基于行为的大规模云工作流模型库高效检索方法研究

黄 华[1),2)] 彭 蓉[1)] 冯在文[1)]

[1)](武汉大学计算机学院软件工程国家重点实验室 武汉 430072)

[2)](景德镇陶瓷大学信息工程学院 江西 景德镇 333001)

摘 要 为了提高流程模型检索效率,现有的基于行为的流程模型检索方法应用基于流程执行路径/任务执行次序关系的索引去过滤大规模流程模型库,以减少候选模型的数量.由于构建与维护基于流程执行路径的索引较为困难,而基于任务执行次序关系的索引并不考虑任务执行次序关系的约束属性(如时间约束),在模型数量巨大且存在大量具有相同或相似行为的流程片段的云工作流模型库中,其过滤能力有限.因此,这些方法并不适合基于行为的大规模云工作流模型库的高效检索.鉴于此,针对云工作流模型库的特征,该文应用带时间约束的任务执行次序关系索引及流程行为精确匹配算法,提出一种改进的基于行为的两阶段流程模型检索方法.该方法在过滤阶段,构建带时间约束的任务执行次序关系索引,通过该索引对大规模云工作流模型库进行过滤,以大大减少候选模型的数量.在精化阶段,应用带时间约束的流程行为精确匹配算法对候选模型集进行精化验证.评估实验结果表明,该方法能显著提高基于行为的大规模云工作流模型库的检索效率.

关键词 云工作流;大规模流程管理;流程检索;时间约束;云计算

中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2018.01238

Efficient Retrieval of Large-Scale Cloud Workflow Model Repositories Based on Behavior

HUANG Hua[1),2)] PENG Rong[1)] FENG Zai-Wen[1)]

[1)](*State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan* 430072)

[2)](*School of Information Engineering, Jingdezhen Ceramic Institute, Jingdezhen, Jiangxi* 333001)

Abstract Cloud workflow system is a workflow management system deployed in cloud computing environment. A large number of tenants can design, configure and run their business processes on it. With the development of cloud computing, cloud service platforms are widely adopted by more and more enterprises and individuals. The underlying cloud workflow systems accumulate large numbers of business process data. Retrieving and recommending the most similar process models according to the tenant's requirements become extremely important, because it is not only beneficial to promote the reuse of the existing model assets, but also helpful to reduce the error rate of the modeling process. However, in cloud service platforms there exist a large number of tenant service systems with the same or similar business background and application scenarios. Consequently, cloud workflow model repositories always have many process models whose process nodes are always having the same or similar labels, and a lot of process fragments with

收稿日期:2016-05-17;在线出版日期:2017-04-28.本课题得到国家重点研发计划(2017YFB0503700,2016YFB0501801)、国家自然科学基金(61100017,61170026)、国家标准研究计划(2016BZYJ-WG7-001),中央高校基本科研业务费专项资金(2012211020203,2042014kf0237)、江西省重点研发计划重点项目(20171ACE50022)、江西省自然科学基金面上项目(20171BAB202011)和江西省教育厅科技项目(GJJ160906,GJJ171524)资助.**黄 华**,男,1981 年生,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为服务计算、业务过程管理.E-mail:jdz_hh@qq.com.**彭 蓉**(通信作者),女,1975 年生,博士,教授,中国计算机学会(CCF)会员,主要研究领域为需求工程、软件工程、服务计算.E-mail:rongpeng@whu.edu.cn.**冯在文**,男,1980 年生,博士,讲师,中国计算机学会(CCF)会员,主要研究方向为业务过程管理.

the same substructure or similar behavior. Therefore, traditional retrieval technology is unable to retrieve the most similar processes efficiently. How to efficiently query large process model repositories in a cloud workflow system is challenging. In order to improve the efficiency of the process model retrieval, the filtering-verification framework is often adopted to reduce the number of process models needed to refined. Generally, in the filtering stage, various indexes are built to locate candidate process models that satisfy the given retrieval conditions; in the verification stage, only the candidate models need to be checked by some refining algorithms. Among the existing process model retrieval approaches based on behavior adopt the index based on process execution path/task executing order relation to filter large-scale process repositories to reduce the number of candidate models. Due to that the construction and maintenance of the index based on process execution path is difficult, and the filtering ability of the index based on task executing order relation that does not consider the constraint attributes (e.g., time constraints) is poor in the cloud workflow model repositories including many process fragments with the same or similar behavior, these approaches are not suitable for the efficient retrieval based on behavior of large-scale cloud workflow model repositories. Therefore, according to the characteristics of cloud workflow model repositories, this paper proposes an improved two-phase process retrieval approach based on behavior. In the filtering stage, by considering the time constraints of process nodes (tasks), the index based on task executing order relation with time constraints is adopted to greatly reduce the number of candidate model in large-scale cloud workflow model repositories and to improve the filtering ability of the index. In the refining phase, based on the ordering relations with time constraints between tasks, a process behavior exactly matching algorithm with time constraints is proposed to improve the efficiency of refining the candidate model set. Finally, to demonstrate the effectiveness and efficiency of the improved two-phase process retrieval approach, based on the large-scale simulation process model library and the actual cloud workflow model repository, extensive experiments are conducted. The valuation experiment results show that the proposed approach can reduce the query response time and significantly improve the retrieval efficiency based on behavior of large-scale cloud workflow model repositories.

