# 基于混合自适应遗传算法的工作流挖掘优化

顾春琴[1,2] 陶 乾[2,3] 吴家培[1] 常会友[2] 姚卿达[2] 衣 杨[2]
(仲恺农业工程学院计算机科学与工程学院 广州 510225)[1]
(中山大学信息科学与技术学院 广州 510275)[2] (广州大学松田学院 广州 511370)[3]

**摘 要** 针对目前工作流挖掘算法采用局部策略而无法保证最优挖掘以及算法对噪声敏感的情况,提出了基于混合自适应遗传算法的工作流挖掘优化算法。首先定义了基本工作流网以及变迁的使能和点火规则,描述了过程模型;然后提出了过程模型转换成基本工作流网的算法,给出了衡量事件日志与过程模型的符合性的适应值评价函数;最后根据进化阶段以及个体相似度设计了混合自适应的交叉率和变异率。仿真试验结果表明,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。

**关键词** 工作流挖掘,过程挖掘,混合自适应遗传算法,基本工作流网,关联矩阵
**中图法分类号** TP18 **文献标识码** A

## Workflow Mining Optimization Based on Hybrid Adaptive Genetic Algorithm

GU Chun-qin[1,2] TAO Qian[2,3] WU Jia-pei[1] CHANG Hui-you[2] YAO Qing-da[2] YI Yang[2]
(College of Computer Science and Engineering, Zhongkai University of Agriculture and Engineering, Guangzhou 510225, China)[1]
(School of Information Science and Technology, Sun Yat-sen University, Guangzhou 510275, China)[2]
(Sontan College, Guangzhou University, Guangzhou 511370, China)[3]

**Abstract** Current workflow mining algorithm using local strategy couldn't ensure that a globally optimal process model was mined. The algorithm was also sensitive to noise. To solve the problems, a hybrid adaptive genetic algorithm (HAGA) was proposed. Firstly, Elementary Workflow net (EW-net) was defined. The enabling and firing rules of EW-net were given, and the process model was described. Secondly, a converting algorithm proposed was used to convert the process model to EW-net, and an evaluating function of the individual fitness was presented in order to measure the compliance between event log and mined process model. Lastly, hybrid adaptive crossover and mutation rates were designed according to evolution stage and parents' similarity. The simulation testing results demonstrate that the new algorithm has noise immunity and is more robust than α algorithm, and it can find better solution and converge faster than the simple genetic algorithm (SGA) employing general genetic strategy.

**Keywords** Workflow mining, Process mining, Hybrid adaptive genetic algorithm, EW-net, Causal matrix

## 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了更高的要求,需要支持跨平台、跨组织、跨地域甚至跨行业的协同工作及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[1]。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。通过工作流挖掘可以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。

Agrawal[2]最早在工作流管理环境下对过程挖掘进行研究。Cook 和 Wolf[3]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3种方法:神经网络、纯算法和 Markov 方法,并且指出后两种方法更有前景。在文献[4]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[5]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[6]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

到稿日期:2009-04-08 返修日期:2009-06-24 本文受国家自然科学基金(60573159)资助。
顾春琴(1979—),女,博士,主要研究方向为工作流挖掘、智能算法,E-mail:guchunqin@gmial.com;陶 乾(1978—),男,博士生,讲师,主要研究方向为网格计算、云计算、工作流调度;吴家培(1960—),男,副教授,主要研究方向为智能算法等;常会友(1962—),男,博士,教授,主要研究方向为协同软件研究等;姚卿达(1937—),男,教授,主要研究方向为数据库、知识挖掘等;衣 杨(1967—),女,博士,副教授,主要研究方向为智能控制理论和算法、基于知识系统的非确定信息的处理等。

提出了能挖掘出工作流网形式过程模型的 α 算法[7]。Medeiros 分析了 α 算法的局限并给出能处理短循环的算法[8]；Wen 改进了 α 算法，提出了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[9]。

上述文献普遍采用概率统计和归纳推理的方法来研究工作流挖掘算法。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[10]。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，目前在数据挖掘领域仍有广泛的应用[11,12]。文献[13]运用基本遗传算法(Simple Genetic Algorithm, SGA)来研究工作流挖掘，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，本文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来优化研究工作流挖掘。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，能显著提高解的质量和收敛速度。

