的服务来完

成，各个服务的 QoS 存在差异，从而使得功能相同的服务在执行不同的任务时候所需要的时间费用及其可靠性都是不同的。调度系统按照多 QoS 的要求来选最优的服务，从而实现网格工作流调度的最优化。

6.3.1 QoS 定义

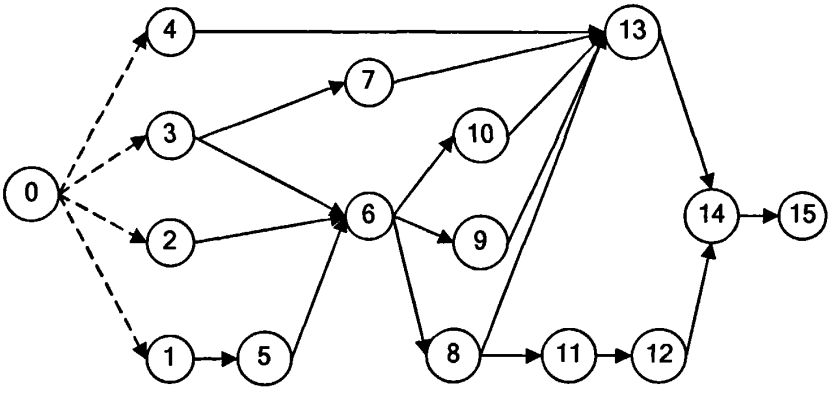


图 6-2 e-Protein 项目中的工作流[105]

图 6-2 所示为 e-Protein 项目中使用 DAG 表示的网格工作流。$|V|=15$，节点 0 为虚拟入口，对于$\forall i \in V$，能够完成该任务的候选服务集称为任务 i 的服务集，记作 $S(i)=(s_1^i,s_2^i,...,s_{l(i)}^i)$；$s_j^i(1 \le j \le l(i))$，$l(i)$为任务 i 对应的候选服务的总数，s_j^i 表示提供给任务 i 的候选服务集中第 j 个候选服务。网格作为协同应用环境，需要面向用户 *multi*-QoS 的资源管理策略与调度策略，下面给出基于有向无环图建模的网格工作流的 QoS 的计算公式。设工作流的调度计划为 sp，$sp=\{i,s_j^i\}$，$(1 \le i \le N, 1 \le s_j^i \le l(i))$，时间、费用和可靠性的计算公式如下：

时间：$T_{sp}=CPA(T_{j1}^1,T_{j2}^2,...,T_{jN}^N) \le T_0$，CPA 为计算关键路径的函数，$T_{jn}^n$ 为关键路径上提供给任务 n 的候选服务集中第 jn 个候选服务的完成时间。执行任务的各个服务需要的时间，在调度过程中关键路径总的时间值不可以超过用户定义的最大时间 T_0；

费用：$C_{SP}=\sum_{i=1}^{N} C_j^i \le C_0(1 \le j \le l(i))$，$C_j^i$ 为执行任务 i 的候选服务 s_j^i 的费用，

在调度过程中总的时间费用不可以超过用户定义的费用上限C_0；

可靠性：$Rel_{sp}=MIN_{1\le j\le N}\{Rel_j^i\}\ge Rel_0(1\le j\le l(i))$，$Rel_j^i$为执行任务的第$j$个候选服务的可靠性。在调度过程中所有选定服务的平均可靠性不可以低于用户对可靠性的最小约束参数Rel_0；

6.3.2 调度问题的数学模型

多 QoS 的网格工作流调度优化问题作为一个重要的的多目标优化问题，大多数情况下各个 QoS 对应的各个子目标是相互冲突的，某个子目标的改善可能会引起其他子目标性能的下降，即同时使得各个子目标均达到最优是不可能的，解决多目标问题的最终手段是在各个子目标之间进行协调和权衡处理，有效地搜索多个非劣解，或称为 Pareto 最优解，所有非劣解的结合构成了多目标意义下的最优解集，也就构成了多目标空间中问题的 Pareto 边界。

将多目标整合为单一目标，是处理多目标优化问题最简单而传统的策略，即将原问题转化为单目标问题来求解。$\min y=(f_1(x_1),f_2(x_2),...,f_n(x_n))$，加权累加函数可以描述如下：$f=\sum_{i=1}^{n}\omega_i f_i(x)$，$\omega_i$为第$i$个目标的非负权值，同时$\sum_{i=1}^{n}\omega_i=1$，在算法的优化过程中，每一组权向量将决定一个搜索方向，为了获得尽可能多的 Pareto 最优解，权值采用随机方式确定。$\omega_i=random_i/\sum_{j=1}^{n}random_j$，$i$=1，2，3。其中$random_k$ 为 0 到 1 之间均匀分布的随机数。如果用户设置了各个 QoS 的权重，则按照用户的自定义进行调整。工作流调度问题的数学模型描述如下：

$$\min fitness=(T_{sp},\ C_{sp},\ Rel_{sp})=\ \omega_1 T_{sp}+\omega_2 C_{sp}+\omega_3/Rel_{sp} \tag{6-1}$$

$$\text{s.t.}\quad T_{sp}=CPA(T_{j1}^1,T_{j2}^2,...,T_{jN}^N)\le T_0 \tag{6-2}$$

$$C_{SP}=\sum_{i=1}^{N}C^i{}_j\le C_0 \tag{6-3}$$

$$Rel_{sp}=MIN_{1\le j\le N}\{Rel_j^i\}\ge Rel_0 \tag{6-4}$$

6.4 克隆选择离散粒子群算法

粒子群算法被提出后，由于简单、高效，引起了众多学者的极大关注，特别是在多目标领域，粒子群算法的高效搜索能力有利于得到多目标意义下的最优解，在种群内可以以并行方式同时搜索多个非劣解；但是基本粒子群算法容易陷入局部极值、早熟收敛，并且在进化的后期的收敛速度慢、精度低。为扩大其应用领域，很多学者致力于从各个方面来提高其性能。多 QoS 约束的网格工作流调度问题作为一个重要的多目标优化问题在实际的网格应用和计算机协同领域中有广泛的应用。文献[106]提出了自适应扩散混合变异机制微粒群算法，但是多 QoS 约束的网格工作流调度问题是离散优化问题，该算法不能直接应用于离散优化领域，需要进一步进行离散化处理。针对这一情况，本章提出了一种克隆选择离散粒子群算法（Clonal Selection DiscreteParticle Swarm Optimization，CSDPSO）来解决多 QoS 约束的网格工作流调度问题。

