第16卷 第9期
2006年9月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol. 16 No. 9
Sep. 2006

基于时间约束Petri网的工作流动态一致性检验

孙智坚,姜 浩
(东南大学 计算机科学与工程系,江苏 南京 210096)

摘 要:工作流系统中的时间管理是工作流建模和分析的重要组成部分。支持动态修改是人们在实际应用中对工作流系统提出的新要求。文中在基于时间约束的Petri网模型基础上,根据时间约束推理规则,提出一种动态修改时间约束时检验工作流一致性的方法,从而丰富了工作流的时间管理功能。

关键词:工作流;时间约束Petri网;动态修改;时间一致性
中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2006)09-0050-03

Verification of Temporal Consistency in Dynamic Modification of Workflow Based on Time Constraint Petri Nets

SUN Zhi-jian, JIANG Hao
(Dept. of Computer Science and Technology, Southeast University, Nanjing 210096, China)

Abstract: Time management is an important part of workflow modeling and analyzing. Supporting dynamic changes, the ability of dynamic adaptability tends to be required when workflow systems are employed. Based on time constraint Petri net model of workflow and temporal reasoning rules, a verification method of temporal consistency of the workflow when time constraints are modified dynamically, which has an important value in enhancing time management functionality is proposed.

Key words: workflow; time constraint Petri net; dynamic modifying; temporal consistency

0 引 言

近年来,工作流管理已经成为支撑企业业务过程建模、重新设计、执行的重要技术。根据工作流管理联盟(WFMC)的定义,工作流是业务过程的全部或部分自动化[1]。设计工作流管理系统的关键在于工作流建模。随着对工作流研究的深入,人们提出了多种工作流建模的方法:有向图、对象模型、语言动作理论、基于约束条件的形式语言文法、Petri网等。

实际业务过程大多具有时间限制,时间违反将造成业务成本增加、工程误期、资源浪费等负面影响。所以,实施工作流管理系统需要解决时间有关的问题,其中,时间约束问题是工作流时间管理的关键问题。另外,随着工作流技术的不断发展以及在实际应用中的不断深入,用户要求工作流具有更强的动态修改能力和可适应能力,提高整个系统的柔性,因而这已成为目前工作流研究的一个热点。文中基于工作流时间约束Petri网模型,提出一种动态修改时间约束并验证时间一致性的方法,可以在工作流执行时对其时间一致性进行检验,并可以动态修改时间约束对一致性进行再检验。

1 工作流的时间约束Petri网模型

Petri网是一种基于状态的建模方法,由于Petri网能够很好地描述离散动态系统的静态和动态行为特性,同时具有有效的数学分析技术,因而Petri网模型方法已被广泛地用来描述工作流模型。将Petri网和工作流相结合,就提出了工作流网(WFN)[2]。

工作流系统的时间管理首先需要解决时间建模问题,工作流时间模型中对时间因素的主要描述方式有:甘特图、赋时工作流图(Timed Workflow Graph)[3]、基于好结构(Well Structure)的工作流时间模型等等[4]。文中采用TCWFN(Timing Constraint Workflow Net)来为引入时间参数的工作流过程建模。在Petri网中引入时间参数的方式有多种[5],有将库所与时间参数相关联的,有将变迁与时间参数相关联的,也有将有向弧与时间参数相关联的。文中采用时间和变迁相关联的方式,将每个变迁和一个时间范围相联系。

定义1(时间约束工作流网)

引入映射函数 TC 的Petri网 $TCWFN=(P,T,F,M_0,TC)$ 是一个时间约束工作流网。如果它满足:

* TCWFN所对应的无时间约束Petri网 $WFN=(P,T,F,M_0)$ 是一个工作流网;

收稿日期:2005-11-25

作者简介:孙智坚(1980-),男,江苏扬州人,硕士研究生,研究方向为Petri网建模与分析、工作流系统;姜 浩,博士,副教授,硕士生导师,研究方向为工作流建模技术、计算机集成制造、Agent技术等。

* TC 是从变迁到一个时间区间$[\alpha,\beta]$的映射,TC:$T\rightarrow[\alpha,\beta],0\leqslant\alpha\leqslant\beta<\infty$。

