工作流模型复杂控制结构构造方法

李海波
(华侨大学计算机科学与技术学院 厦门 361021)

摘 要 当工作流模型中的业务规则语义复杂时,控制结构的正确性很难由人工保证。然而自动构造的方法尚少,因此首先引入层次清晰、能满足工作流模型合理性许多性质的块结构化(Block Structured)控制结构,通过分析工作流模型中块的拓扑结构,分别定义出控制结构的分支和汇合应遵循的语法规则,根据业务规则语义给出分支控制结构的构造算法;再基于此,采用权重法给出汇合控制结构的构造算法,最后通过示例验证构造过程。结果显示提出的方法具有很强的通用性,不依赖于具体的建模方法。

关键词 工作流模型,控制结构,块结构

中图法分类号 TP311 **文献标识码** A

Construction Method of Workflow Model Complex Control Structure

LI Hai-bo
(College of Computer Science&Technology, Huaqiao University, Xiamen 361021, China)

Abstract It is difficult to guarantee correctness of workflow model control structure by artificial, when case semantic of business rules in workflow model changes more complicated. Methods of construction automatically are less. At first well block structured control structure was introduced, which has lots of useful properties in model soundness. By analyzing topology of blocks in workflow models, branch and merge control structures and syntax rules they should follow were defined respectively. Algorithms of branch control structure were proposed according to semantics of business rule. Then based on this, algorithms of merge ones were proposed using all branch weights. Finally a practical example was given to verify the constructing process of correct block structure. This method is independent of specific modeling method, so that has three advantages, complete commonality, modeling-independent and wide applicability.

Keywords Workflow model, Control structure, Block structure

1 引言

模型驱动的企业应用软件的优势在于当用户需求不断变化时,更多地是修改模型而不是软件,最终通过模型到可执行软件的层层映射来满足。其中,工作流技术已经成为实现业务过程自动化的核心技术,把过程逻辑从过程执行中独立出来描述成模型是工作流管理系统的主要特点,模型最终映射成工作流运行时。构造一个复杂的、正确的工作流模型往往需要两个步骤:建模和模型验证。工作流模型的拓扑结构直接取决于各种控制结构的组合,且间接取决于不同的业务规则。一旦业务规则语义变得复杂,其正确性就很难由人工保证,需要进一步验证。在过去的研究中[1]曾列举了几类导致错误控制结构的语义关系,如语义遗漏、语义重叠、语义冗余,它们可验证语义到模型结构之间映射的完整性,但还不能验证控制结构的正确性,即并发、选择等控制结构。而结构正确性通常验证的是工作流模型中出现的死锁、无同步、无终止活动或无初始活动等结构冲突[2]。但是,目前的方法基本不考虑业务规则的语义,而是直接进行结构性验证,这无益于模型驱动的软件开发方法。

工作流控制结构的构造方法中,目前最多的就是基于日志挖掘的方法,如 Hammori 等人[3]提出的 MergeSeq、SplitSeq 和 SplitPar 算法。Aalst 等人[4]采用过程挖掘的方法处理任务之间的依赖关系,但其无法显示出 Split 和 Join 等关系,也无法处理重复任务。此外,基于启发式规则的算法也取得了很好的效果,如文献[5]的方法可处理非自由选择结构;文献[6]能处理重复任务,但其不能处理复杂结构;文献[7]的方法在准确性方面还有待提高。其他还有基于遗传算法的方法,如文献[8,9]把问题转化为优化问题挖掘最符合日志的模型,但模型具有准确性方面的问题。工作流模型验证方法的研究主要包括基于图形展开及图形归约的验证方法[10,11]、基于 Petri 网的验证方法[12,13]、基于逻辑的验证方法[14]等。这些验证方法可以通过不同的形式检验工作流模型的不合理部分,但大多不适合复杂的工作流模型。文献[15]最早给出了不同类型控制结构的正确性验证方法,但其不能支持异种控

到稿日期:2012-01-07 返修日期:2012-04-08 本文受福建省高校产学研合作科技重大项目(2010N5008),厦门市科技计划创新项目(3502Z20110013),泉州市科技计划项目(2011G5),华侨大学基本科研业务费专项基金(JB-ZR1147),国务院侨办科研基金项目(10QZR07)资助。

李海波(1972—),男,博士,副教授,主要研究方向为工作流、服务计算技术,E-mail:lihaibo@hqu.edu.cn。

制结构嵌套的情形，也不能验证块结构中汇合控制结构，只能构造简单的控制结构正确性。文献[16]中也给出了不同类型Block的识别算法，但其没有考虑到条件无关性和并发执行的情况，因此适用范围小。