6.4.1 粒子编码

编码问题是构造微粒移动规则的前提，粒子编码的维数为有向无环图中节点的个数，每一维对应一个任务，每一维对于一个候选服务集，图 6-2 所示的网格工作流中共有 16 个结点（任务），粒子编码采用 16 维编码。由于每一个结点（任务）的候选服务数量为[1, $l(k)$]，如表 6-1 所示，则粒子编码空间中解的个数为 $\prod_{k=1}^{N} l(k), N = |\nu|$。

表 6-1 粒子编码

任务	1	2	...	16
粒子 i 的维度	1	2	...	16
候选服务集	$s_1^1, s_2^1, ..., s_{l(1)}^1$	$s_1^2, s_2^2, ..., s_{l(2)}^2$	...	$s_1^{16}, s_2^{16}, ..., s_{l(16)}^{16}$

采用上述编码方式，对有 16 个节点的网格工作流，使用克隆选择离散粒子

群算法，种群规模为 *PNum*，粒子从 16 维空间对 16 个候选服务进行搜索，如图 6-3 所示。

$$\begin{cases} x_i = (x_{i1}, x_{i2}, \ldots, x_{iN}) \\ v_i = (v_{i1}, v_{i2}, \ldots, v_{iN}) \\ pBest_i = (p_{i1}, p_{i2}, \ldots, p_{iN}) \\ gBest = (p_{g1}, p_{g2}, \ldots, p_{gN}) \end{cases} \tag{6-5}$$

式(6-5)中，N 表示粒子的维数，x_i 表示粒子 i 的当前位置，v_i 表示粒子 i 的飞行速度，$pBest_i$ 表示粒子 i 飞行中搜索到的最优解，$gBest$ 表示整个种群在当前搜索到的最优解。

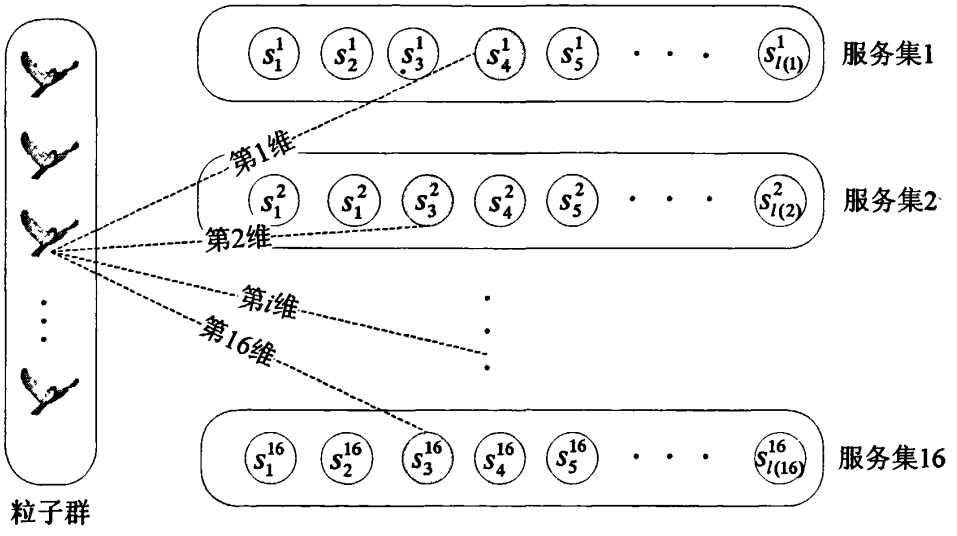


图 6-3 粒子群 16 维空间搜索

粒子的位置定义为候选服务的编号，为离散量，第 j 个粒子的位置记为 $x_j = (x_{j1}, x_{j2}, \ldots, x_{jN})$，$x_{jk} \in \{1, 2, \ldots, l(k)\}, 1 \le k \le N$，$l(k)$ 为节点 k 对应的候选服务集的长度，值取整数。粒子的速度 v 的作用是改变粒子的位置，定义为对粒子位置的一种变换，记为 $v_j = (v_{j1}, v_{j2}, \ldots, v_{jN})$，$v_{\min,k} \le v_{jk} \le v_{\max,k}$，速度更新公式如式(6-6)所示，速度定义为浮点数，对速度的修正如式(6-8)。

$$v_i(t+1) = \omega v_i(t) + c1 \cdot r1(pBest_i - x_i) + c2 \cdot r2(gBest_i - x_i) \tag{6-6}$$

$$\omega = 0.9 - t * 0.5 / GEN \tag{6-7}$$

$$v_{ij} = \min\left(v_{\max,j}, \max(v_{\min,j}, v_{ij})\right) \tag{6-8}$$

6.4.2 相关算子

（1）位置离散化方法

结合速度和位置之间的关系文中定义了一种“轮转击发”规则，利用该规则实现粒子群离散化。图 6-4 为任务 i 的候选服务选择示意图。以长度为 $l(i)$的循环队列为数据结构，粒子在第 i 维空间上进行轮转选择。

“轮转击发”规则如下：

针对粒子 i 维对应的是第 i 个候选服务集，位置变化如下，速度可以取正数和负数，取正数则表示位置对应的候选服务的编号在增加，取负数说明位置对应的候选服务的编号在减少。增加和减少以 1 或者 1 的倍数变化，1 为步长。候选服务的编号采用循环队列的形式：

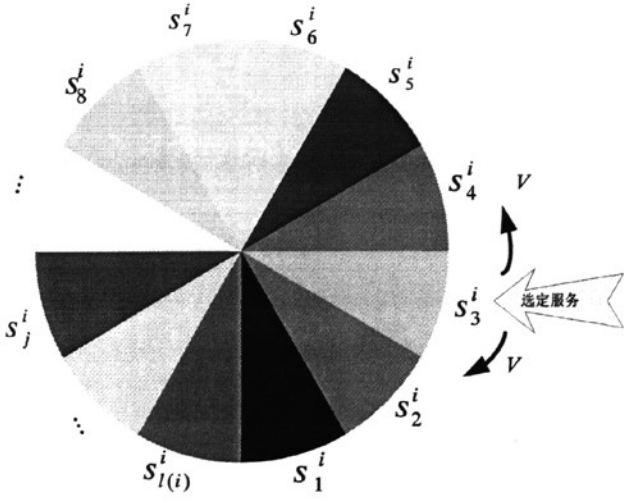


图 6-4　候选服务选择图

根据“旋转击发”规则，位置随着速度变化而离散变化的更新示意图如图 6-5 所示。具体的公式如式(6-9)所示。

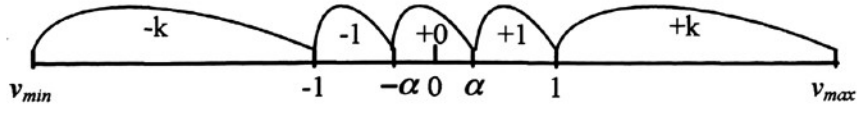


图 6-5　位置离散化示意图

$$x_{ij}(t+1)=\begin{cases}x_{ij}(t) & ,-\alpha \le v_{i,j}(t+1)\le \alpha \\ x_{ij}(t)+1 & ,\alpha < v_{i,j}(t+1)\le 1 \\ x_{ij}(t)-1 & ,-1\le v_{i,j}(t+1)<-\alpha \\ x_{ij}(t)+k & ,1<v_{ij}(t+1)<v_{\max,j} \\ x_{ij}(t)-k & ,v_{\min,j}\le v_{ij}(t+1)<-1\end{cases} \tag{6-9}$$