这里 WFN = (P,T,F,M_0) 称为与 TCWFN = (P,T,F,M_0,TC) 相对应的无时间约束工作流网。在工作流时间约束的Petri网中,库所对应过程中的条件,变迁对应过程中的可执行活动,TC 映射到变迁上的时间区间$[\alpha,\beta]$表示活动的持续的时间范围,即活动的持续时间大于α,小于β。$\beta-\alpha$称为活动A执行的松弛时间。通过这个模型,可以在工作流过程定义中添加时间约束,以便进行时间维上的验证和分析。

2　时间约束分类

工作流系统时间约束的表示方式一般分为绝对时间(absolute time)约束和相对时间(relative time)约束。绝对时间就是日历时间,绝对时间约束是限制某一事件发生的绝对时间范围。相对时间是基于某一时间参考点的时间,相对时间约束表示两事件之间相对时间的约束。

用一个二元关系运算符"$\prec$"来描述事件发生的先后关系。假设有事件e_1和e_2,$e_1\prec e_2$表示如果事件e_1和e_2都发生,则e_1一定在e_2之前发生。称e_1为"$\prec$"的前件,e_2为"$\prec$"的后件。相对时间约束可以表示为:$e_1\prec^{\beta}_{\alpha}e_2$,$e_1$和$e_2$发生的最小时间间隔为$\alpha$,最大为$\beta$。类似地,绝对时间约束可以表示为:$T_1\prec e\prec T_2$,事件$e$必须在绝对事件$T_1$到$T_2$之间发生。

定义2(内部时间约束)

对于变迁t,如果有时间约束$e_1\prec e_2,e_1\in E_t\wedge e_2\in E_t$,其中$E_t$表示与变迁$t$相关的事件集合,则此时间约束称为内部时间约束。

定义3(间接时间约束、直接时间约束)

对于变迁t_1和t_2,如果存在库所$p\in P$,使得$p\in t_1\cdot\wedge p\in\cdot t_2$,则称时间约束$e_{t_1}\prec e_{t_2}$为直接时间约束,否则称为间接时间约束。

时间约束的分类如图1所示,流程活动中常见的时间约束都可以根据以上的定义分类。

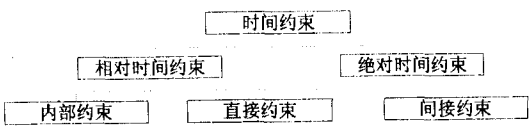

图1　时间约束分类

3　时间约束推理规则

WFMC定义了工作流6种基本控制结构:顺序(Sequence)、与分叉(And-Split)、与合并(And-Join)、或分叉(Or-Split)、或合并(Or-Join)和循环(Iteration),由基本控制结构组成的4种常用的流程结构:顺序结构、并行结构、选择结构、循环结构都可以看作一个等价结点[6],循环结构存在不确定因素常常看作一个黑盒[7],当工作流执行过程中一些结果逐渐明朗时,才决定黑盒内部的具体意义,从外部来看,它的时间约束就是从循环结构开始执行到执行结束的时间。如图2所示。在工作流过程定义中加入了时间约束之后,除了要满足流程控制逻辑的要求,还要考虑时间因素的影响。下面引入时间约束推理规则:

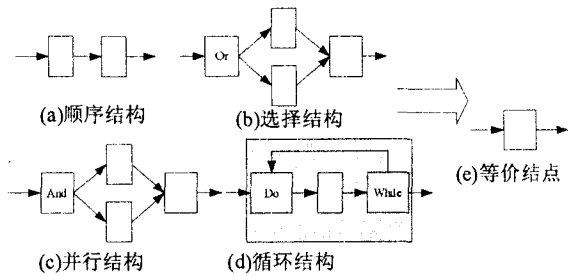

图2　基本流程结构

规则1:顺序结构的两个事件e_i和e_j的时间约束为(α_1,β_1)和(α_2,β_2),则等价结点的时间约束为$(\alpha_1+\alpha_2,\beta_1+\beta_2)$。

规则2:并行结构是由与分叉和与合并结构组成的,并行分支有k条路径,第l条路径的时间约束为(α_l,β_l),则等价结点的时间约束为$(\max(\alpha_l),\min(\beta_l))$,其中$l=1,2,\cdots,k$。

规则3:选择结构是由或分叉和或合并结构组成的,并行分支有k条路径,第l条路径的时间约束为(α_l,β_l),则等价结点的时间约束为$(\min(\alpha_l),\max(\beta_l))$,其中$l=1,2,\cdots,k$。

4　动态修改时间约束的一致性验证

在业务流程的定义中如果包含多个时间约束,就有可能发生冲突。相对约束和绝对约束都可能发生冲突,它们之间也有可能发生约束冲突。如果时间约束限制的事件,其发生顺序和控制流顺序不一致,则可能造成循环等待问题。因此,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。所以,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。