针对上述研究的不足，基于过去的研究基础[1]，本文根据ECA描述的业务规则直接构造复杂的工作流控制结构。该方法通过分析模型中业务规则间的语义关系，采用算法构造复杂的嵌套块结构，而且支持异种类型结构的嵌套，方法具有通用性。

2 工作流模型控制结构的拓扑分析

2.1 控制块

块结构化(Block Structured)不是一个新概念，它因能保证工作流模型具有良好的结构(well-structured)而经常被采用。Aalst给出了EPC模型的结构良好性(Well-structured)的定义：An event-driven process chain is well-structured iff for any pair of connectors $c_1 \in C_S$ and $c_2 \in C_J$, such that one of the nodes is in $C^{\wedge}$ and the other in C_{XOR} and for any pair of elementary paths p_1 and p_2 leading from c_1 to c_2, $\alpha(p_1) \cap \alpha(p_2) = \{c_1, c_2\} \Rightarrow p_1 = p_2$. The alphabet operator α is defined as follows. If $p = \langle n_1, n_2, \cdots, n_k \rangle$, then $\alpha(p) = \{n_1, n_2, \cdots, n_k\}$。现实中的业务过程应该尽可能地避免非结构良好(non-well-structured)的工作流模型[16]。

控制结构代表工作流的各种执行模式，工作流管理联盟(WfMC)定义了sequence、iteration、AND (splits)、XOR (splits)、AND (joins)和XOR(joins) 6种控制结构[17]，分别为顺序、循环、并发分支、排他分支、并发汇合和排他汇合，并且受到多数工作流管理系统的支持。另外，还有一种常见的控制结构OR，其使用方法相当于WfMC中带条件的AND结构。由于OR类型总可以通过无条件的并发结构AND和排他结构XOR的组合以及条件cond表示[2]，即OR=f(AND, XOR, cond)，因此后面的讨论只涉及AND和XOR两种控制结构类型，即AND是无条件的并发结构，XOR是多选一结构。工作流模型描述的是活动间的时序关系，构成了以活动为节点的有向图，并由控制结构及条件协调活动的执行。

2.2 工作流模型定义

各种Block之间可以嵌套，AND Block和XOR Block可以被完全地嵌套在其他的AND Block和XOR Block里，但是具有嵌套关系的两层非顺序(none-sequential)的Block不能有重叠部分，这称作工作流模型的嵌套规则[16]。我们的研究就是建立在的嵌套块结构之上，如图1(e)、(f)所示的嵌套块结构。

定义1(工作流模型图) 工作流模型图表示成一个有向图$G_M = \{V, E, C, COND\}$，V表示活动节点的集合；E是有向边$\langle a_i, a_j \rangle$的集合，表示活动a_i先于a_j执行；$C = \{C_{AND}, C_{XOR}\}$是两类控制结构C_{AND}和C_{XOR}的超集，类型分别表示为AND和XOR，对于任意一个控制结构$c \in C$，类型$T(c) \in \{AND, XOR\}$，并且满足$C = C_{AND} \cup C_{XOR}$；COND是条件谓词集合，$cond_i \in COND$标识在$\langle a_i, a_j \rangle$上表示活动a_i执行完后依据条件谓词$cond_j$的结果来确定是否执行a_j，表示成 $a_i \xrightarrow{cond_j} a_j$。

这种行为规则也适合采用ECA模型来描述，如 $a_i \xrightarrow{cond_j} a_j$相应于两条ECA描述的业务规则的触发。在块结构的工作流模型图G_M中，对于任意一个控制块B，都有 $a \xrightarrow{cond_i} a_i$，其中$\langle a, a_i \rangle \in E, a \notin B, a_i \in B, i = 1, \cdots, n$，此时可以表示成 $a \xrightarrow{Cond} B$，其中$Cond = \{cond_i \mid i = 1, \cdots, n\}$。控制块$B$也可以由规则集合$R = \{r_i \mid i = 1, \cdots, n\}$描述，本文分支控制结构的构造就是建立在ECA模型描述的业务规则集合基础上的。

相应地，结构良好的工作流模型图的控制结构应该满足如下语法规则：每一个非顺序的(non-sequential)Block都存在一个起始控制结构c_1和结束控制结构c_2，并且$|succ(c_1)| > 1$，$|pred(c_2)| > 1$，$T(c_1) = T(c_2)$。succ(n)表示n的输出节点的集合，称作后置集；pred(n)表示n的输入节点的集合，称作前置集；T是模型中控制结构到类型的映射，即$T \in C \rightarrow \{AND, XOR\}$。图1(a)、(b)是错误的控制结构，(c)－(f)是结构良好的控制块。