其中，

$$v_{\max,j}=l(j),l(i)>=2,\alpha=\frac{v_{\max,j}}{ALPHA},ALPHA\in\{8,10,20\},k=\begin{cases}\lceil v_{ij}(t+1)\rceil,v_{ij}(t+1)>0\\ \lfloor v_{ij}(t+1)\rfloor,v_{ij}(t+1)<0\end{cases}$$

（2）克隆选择

Burnet[107]提出了克隆学说，认为只有能完全匹配或者部分匹配抗原，具有较高亲和力的 B 细胞，才能被免疫系统选中并对其进行克隆复制并产生后代，后代通过体细胞高频变异实现亲和力的成熟，而那些亲和力低 B 细胞将无法获得克隆。这种机制称为克隆选择。

粒子经过成比例克隆、克隆高频变异后，从父代个体与子代个体中，选择一个适应度最高的最优个体作为下一代个体。文中采用柯西变异，柯西密度函数为：

$$Cauch(x)=\frac{1}{\pi(x^2+t^2)} \tag{6-10}$$

其中 t 是尺度参数，$t>0$，经过柯西变异后，粒子 p_i 变为 p'_i。

（3）CSDPSO 算法流程

算法流程如图 6-6 所示。

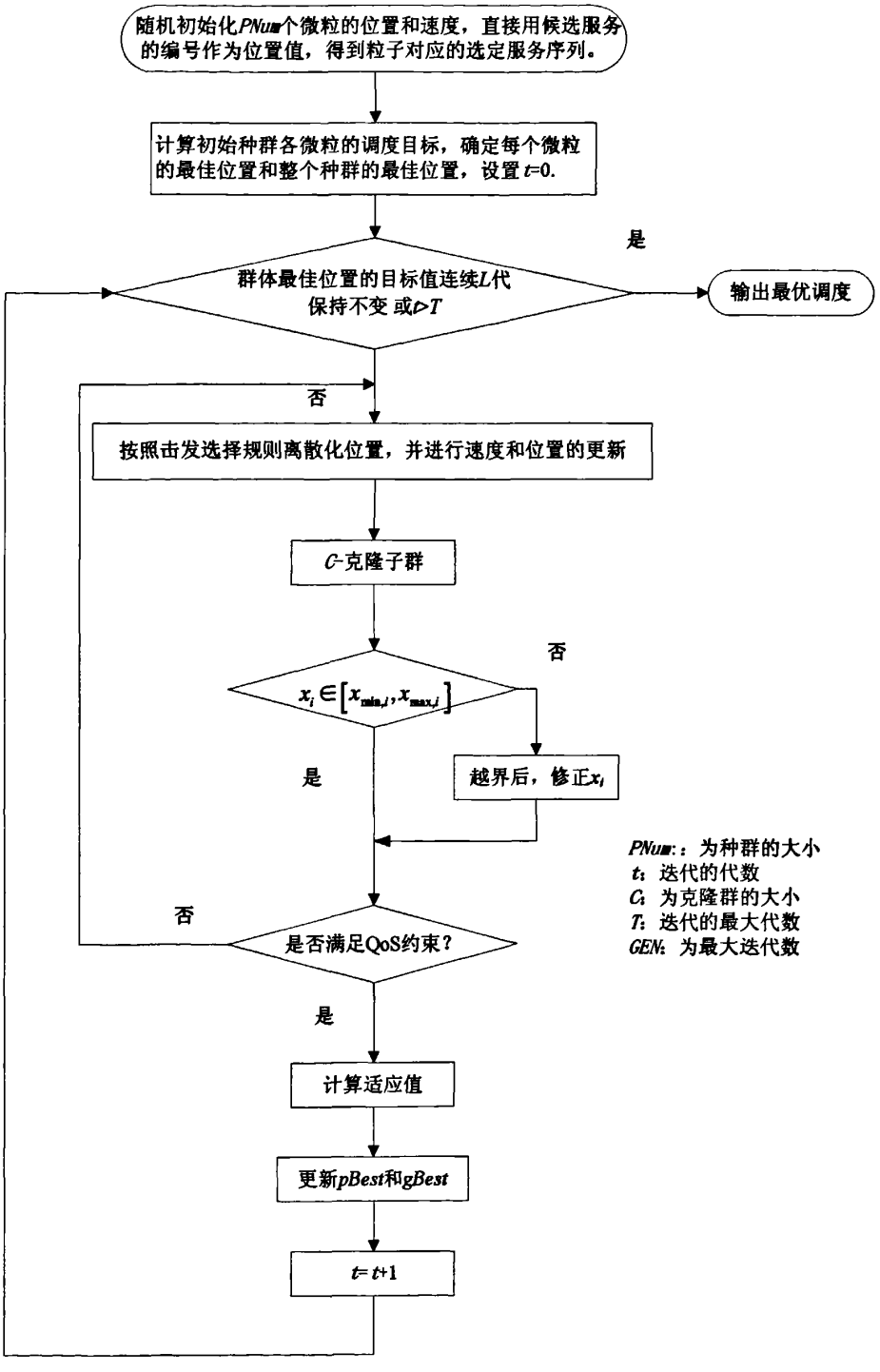


图 6-6 CSDPSO 算法流程图

6.5 仿真实验及结果分析

以图 6-2 所示的 e-Protein 项目中的工作流为例，表 6-2 给出了各个任务对应的候选服务的时间、费用和可靠性数据。

表 6-2 任务的候选服务集

	1			2			3			4			5			6		
	T	*C*	*Rel*	*T*	*C*	*Rel*	*T*	*C*	*Rel*	*T*	*C*	*Rel*	*T*	*C*	*Rel*	*T*	*C*	*Rel*
S_1(3)	3	6	0.9	6	5	0.7	5	5	0.8									
S_2(6)	3	4	0.8	6	5	0.9	3	7	0.8	4	2	0.6	3	5	0.7	4	5	0.8
S_3(5)	6	4	0.9	4	5	0.6	5	5	0.8	6	2	0.9	6	5	0.7			
S_4(4)	5	8	0.9	6	5	0.7	4	8	0.8	3	2	0.7						
S_5(3)	3	5	0.7	6	8	0.7	5	7	0.6									
S_6(2)	3	4	0.9	5	5	0.8												
S_7(3)	3	5	0.6	6	5	0.7	5	5	0.8									
S_8(5)	6	5	0.7	5	5	0.8	3	5	0.8	4	7	0.6	5	8	0.8			
S_9(4)	5	3	0.9	6	5	0.7	5	5	0.8	3	3	0.7						
S_{10}(3)	6	5	0.6	4	5	0.7	5	5	0.8									
S_{11}(2)	3	6	0.9	6	5	0.7												
S_{12}(3)	6	4	0.9	6	5	0.7	5	4	0.8									
S_{13}(3)	5	4	0.8	2	5	0.8	5	5	0.8									
S_{14}(4)	5	4	0.9	6	5	0.7	5	5	0.8	4	5	0.7						
S_{15}(3)	5	2	0.8	6	5	0.7	5	5	0.8									

