基于时间约束Petri网的一致性验证算法

刘林钢,姜 浩
(东南大学 计算机科学与工程学院,江苏 南京 210096)

摘 要:时间约束的一致性验证是保证工作流时间模型正确工作的前提,因而一致性验证的算法的精确度和复杂度关乎整个工作流时间模型的运行效率。文中简要介绍了时间约束一致性定义及约束关系的推理规则,提出了一种简洁有效的时间约束一致性验证算法并分析了算法的时间复杂度。该算法借助于T-组件网和时间约束流图,能有效验证时间约束Petri网中存在的各种时间冲突,以保证工作流时间约束模型的建立及运行等各个阶段的正确性,对业务流程的建立、维护和优化都具有重要的参考意义。

关键词:工作流;时间约束一致性;时间约束工作流网;验证算法
中图分类号:TP301.4 **文献标识码**:A **文章编号**:1673-629X(2010)01-0058-05

Verification Algorithm of Consistency Based on Time Constraint Petri Nets

LIU Lin-gang,JIANG Hao
(Dept. of Computer Sci. and Eng.,Southeast University,Nanjing 210096,China)

Abstract:To verify the consistency of time constraints is to ensure that the time model of workflow work correctly. The accuracy and complexity of the algorithm is closely related to the operating efficiency of the entire time model. Firstly introduced briefly the definition for consistency of time constraints and the reasoning rules of constraints relation. Then propose a simple and effective verification algorithm for time constraints, and the time complexity analysis was presented also. To use the T-component nets and time constraints flow graph, the algorithm can detect all kinds of existing time conflicts in the time constraints workflow net (TCWFN). To ensure the correctness in various stages about time constraints model, it can promote the establishment, maintenance and optimization for the business process.

Key words:workflow;consistency of time constraints;TCWFN;verification algorithm

0 引 言

许多业务流程都有相对时间约束,包括活动的执行时间界限以及子进程和活动执行的绝对时间限制[1]。工作流系统中引入时间参数是业务流程管理的需要。但时间参数的引入不仅影响了原有系统的运行调度方式,而且可能出现时间约束的冲突,破坏流程定义的有效性[2,3]。一个工作流模型,即便具有正确的控制流逻辑,它也可能含有不一致的时间约束。因此,不论在过程的设计阶段,还是在过程的执行阶段,都需要对过程定义中的时间约束进行检验,以保证业务过程可以被有效地执行。

收稿日期:2009-05-25;**修回日期**:2009-08-12
作者简介:刘林钢(1983-),男,江苏泰州人,硕士研究生,研究方向为Web Service和工作流;姜 浩,副教授,研究方向为Web Service和工作流。

1 时间约束一致性及约束关系的推理

对于工作流过程定义中的时间约束,需要知道它们能否保证其一致性,也就是多个时间约束之间能否保证不产生冲突。先给出时间约束一致性的定义。

[定义1](时间约束一致性)[4]:

一个时间约束集与某一给定的工作流模型是一致的,当且仅当基于工作流时序控制逻辑及流程的运行状态,该集合所包含的所有时间约束是满足的。

这里的一致性仅仅是指存在活动可调度的一种可能性,即存在一种满足所有时间约束的可能性。这意味着当前某一时刻满足一致性要求的工作流程,未必将来也一定能保证执行结果的有效性。但是当前不满足一致性要求的工作流程必然会出现问题,从而造成资源的浪费。

从[定义1]中可以看出,时间约束的一致性是建立在与控制流一致的基础上的。对于变迁 t_1 和 t_2,如

果它们的实施顺序满足控制流要求,t_1 在 t_2 之前实施,则在时间约束 $e_{t_1} \prec^{\max}_{\min} e_{t_2}$ 中,$\max \geqslant \min \geqslant 0$。同时时间约束集本身也要保证互相满足,不产生冲突。

由于在过程定义中并不需要对所有活动施加时间限制,这就使得一致性验证所要提供的信息不完备,所以需要通过已知的时间信息进行推理,得出未知的时间信息。例如:对于三个事件 e_i, e_j, e_k,存在时间约束 $e_i \prec^{L'}_{S'} e_j$ 和 $e_j \prec^{L''}_{S''} e_k$,如果根据此约束进行推理,得出 $e_i \prec^{L}_{S} e_k$ 成立,即 $L > S$,则可以断定时间约束 $e_i \prec^{L'}_{S'} e_j$ 和 $e_j \prec^{L''}_{S''} e_k$ 是没有冲突的。下面给出几个基本的推理规则。