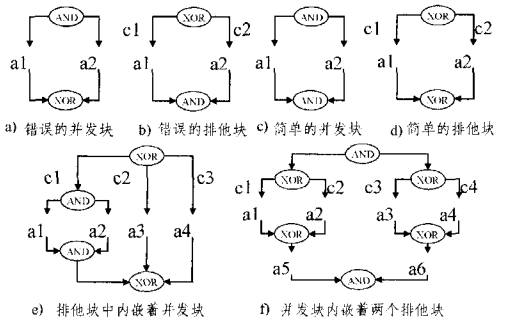


图1 各种结构良好的控制块

根据结构良好性规则，应该首先验证控制结构分支部分的正确性，再验证汇合部分的正确性。分支部分的验证就是依据两种控制结构AND和XOR，以及XOR相应的条件集合，来判断这几种元素是否组合出正确的拓扑结构。汇合部分验证是否与分支部分的拓扑结构匹配，并满足结构良好性规则。

3 控制块的构造

3.1 分支结构的构造

控制块的分支可以是简单的，如图1(c)、(d)所示，也可能是复杂的，如图1(e)、(f)所示的嵌套控制块，其拓扑结构取决于所有控制结构的类型和条件。工作流模型图G_M可以看成是由这些控制块组成的顺序结构，首先定义该控制块中的分支集合。

定义2(控制块的分支结构B_{split}) 对于工作流模型图中任意控制块$B_i \in G_M$，称 $a_i \xrightarrow{Cond_i} B_i$ 为分支结构B_{split}，其中$Cond_i \subseteq COND$，$Cond_i = \{cond_i \mid i = 1, \cdots, n\}$，活动$a_{ij} \in B_i$，$j = 1, \cdots, n$。

例如有如下业务规则集合：$R = \{r_0, r_1, r_2, r_3, r_4\}$，$r_0$表示订单变更时要修改订单；$r_1$表示当订单已生产时，成品和原材料退库；$r_2$表示当订单还未生产时，调整生产计划；$r_3$表示当订单还未生产时，调整采购计划；$r_4$表示当订单还未生产时，调整发货计划。依据条件活动的执行序列如图2所示，显然a_1, a_2, a_3, a_4组成一个控制块，但块内的逻辑关系需要用控制结构正确地表达出来。此时，把不包含控制结构的Block记为B，用控制结构表达出来的Block记为B'。

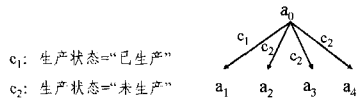


图 2 分支规则集合示意图

B_{split} 中的条件集合可能形成嵌套的控制结构，除图 1 所示的几类简单和复杂结构外，还有以下两种基本情况：

(1)存在与条件无关的规则节点。这种规则节点具有条件无关性，应该和其他节点并发执行，如图 3(a)所示的嵌套结构。

(2)相同的条件下触发了不同的规则。如图 3(b)所示，a_1 和 a_2 并发被执行，此时需要两种控制结构级联使用。

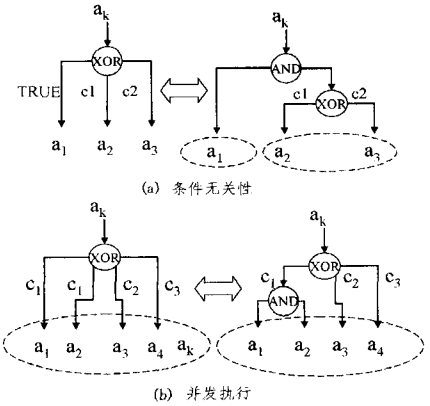


图 3 不同类型控制结构的嵌套

文献[16]给出了不同类型 Block 的识别算法，但是它没有考虑到上述两种情况，具有局限性。因此，本文给出 Block 的分支控制结构验证算法，其通过构造出完整的分支控制拓扑结构来验证业务规则语义到模型的正确性。