Keywords　cloud workflow; management of large process repositories; process retrieval; time constraints; cloud computing

1　引　言

云工作流是工作流管理系统在云计算环境下的一种新的应用模式，按照工作流在云计算环境中的应用方式，可分为“云之上”和“云之中”两种不同的层次结构[1-3]. “云之上”的云工作流是运用工作流思想来定义、建模、重设计、自动执行云计算应用项目；而“云之中”的云工作流用于不同租户在线设计、配置、运行各自的业务流程，并达到三个层次的隔离：数据隔离、性能隔离、执行隔离. 本文研究的云工作流属于后者，是一种支持多租户的工作流管理系统，不同的租户在云服务平台中开发和部署业务流程，包括业务流程的建模、执行、监控以及资源管理[3].

在云服务平台中，大量租户的服务系统具有相同或相似业务背景及应用场景. 因此，与传统工作流模型库相比，本文研究的云工作流模型库有如下特征：(1) 存在多种模型格式；(2) 流程模型数量巨大(通常达到 10 万以上)且日益增长；(3) 存在大量具有相同或相似标签的流程节点；(4) 存在大量具有相同或相似子结构及行为的流程片段. 由于上述特征，在大规模云工作流模型库中应用传统的流程检索技术找到最为相似的流程并反馈给租户将极其耗时[3]，租户不可能等待时间漫长的查询响应. 因此，如何实现大规模云工作流模型库的高效检索就是挑战之一[4-5].

流程检索就是通过输入特定格式的查询条件去

检索流程模型库,并返回一个满足用户需求的流程模型集.根据文献[6]所述,当前流程检索方法可分成以下5类:基于图结构的流程精确检索方法、基于图结构的流程相似检索方法、基于行为的流程精确检索方法、基于行为的流程相似检索方法及基于操作语义的流程相似检索方法.本文主要关注于基于行为的流程高效检索方法的研究.

流程模型除了具有静态拓扑结构外,模型中还包含了流程的执行语义,也称之为流程行为.流程行为体现实际运行中流程中各个活动的依赖关系,它是流程模型的一个重要属性.基于行为的流程检索就是在流程模型库中查询得到一些满足流程行为需求的流程模型[7].为了提高基于行为的流程模型检索效率,现有的方法大都采用过滤-验证框架来减少需要进行流程行为匹配的模型数量.在过滤阶段,构建各种行为索引去得到可能满足给定查询模型行为的候选流程模型集;在精化验证阶段,通过基于流程行为的相似度计算算法或流程行为匹配算法对候选模型集进行精化验证.在这方面,文献[4]提出的两阶段流程检索方法是一个比较典型的基于行为的大规模流程模型库的查询方法.该方法在过滤阶段,通过TARIndex索引对大规模流程模型库进行筛选;在精化验证阶段,应用基于TAR集的流程行为相似度计算算法对过滤后的候选模型集进行精化验证.实验表明该方法能有效提高基于行为的大规模流程模型库的检索效率,因此,它将作为本文比较的基准方法.

在基于行为的流程相似度计算及流程检索的研究中,流程的行为基本上是通过其行为剖面(behavioral profile[5])来描述,而流程的行为剖面则是由任务间的执行次序关系来定义的.通常存在三种基本的次序关系:严格的顺序关系(strict order relation)或叫因果关系(causal relation)、并行关系(parallel relation)、冲突关系(conflict relation).在文献[4]提出的基于行为的两阶段流程检索方法中,过滤阶段应用的TARIndex索引只考虑任务的紧邻关系且不考虑执行次序关系的约束条件(如时间约束,而在实际的流程中,如科学工作流、医疗流程、陶瓷制作工艺流程,需要对任务的时间约束进行建模),其过滤能力有限.

例如,在一个由图1所示的两个用Timed WF-net(Timed WF-net从Petri-nets扩展而来,由于Petri-nets[8-9]具有较强的形式化验证理论,因此,Timed WF-net也将适合去分析带时间约束的流程行为,Timed WF-net的详细介绍可参见定义5)表示的流程模型P1,P2组成的流程库中,当检索包含顺序执行关系:B→C,且C必须在B触发后的2到3小时内开始执行的流程模型时,如果用TARIndex索引进行过滤,则候选模型集是{P1,P2}.显然,TARIndex索引并没有过滤掉任何模型.也就意味着,当一