规则 1:顺序发生事件的时间关系推理。

令 e_i, e_j, e_k 为三个顺序发生的事件,如有 $e_i \prec^{L_{ij}}_{S_{ij}} \wedge e_j \prec^{L_{jk}}_{S_{jk}} e_k$,则 $e_i \prec^{L}_{S} e_k = e_i \prec^{L_{ij}+L_{jk}}_{S_{ij}+S_{jk}} e_k$。

规则 2:并行发生事件的时间关系推理。

这是一种"AND" 的时间关系,产生于工作流过程的并行结构(And - Split 和 And - Join 结构),其特点是,开始事件与结构的开始变迁相关,结束事件与结构的结束变迁相关。令 e_i 为并行分支的开始事件,e_j 为并行归并的结束事件,并行结构中有 k 条路径,如第 l 条路径的时间约束为 $e_i \prec^{L_l}_{S_l} e_j$,则整个并行结构的时间约束为 $e_i \prec^{\max(L_l)}_{\min(S_l)} e_j, l = 1,2,\cdots,k$。

规则 3:选择发生事件的时间关系推理。

这是一种"OR" 的时间关系,产生于工作流过程的选择结构(Or - Split 和 Or - Join 结构),其特点是,开始事件与结构的开始变迁相关,结束事件与结构的结束变迁相关。令 e_i 为并行分支的开始事件,e_j 为并行归并的结束事件,并行结构中有 k 条路径,第 l 条路径的时间约束为 $e_i \prec^{L_l}_{S_l} e_j$,则整个并行结构的时间约束为 $e_i \prec^{\max(L_l)}_{\min(S_l)} e_j, l = 1, 2,\cdots, k$。

通过以上几个基本的推理规则,可以得到过程活动之间未显式给出的时间约束,以便进一步进行时间约束一致性的验证。

2　时间约束流图及其循环的检查

要保证时间约束的一致性,其基本前提是工作流过程的控制流和时间约束流必须保持一致。因此,为了验证时间约束的一致性,首先需要得到过程模型中完整的时间约束关系。这一关系可以由时间约束流图来描述。

[定义 2](时间约束流图):

时间约束流图是一个二元组 $\sum = (V, E)$,其中 V 是活动的集合,E 为活动中事件的时间约束关系的集合,图中的路径表示相对时间约束的传递关系。

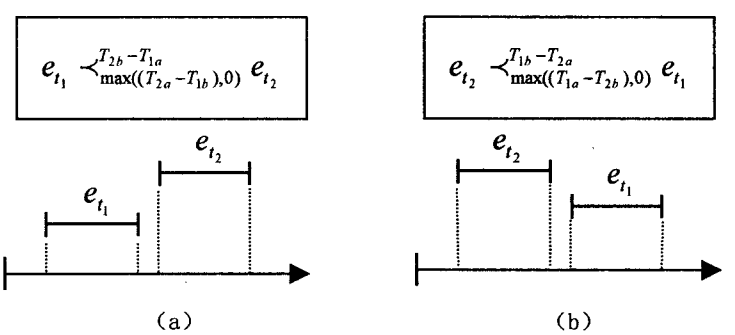
在时间约束流图中,边仅仅反映相对时间约束中的直接约束和间接约束关系。对于一个内部时间约束 $t_{start} \prec t_{end}$ 而言,其合理性就是要保证 $\max \geqslant \min \geqslant 0$,这已经隐含在图的结点内部。同时,图中也不反映绝对时间约束。将时间约束流图中某一活动的所有前驱活动的集合记为 $\cdot v$,它的后序活动的集合记为 $v \cdot$。时间约束流图对 TCWFN 和定义在其上的时间约束集通过以下步骤来得到[5]。

步骤 1:顶点复制。

将 TCWFN 中的顶点集合复制到 $\sum$ 中,即 $V = T$;