表中 S_1-S_{15} 表示候选服务集，*T* 表示候选服务的时间，*C* 表示候选服务的费用，*Rel* 表示候选服务的可靠性。1-6 表示候选服务集中第 1 到第 6 个候选服务。

算法使用 VC++实现，在 Pentium D 3.00GHz/1024MB 的微机上运行。求得的最优解为 bestx={1，1，4，4，1，1，3，3，1，3，1，3，1，1，1}。其中 T_0=45，C_0=85，Rel_0=0.7，种群规模 $PNum \in \{20, 25, 30, 35, 40\}$，$c_1$=$c_2$=2，*ALPHA*={8,10,20}，最大迭代次数为 200。

图 6-7 给出了在不同种群规模下，*ALPHA* 参数的取值对最优适应值的影响对比图。由图 6-7 可以看出在粒子种群规模为 40 时，ALPHA 取值 20 可以使得调度最优适应值达到最优。

图 6-8 给出了 CSDPSO 和 DPSO 的目标函数最优值走势图，从图中可以看出 CSDPSO 比 DPSO 具有更好的收敛效果，并且可以取得更优的目标函数。表给出了分别运用 CSDPSO 和 DPSO 得出的 QoS 数据。

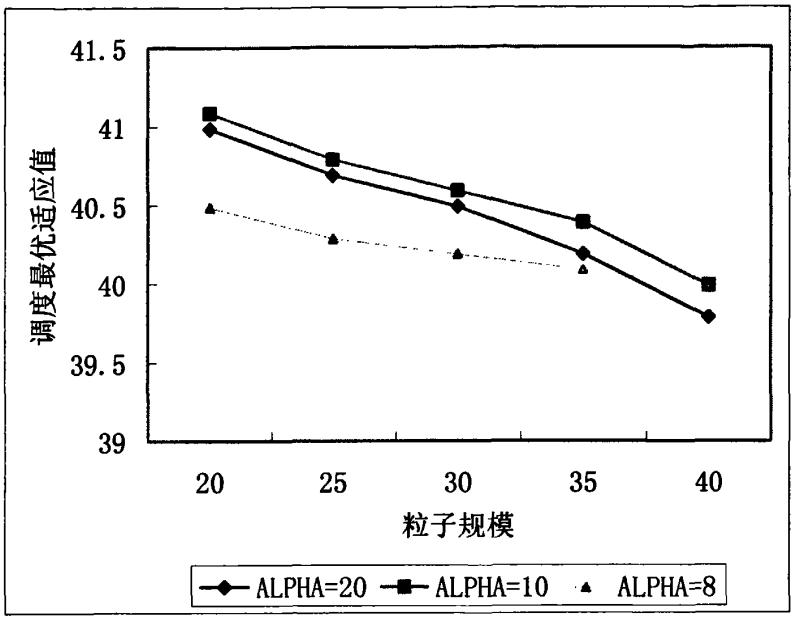


图 6-7 不同规模下 ALPHA 参数的影响

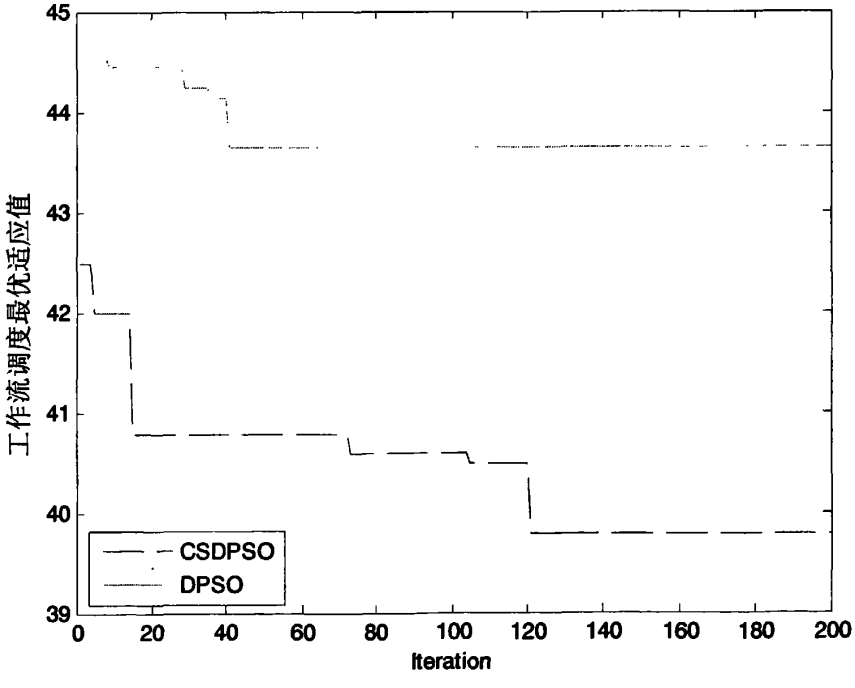


图 6-8 CSDPSO 与 DPSO 最优调度走势图

表 6-3 CSDPSO 与 DPSO 的 QoS 比较

	CSDPSO				DPSO			
粒子规模	目标函数	*T*	*C*	*Rel*	目标函数	*T*	*C*	*Rel*
20	40.9857	34	61	0.7	44.5400	33	69	0.7
25	40.6857	33	61	0.7	44.2400	32	69	0.7
30	40.4857	29	63	0.7	44.0400	33	68	0.7
35	40.1857	33	60	0.7	43.9400	31	69	0.7
40	39.7857	30	61	0.7	43.6400	30	69	0.7

6.6 本章小结

调度问题是一个复杂的组合优化问题，随着问题规模的增长，算法复杂性呈指数级增长。工作流调度在资源丰富的网格环境下更凸显其重要性，一直是工作流领域研究的热点。

本章提出了 CSDPSO 算法，并运用该算法解决网格工作流调度问题，取得了很好的效果。文中对CSDPSO和DPSO进行了对比分析，得出CSDPSO比DPSO具有更快的收敛速度，更高求解精度。