在工作流的执行过程中,假设某个活动正在执行,以此活动作为检查点,对时间约束一致性进行动态验证,如果出现时间冲突,应当采取适当措施重新获得一致状态,并尽量地减小可能的损失。常用的措施有:触发异常处理,修改时间约束,删除后续选择性活动,请求人为干预等[8]。下面提出一个动态执行时修改时间约束并检验一致性的算法,算法中的时间约束关系通过时间约束流图来表示。

定义4(时间约束流图)

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

时间约束流图可以通过以下步骤来得到:1)顶点复制。将TCWFN中的顶点集合复制到Σ中,即$V=T$;2)绝对时间约束的处理。将绝对时间约束转化,两两之间形

成相对约束,并加入到 TCWFN 的相对时间约束集合中;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 Σ 中的顶点 t_1 和 t_2 之间建立有向边(t_1,t_2);4) 相邻活动之间按流程结构建立有向边。时间约束流图的例子如图 3 所示。

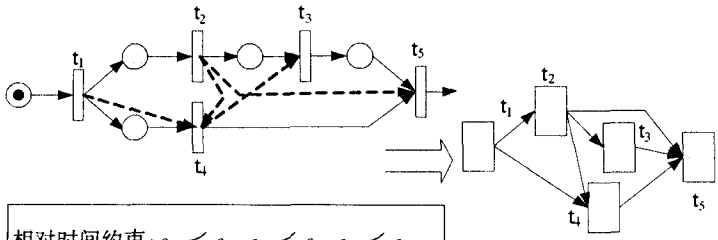

图 3 时间约束流图示例

时间约束一致性检查分定义时(build - time)检查、运行时(run - time)检查[8],动态修改时间约束并检查一致性必须要用到前面的检查结果,所以首先引入以下几个算法。

算法 1 (定义时检查时间约束)

初始化:输入时间约束工作流网 TCWFN 及其定义在上面的时间约束集 $TC = \{TC_A, TC_D, TC_{ID}, TC_I\}$。

第一步:对于每一个绝对时间约束 $T_{i1} \prec e_i \prec T_{i2} \in TC_A$,如果 $|TC| \geqslant 2$,根据其相互关系建立相对时间约束 TC',并加入到现有时间约束集中,$TC = TC + TC'$。

第二步:构造 $N = (P, T, F, M_0, TC)$ 的时间约束流图 $\Sigma = (V, E)$。将 N 中的变迁集作为 Σ 的顶点集,$V = T$。对于每一个时间约束 $e_{t_i} \prec e_{t_j} \in TC_D \cup TC_{ID}$,相应地在 Σ 中添加有向边,$E = E + \{(t_i, t_j)\}$。对于相邻但没有直接时间约束的变迁,在 Σ 中直接添加边。对于未定义内部时间约束的变迁 t_i,以缺省值补充,$TC = TC + \{t_{i\text{-start}} \prec_{\min}^{\max} \prec t_{i\text{-end}}\}$。

第三步:检查 Σ 中是否存在循环。初始化栈 S。将 Σ 的起始结点 a 压栈,PUSH(a, S)。当 S 非空时循环:弹栈,x = POP(S),如果 x 有后继 y,y 未被访问过且 y 不是栈中元素,压栈 PUSH(y, S),否则有循环,算法结束。当 S 为空,循环结束,进入下一步。

第四步:对 Σ 中结点进行编号。初始化队列 Q。将 Σ 的起始结点 a 入队列,ENQ(a, Q),设 a 的编号为 1。当 Q 非空时循环:出队列,x = DEQ(Q),将 x 的每个后继 y 依次入队列,ENQ(y, Q),y 的编号依次加 1。当 Q 为空,循环结束,进入下一步。

第五步:按 Σ 中结点的编号依次计算每个事件的累计时间约束,检验合理性。设起始结点 a 启动时间为 0,即 t_1 的累计相对时间约束为其内部时间约束。对结点 $t_i (i = 2, \cdots, |V|)$ 依次检查:对 t_i 的每个前驱 $v \in \cdot t_i$,如果时间约束的后件为 $t_{i\text{-start}}$,按照顺序推理规则计算累计相对时间约束 $tc_{i,k}$,按并行推理规则计算满足每个 $tc_{i,k}$ 的时间区间$[S, L]$。如果 $S > L$,有冲突,算法结束;否则继续,对 t_i 的每个前驱 $v \in \cdot t_i$,如果时间约束的后件为 $t_{i\text{-end}}$,按照顺序推理规则计算累计相对时间约束 $tc'_{i,k}$,按并行推理规则计算满足 $tc'_{i,k}$ 的时间区间$[S', L']$。如果 $S' > L'$,有冲突,算法结束;否则继续。直到所有结点检查完,未发现冲突,说明 TCWFN 定义的时间约束是一致的,算法结束。