步骤 2:绝对时间约束的处理。

单个事件的绝对时间约束无法在过程的定义阶段判断其合理与否,只有在过程实例启动执行时才可能确定此时间约束能否被满足。但是当有多个绝对时间约束同时被定义时,却可以依据过程的控制流关系检查它们两两之间是否隐藏着不一致。对分别属于两个不同活动 t_1, t_2 的事件 e_{t_1} 和 e_{t_2},如果定义了它们各自的绝对时间约束 $T_{1a} \prec e_{t_1} \prec T_{1b}$、$T_{2a} \prec e_{t_2} \prec T_{2b}$,则可以建立两事件之间的相对时间约束。如果有 $(T_{2b} - T_{1a}) \geqslant \max((T_{2a} - T_{1b}), 0)$,那么,两事件之间相对时间约束为 $e_{t_1} \prec^{T_{2b}-T_{1a}}_{\max((T_{2a}-T_{1b}),0)} e_{t_2}$,否则为 $e_{t_2} \prec^{T_{1b}-T_{2a}}_{\max((T_{1a}-T_{2b}),0)} e_{t_1}$。图 1 说明了定义了两个事件的绝对时间约束后它们的相对时间约束,其中(a) 表示 e_{t_1} 在 e_{t_2} 之前发生,(b) 表示 e_{t_1} 在 e_{t_2} 之后发生。



(a)　　(b)

图 1　绝对时间约束事件的相对约束关系

这样,对 TCWFN 中的所有绝对时间约束,两两之间形成相对时间约束,并被加入到 TCWFN 的相对时间约束集合中。

步骤 3:相对时间约束的处理。

对于 TCWFN 中的任意两个变迁 t_1 和 t_2,如果存在相对时间约束 $e_{t_1} \prec e_{t_2}$,则在 $\sum$ 中的顶点 t_1 和 t_2 之间建立有向边 (t_1, t_2)。

步骤 4:相邻活动之间未定义时间约束时的处理。

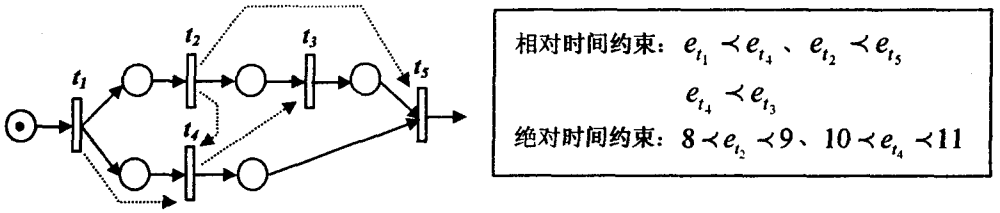


图 2 时间约束流图的构造

TCWFN 中任意两个相邻变迁 t_1 和 t_2 之间如果没有相对时间约束，则按照控制流逻辑隐含的相对时间约束，在 $\sum$ 中的顶点 t_1 和 t_2 之间建立有向边(t_1，t_2)，并添加时间约束 $t_{1\_end} \prec_0^{\infty} t_{2\_start}$。

按照上述步骤，可通过 TCWFN 和时间约束集得到时间约束流图。图 2 表示在 TCWFN 中添加了时间约束关系，图 3 为变换后的时间约束流图。

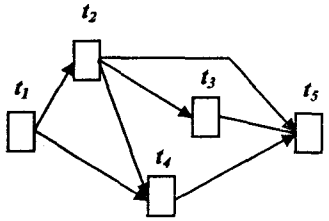


图 3 时间约束流图

比较时间约束工作流图 TCWFN 和相应的时间约束流图 $\sum$ 可以看出，两个图中结点个数相同。TCWFN 中的每一条边，在 $\sum$ 中依然存在，但是，由于要反映间接时间约束和绝对时间约束，在 $\sum$ 中增添了许多的边，所以 $\sum$ 的边数一般多于 TCWFN 中的边数。此外，对 TCWFN 任意相对时间约束 $e_{t_i} \prec e_{t_j}$，在 $\sum$ 中都有边(t_i，t_j)。

如果在时间约束流图中出现循环，则说明必存在时间约束的冲突。检查此种冲突可以采用图的深度优先遍历方法，在遍历时如果发现某个已经遍历过的结点再次被遍历，说明图中存在循环。造成循环的原因主要是：