算法 1 CreateSplitB——构造分支类型的控制结构

输入：不包含控制结构的块 B_{split}；

输出：用控制结构表达出来的块 $B_{split'}$；

```
Begin
  Bsplit'←Bsplit; //为所有的 Bsplit 赋初值
  Cond←{condi|condi∈Bsplit}; //为所有的条件集合 COND 赋初值
  For (all ai⊆Bsplit) do //处理每个 Bsplit
If (condi=TRUE && |condi|=1) then //对于无条件触发一个节点的处理
    Bsplit'←Bsplit'−{⟨r,Rk⟩|Rk(RSplit−i};
    Bsplit'←Bsplit'∪{⟨r,rk⟩|rk∈Rk,Rk∈RSplit−i, |Rk|=1};
Endif
  If (condi=TRUE && | condi|>1) then //对于无条件触发多个节点的处理
    Bsplit'←Bsplit'∪{c|T(c)=AND};
Endif
    If (condi≠TRUE) then //带条件触发的情形
  Bsplit'←Bsplit'∪{c|T(c)=XOR};
  If (|condi|=1) then Bsplit'←Bsplit'∪{⟨c,rk⟩|c∈CXOR,rk∈Rk};
    If (|condi|>1) then
  Bsplit'←Bsplit'∪{c|T(c)=XOR,};
  Bsplit'←Bsplit'∪{c'|T(c)=AND};
Endif
  Endif
Endfor
End CreateSplitB
```

算法中|*cond*|表示条件 *cond* 出现的次数。算法的时间复杂度为 O(n)，n 表示集合 B_{split} 中活动节点的个数。例如，以图 2 为例，经过算法 SplitBSplitB 转换后，产生如图 4 所示的控制结构嵌套。

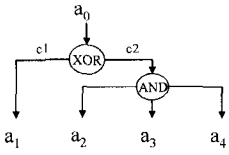


图 4 控制结构构造示例

3.2 汇合结构的构造

由于汇合控制结构不带有条件，只与分支控制结构相呼应，因此，依据条件谓词确定汇合节点十分困难。虽然文献[16]给出了 Block 的识别算法，但方法没有考虑控制结构嵌套的情况，也就不能准确地构造出 Block 中相应的汇合控制结构，这与我们提出的构造方法有所区别。

结构化块要求分支汇合结构必须满足：每一个分支控制结构都应该对应一个同类型的汇合控制结构[4,11]。图 1(a)、(b)分别显示了错误的 Block：后序活动会处于永久等待状态或者丢弃了一个活动的执行结果。下面讨论用于汇合的控制结构应该遵循的语法规则。

定义 3 控制块的汇合结构 B_{join} 对于工作流模型图中任意控制块 $B_i \in G_M$，称 $B_i \rightarrow a_i$ 为汇合结构 B_{join}，其中 $a_{ij} \in B_i$，$j=1,\cdots,n$。

汇合结构是对分支结构的对应，因此不需要考虑条件因素。汇合规则集合同样需要考虑控制结构嵌套的情况，如图 5 所示。在这两种情况下，汇合控制结构不可能是一个，也不会是同一类型。按照嵌套规则，图 5(a)的每个图中都存在两个分支控制结构，所以也应该存在两个汇合控制结构与之对应，如图 5(b)所示。

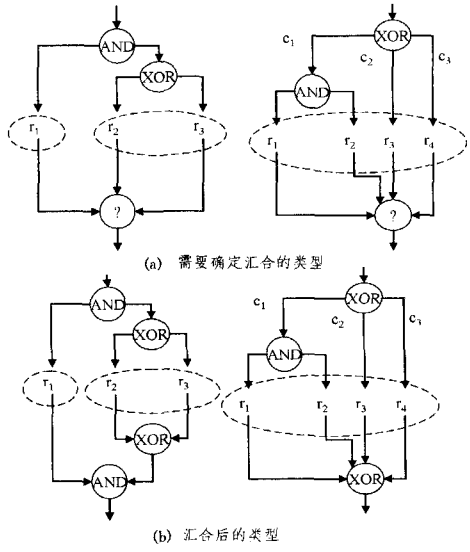


图 5 不同类型的控制结构

下面给出汇合结构的构造算法，算法的思路如下：由于通过算法 CreateSplitB 已经构造出分支控制结构，因此不必采取文献[16]中为所有节点都分配一个权重(water-level)的方法，只需要给分支控制结构分配权重即可；然后依据分支控制结构和权重值确定汇合节点的位置和类型。再介绍权重计算方法：权重用来标识一个网络图节点所在的层次，节点出现分

支后，所有后继节点的权重就会变小。同样，汇合后的权重又会增加。如果节点的权重为 w，存在 k 个后继分支节点，则每个后继分支节点的权重为 w/k。同样，一个节点的权重等于所有直接前置节点权重之和。下面给出权重的计算方法 Weight，其中 PutSet(SET，s)是集合基本操作，表示放元素到集合中。