第7章 总结与展望

本论文主要研究了工作流挖掘、工作流时间性能分析以及工作流调度问题。主要的创新与贡献包含有如下几个方面：

（1）提出了发现多种复杂任务的工作流挖掘算法。为了有效挖掘含有多种复杂任务的过程模型，对含有循环任务和重复任务的事件日志进行了研究，提出发现循环、重复和同一任务的启发式规则，并且给出证明；改进了 α 算法的关联关系定义，在此基础上提出了 τ 算法。实例验证表明该方法是有效的，对循环、重复和同一任务的判定是正确的。挖掘出的模型的仿真分析表明其产生的日志和原始日志具有逻辑等价性。

（2）提出了基于混合自适应遗传算法的工作流挖掘算法。为了解决目前工作流挖掘算法大都采用局部策略因而无法保证最优挖掘以及算法对噪声敏感的情况，提出基于混合自适应遗传算法的工作流挖掘优化算法。该算法与启发式算法相比具有更高的鲁棒性和对噪声的抗干扰性；与基本遗传算法相比，该算法能显著提高解的质量和收敛速度。

（3）解决了工作流时间性能分析问题。工作流性能分析是对工作流进行评价和优化的基础，时间性能则是衡量工作流性能的一个重要指标。利用概率论中关于服从指数分布的随机变量的分布函数、密度函数及数学期望的基本性质，详细地研究了组成 SPN 模型的串行、并行、选择和循环四种基本结构的平均延迟时间，得出了通用的 SPN 模型平均延迟时间公式。通过对复杂 SPN 模型的等效化简，实现对工作流时间性能的分析。最后，通过实例说明了该方法的可行性和有效性。

（4）提出了基于克隆选择离散粒子群的工作流调度算法。为了解决工作流调度优化问题，以 QoS 为优化目标，提出了基于克隆选择的离散粒子群算法，通过该算法进行工作流调度优化研究。该算法增加了种群的多样性，提高了算法精确度，加快了算法的收敛速度，克服了离散粒子群算法早熟收敛和局部极值等问题，在工作流调度中有良好的性能。

本论文提出的算法无论在理论上还是实际应用中都具有十分重要的意义和

参考价值。

以本论文为基础，下一步工作主要包括两个方面：

（1） 在工作流调度方面，对 QoS 文中只考虑的时间、费用和可靠性。可以扩展 QoS 的指标，从而提出更全面的 QoS 评价体系。

（2） 另外还可以加入考虑资源的负载均衡问题，这样全面地考虑问题，对于实际工作流调度应用具有更加现实地指导意义。

参考文献

[1] 史美林, 向勇, 杨光信等. 计算机支持协同工作理论与应用[M]. 北京: 电子工业出版社, 2000.

[2] T. M. Kouropoulos. The Workflow Imperative[M]. New York: Van Nostrand Reinhold, 1995.

[3] 范玉顺. 工作流管理技术基础——实现企业业务过程重组、过程管理与过程自动化的核心技术[M]. 北京: 清华大学出版社, 2005.

[4] J. Li, Y. Fan, M. Zhou. Performance Modeling a nd Analysis of Workflow[J]. IE EE Transactions on System, Man, and Cybenretics, 2004, 34: 229-242.

[5] C. A. Petri. Kommunikation mit Automaten[D]. PhD thesis, Institue für Instrumentelle Mathmatik, BBonn, Germany 1962. In German.

[6] 袁崇义. Petri 网原理与应用[M]. 北京: 电子工业出版社, 2005.

[7] T. Murata. Petri Nets: Properties, Analysis and Applications[C]. Proceedings of the IEEE,1989, 77(4): 541-580.

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

[9] 周明, 孙树栋. 遗传算法原理及应用[M]. 北京: 国防工业出版社, 1999.

[10]J. H. Holland. Adapation in Natural and Artificial Systems[M]. Ann Arbor: University of Michigan press, 1975.

[11]D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning[M]. Addison-Wesley, 1989.

[12]王小平, 曹立明. 遗传算法——理论、应用与软件实现[M]. 西安: 西安交通大学出版社, 2002.

[13]M. Srinivas, L. M. Patnailk. Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms[J]. IEEE Transaction on System, Man and Cybernetics. 1994, 24(4): 656-667.

[14]J. Kennedy, R. C. Eberhart. Particle Swarm Optimization[C]. Proceedings of IEEE International Conference on Neural Networks, 1998: 1942–1948.

[15]曾建潮, 介婧, 崔志华. 微粒群算法[M]. 北京: 科学出版社, 2004.

[16]J. Kennedy, R. C. Eberhart. A Discrete Binary version of the Particle Swarm Optimization[C]. Proceedings of the IEEE conference on Systemics, Cybernetics and Informatics. 1997: 4104-4109.

[17]Workflow Management Coalition. Workflow management coalition terminology and glossary[R]. Technical Report, WfMCTC-1011, Brussels: Workflow Management Coalition, 1996.

[18]汪涛, 吴耿锋, 黄力芹. 工作流管理的现状和未来趋势[J]. 小型微型计算机系统, 2001, 22 (2): 23 2-236.

[19]M. Voorhoeve. Compositional Modeling and Verification of Workflow Processes[G]. In: W. Aalst, J. Desel, A. Oberweis eds.. Business Process Management: Models, Techniques, and Empirical Studies. Berlin: Springer-Verlag, 2000: 184-200.

[20]周建涛, 史美林, 叶新铭. 工作流过程模型中的形式化验证技术[J]. 计算机研究与发展, 2005, 42(1):1-9.

[21]王斌君. 工作流过程模型的层次研究及其分析[D]. 博士学位论文, 西北大学, 2002.

[22]肖志娇. 工作流资源管理技术研究[D]. 博士学位论文,中山大学, 2006.

[23]孙雪冬, 徐晓飞, 王刚. 基于有向超图的工作流资源分配均衡优化方法[J]. 电子学报, 2005, 33(8): 1370-1374.

[24]张健, 孙吉贵, 李妮娅等. 工作流系统中一个基于多权角色和规则的条件化RBAC安全访问控制模型[J]. 通讯学报, 2008, 29(2): 8-16.

[25]J. X. Liu, H. Y. Chen, M. D. Tang. A Role-based Semantic Authorization Framework for Workflow Management Systems[J]. Chinese Joumal of Electronics, 2006, l5(1): 55-59.

[26]谢玉凤, 杨光信, 史美林. 基于条件化有向图的工作流过程优化[J]. 计算机学报, 2001, 24(7): 720-526.