(1)控制流和时间约束流不一致，如图 4 所示。这是一种与控制流相关的循环，其中，实线箭头表示控制流，虚线箭头表示时间约束流。

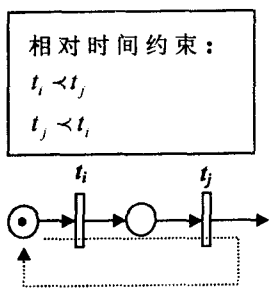


图 4 与控制流相关的循环

(2)定义了不合理的时间约束，如图 5 所示。这是一种与控制流不相关的循环。

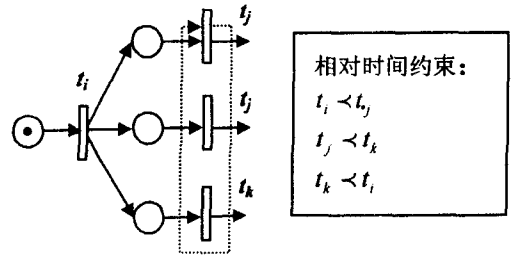


图 5 与控制流不相关的循环

3 时间约束一致性的验证

为了对相对时间约束进行比较，需要建立统一的时间参考点[6]。这里引入事件的累计相对时间约束的概念。对于工作流过程中任一发生的事件 e，它的累计相对时间约束是指从过程的启动事件发生到 e 发生的时间间隔。例如，对于顺序发生的事件 $e_0, e_1, e_2, \cdots, e_i, \cdots$，它们之间的事件约束为 $e_0 \prec_{S_1}^{L_1} e_1, e_1 \prec_{S_2}^{L_2} e_2, \cdots, e_{i-1} \prec_{S_i}^{L_i} e_i, \cdots$，如果过程启动事件 e_0 的时间为 0，则事件 e_i 的累计相对时间约束为：$e_0 \prec_{S_1+S_2+\cdots+S_i}^{L_1+L_2+\cdots+L_i} e_i$

对时间约束一致性验证的一般过程描述如下：

1. 将工作流网分解为一个 T - 组件网的集合 $\Theta = \{N_1, N_2, \cdots, N_k\}$[7]。

2. 去掉无意义的时间约束。这类约束一般是相对时间事件约束，且约束的前件和后件不属于同一个 T - 组件网。

3. 对每一个 T - 组件网 N_i，验证其时间约束一致性。这一过程首先需要根据 N_i 和时间约束集合 $TC = \{TC_A, TC_D, TC_{ID}, TC_I\}$ 构造相应的时间约束流图 $\sum_i$；然后检查 $\sum_i$ 中有无循环，如果 $\sum_i$ 中存在循环，说明时间约束有冲突；在 $\sum_i$ 中不存在循环的情况下，可进一步对 $\sum_i$ 中的结点从 1 开始进行编号，以保证时间约束推理顺序的正确。编号方法为：从 $\sum_i$ 的开始活动顶点出发，用类似于广度优先遍历的方法对网中所有结点编号，与广度优先遍历方法不同的是，可能对一个结点多次遍历，每次遍历到一个结点时，都将其编号设置为遍历时的前驱结点的编号加 1；最后，按顶点编号对 $\sum_i$ 中的所有可能发生的事件依据第二部分

中给出推理规则进行时间约束推理,在推理过程中如发现某一时间约束 $e_i \prec_S^L e_j$ 不能满足 $0 \leqslant S \leqslant L \leqslant \infty$,则说明时间约束冲突,时间约束集合 TC 不符合一致性的要求。

假设需要验证的时间约束工作流网为 TCWFN,定义在其上的时间约束集合为:$TC = \{TC_A, TC_D, TC_{ID}, TC_I\}$,其中 TC_A 为绝对时间约束集合,TC_D 为直接约束集合,TC_{ID} 为间接约束集合,TC_I 为内部约束集合;TCWFN 分解出的 T - 组件网集合为 $\Theta = \{N_1, N_2, \cdots, N_k\}$,每个 T - 组件网对应一个时间约束流图,所有时间约束流图的集合为 $\Psi = \{\sum_1, \sum_2, \cdots, \sum_k\}$。[算法 1] 描述了时间约束一致性的验证过程。