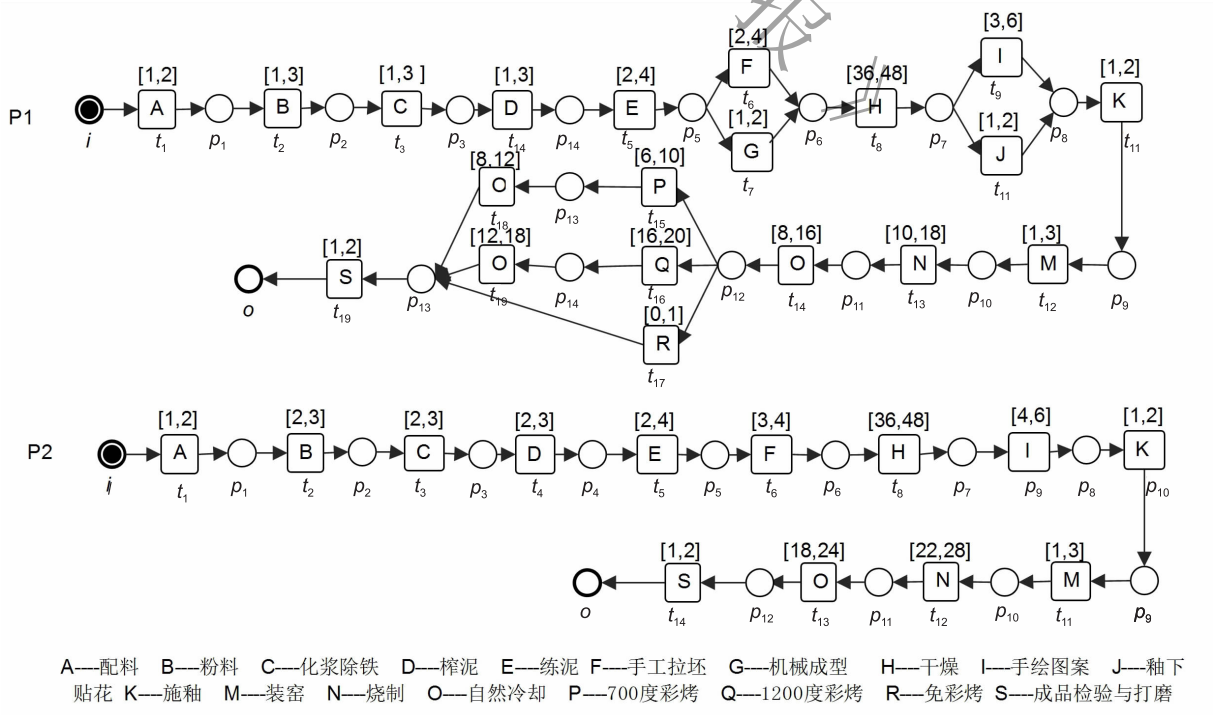


图1 两个用Timed WF-net表示的粉彩瓷制作工艺流程

个流程模型库(如云工作流模型库)中存在大量具有相同或相似标签的流程节点时,TARIndex 索引的过滤能力将大大降低,其精化阶段采用的基于 TAR 集的流程行为相似度计算算法的精化验证效率也将大打折扣.鉴于此,为了提高基于行为的大规模云工作流模型库检索的效率,我们在过滤阶段,构建带时间约束的任务执行次序关系索引,以提高索引的过滤能力;在精化阶段,应用带时间约束的流程行为匹配算法,以提高候选模型的精化验证效率.本文的贡献主要有以下 3 点:

(1)通过考虑流程任务执行的时间约束对任务执行次序关系进行量化,并提出带时间约束的流程行为匹配算法;

(2)提出一种改进的基于行为的两阶段(过滤阶段和精化阶段)流程模型检索方法,并将其应用在基于行为的大规模云工作流模型库的高效检索中;

(3)在大规模模拟流程模型集及真实的云工作流模型库中做了大量验证实验,验证了方法的有效性.

本文第 2 节给出相关研究工作;第 3 节对相关基础知识进行介绍;第 4 节给出带时间约束的流程任务执行次序关系的相关定义及计算过程的细节描述;第 5 节详细描述基于行为的流程查询过程,包括带时间约束的任务执行次序关系索引的构建及流程行为匹配算法;第 6 节是在模拟生成的大规模流程模型集及真实的云工作流模型库中进行实验评估及结论分析,以验证该方法的有效性;第 7 节是本文的总结及下一步工作展望.

2　相关工作

由于流程检索的核心就是找到最为相似的流程并反馈给用户,所以,在基于行为的流程检索方面,大量工作都是围绕着流程行为相似度计算进行的,例如,基于流程模型完全触发序列的流程行为相似度计算方法[10-11]、基于流程模型中任务依赖关系的流程行为相似度计算方法[12-13]及基于 Artifact 的流程行为相似度计算方法[14].而在基于行为的流程检索方面虽有一些研究,但研究工作相对较少.例如,文献[15]提出一种不依赖任何具体建模语言的基于流程语义的检索算法.文献[16]提出了一种基于行为的流程检索算法,其中定义了三种次序关系:因果关系、并发关系、冲突关系,根据给定的行为需求描述来提取每种基本关系,并建立索引,将所有满足需求的流程模型通过检索反馈给用户.文献[17]提出一种 BQL(Behavior Query Language)语言去精确描述查询的行为需求,并通过展开技术去有效地计算业务流程的行为特征以实现基于流程行为的检索,但该方法的检索效率还有待提高.后来,文献[4]使用其定义的 TAR(Task Adjacency Relations)来计算基于行为的流程之间的相似度,并通过建立名为 TARIndex 的索引结构来提高检索效率.但由于 TARIndex 只考虑直接相邻任务间的依赖关系且不考虑任务间的其他约束属性(如时间约束),因此,该方法的 TARIndex 索引的过滤能力有限,在实际应用中,该方法有一定的局限性.

基于行为的流程检索大都基于流程行为相似度计算或者流程节点执行次序关系的抽取,由于这些方法并不考虑行为的其他约束条件(如时间约束),而在云工作流模型库中,业务流程模型数量巨大且存在大量相同或相似标签的流程节点及大量相同或相似行为的流程片段,因此,当对大规模云工作流模型库进行基于行为的检索时,这些方法的检索效率及精确性还有待提高[18-21].

鉴于此,针对云工作流模型库的特征,本文提出了一种改进的基于行为的两阶段(过滤阶段和精化阶段)流程模型检索方法,并将其应用在基于行为的大规模云工作流模型库的高效检索中.