[27]S. Wang, W. Shen, Q. Hao. Agent Based Workflow Ontology for Dynamic Business Process Composition[C]. The 9th International Conference on Computer Supported Cooperative Work in Design, Coventry, UK. 2005: 452-457.

[28]闻立杰. 基于工作流网的过程挖掘算法研究[D]. 博士学位论文, 清华大学, 2006.

[29]苑迎春, 李小平, 王茜. 基于逆向分层的网格工作流调度算法[J]. 计算机学报, 2008, 31(2): 282-290.

[30]胡志刚, 胡周君. 基于多QoS目标的工作流任务调度算法[J]. 计算机工程, 2008, 34(10): 126-127.

[31]E. Cook, A. L. Wolf. Discovering Models of Software Processes from Event-based Data[J]. ACM Transactions on Software Engineering and Methodology, 1998, 7(3): 215- 249.

[32]S.S. Pinter, M. Golani. Discovering Workflow Models from Activities' lifespans[J]. Computers in Industry, 2004, 53(3): 283-296.

[33]R. Agrawal, D. Gunopulos, F. Leymann. Mining Process Models from Workflow Logs[G]. Lecture Notes In Computer Science, 1998,1377: 469-483

[34]G. Schimm. Mining Exact Models of Concurrent Workflows[J]. Computers in Industry, 2004, 53(3): 265 -281.

[35]W. M. P. van der Aalst, T. Weijters, L. Maruster. Workflow Mining: Discovering Process Models from Event Logs[J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128-1142.

[36]A.K.A. de Medeiros, W.M.P. van der Aalst, A.J.M.M. Weijters. Workflow Mining: Current Status and Future Directions[G]. Lecture Notes in Computer Science, 2003, 2888: 389-406.

[37]A.J.M.M. Weijters, W.M.P. van der Aalst, Process Mining: Discovering Workflow Models from Event-based Data[C], in: B. Kroose, M. de Rijke, G. Schreiber, M. van Someren (eds.), Proceedings of the 13th Belgium–Netherlands Conference on Artificial Intelligence (BNAIC 2001), 2001: 283–290.

[38]A.J.M.M. Weijters, W.M.P. van der Aalst, Workflow Mining: Discovering Workflow Models From Event-based Data[C], in: C. Dousson, F. Hooppner, R.Quiniou (Eds.), Proceedings of the ECAI Workshop on Knowledge Discovery and Spatial Data, 2002: 78-84.

[39]L. Maruster, A.J.M.M. Weijters, W.M.P. van der Aalst, A. van den Bosch. Process Mining: Discovering Direct Successors in Process Logs[C], in: Proceedings of the 5th International Conference on Discovery Science (Discovery Science 2002), Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin, 2002, 2534: 364–373.

[40]A. Colombo, E. Damiani, G. Gianini. Discovering the Software Process by Means

of Stochastic Workflow Analysis[J]. Journal of Systems Architecture, 2006, 52: 684-692.

[41]S. Dustdar. Caramba – A Process-Aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Teams. Distributed and Parallel Databases, 2004, 15: 45-56.

[42]S. Dustdar, T. Hoffmann, W. M. P. van der Aalst. Mining of Ad-hoc Business Processes with TeamLog. Data & Knowledge Engineering, 2005, 55: 129-158.

[43]W. Gaaloul, S. Bhiri, C. Godart. Discovering Workflow Transactional Behavior from Event-based Log[J]. Lecture Notes in Computer Science, vol. 3290, 3~18, 2004.

[44]W. Gaaloul, C. Godart. Mining Workflow Recovery from Event-based Logs[G]. Lecture Notes in Computer Science, 2005, 3649: 169-185.

[45]P. Zhang, N. Serban. Discovery, Visualization and Performance Analysis of Enterprise Workflow[J]. Computational Statistics & Data Analysis, 2007, 51: 2670-2687.

[46]赵静, 赵卫东. 基于工作流日志挖掘的流程角色识别[J]. 计算机集成制造系统, 2006, 12(11): 1916-1920.

[47]S. Subramaniam, V. Kalogeraki, D. Gunopulos. Improving Process Models by Discovering Decision Points[J]. Information Systems, 2006.

[48]A. Rozinat, W. M. P. van der Aalst. Dicision Mining in ProM[G]. Lecture Note in Comuter Science, 2006, 4102: 420-425.

[49]W. M. P. van der Aalst, B. F. van Dongen, J. Herbst, L. Maruster. Workflow Mining: A Survey of Issues and Approaches[J].Data & Knowledge Engineering, 2003, 47: 237-267.

[50]B. F. van Dongen, W. M. P. van der Aalst. Emit: A Process Mining Tool[J]. ICATPN 2004, LNCS 3099, 2004:454-463.

[51]A.J. M. M. Weijters, W. M. P. van der Aalst. Rediscovering workflow models form event-based data using little thumb[J]. Integrated Computer-Aided Engineering, 2003, 10: 151-162.

[52]J. Herbst, D. Karagiannis. Workflow Mining with InWoLvE[J]. Couputers in Industry, 2004, 53: 245-264.

[53]G. Schimm. Process Miner – A Tool for Mining Process Schemes from Event-Based Data[G]. Lecture Notes in Computer Science, 2002, 2424: 525-528.

[54]H. S. Jin, H. K. Myoung. Improving the Performance of Time-constrained Workflow Processing[J]. Journal of Systems and Software, 2001, 58(3): 211-219.

[55]林闯, 田立勤, 魏丫丫. 工作流系统模型的性能等价分析[J]. 软件学报, 2002, 13(8): 14 72-1480.

[56]李建强, 范玉顺. 一种工作流模型的性能分析方法[J]. 计算机学报, 2003, 26(5): 513-523.

[57]姜浩, 董逸生. 一种基于扩展时间 Petri 网的工作流时间性能评价方法[J]. 计算机研究与发展, 2005, 42(5): 849~855.

[58]Y. Pan, Y. Tang, J. Xu, K. Wu. Time Performance Evaluation for Workflow Based on Extended FTWF-nets[C]. Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design[C], 2006:1-5.

[59]H. Jiang. An Approach for Workflow Performance Evaluation based on Discrete Stochastic Petri net[C]. IEEE International Conference on e-Business Engineering (ICEBE), 2007: 327-330.

[60]R. Buyya, D. Abramson, J. Giddy, H. Stockinger. Economic Models for Resource Management and Scheduling in Grid computing[J]. Concurrency and Computation: Practice and Experience Journal, 2002, 14(13-15):1507-1542.

[61]D. Abramson, R. Buyya, J. Giddy. A Computational Economy for Grid Computing and its Implementation in the Nimrod-G Resource Broker[J]. Future Generation Computer Systems Journal, 2002, 18(8):1061-1074.