[算法 1] TCWFN_ TC_ Consist。

输入:时间约束工作流网 TCWFN 及其定义在其上的时间约束集。

$TC = \{TC_A, TC_D, TC_{ID}, TC_I\}$。

输出:0 表示存在时间约束的冲突,1 表示不存在。

begin

// Step1: T - 组件网分解[7] //

begin

将工作流网分解为一个 T - 组件网的集合 $\Theta = \{N_1, N_2, \cdots, N_k\}$

end

// Step2:从 TC_{ID} 中删除所有跨 T - 组件网的相对时间约束,设

$N_1 = (P_l, T_l, F_l), l = 1,2,\cdots k$ //

for every $e_{t_i} \prec e_{t_j} \in TC_{ID}$

if $t_i \in T_m \wedge t_j \in T_n \wedge m \neq n$

$TC_{ID} = TC_{ID} - \{e_{t_i} \prec e_{t_j}\}$

endif

endfor

// Step3: 验证每一个组件网的时间约束一致性 //

for every $N' = (P', T', F') \in \Theta$

// Step3.1 将绝对时间约束两两转化为相对时间约束 //

for every $T_{11} \prec u_1 \prec T_{21} \prec u_2 \prec T_{22} \in TC_A \wedge T_{11} \prec u_1 \prec T_{12} \neq T_{21} \prec u_2 \prec T_{22}$ do begin

if $u_1 \in E_{t_i} \wedge u_2 \in E_{t_j} \wedge t_i, t_j \in T'$ //事件 u_1, u_2 所归属的活动均在 T - 组件网 T' 的变迁集中 //

根据 $T_{11} \prec u_1 \prec T_{12}, T_{21} \prec u_2 \prec T_{22}$,建立相对时间约束 $u_1 \prec u_2$(或 $u_2 \prec u_1$)

$TC = TC + \{u_1 \prec u_2\}$(或 $TC = TC + \{u_2 \prec u_1\}$)

endif

endfor

// Step3.2: 构造时间约束流图 //

begin

构造 $N' = (T', P', F')$ 的时间约束流图 $\sum' = (V', E') \in \Psi$

end

// Step3.3: 检查 $\sum'$ 中是否存在循环 //

begin

初始化:栈 S

$x = \sum'$ 的开始结点,PUSH(x, S) //x 入栈 //

while (S 不空) do

x = POP(S) //S 的出队元素赋给 x//

if x 存在后继 y,y 未被访问过,且 y 不是栈中元素 PUSH(y, S)

else

return(0) //图 $\sum'$ 中存在循环,结束检查 //

endif

endwhile

end

// Step3.4: 对时间约束流网中的结点进行编号 //

begin

初始化:队列 Q

$x = \sum'$ 的开始结点,x 的编号设为 1,把 x 放入队列 Q

while (Q 不空) do

x = deq(Q)//Q 的出队元素赋给 x//

for 所有 x 的后继 y

y 的编号设为 x 的编号加 1,把 y 放入队列 Q

endfor

endwhile

end

// Step3.5 按图 $\sum'$ 中顶点编号的顺序计算每个事件的累计相对时间约束,并检查其合理性,$\sum'$ 中顶点的数量为:$|V'| = Num'$ //

begin

//设过程的启动事件为 e_s,启动时间为 0,$\sum'$ 中起始活动的内部时间约束区间为 $[a, b]$// 设定 0 为顶点 $t_1 \in V'$ 所代表活动的启动事件的发生时间

t_1 活动的结束时间的累计相对时间约束为 $e_s \prec_a^b t_{1\_end}$

for g = 2 to Num'

begin

for every $v \in \cdot t_g$ //对 t_g 的所有前驱活动 //

if 时间约束的后件为 t_g 的启动事件

按时间约束的顺序推理规则计算 t_g 启动事件的累计相对时间约束 tc_i

endif

endfor

按时间约束的并行推理规则计算 t_g 启动事件满足所有累计相对时间约束 tc_i 的时间区间 $[S,L]$ //$i = 1,2,\cdots,d_1$//

if $L < S$

return(0) //时间约束冲突,结束检查 ///

endif

根据活动 t_g 的内部时间约束计算 t_g 结束事件的累计相对时间约束 tc_0

for every $v \in \cdot t_g$ //对 t_g 的所有前驱活动 //

if 时间约束的后件为 t_g 的结束事件

按时间约束的顺序推理规则计算 t_g 结束事件的累计相对时间约束 tc_j

endif

endfor

按时间约束的并行推理规则计算 t_g 结束事件满足所有累计相对时间约束 tc' 的时间区间 $[S',L']$ //$i = 1,2,\cdots,d_2$//

if $L' < S'$

return(0) //时间约束冲突,结束检查 ///

endif

end

endfor

end

// 未发现时间约束的冲突,说明 TCWFN 在上定义的时间约束是一致的 //

return(1)