算法 2(运行时检查时间约束)

初始化:输入算法 1 的 TCWFN 中所有事件的累计相对时间约束和绝对时间约束集合 TC_A。

对于每个绝对时间约束 $T_{i1} \prec e_i \prec T_{i2} \in TC_A$,设工作流起始事件的发生的绝对时间为 T_B,根据累计相对时间约束计算出绝对时间约束 $e_1 \prec_L^S \prec e_i$ 进行检查,如果 $T_{i1} > S \vee T_{i2} < L$,即时间区间不相交,则有冲突,算法结束。当所有绝对时间约束检查完毕未发现冲突,则运行时时间约束是一致的,算法结束。

算法 3(动态修改时间约束并检查一致性)

初始化:输入时间约束工作流网 TCWFN 及时间约束集 $TC = \{TC_A, TC_D, TC_{ID}, TC_I\}$,时间约束流图 Σ,到目前为止最后完成的活动 t_{last},提交的待修改的时间约束集 TC_M。

第一步:去掉不起作用的时间约束。对每个时间约束 $tc_i \in TC \cup TC_M$,如果约束前件和后件事件都已完成,则从集合里去除。

第二步:对提交的待修改的时间约束集 TC_M 进行处理。对于每个待修改时间约束 $tc_i \in TC_M$ 进行检查:如果 tc_i 是绝对时间约束,由算法 2 判断是否合理。如果发现冲突,算法结束,否则将其转换成相对时间约束;如果 tc_i 是相对时间约束,则对 TC 进行更新,进行相应的增加、减少、更改时间约束,并相应更改累计时间约束。

第三步:检查一致性。设工作流起始事件的发生的绝对时间为 T_B,t_{last} 完成的绝对时间为 T_E,令 $T = T_E - T_B$,设 s 为 t_{last} 的编号加 1。对结点 $t_i (i = s, \cdots, |V|)$ 依次进行检查:对 t_i 的每个前驱 $v \in \cdot t_i$,如果时间约束的后件为 $t_{i\text{-start}}$,且若 v 已经完成,累计相对时间约束 $tc_{i,k}$ 为原有累计时间约束减去 T,否则按照顺序推理规则计算累计相对时间约束 $tc_{i,k}$,按并行推理规则计算满足每个 $tc_{i,k}$ 的时间区间$[S, L]$。如果 $S > L$,有冲突,算法结束;否则继续,对 t_i 的每个前驱 $v \in \cdot t_i$,如果时间约束的后件为 $t_{i\text{-end}}$,且若 v 已经完成,累计相对时间约束 $tc'_{i,k}$ 为原有累计时间约束减去 T,否则按照顺序推理规则计算累计相对时间约束 $tc'_{i,k}$,按并行推理规则计算满足 $tc'_{i,k}$ 的时间区间$[S', L']$。如果 $S' > L'$,有冲突,算法结束;否则继续。直到所有结点检查完,未发现冲突,当前提交的动态修改后的时间约束是一致的,算法结束。

(下转第 55 页)

作用是判断此组数据能否遍历指定路径,即若某个体的适值函数值为0,则它能遍历此路径,此时的数据即为生成的测试数据。

由于每个个体的适值函数值不为0,而且1,3,5号个体的适值函数值较大,2,7,9号适值函数值较小,所以淘汰适值函数大的个体,小的个体自身复制。

因此,第1代个体如表4所示。

表4　第1代个体

个体编号	父个体	子个体(新个体)	适值函数值
1	0111 0101100010110010	0111 0010010010111000	5+0
2	0111010 1100010110010	0111010 0110011001111	0+4
3	1001 0010010010111000	1001 0101100010110010	0+4
4	1011011011000101 101	1011011011000101 111	0+14
5	1001110 0110011001111	1001110 1100010110010	0+4
6	1001110011001101 111	1001110011001101 101	0+0
7	1 0101001101100111001	1 1101011101101010101	15+8
8	0110101110110 1010101	0110101110110 1001110	0+0
9	0 1101011101101010101	0 1001011101101010101	0+0
10	1001000111100 1001110	1001000111100 1010101	9+0