[62]H. Casanova, A. Legrand, D. Zagorodnov, F. Berman. Heuristics for Scheduling Parameter Sweep Applications in Grid Environments[C]. Procceding of the 9th heterogeneous Computing Workshop (HCW'00), Cancun, Mexico, 2000:349-363.

[63]X. He, X. Sun and G. Laszewski. A QoS Guided Min-Min Heuristic for Grid Task Scheduling[J]. Journal of Computer Science and Technology, 2003, 18: 442-451.

[64]胡春华, 吴敏, 刘国平. Web服务工作流中基于信任关系的QoS调度[J]. 计算机学报, 2009, 32(1): 43-54.

[65]张坚, 刘春林, 谭庆平. 一种分布式工作流中基于负载平衡的调度算法[J]. 计算机科学, 2006, 33(7): 115-118.

[66]T. Chen, B. Zhang, X. Hao, Y. Dai. Task Scheduling in Grid based on Particle

Swarm Optimization[C]. Proceedings of the 5th International Symposium on High Parallel and Distributed Computing (ISPDC'05), 2006, 238-245.

[67]L. Zhang, Y. Chen, R. Sun, S. Jing, B Yang. A Task Scheduling Algorithm based on PSO for Grid Computing[C]. International Journal of Computational Intelligence Research, 2008, 4(1): 37-43.

[68]A. Abraham, H. Liu, W. Zhang, T. Chang. Scheduling Jobs on Computational Grids Using Fuzzy Particle Swarm Algorithm[G]. Lecture Notes in Computer Science, 2006, 4252: 500-507.

[69]Q. Kang, H. He, H. Wang, C. Jiang. A Novel Discrete Particle Swarm Optimization Algorithm for Job Scheduling in Grids[C]. Proceedings of the 4th International Conference on Natural Computation (ICNC'08), 2008,401-405.

[70]张晓东, 李小平, 王茜等. 服务工作流的混合粒子群调度算法[J]. 通信学报. 2008, 29(8): 87-94.

[71]王勇, 胡春明, 杜宗霞. 服务质量感知的网格工作流调度[J]. 软件学报, 2006, 17(11): 2341-2351.

[72]V. D. Martino, M. Mililotti. Scheduling in a Grid Computing Environment Using Genetic Algorithms[C]. Proceedings of International Parallel and Distributed Processing Symposium (IPDPS'02), 2002, 235-239.

[73]Y. Gao, H. Rong, J. Z. Huang. Adaptive Grid Job Scheduling with Genetic Algorithms[J]. Future Generation Computer Systems, 2005, 21(1):151-161.

[74]S. Song, Y. Kwok, K. Hwang. Security-driven Heuristics and a Fast Genetic Algorithm for Trusted Grid Job Scheduling[C]. Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), 2005.

[75]S. Lorpunmanee, M. N. Sap, A. H. Abdullah, C. Chompoo-inwai. An Ant Colony Optimization for Dynamic Job Scheduling in Grid Environment[C]. Proceedings of World Academy of Science, Engineering and Technology, 2007, 314-321.

[76]W. Chen, J. Zhang. An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem with Various QoS Requirements[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2009, 39(1): 29-43.

[77]范玉顺, 吴澄. 工作流管理技术研究与产品现状及发展趋势[J].计算机集成制

造系统, 2000, 6(1):1-8.

[78]范玉顺, 吴澄. 基于协调理论的工作流建模方法[J]. 计算机集成制造系统, 2001, 7(4): 1-6.

[79]邓水光, 俞　镇, 吴朝辉. 动态工作流建模方法的研究与设计[J]. 计算机集成制造系统, 2004, 10(6): 601-608.

[80]战德臣, 王忠杰, 徐晓飞等. 面向企业资源计划全生命周期的建模方法及工具[J]. 计算机集成制造系统, 2006, 12(9): 1345-1373.

[81]S. Y. Hwang, W. S. Yang. On the Discovery of Process Models from there instances[J]. Decision Support System, 2002, 34(1): 41-57.

[82]李嘉菲, 刘大有, 杨博. 过程挖掘中一种能发现重复任务的扩展a算法[J]. 计算机学报, 2007, 30(8): 1436-1445.

[83]W M P van der Aalst, B F van Dongen, et al. Workflow mining: a survey of issues and approaches [J]. Data and Knowledge Engineering, 2003, 47(2): 237-267.

[84]J. E. Cook, A. L. Wolf. Event-based Detection of Concurrency [A]. The 6th ACM SIGSOFT International Symposium on Foundations of Software Engineering [C]. Lake Buena Vista, Florida, United States. New York: ACM Press, 1998:35-45.

[85]J. E. Cook, A. L. Wolf. Software Process Validation: Quantitatively Measuring the Correspondence of a Process to a Model[J]. ACM Transaction Software Engineering and Methodology, 1999, 8(2): 147-176.

[86]J. Herbst, D. Karagiannis. Integrating Machine Learning and Workflow Management to Support Acquisition and Adaptation of Workflow Models[J]. International Journal of Intelligent Systems in Accounting, Finance and Management, 2000, 9(2): 67-92.

[87]A. K. A. de Medeiros, B. F. van Dongen, W. M. P. van der Aalst, A. J. M. M. Weijters. Process Mining: Extending the a-algorithm to Mine Short Loops[M]. BETA Working Paper Series WP 113, Eindhoven University of Technology, Eindhoven, 2004.

[88]L. Wen, J. Wang, J. Sun. Detecting implicit dependencies between tasks from event logs [A]. The 8th Asia-Pacific Web Conference[C]. Lecture Notes in Computer Science 3841, Berlin: Springer-Verlag, 2006: 591-603.

[89]D. Hochbaum. Approximation Algorithms for NP-hard Problems[M]. Berkeley, CA: PWS Publishing Company, 1997.

[90]高贵, 周蝶飞, 等. 基于遗传算法的SAR图像目标鉴别特征选择[J].电子学报, 2008, 36(6): 1041-1046.

[91]李楠, 潘建, 等. 移动IP网络中基于遗传算法的多步寻呼策略及性能分析[J]. 电子学报, 2008, 36(12): 2333-2338.

[92]常会友, 衣杨, 汪定伟. 软计算求解盟友选择问题[J]. 系统仿真学报, 2003, 15(12): 1756-1758.

[93]A. K. A. de Medeiros, A. J. M. M. Weijters, W. M. P. van der Aalst. Using Genetic Algorithms to Mine Process Models: Representation, Operators and Results[M]. BETA Working Paper Series, WP 124, Eindhoven University of Technology, Eindhoven, 2004.

[94]J Reed. Simulation of biological evolution and machine learning[J]. Journal of Theoretical Biology, 1967, 17(3): 319-342.