算法 2 Weight-计算工作流模型图各节点的权重

```
输入：只包含分支控制结构的工作流模型图 GM
输出：各分支控制结构的权值集合 w-list
Begin
for all v do w(v)=0.0;//初始化所有节点权重为 0
w(s)=1.0; w-list={1.0}; QUEUE={s};//准备初始节点
while (QUEUE≠Ø) do
    v←QUEUE;//处理节点 v
    mark(v);//标记节点 v
    for (all v'∈succ(v)) do// v 的所有后继节点 v'均为某 block 中
       的节点
    w(v')= w(v')+ w(v)/|succ(v)|;//以平均数计算权重
    if (w(v')∉w-list) then w-list(w(v');
    if (all prede(v') are marked) then QUEUE←v';
    end for
end while
end for
end Weight
```

图 6 的实例表示了所有节点的权值，比如最内层的 Block 中，节点 a_5，a_6，a_7 的权重均为 0.5/3=0.17。由于采取了深度优先搜索方法，并且标记了访问过的节点，因此算法 Weight 的时间复杂度为 $O(n^2)$，n 表示节点个数。与文献[16]不同的是，分支控制结构节点也标记上了权值。优点：1)分支控制结构的确定本身就已经考虑了嵌套的情况；2)利用分支控制结构去构造汇合控制结构，可以简化汇合控制结构构造算法的复杂度，可以更准确地利用节点的权重进行识别，然后向外层扩展。

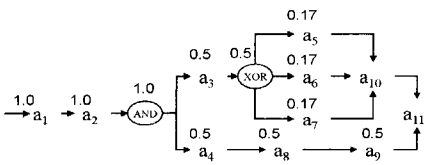


图 6 工作流模型图权重的计算

下面依据带权值的工作流模型图 G_M，给出汇合(AND-Join 和 XOR-Join)类型的控制结构验证算法。

算法 3 CreateJoinB——构造汇合类型的控制结构

```
输入：经算法 CreateSplitB 转换的工作流模型图 GM，分支控制结构的
      权值集合 w-list;
输出：包含汇合控制结构 AND 和 XOR 的工作流模型图 GM';
Begin
while w-list≠Ødo
b=1.0; //初始化成最大权值单位 1.0
for (all v∈GM && (v∈CAND || v∈CXOR) && v∈CS&& c is not
   marked) do//处理模型图中所有的分支控制节点
  if (b>w(v)∈w-list) do b= w(v);//根据权值查找汇合节点
end for
  if (b=1.0) then GM'←GM//权值为 1 表示已处理到最外层 block
    return GM'
  end if
//确定汇合节点的类型，是 AND 还是 XOR
for (all v∈GM' && (v∈CAND || v∈CXOR) && b= w(v) && v∈
    CS) do C←v; //根据模型图中所有的分支控制节点集合 C 进行
    处理
for (all c∈C) do //处理所有分支控制节点
V←succ(c);
for (all v∈V ) do //查找 v 之后、控制节点之前的顺序路径上最后规
   则节点
  if (|succ(v)|=1 && |pred(succ(v))|=1 && v∉CAND&& v∉
    CXOR)) then v'←succ(v);
    V←V-{v};//节点 v 已处理，舍弃
    V←V∪{v'};//增加最后的节点 v'
  end if
end for
mark(c); //节点 c 标记成已处理过
//增加同类型控制结构节点到模型 GM'中
for all v∈V do GM' ←⟨c,succ(v')⟩;
GM'←⟨ v',c⟩;
end for
  end for
  for all (w(v)∈w-list && w(v)=b) do w-list=w-list-{w(v)};
    //处理下一个规则节点的权值
end while
end CreateJoinB
```

算法 CreateJoinB 采取了深度优先搜索方法，只要找到结束规则节点，就增加控制结构节点，因此算法只需要标记访问过的分支控制结构节点，时间复杂度为 $O(n^2)$，n 表示分支控制结构节点个数。算法 CreateJoinB 还考虑了 Block 之间同步的情况，从时间复杂度上看优于文献[16]的方法。由于算法按照工作流模型图中节点进行递归计算，节点个数有限，因此递归次数也有限，能正确终止。图 7 显示了算法被调用的过程。