由于个体6,8,9的适值函数值为0,即得测试数据 $i=19,j=19,l=6,k=29$;$i=13,j=14,l=26,k=14$;$i=9,j=14,l=26,k=21$。

由表4可以看出,有3个个体的适值函数值为0,说明有3组数据可以遍历给定的路径。将个体按照初始化时的方式进行拆分:$i=10011,j=10011,l=00110,k=11101$;$i=01101,j=01110,l=11010,k=01110$;$i=01001,j=01110,l=11010,k=10101$。再将每个变量的0,1串转换成十进制数。$i=19,j=19,l=6,k=29$;$i=13,j=14,l=26,k=14$;$i=9,j=14,l=26,k=21$,即是生成的测试数据,它们能遍历指定的路径 $P=1-2-4-5-6-7$。

3　总　结

测试数据的自动生成是测试阶段最关键的技术问题,改进软件测试方法,对提高软件测试的自动化程度,具有十分重要的现实意义。文中应用遗传算法和函数极小化策略解决软件结构测试数据生成问题,克服了传统的以测试数据为核心的测试方法的不足和缺陷,对软件测试中的测试数据自动生成具有很强的使用价值。

参考文献:

[1] 朱　鸿,金凌紫.软件质量保障与测试[M].北京:科学出版社,1997.

[2] 伦立军,丁雪梅,李英梅.路径覆盖自动生成技术研究[J].计算机工程与应用,2003,39(16):123-125.

[3] 伦立军,丁雪梅,李英梅.基于遗传算法的测试数据生成研究[J].计算机工程,2005,31(23):82-84.

[4] Pargas R, Harrold M, Peck R. Test-Data Generation Using Genetic Algorithms[J]. Journal of Software Testing, Verification & Reliability, 1999, 9(4): 263-282.

[5] Korel B. Automated Software Test Data Generation[J]. IEEE Trans on Software Eng, 1990, 16(8): 870-879.

[6] Korel B. Dynamic Method of Software Test Data Generation [J]. Journal of Software Testing, Verification & Reliability, 1992, 2(4): 203-213.

[7] Christoph C, Gary M, Michael S. Generating Software Test Data by Evolution[J]. IEEE Trans on Software Eng, 2001, 27(12): 1085-1110.

[8] Sthamer H. The Automatic Generation of Software Test Data Using Genetic Algorithms [D]. Great Britain: University of Glamorgan, 1996.

(上接第52页)

至此已经给出了静态验证和动态修改并验证的算法,根据以上的算法,可以在过程的设计阶段和执行阶段随时检查时间约束一致性,保证业务过程能够被有效地执行。

5　结　论

将时间参数引入工作流模型中并对模型进行时间维分析是工作流技术的重要研究内容。文中采用工作流时间约束Petri网模型,对时间约束进行分类,在此基础上提出了动态修改时间约束并判断一致性的算法,该算法便于动态执行时对时间冲突进行违反处理,也可以用于动态运行时检查时间一致性。

参考文献:

[1] Chen Jinjun, Yang Yun, Chen T Y. Dynamic Verification of Temporal Constraints on-the-fly for Workflow Systems [A]. 11th Asia-Pasific Software Engineering Conference, (APSEC'2004)[C]. Busan, Korea: [s. n.], 2004. 30-37.

[2] 范玉顺.工作流管理技术基础—实现企业业务重组[A].过程管理与过程自动化的核心技术[C].北京:清华大学出版社,2001.

[3] Eder J, Panagos E. Time management in workflow systems [A]. BIS'99 3rd International Conference on Business Information Systems [C]. [s. l.]: Springer Verlag, 1999. 265-280.

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

[5] Bowden F D J. A brief Survey and Synthesis of the Roles of Time in Petri nets[J]. Mathematical and Computer Modeling, 2000, 31: 55-68.

[6] 孙瑞志,史美林.一个支持动态变化的工作流元模型[J].电子学报,2002(12A):2052-2056.

[7] 孙瑞志,史美林.支持动态变化的工作流过程元模型[J].软件学报,2003,14:62-67.

[8] Eder J, Panagos E. Time Constraints in Workflow Systems [A]. 11th Conference on Advanced Information Systems Engineering (CAISE99) [C]. Heidelberg, Germany: [s. n.], 1999. 1-14.