为了提高基于行为的大规模云工作流模型库的检索效率,主要从以下两个方面进行改进:

(1)提高过滤阶段索引的过滤能力.在过滤阶段,考虑流程任务执行的时间约束属性,对流程行为也即流程任务节点间的执行次序关系进行量化,然后,基于定量化的任务执行次序关系构建过滤能力更强的关系索引.通过带时间约束的关系索引能过滤掉更多的不满足检索需求的流程模型,这将大大缩减候选模型集的规模,减少精化验证阶段的时间消耗.

(2)提高精化阶段的验证效率.在精化阶段,利用定量化的流程任务执行次序关系集,提出一种更精确有效的基于行为的流程行为匹配算法,并应用该算法对候选模型集进行精化验证.

为了更好的理解本文提出的方法,下面将给出与本文研究方法相关的一些预备知识.

3　预备知识

为了建模具有时间约束的业务流程模型,本文对 Petri-nets 模型进行扩展得到 Timed WF-nets.下面将给出相关模型和概念的定义及描述[22-25].

定义 1(Petri nets).

一个 Petri net 模型 PN 由一个 3 元组表示，$PN=(P,T,F)$. 其中：

(1) P 表示库所的有限集合，一个库所是用来定义任务的前置条件和后置条件；

(2) T 表示变迁的有限集合，并满足 $P\cup T\neq\varnothing$ 且 $P\cap T=\varnothing$，一个变迁对应一个任务(task)；

(3) $F\subseteq(P\times T)\cup(T\times P)$ 表示流关系的集合，一个流关系是用连接库所与变迁的有向边来表示.

通常定义 $X=(P\cup T)$，用 X 表示一个包含模型 PN 中所有节点的集合. 对于每一节点 $x\in X$，x 的所有直接前驱节点集用 ${}^\bullet x$ 表示，${}^\bullet x=\{y|(y,x)\in F\}$，$x$ 的所有直接后继节点集用 $x^\bullet$ 表示，$x^\bullet=\{y|(x,y)\in F\}$. 一个节点 $x\in X$ 是另一个节点 $y\in X$ 的输入(输出)节点，当且仅当：$x\in{}^\bullet y$ ($x\in y^\bullet$). 对于 X 的任一子集 $Z\subseteq X$，可以得到 ${}^\bullet Z=\bigcup_{x\in z}{}^\bullet x$ 和 $Z^\bullet=\bigcup_{x\in z}x^\bullet$. 此外，一个被标识的 Petri net 可用 (PN,M) 来表示，其中 $PN=(P,T,F)$ 表示一个 Petri net 模型，$M:P\rightarrow N_0$ 表示 PN 状态标识函数，N_0 是自然数集合，$M(p)$ 表示库所 p 中托肯的个数. 关于 Petri net 更多语义描述可参见文献[8-9]. 为了更好的描述流程的任务，下面给出了标签 Petri nets 及标签 WF-nets 的定义.

定义 2(标签 Petri nets).

一个标签 Petri net 模型 PN 由一个 5 元组表示：$PN=(P,T,F,L,lf)$，其中 (P,T,F) 表示一个 Petri net 模型，L 表示模型中任务标签的集合，$lf:T\rightarrow L$ 表示标签映射函数，对于每一任务节点 $t\in T$，如果其标签为 $l\in L$，则 $lf(t)=l$.

定义 3(标签 WF-nets).

一个标签 Petri net 模型 $PN=(P,T,F,L,lf)$ 是一个标签工作流网(简称标签 WF-nets)模型，当且仅当：

(1) PN 有且只有一个源库所 i 且 ${}^\bullet i=\varnothing$；

(2) PN 有且只有一个汇结库所 o 且 $o^\bullet=\varnothing$；

如果给 PN 增加一个变迁 τ 并满足 ${}^\bullet\tau=o$ 及 $\tau^\bullet=i$ 得到一个新的模型 PN'，PN' 将是强连通的，也即 PN' 中任意两节点间都至少有一条可达的路径.

一个 WF-net 模型是正确的，表明在该模型的任一标识状态中每一库所中的托肯数小于等于 1. 为了便于后面的分析，在文中假定所有的 WF-net 模型都是正确的流程模型(也就是 Sound WF-net). 同时，给每一任务增加一个时间约束，也即任务执行的最小时间与最大时间的时间区间(Time Interval)，并构建 Timed WF-nets 去描述时间依赖的流程模型. 下面给出时间区间标签及 Timed WF-nets 相关定义.

定义 4(时间区间标签).

文中用最小时间与最大时间的一个时间区间来表示一个任务执行的时间约束. 给定一个变迁 t，其时间约束变量由一个时间区间标签 $v=[a,b]$ 来表示，其中，变迁 t 的最小完成时间：$t_{\min}(v)=a$，最大完成时间：$t_{\max}(v)=b$，a,b 是两个大于 0 的实数且 $a\leq b$.

定义 5(Timed WF-nets)

一个 Timed WF-net 模型 TN 由一个 7 元组表示：$TN=(P,T,F,L,lf,V,tf)$，其中，(P,T,F,L,lf) 表示一个标签 WF-nets，V 表示一个时间区间标签集，$tf:T\rightarrow V$ 是时间区间标签映射函数. 对于每一任务节点 $t\in T$，如果其时间区间标签为 $v\in V$，则 $tf(t)=v$.

为了更好的理解 Timed WF-nets 的定义及描述，下面给出一个具体示例，如例 1 所示.