end

end //算法结束 //

[算法 1]对时间约束工作流网 TCWFN = (P,T,F,M_0,α,β) 及其在其上定义的时间约束集合 $TC = \{TC_A, TC_D, TC_{ID}, TC_I\}$ 进行验证,其时间复杂度为多项式级别(次方数不超过 3),具体分析如下:

Step1 部分的作用是分解 T - 组件网,时间复杂度为 $O((|P|+|T|)\times|\Theta|)$,其中 $\Theta = \{N_1, N_2, \cdots, N_k\}$ 是从 TCWFN 分解出的 T - 组件网的集合。

Step2 删除所有跨 T - 组件网的相对时间约束。此部分需要对所有的相对时间约束进行检查,时间复杂度为 $O(|TC_D|+|TC_{ID}|)$。

Step3 对每一个 T - 组件网的时间约束一致性进行验证。其中,Step3.1 将绝对时间约束两两转化为相对时间约束,时间复杂度为 $O(|TC_A|^2)$;Step3.2 构造时间约束流图,时间复杂度为 $O(|TC_{ID}|+|TC_D|+|TC_A|^2+|T|^2)$;Step3.3 检查时间约束流图中是否存在循环,时间复杂度为 $O(|V|+|E|)$;Step3.4 对时间约束流图中的顶点进行编号,时间复杂度为 $O((|P|+|T|)\times d)$,其中 d 为网 TCWFN 中出度最大的结点的出度数;Step3.5 通过时间约束推理检查时间约束一致性,时间复杂度为 $O(|E|)$,所以,Step3 部分具有多项式(次方数不超过 3)的时间复杂度。

4 结束语

时间约束一致性验证的目的,是在时间约束工作流网上检查各种时间约束的冲突,包括:相对时间约束的冲突、循环等待、相对时间约束与绝对时间约束的冲突、绝对时间约束之间的冲突以及无意义的时间约束[8]。文中提出的算法,不仅对时间约束模型的建立过程提供了验证方法,对时间约束模型的运行时验证也有重要的参考意义。

参考文献:

[1] Eder J, Panagos E, Rabinovich M. Time constraints in workflow systems[C]//In: Jarke M, Oberweis A. Proc. of the 11th Conf. on Advanced Information Systems Engineering. Heidelberg: Springer - Verlag, 1999:286 - 300.

[2] 李慧芳,范玉顺. 基于时间 Petri 网的工作流模型分析[J]. 软件学报,2004(1):17 - 26.

[3] 孙智坚,姜 浩. 基于时间约束 Petri 网的工作流动态一致性检验[J]. 计算机技术与发展,2006,16(9):50 - 56.

[4] Tsai J, Yang S J. Timing constraint Petri nets and their application to schedulability analysis of real - time system specifications[J]. IEEE Trans. on Software Engineering, 1995, 21(1):32 - 49.

[5] 李慧芳,范玉顺. 时间约束工作流模型的可调度性分析算法[J]. 计算机集成制造系统,2002(7):527 - 532.

[6] 李 丹,陈启璋,刘 强. 一种基于 Petri 网的时间工作流模型的研究与验证[J]. 计算机工程,2007,33(7):78 - 80.

[7] Dai Xuemei, Qin Kai, Jiang Hao. An Approach to Decomposition of Workflow Models Based on Petri Net[C]//Computational Intelligence and Design, 2008. ISCID '08. [s.l.]:[s.n.], 2008:503 - 506.

[8] 冯 林,姜 浩. 基于时间约束 Petri 网的工作流可调度性分析[J]. 计算机技术与发展,2006,16(11):34 - 37.