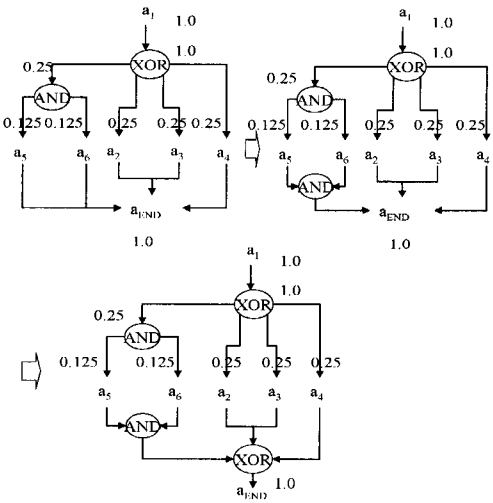


图 7 汇合控制结构构造示例

3.3 循环结构的构造

工作流模型图中，结构的良好性保证了循环结构不允许跨控制块。同时，它属于一种特殊的 XOR 控制块，多个分支汇合于至少 2 个以上不同的汇合控制结构上，由此形成新一层控制块。之前的分支和汇合构造算法已经支持这种情况。

4 实验验证

以客户订单变更为例，表1给出了相关条件谓词的形式化表示，表2给出了业务规则及形式化表示。当客户订单发生变更时，视表2列出的不同情况确定要执行的活动，因此这是一个复杂嵌套的控制块。

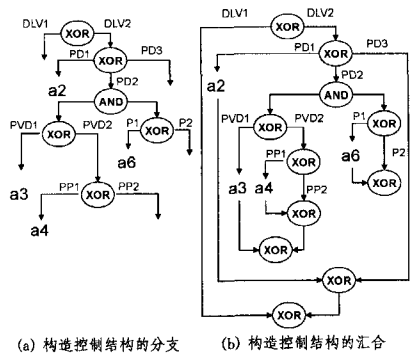
表1 客户订单变更业务涉及到的条件谓词及形式化表示

条件谓词语义	表达式
是否已发货	DLV1：发货状态＝已发货；DLV2：发货状态＝未发货
是否已生产	PD1：生产状态＝已生产未完成；PD2：生产状态＝未生产；PD3：生产状态＝已生产完
供应商是否已发货	PVD1：供应商状态＝供应商已发货；PVD2：供应商状态＝供应商未发货
是否已列采购计划	PP1：采购计划状态＝已列两月采购计划或者周采购计划； PP2：采购计划状态＝未列两月采购计划或者周采购计划
是否已列生产计划	P1：生产计划状态＝已列两月生产计划或者周生产计划； P2：生产计划状态＝未列两月生产计划或者周生产计划

表2 订单变更业务规则

规则	约束条件	执行的活动	
r1	客户要求订单变更时，企业已经给客户发货	无	DLV1
r2	企业还没给客户发货，已经开始生产但还未完成	生产变更 a2	DLV2 and PD1
r3	企业未发货，未生产，但是采购供应商已经发货	采购无法变更 a3	DLV2 and PD2 and PVD1
r4	企业未发货、未生产、采购供应商还未发货，但已经列入采购计划（两月或者周采购计划之一）	调整采购计划 a4	DLV2 and PD2 and PVD2 and PP1
r5	企业未发货、未生产、采购供应商还未发货，也未列入采购计划	无	DLV2 and PD2 and PVD2 and PP2
r6	企业未发货，未生产，但已经列入生产计划（两月或者周生产计划之一）	调整生产计划 a6	DLV2 and PD2 and P1
r7	企业未发货，未生产，也未列入生产计划	无	DLV2 and PD2 and P2
r8	企业未发货，已生产完	无	DLV2 and PD3

第1步，通过算法 CreateSplitB 创建分支结构，得到如图8(a)所示的拓扑结构。



(a) 构造控制结构的分支　(b) 构造控制结构的汇合

图8 分支和汇合控制结构构造示例

第2步，通过算法 CreateJoinB 构造汇合部分的控制结构，得到如图8(b)所示的拓扑结构。

以上两个步骤得到的拓扑结构，经过文献[1]的方法可验证其完整性(本文略)，进而实现本例中的复杂控制结构。

下面讨论分支(算法1)和汇合(算法3)这两个主要算法的效率。算法1虽然要对每个控制块进行处理，但效率还是取决于活动节点个数。算法3基于权值的预处理，加之对处理后的节点进行了标记，因此其效率实际上取决于深度优先算法的效率。分别考虑6个节点(客户订单变更例子)、30个节点(模拟)和100个节点(模拟)的规模下，算法1和算法3的执行时间(运行环境：JDK6.0，windows 7，Intel Core(TM) CPU 2.4GHz，内存2GB，32位操作系统)，如图9所示。对比结果表明，节点规模增大后，算法3的时间开销显著增大，但活动节点少于30，算法效率仍在可接受范围内，因为实际应用中也不建议工作流模型过于复杂，过于复杂的业务过程通常拆分成若干个工作流模型进行描述。