例 1. 图 2 给出了一个用 Timed WF-net 表示的流程模型 TN_0. 在 TN_0 中，每一变迁的时间约束通过其上方的时间区间来描述，表示当前任务的执行时间及为其后继节点生成托肯的时间之和必须在该时间区间中. 例如，在模型 TN_0 中，变迁 t_2 的时间约束为[1,2]，表示变迁 t_2 的执行时间及为其后继节点 p_2，p_6 生成托肯的时间之和在 1 到 2 小时之间.

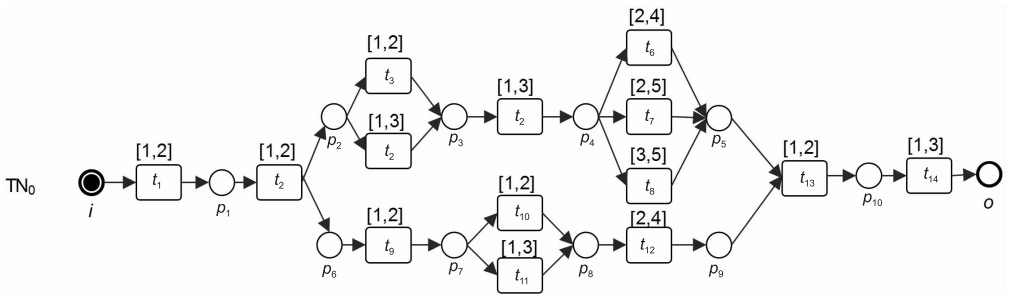


图 2 用 Timed WF-nets 表示的模型示例

为了抽取各任务间的执行次序关系，文中用可覆盖树(可覆盖树的详细介绍可参见文献[8])去构建流程模型的完全触发序列. 下面给出了触发序列(firing sequence)及完全触发序列(complete firing sequence)的定义及描述.

定义 6(触发序列).

给定一个 Timed WF-net 模型 $TN=(P,T,F,L,lf,V,tf)$，用 $t\in T$ 表示一个变迁(任务)，M_1 表示 TN 的一个标识. 当变迁 t 在标识 M_1 下是使能的，也就是对于变迁 t 的任一前置条件(库所) $p\in {}^\bullet t$ 都有 $M_1(p)=1$，可以定义以下符号表示：

(1) $M_1[t>M_2$ 表示变迁 t 在标识 M_1 下触发后，流程状态就从标识 M_1 变化得到标识 M_2，也就是对于任一 $p\in {}^\bullet t\setminus t^\bullet$，有 $M_2(p)=M_1(p)-1$，对于任一 $p\in t^\bullet\setminus {}^\bullet t$，有 $M_2(p)=M_1(p)+1$，对于任一其他库所 p，有 $M_2(p)=M_1(p)$；

(2) $\sigma=\langle t_1,t_2,\cdots,t_n\rangle\in T^*$($T^*$表示 T 的反射传递闭包)是一个从标识 M_1 到标识 M_{n+1} 的触发序列，该触发序列可通过 $M_1[\sigma>M_{n+1}$ 来表示，其中：标识 $M_2,M_3,\cdots,M_n$ 满足 $M_1[t_1>M_2[t_2>\cdots>M_n[t_n>M_{n+1}$.

定义 7(完全触发序列).

给定一个 Timed WF-net 模型 $TN=(P,T,F,L,lf,V,tf)$，其源库所为 i，汇结库所为 o. 用 $\sigma=\langle t_1,t_2,\cdots,t_n\rangle\in T^*$ 表示一个触发序列，M_i,M_o 分别表示模型 TN 的初始标识及结束标识. 当满足以下条件：

(1) $M_i(i)=1$ 且 $M_i(p)=0$ if $p\neq i$；

(2) $M_o(o)=1$ 且 $M_o(p)=0$ if $p\neq o$；

(3) $M_i[\sigma>M_o$.

此时，可认为 σ 就是模型 TN 的一个完全触发序列. 事实上，当一个流程中存在回路(环)时，它可能存在无穷多个完全触发序列. 为了便于分析，在文中抽取任务节点执行次关系时，只考虑不存在回路(无环)的流程模型. 因此，每一流程模型的完全触发序列集都是一个有限集合. 基于可覆盖树构建完全触发序列的详细过程及算法可参见文献[18]. 根据模型 TN_0 的可覆盖树，可构建图 2 所示模型 TN_0 的有限完全触发序列集(用 S 表示)：$S=\{\langle t_1,t_2,t_3,t_9,t_5,t_6,t_{10},t_{12},t_{13},t_{14}\rangle,\langle t_1,t_2,t_3,t_9,t_5,t_6,t_{11},t_{12},t_{13},t_{14}\rangle,\langle t_1,t_2,t_3,t_9,t_5,t_7,t_{10},t_{12},t_{13},t_{14}\rangle,\langle t_1,t_2,t_3,t_9,t_5,t_7,t_{11},t_{12},t_{13},t_{14}\rangle,\cdots,\langle t_1,t_2,t_9,t_{11},t_{12},t_4,t_5,t_8,t_{13},t_{14}\rangle\}$，且 $|S|=240$，表示集合 S 中共有 240 个元素，也即模型 TN_0 共有 240 个完全触发序列.

为了形式化定义任务执行次序关系，可用弱顺序关系来表示直接相邻或间接相邻的两个任务顺序执行关系，其具体描述如定义 8.

定义 8(弱顺序关系).