## 2 基本网系统

**定义 1** P/T 系统是一个六元组 $\Sigma=(P,T,F,K,W,M)$。其中，

(1)$N=(P,T,F)$是一个网，称为 $\Sigma$ 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)$K:P\to\{1,2,3,\cdots\}$是库所集上的容量函数，若 $K(p)=w$，则表示库所 $P$ 的容量为无穷；

(3)$W:F\to\{1,2,3,\cdots\}$是弧集上的权函数；

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

在网中，库所节点用圆形表示，变迁节点用矩形表示。节点之间通过有向弧连接，圆形内的小黑点表示库所的托肯(token)。

**定义 2** 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 $\forall p\in p:M(p)\leqslant 1$，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 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 ${}^{\cdot}i=\varnothing$ 且 $M(i)=1$，库所 $o$ 是结束库所，即 $o^{\cdot}=\varnothing$；

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

图 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。

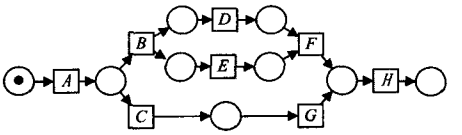


图 1 EW-net 实例

**规则 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 $M[t>$，也称 $M$ 授权 $t$ 点火或 $t$ 在 $M$ 授权下点火。

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

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

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

图 1 所示的 EW-net 中，由规则 1 可知 $A$ 是使能的；根据规则 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 2 是 3 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 $B,C$ 都是使能的，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点火。在并行路由结构中，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，并先后点火。

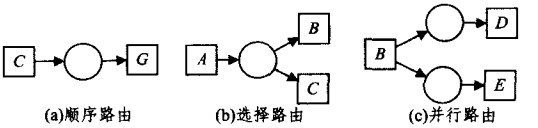


图 2 3 种基本路由结构图

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

图 1 所示的 EW-net 具有 3 个点火序列，分别为 $\sigma_1=ABDEFH$，$\sigma_2=ABEDFH$，$\sigma_3=ACGH$。

## 3 混合自适应遗传算法的设计

### 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)可具体描述为：

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

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，$CMatrix = [c_{ij}]_{n\times n}$，$c_{ij} = \begin{cases} 1, A_i, A_j \text{ 关联}, A_i \text{ 是起点}, A_j \text{ 是终点} \\ 0, \text{没有关联} \end{cases}$，若 $c_{ij}=1$，则 $A_i$ 是 $A_j$ 的前驱活动，记为 $A_i = PreActivity(A_j)$，且 $A_j$ 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。

图 1 所示的 EW-net 的过程模型定义如下：

$ASet = \{A, B, C, D, F, E, G, H\}$；

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

$OSet = \{\{BC\}, \{D, E\}, \{G\}, \{F\}, \{F\}, \{H\}, \{H\}, \{\emptyset\}\}$；

$ISet = \{\{\emptyset\}, \{A\}, \{A\}, \{B\}, \{B\}, \{D, E\}, \{C\}, \{FG\}\}$。

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 1**　$CreateOSet$

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 $O_i = \{\emptyset\}$；

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。算法如下：

**算法 2**　$CreateBoolExpr$

输入：$ODegree(A_i)$，$PostActivity(A_i)$

输出：$OSet$

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($i=1; i<k; i++$){

$a = \emptyset$；

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

若 $m_{ij}=1$，则 $a += B_j$；

若 $OSet$ 中不包含 $a$，则加入 $a$；}

$ISet$ 生成算法和 $OSet$ 类似，在此省略。

### 3.2　适应值函数

在过程挖掘中，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对比，即判断事件日志同过程模型是否吻合。目前对过程挖掘的符合性测试方法有 Medeiros[13] 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 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：

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

$$Punishment = \frac{\sum_{i=1}^{k} n_i \times NumArtificalToken_i}{\sum_{i=1}^{k} n_i \times NumActivity_i} \quad (2)$$

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及库所和变迁之间的有向弧。转换的主要算法如下：

**算法 3**　$ActivityConvertToWorkflow$

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 $P_{k+1}$（$k$ 为当前 $EW\text{-}net$ 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 $T_{ir}\rightarrow P_k$，$P_k\rightarrow T_{ii}$；