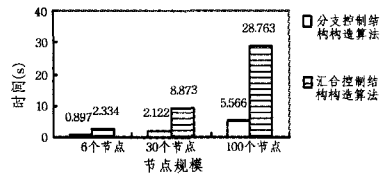


图9 算法效率分析

结束语 块结构化的控制结构由于层次清晰并具有很多优良的性质，因此一直受到各种工作流建模方法的推崇。本文在过去的研究基础上，提出了复杂嵌套的控制结构的构造方法。方法突破了文献[16]只能处理简单控制块的限制，可用于各类建模工具以及模型验证工具，是对结构化工作流建模验证技术的重要补充。下一步的工作拟采用模型化简的方法进一步提高算法3的运行效率。

参考文献

[1] 李海波，战德臣，徐晓飞. 工作流业务规则语义的完整性验证技术[J]. 计算机研究与发展，2009，46(7)：1143-1151

[2] Van der A W M P，Ahm T H，Kiepuszewskib，et al. Workflow patterns[J]. Distributed and Parallel Databases，2003，14(3)：5-51

[3] Hammori M，Herbst J，Kleiner N. Interactive workflow mining requirements，concepts and implementations [J]. Data and Knowledge Engineering，2006，56(1)：41-63

[4] van der A W M P，Weijters A J M M，Maruster L. Workflow mining：discovering process models from event logs[J]. IEEE Trans on Knowledge and Data Engineering，2004，16(9)：1128-1142

[5] Wen Li-jie，van der AalstW M P，Wang Jian-min. Mining process models with non-free-choice constructs[J]. Data Mining and Knowledge Discovery，2007，15(10)：145-180

[6] 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 α 算法[J]. 计算机学报，2007，30(8)：1436-1445

[7] 陈信敏，滕少华. 一种发现复杂工作流结构的扩展 α 算法[J]. 计算机应用研究，2011，28(2)：536-540

[8] de Medeiros A L V E S A K，Weijters A J M M，van der Aalst W M P. Genetic process mining：an experimental evaluation[J]. Data and Knowledge Engineering，2007，14(2)：245-304

[9] Song Wei，Gao Dian-fang，Liu Qiang. Study on complicated workflow structure mining[J]. Journal of Software，2008，19(zk)：104-111

[10] 宋宝燕，王菊英，于戈. 基于图形展开及图形归约的过程模型验证方法[J]. 小型微型计算机系统，2005，26(6)：1073-1078

(下转第136页)

较大。这是很显然的，因为随着 k 值的增加，所需的路网扩展也增加。然而，不论 k 值是大还是小，CBRkNN 的性能均优于 NAV_B。

图 10 研究了对象的可移动性对算法运行时间的影响。当对象的可移动性增大时，CBRkNN 算法的处理时间将变长。这一点显而易见，因为对象频繁的位置更新会导致结果集的较大部分失效，从而增加维护代价。而 NAV_B 算法在对象的可移动性变化时，处理时间没有变化。这是因为 NAV_B 算法在每一个时间点都重新计算结果集，每一个时间点对它而言都是全新的。同样地，NAV_B 算法在不同的查询点可移动性下，其处理时间维持不变。

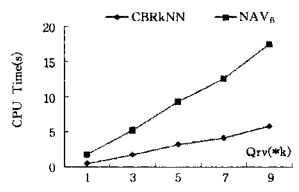


图 8 查询数的变化对算法性能的影响

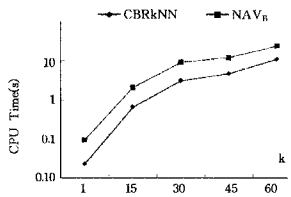


图 9 k 值变化对算法性能的影响

图 11 给出了各算法对于不同的查询点可移动性所表现出来的特征。如图 11 所示，当发生位置更新的查询点数目增加时，CBRkNN 算法的处理时间也增加。这是因为当查询点移动后，会使得它的 DLM 树部分或全部失效，从而触发失效部分的重建操作。