给定一个 Timed WF-net 模型 $TN=(P,T,F,L,lf,V,tf)$，用 $x,y\in T$ 表示模型 TN 的两个变迁，S 表示模型 TN 的完全触发序列集，可以定义以下符号表示：

(1) 变迁 x,y 满足弱顺序关系：$x>y$，其中 $>\subseteq T\times T$，当且仅当：$\exists\sigma=\langle t_1,t_2,\cdots,t_n\rangle\in S$，且在 σ 中存在 $t_j=x,t_k=y,1\leq j<k\leq n$；否则，变迁 x,y 不是弱顺序关系，也即 $!(x>y)$；

(2) $!(x>y)$ 表示在模型 TN 的完全触发序列集中不存在任何一条变迁 x 在变迁 y 之前执行的完全触发序列.

例 2.　在图 2 所示的模型 TN_0 中选 3 对变迁：t_3 与 t_6，t_5 与 t_{10}，t_3 与 t_4 作为示例来说明它们之间的弱顺序关系. 根据定义 8，基于 TN_0 的完全触发序列集可以得到：变迁 t_3 与 t_6 的关系满足：$(t_3>t_6)$ 且 $!(t_6>t_3)$；变迁 t_5 与 t_{10} 的关系满足：$(t_5>t_{10})$ 且 $(t_{10}>t_5)$；变迁 t_3 与 t_4 的关系满足：$!(t_3>t_4)$ 且 $!(t_4>t_3)$.

在文中，假定所有的 Timed WF-nets 都是无循环的. 根据弱顺序的定义并基于完全触发序列，下面给出任务执行次序关系的形式化定义.

定义 9(次序关系).

给定一个 Timed WF-net 模型 $TN=(P,T,F,L,lf,V,tf)$，用 $x,y\in T$ 表示模型 TN 的两个变迁，那么，$(x,y)\in T\times T$ 将是下面 3 种关系之一：

(1) 严格的顺序关系. $(x,y,\rightarrow)$，当且仅当：$x>y$ 且 $!(y>x)$；

(2) 并行关系. $(x,y,\|)$，当且仅当：$x>y$ 且 $y>x$；

(3) 冲突关系. $(x,y,\#)$，当且仅当：$!(x>y)$ 且 $!(y>x)$.

例 3.　在图 2 所示的模型 TN_0 中仍选 3 对变迁：t_3 与 t_6，t_5 与 t_{10}，t_3 与 t_4 作为示例来说明它们之间的执行次序关系. 根据例 2，有 $(t_3>t_6)$ 且 $!(t_6>t_3)$，表明变迁 t_3 与 t_6 的执行次序关系是严格的顺序关系，也即 $(t_3,t_6,\rightarrow)$；有 $(t_5>t_{10})$ 且 $(t_{10}>t_5)$，表明变迁 t_5 与 t_{10} 的执行次序关系是并行关系，也即 $(t_5,t_{10},\|)$；有 $!(t_3>t_4)$ 且 $!(t_4>t_3)$，表明变迁 t_3 与 t_4 的执行次序关系是冲突关系，也即 $(t_3,t_4,\#)$.

4 带时间约束的任务执行次序关系计算

4.1 带时间约束的任务执行次序关系的定义

在检索流程时,任务的执行时间及任务间执行次序关系的时间约束对于用户选取一个合适的业务流程是非常关键的.因此,需要对任务执行次序关系进行的定量化计算,也即抽取带时间约束的任务执行次序关系.为了形式化定义带时间约束的任务执行次序关系,下面给出了两个任务执行的间隔时间的定义及其计算公式,如定义 10 所示.

定义 10(间隔时间).

给定一个 Timed WF-net 模型 $TN=(P,T,F,L,lf,V,tf)$,用 $x,y\in T$ 表示模型 TN 的两个变迁(也即两个任务),$\sigma=\langle t_1,t_2,\cdots,t_n\rangle$ 表示模型 TN 的一个完全触发序列.在 σ 中,变迁 x,y 执行的最小间隔时间及最大间隔时间分别用 $TC_{\min}(x,y,\sigma)$ 和 $TC_{\max}(x,y,\sigma)$ 表示.$TC_{\min}(x,y,\sigma)$ 和 $TC_{\max}(x,y,\sigma)$ 的计算公式如式(1)及式(2)所示.

$$TC_{\min}(x,y,\sigma)=\begin{cases}\sum_{i=j}^{k-1}t_{\min}(tf(t_i)), & \text{若}\sigma\text{满足}x\succ y,\text{且}t_j=x,t_k=y,\\ & 1\le j<k\le n\\ 0, & \text{否则}\end{cases}\tag{1}$$

$$TC_{\max}(x,y,\sigma)=\begin{cases}\sum_{i=j}^{k-1}t_{\max}(tf(t_i)), & \text{若}\sigma\text{满足}x\succ y,\text{且}t_j=x,t_k=y,\\ & 1\le j<k\le n\\ 0, & \text{否则}\end{cases}\tag{2}$$

例 4. 在图 2 所示的模型 TN_0 中存在一个完全触发序列 $\sigma_1=\langle t_1,t_2,t_3,t_9,t_5,t_6,t_{10},t_{12},t_{13},t_{14}\rangle$.在完全触发序列 σ_1 中,变迁 t_3,t_6 执行的最小间隔时间及最大间隔时间分别为:$TC_{\min}(t_3,t_6,\sigma_1)=t_{\min}(tf(t_3))+t_{\min}(tf(t_9))+t_{\min}(tf(t_5))=1+1+1=3$ 及 $TC_{\max}(t_3,t_6,\sigma_1)=t_{\max}(tf(t_3))+t_{\max}(tf(t_9))+t_{\max}(tf(t_5))=2+2+3=7$.