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

### 3.3 混合自适应遗传算法

#### 3.3.1 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为：

$$P(a_i)=Fitness(a_i)/\sum_{j=1}^{PopSize}Fitness(a_j) \tag{3}$$

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。

#### 3.3.2 混合自适应交叉变异操作

文献[13]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[14]。图3描述了在进化初始期、半成熟期、成熟期、后成熟期4个阶段交叉率和变异率的自适应调整曲线。其中 $\Delta P_x\in[-(1-P_x),1-P_x]$ 且 $\Delta P_m\in[-(1-P_m),1-P_m]$。

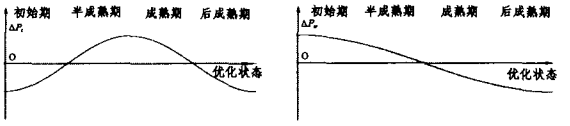


图3 在不同的进化阶段 $\Delta P_x$ 和 $\Delta P_m$ 的调整曲线[14,16]

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。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。个体相似度是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。设种群中的染色体为 $X$ 与 $Y$，其相似度为：

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

$X_i$ 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[15]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，但此算法在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。交叉率和变异率的自适应调整公式如式(5)—式(8)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。交叉后若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。式(5)和式(7)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。

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，如果活动的出、入逻辑与活动关联矩阵不一致，则修补活动的出、入逻辑。

$$P_c=\begin{cases}\dfrac{k_1(f_{max}-f')}{f_{max}-f_{avg}}, & f\geqslant f_{avg}\\ k_2, & f<f_{avg}\end{cases} \tag{5}$$

$$P_{ac}(X,Y)=\begin{cases}0, & \text{Similarity}(X,Y)\in[0.9,1]\\ p_c+\Delta P_c, & \text{Similarity}(X,Y)\in(0.1,0.9)\\ (P_c+\Delta P_c)(1-\text{Similarity}(X,Y)), & \\ & \text{Similarity}(X,Y)\in[0,0.1]\end{cases} \tag{6}$$

$$P_m=\begin{cases}\dfrac{k_3(f_{max}-f)}{f_{max}-f_{avg}}, & f\geqslant f_{avg}\\ k_4, & f<f_{avg}\end{cases} \tag{7}$$

$$P_{am}(X)=P_m+\Delta P_m \tag{8}$$

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。

### 3.4 混合自适应遗传算法流程

算法流程如图4所示。

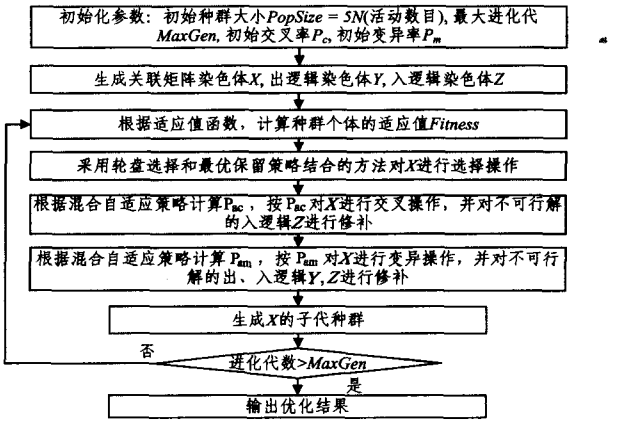


图4 混合自适应遗传算法流程

## 4 仿真试验与算法比较

为了验证算法的效果，进行了大量的计算机仿真试验。绝大多数结果都是令人非常满意的。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CPNTOOL分别对图5—图7所示的EW-net进行仿真，生成测试基础日志。在测试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 $\alpha=0.3$，$\beta=0.7$，$\gamma=0.2$，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8—图10是对不同规模的事件日志进行挖掘优化的最优适应值的走势曲线。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。