[95]J Zhang, H. S. H. Chung W. L. Lo. Clustering-Based Adaptive Crossover and Mutation Probabilities for Genetic Algorithms[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(3): 326-335

[96]A. K. A. de Medeiros, C. W. Gunther. Process Mining: Using CPN tools to create test logs for mining algorithms[A]. The 6th Workshop and Turorial on Practical Use of Coloured Petri Nets and the CPN Tools, 2005.

[97]M. F. Neuts. Matrix Geometric Solutions in Stochastic Models[M]. Johns Hopkins University Press, Baltimore, 1981.

[98]Foster and C. Kesselman, The grid: Blueprint for a Future Computing Infrastructure[M]. San Mateo, CA: Morgan Kaufmann, 1999.

[99]M. Aggarwal, R. D. Kent, A. Ngom. Genetic Algorithm Based Scheduler for Computational Grids[C]. Proceedings of the 19th International Symposium on High Performance Computing Systems and Applications (HPCS'05), 2005: 209-215.

[100]M. R. Garey, D. S. Johnson. Computers and Intractability: a Guide to the Theory of NP-completeness[M]. New York: Freeman, 1979.

[101]Foster, C. Kesselman, J. M. Nick, S. Tuecke. Grid Service for Distributed System Integration[J]. IEEE Computer, 2002, 35(6): 37-46.

[102]金海, 陈汉华, 吕志鹏等. CGSP作业管理器合成服务的QoS 优化模型及求解[J]. 计算机学报, 2005, 28(4): 578-588.

[103]D. Kyriazis, K. Tserpes, A. Menychtas, A. Litke, T. Varvarigou. An Innovative Workflow Mapping Mechanism for Grids in the Frame of Quality of Service[J]. Future Generation Computer System, 2008, 24(6): 498-511.

[104]E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, et al.. Mapping Abstract Complex Workflows onto Grid Environments[J]. Journal of Grid Computing, 2003, 1(1): 25-39.

[105]A. O'Brien, S. Newhouse, J. Darlington. Mapping of Scientific Workflow within the E-protein Project to Distributed Resources[C]. Proceedings of the UK e-Science All Hands Meeting 2004, 2004: 404-409.

[106]吕艳萍, 李绍滋, 陈水利, 郭文忠, 周昌乐. 自适应扩散混合变异机制微粒群算法[J]. 软件学报, 2007, 18(11): 2740-2751.

[107]F. M. Burnet. The Clonal Selection Theory of Acquired Immunity[M]. Cambridge University Press, 1959.

致谢

博士学位论文完成了，此时此刻感慨万千。首先，我要衷心感谢我的指导老师常会友教授。常老师渊博的学识、严谨的治学态度、忘我的工作热情给我留下了深刻的印象，使我十分敬佩，终身受益。常老师在学术研究上给了我许多指导，这是我学习和科研工作顺利进行的基础。本文正是在常老师的引导和帮助下，才得以完成的。常老师为我提供了良好的研究环境，给我创造了很多学习和实践机会，使我在理论和实践方面都取得了很大的进步。导师对我的帮助和影响之大，无法言尽。在此，我再次向我的导师表示崇高的敬意和由衷的感谢！

感谢衣杨老师对我科研学习的指导和启发，以及对我生活的亲切关怀。衣老师的才思敏捷和严谨的治学精神令我难忘，是我终身学习的榜样。感谢路永和老师给予我的关心和帮助！感谢网络中心的郭清顺主任对我论文的指导。感谢实验室的张锋老师和孙雪冬老师，你们对科研的忘我钻研精神值得我学习，与你们的讨论交流，让我受益匪浅。感谢实验室陶乾同学，在实验室与你合作的过程中，当我们遇到学术难关时，你的钻研精神及追求完美的精神给我留下了深刻的印象，使我终生受益。同时也感谢实验室其他同学给予我的帮助。

特别感谢我的父母，谢谢你们给予我无私的爱。

感谢我的母校中山大学和母校里辛勤工作的各位老师和工作人员，谢谢你对我的关心和帮助。

感谢论文各位评审老师，你们的审稿意见有助于本文的完成。

最后感谢所有帮助过我的可敬的师长、同学和朋友！

附录 I 攻读博士学位期间发表的论文

[1] 顾春琴, 常会友, 衣杨, 路永和. 可解决多种复杂任务的过程挖掘算法[J]. 计算机集成制造系统. (已录用)

[2] 顾春琴, 陶乾, 常会友, 姚卿达, 衣杨. 基于混合自适应遗传算法的工作流挖掘优化 [J]. 计算机科学. (已录用)

[3] 顾春琴, 常会友, 衣杨, 许龙飞. 基于 SPN 的工作流时间性能分析方法[J]. 计算机研究与发展(suppl.), 2008, 10A:413-417.

[4] 顾春琴, 衣杨, 常会友, 容福丽, 王真. 时间约束工作流模型的可调度性验证研究[J]. 系统仿真学报. (已录用)

[5] Chunqin Gu(顾春琴), Huiyou Chang, Yang Yi. Overview of workflow mining technology. International Conference on Granular Computation, 2007, 347-350.

[6] Chunqin Gu(顾春琴), Huiyou Chang, Yang Yi. Workflow mining: extending the a-algorithm to mine the duplicate tasks. International Conference on Machine Learning and Cybernetics, 2008, 361-368.

[7] Yang Yi, Xiaoming Li, Chunqin Gu(顾春琴). Improved Particle Swarm Optimization for Multi-Objective Resource Allocation. Journal of Systems Engineering and Electronics, 2008, 19(5):959-964.

[8] Qian Tao, Huiyou Chang, Yang Yi, Chunqin Gu(顾春琴), Yang Yu. QoS Constrained Grid Workflow Scheduling Optimization Based on a Novel PSO Algorithm, The 8th International Conference on Grid and Cooperative Computing, 2009.(已录用)

[9] 顾春琴, 常会友, 衣杨, 陶乾. 基于克隆选择离散粒子群算法的工作流调度, 计算机研究与发展（在审）

附录 II 攻读博士学位期间参与的项目

[1] 广州市公安局信息化改造规划项目，2007－2008；(核心成员)

[2] 粤港关键重点突破项目：面向制造业信息化和电子政务领域的软件构件库平台 (编号：2006Z1-D6021)，2006－2008； (核心成员)

[3] 国家自然科学基金项目：基于人工生命计算的计算协同关键问题研究 (编号：60573159)，2006－2008；(主要成员)

[4] 广东省自然科学基金重点项目：计算机协同工作关键问题研究 (编号：05200302)，2005－2008；(主要成员)

[5] 广东省科技攻关项目：基于企业业务流程重构的综合管理平台软件 (编号：2004A10204007)，2005－2007；(主要成员)

[6] 广州市科技攻关项目：ERP 结合的 SCM&CRM 系统平台 (编号：200523-D0301)， 2005－2006；(主要成员)