定义 11(带时间约束的执行次序关系).

给定一个 Timed WF-net 模型 $TN=(P,T,F,L,lf,V,tf)$,用 $x,y\in T$ 表示模型 TN 的两个变迁(任务),S 表示模型 TN 的完全触发序列集,$rt\in\{\rightarrow,\|,\#\}$ 表示 x,y 的次序关系的类型,v 表示 x 被触发的时刻到 y 被触发的时刻之间的间隔时间.任务 x,y 间带时间约束的执行次序关系的形式化定义如下:

(1)如果 x 和 y 满足 $(x,y,\rightarrow)$,那么,x,y 间带时间约束的执行次序关系可表示为 $\mathrm{ortc}(x,y,rt,v)=(x,y,\rightarrow,[tc_{\min},tc_{\max}])$,其中 $tc_{\min},tc_{\max}$ 的计算公式如式(3)及式(4)所示,$(x,y,\rightarrow,[tc_{\min},tc_{\max}])$ 表示任务 x 和 y 是严格的顺序关系并且 x 被触发的时刻到 y 被触发的时刻之间的最小间隔时间及最大间隔时间分别为 $tc_{\min}$ 及 $tc_{\max}$:

$$tc_{\min}=\min_{\sigma\in S}\{TC_{\min}(x,y,\sigma)\}\tag{3}$$

$$tc_{\max}=\max_{\sigma\in S}\{TC_{\max}(x,y,\sigma)\}\tag{4}$$

(2)如果 x 和 y 满足 $(x,y,\|)$,那么,x,y 间带时间约束的执行次序关系可表示为 $\mathrm{ortc}(x,y,rt,v)=(x,y,\|,0)$.$(x,y,\|,0)$ 表示任务 x 和 y 是并行关系,x 被触发的时刻到 y 被触发的时刻之间的间隔时间可认为是 0;

(3)如果 x 和 y 满足 $(x,y,\#)$,那么,x,y 间带时间约束的执行次序关系可表示为 $\mathrm{ortc}(x,y,rt,v)=(x,y,\#,\infty)$,表示任务 x 和 y 是冲突执行关系,它们不可能在同一完全触发序列中出现,也就是它们被触发的时刻之间的间隔时间为无穷大.

为了更好地理解带时间约束的执行次序关系的定义,仍用一个示例来说明,如例 5 所示.

例 5. 在图 2 所示的模型 TN_0 中仍选 3 对变迁:t_3 与 t_6,t_5 与 t_{10},t_3 与 t_4 作为示例来说明它们间带时间约束的执行次序关系.基于上述的例 3,可以得出:变迁 t_3 与 t_6 是严格的顺序执行关系,变迁 t_5 与 t_{10} 的并行执行关系,变迁 t_3 与 t_4 的冲突执行关系.根据定义 11,它们之间带时间约束的执行次序关系可分别表示为 $(t_3,t_6,\rightarrow,[3.3,6.5])$,$(t_5,t_{10},\|,0)$,$(t_3,t_4,\#,\infty)$.

4.2 带时间约束的任务执行次序关系的计算算法时间

根据上述定义及相关描述,为了抽取流程中各任务节点间执行次序关系,可以先用可覆盖树去构建 Timed WF-nets 模型的完全触发序列集,然后,基于得到的完全触发序列集,就可计算得到模型中各任务节点间带时间约束的执行次序关系.具体的实现过程如算法 1 所示.

算法 1. 带时间约束的任务执行次序关系的计算算法(用 ORTC-CFS 表示).

输入:$TN=(P,T,F,L,lf,V,tf)$ //Timed WF-nets 流程模型(无环)

S //TN 的完全触发序列集

输出:$ORTC\_Set$ //带时间约束的任务执行次序关系集

//初始化

1. $WR\_Set=\varnothing$; //任务节点间弱顺序关系集
2. $OR\_Set=\varnothing$; //任务节点间执行次序关系集

```
3. ORTC_Set=∅; //任务节点间带时间约束的执行次序关系集
//根据定义 8 得到各任务节点间的弱顺序关系
4. FOR EACH t_i,t_j∈T∧t_i≠t_j DO
5. {IF ∃σ=⟨t_1,t_2,…,t_n⟩∈S, such that t_l=t_i and t_k=t_j, 1≤l<k≤n THEN
6. r=t_i≻t_j;
7. ELSE
8. r=!(t_i≻t_j);
9. add r to WR_Set;
10.}
//根据定义 9 得到各任务节点间的执行次序关系
11. FOR EACH t_i,t_j∈T∧t_i≠t_j DO
12.{IF t_i≻t_j∈WR_Set and !(t_j≻t_i)∈WR_Set THEN
13.    add (t_i,t_j,→) to OR_Set;
14. IF !(t_i≻t_j)∈WR_Set and !(t_j≻t_i)∈WR_Set THEN
15.    add (t_i,t_j,#) to OR_Set;
16. IF t_i≻t_j∈WR_Set and t_j≻t_i∈WR_Set THEN
17.    add (t_i,t_j,‖) to OR_Set;
18.}
//根据定义 10 及定义 11 得到各任务节点间带时间约束的执行次序关系
19. FOR EACH t_i,t_j∈T∧t_i≠t_j DO
20.{IF (t_i,t_j,→)∈OR_Set THEN
21.  {TCS=∅; TCS′=∅;
22.   FOR EACH σ∈S DO
23.   { set the value of tc according to Eq.(1);
24.     set the value of tc′ according to Eq.(2);
25.     add tc to TCS; add tc′ to TCS′;
26.   }
27.   tc_1=min(TCS); tc_2=max(TCS′);
28.   add (t_i,t_j,→,[tc_1,tc_2]) to ORTC_Set;
29.  }
30.  IF (t_i,t_j,#)∈OR_Set THEN
31.  { add (t_i,t_j,#,∞) to ORTC_Set;
32.  }
33.  IF (t_i,t_j,‖)∈OR_Set THEN
34.  { add (t_i,t_j,→,0) to ORTC_Set;
35.  }
36.}
37.  return ORTC_Set;
```