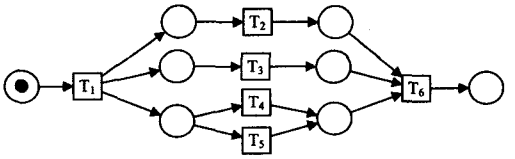


图5 含6个变迁的EW-net

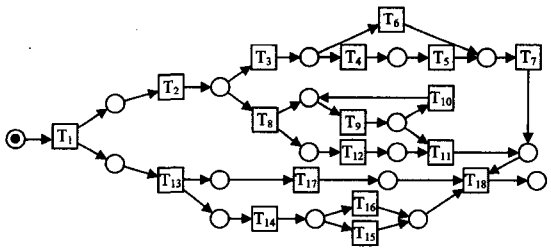

图 6　含 18 个变迁的 EW-net

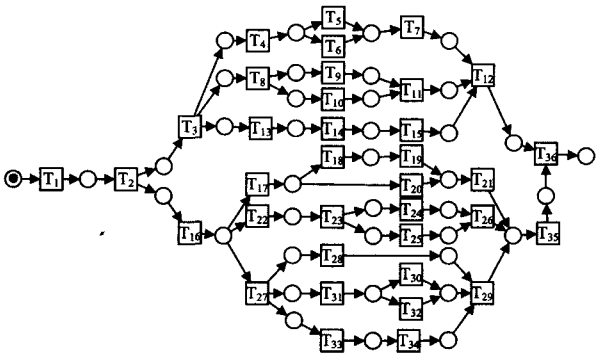

图 7　含 36 个变迁的 EW-net

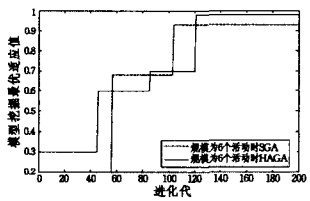

图 8　含 6 个活动的模型挖掘最优适应值走势图

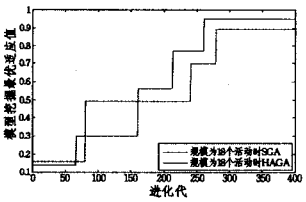

图 9　含 18 个活动的模型挖掘最优适应值走势图

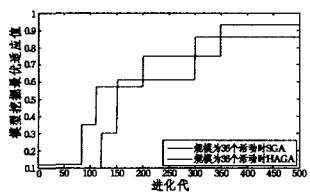

图 10　含 36 个活动的模型挖掘最优适应值走势图

表 1 对比了运用 $\alpha$ 算法[7]、Medeiros[13]提出的 SGA 算法以及本文提出的混合自适应遗传算法(HAGA)得到的模型挖掘优化最优值、达优率和执行时间。从表 1 中可以看出，HAGA 比 $\alpha$ 算法、SGA 获得了更好的最优值；对比 SGA，HAGA 具有更高的达优率，且执行时间也有所降低。

表 1　工作流挖掘优化算法比较

| 规模 | 最优适应值 | | | 达优率 | |
|---|---|---|---|---|---|
| | $\alpha$ | 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 |

**结束语**　目前工作流挖掘算法大都采用局部策略并且存在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 4 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，这就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，提出了用混合自适应遗传算法进行工作流挖掘优化。仿真试验表明，本算法与 $\alpha$ 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，本算法在提升解的收敛速度和质量方面有明显的优势。

## 参 考 文 献


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

[2] Agrawal R, Gunopulos D, et al. Mining process models from workflow logs [A]// The 6th International Conference on Extending Database Technology [C]. Valencia, Spain. Berlin: Springer-Verlag, 1998: 469-48

[3] Cook J E, Wolf A L. Discovering models of software processes from event-based data [J]. ACM Transaction on Software Engineering and Methodology, 1998, 7(3): 215-249

[4] Cook J E, Wolf A L. 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

[5] Cook J E, Wolf A L. 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

[6] Herbst J, Karagiannis D. 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

[7] van der Aalst W M P, Weijters A J M M, et al. Workflow mining: discovering process models from event logs [J]. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128-1142

[8] de Medeiros A K A, van Dongen B F, et al. Process mining: extending the α-algorithm to mine short loops [M]. BETA Working Paper Series WP 113, Eindhoven: Eindhoven University of Technology, 2004

[9] Wen Li-jie, Wang Jian-min, et al. 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

[10] Hochbaum D. Approximation algorithms for NP-hard problems [M]. Berkeley, CA: PWS Publishing Company, 1997

[11] 许文杰，刘希玉. 基于改进免疫遗传算法的聚类分析研究与应用[J]. 计算机科学，2008，35(1)：204-206

[12] 戴文华，焦翠珍，何婷婷. 基于并行遗传算法的 K-means 聚类研究[J]. 计算机科学，2008，35(6)：171-174

[13] de Medeiros A K A, Weijters A J M M, et al. Using genetic algorithms to mine process models: representation, operators and results [M]. BETA Working Paper Series, WP 124, Eindhoven: Eindhoven University of Technology, 2004

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

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

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