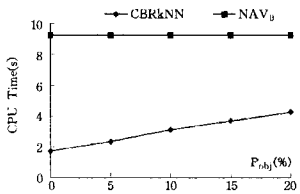


图 10 对象的可移动性对算法性能的影响

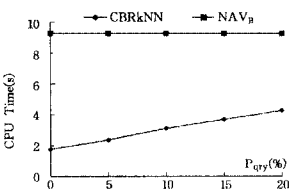


图 11 查询点的可移动性对算法性能的影响

平均来说，所提 CBRkNN 的性能优于参照算法 NAV_B 1.95 倍。而且，CBRkNN 对 k 值、对象数、查询数、对象和查询的可移动性等参数的变化都是健壮的。因此，它是高效和可扩展的。

结束语 本文首次讨论了路网中双色集上反向 k 近邻查询连续监控问题(CBRkNN)。所提 CBRkNN 查询处理方法改造了文献[2]所提出的两层多路树(DLM-tree)，来表示 CBRkNN 查询 q_A 的反向 k 近邻的有效监控范围。为进一步提高算法的效率，提出了相关引理来尽可能地削减查询空间，并提出一系列技术来高效地处理范围树和验证树的更新。实验结果表明，所提出的方法比直接方法高效约 1.95 倍。

参 考 文 献

[1] Sun Huan-liang, Jiang Chao, Liu Jun-ling, et al. Continuous Reverse Nearest Neighbor Queries on Moving Objects in Road Networks[C]// The Ninth International Conference on Web-Age Information Management. 2008:238-245

[2] Li Guo-hui, Li Yan-hong, et al. Continuous Reverse k Nearest Neighbor Monitoring on Moving Objects in Road Networks [J]. Information Systems, 2010, 35(8):860-883

[3] Korn F, Muthukrishnan S. Influence Sets Based on Reverse Nearest Neighbor Queries[C]// Proc. ACM SIGMOD '00. 2000: 201-212

[4] Yang C, Lin K-I. An Index Structure for Efficient Reverse Nearest Neighbor Queries [C]// Proc. 17th Int'l Conf. Data Eng. (ICDE' 01). 2001:485-492

[5] Stanoi I, Agrawal D, El Abbadi A. Reverse Nearest Neighbor Queries for Dynamic Databases[C]// ACM SIGMOD Wo-rkshop Research Issues in Data Mining and Knowledge Discovery (DMKD). 2000:44-53

[6] Tao Y, Papadias D, Lian X. Reverse kNN Search in Arbitrary Dimensionality[C]// Proc. 30th Int'l Conf. Very Large Data Bases (VLDB '04). 2004:744-755

[7] Xia T, Zhang D. Continuous Reverse Nearest Neighbor Monitoring [C]// Proc. 22nd Int'l Conf. Data Eng. (ICDE '06). 2006: 77

[8] Kang J M, Mokbel M F, Shekhar S, et al. Continuous Evaluation of Monochromatic and Bichromatic Reverse Nearest Neighbors [C]// ICDE. 2007:806-815

[9] Wu Wei, Yang Fei, Chan C-Y, et al. Continuous Reverse k-Nearest-Neighbor Monitoring[C]// The 9th International Conference on Mobile Data Management. 2008:132-139

[10] Yiu M L, Papadias D, Mamoulis N, et al. Reverse Nearest Neighbors in Large Graphs[J]. TKDE, 2006, 18(4):540-553

[11] Safar M, Al-Saleh A. PINE based RNN queries in Road Networks[C]// Campos do Jordão, São Paulo, Brazil. VII Brazilian Symposium on Geoinformatics, INPE. November 2005:356-367

[12] http://www.fh-oow.de/institute/iapg/personen/brinkhoff/generator/

(上接第 110 页)

[11] 王金朋，侯贵宾，邓成玉，等. 基于 Pi-演算的扩展有向图工作流模型及验证[J]. 计算机工程与设计，2010，31(10)：2399-2404

[12] 周建涛，史美林，叶新铭. 一个基于 Petri 网化简的工作流过程语义验证方法[J]. 软件学报，2005，16(7)：1242-1251

[13] 胡乃静，赵亮，胡金化. 基于 Petri 网的工作流结构正确性化简验证方法[J]. 小型微型计算机系统，2007，28(6)：1076-1079

[14] Henry H B, Zhao LJ. Applying propositional logic to workflow verification [J]. Information Technology and Management, 2004, 5(3/4):293-318

[15] Sadiq W, Orlowska M. Modeling and verification of workflow graphs[R]. Tech. Rep. 386. Department of Computer Science, University of Queensland, 1996

[16] Bae J, Caverlee J, Liu Ling, et al. Process Mining by Measuring Process Block Similarity [C]// Business Process Management Workshops. 2006:141-152

[17] Lawrence P. Workflow Handbook 1997[M]// John Wiley and Sons. Workflow Management Coalition, New York, 1997