在算法 1 中，主要有四个步骤：首先，将 WR_Set，OR_Set，$ORTC\_Set$ 初始化为空集（第 1～3 行），然后，根据定义 8 得到模型中各任务节点间的弱序关系集 WR_Set（第 4～10 行）；接下来，根据定义 9 得到各任务节点间的执行次序关系集 OR_Set（第 11～18 行）；最后，根据定义 10 及定义 11 计算得到各任务节点间带时间约束的的执行次序关系集 $ORTC\_Set$，并最终返回 $ORTC\_Set$（第 19～37 行）.

根据目前我们最好的知识，文献[19]是对任务节点间执行次序关系进行精化的唯一研究工作，因此，本文将文献[19]提出的精化任务节点间执行次序关系的算法（用 ORU-CPU 表示）作为与算法 1 比较的基准算法. 与只对任务执行次序关系做定性分析与精化的算法 ORU-CPU 相比，本文所提的算法 1（用 ORTC-CFS 表示）是对任务执行次序关系进行的定量化计算，可以有效获取各任务节点间带时间约束的执行次序关系. 此外，算法 ORU-CPU 采用完全前缀展开技术去计算各任务节点间执行次序关系，它的时间复杂度是 $O(n^4)$，其中，n 表示模型中变迁个数；而算法 ORTC-CFS 基于完全触发序列对计算各任务节点间带时间约束的的执行次序关系，其中算法的最后部分（第 19～36 行）占主要花费时间. 由于完全触发序列集的大小约为 $O(2^k)$ 且模型中所有变迁对数约为 $O(n^2)$，则算法 ORTC-CFS 的时间复杂度是 $O(2^k n^2)$，其中，k 表示模型中处于冲突关系的变迁对数，n 表示模型中变迁个数. 在实际的流程模型中，通常有 $k<2\times\log_2 n$，也即 $2^k<n^2$，可以得出：$O(2^k n^2)<O(n^4)$. 因此，与算法 ORU-CPU 相比，算法 ORTC-CFS 在计算各任务节点间执行次序关系方面的效率更好. 换言之，算法 ORTC-CFS 能有效计算模型中各任务节点间带时间约束的执行次序关系.

5　基于行为的流程查询过程

为了提高基于行为的大规模云工作流模型库的检索效率，在过滤阶段构建过滤能力更强的带时间约束的任务执行次序关系索引，并用它去过滤大规模云工作流模型库，以大大减少候选模型集的规模；在精化阶段，利用带时间约束的任务执行次序关系集，提出一种更精确有效的流程行为匹配算法，并应用该算法对候选模型集进行精化验证. 下面将详细介绍带时间约束的任务执行次序关系索引（用 ORTCIndex 表示）的构建及基于带时间约束的任务执行次序关系的流程行为匹配算法.

5.1　构建带时间约束的任务执行次序关系索引

在文中，用 ORTCIndex 表示带时间约束的任务执行次序关系索引. ORTCIndex 索引是通过（ortc，model list）二元组的形式来建立流程模型与流程中两个任务间带时间约束的执行次序关系的对应关系，其中 ortc 表示两个任务间带时间约束的执

行次序关系(其详细描述如定义 11 所示),model list 表示包含该关系 ortc 的所有流程模型的集合.

为了构建带时间约束的任务执行次序关系索引,首先应用算法 1 抽取每一模型(假定为 TN)中各任务节点间带时间约束的执行次序关系;然后,判断每一关系 ortc 是否在 ORTCIndex 索引表中有记录;如果 ortc 不在 ORTCIndex 索引表中,则将当前关系 ortc 及当前模型 TN 增加到 ORTCIndex 索引表中;如果 ortc 在 ORTCIndex 索引表中,则将当前模型增加到当前关系 ortc 对应的模型列表(model list)中.很显然,当一个新流程模型增加到模型库时,ORTCIndex 索引表也可以快速更新.由于一个模型的 ORTCIndex 构建过程较为耗时,并且云工作流模型库中模型数量巨大,因此,云工作流模型库的 ORTCIndex 索引都是事先以离线方式在 Apache Lucene 中进行构建与维护的.

5.2 带时间约束的流程行为匹配算法

为了验证一个查询模型的行为与候选模型的某一子流程行为是否一致,需要以查询模型为基准来计算候选模型与查询模型的行为(任务执行次序关系)匹配度.为了更好描述候选模型与查询模型间行为匹配度的相关定义,可将从候选模型中得到的各任务间带时间约束的执行次序关系(集)称为候选关系(集),从查询模型中得到的各任务间带时间约束的执行次序关系(集)称为查询关系(集).下面将给出候选模型与查询模型的行为匹配度的相关定义和描述.

文中通过时间区间的绝对距离来量化候选关系的时间约束与查询关系的时间约束间的匹配度,以过滤掉可能不满足查询关系中相应时间约束的候选关系.

定义 12(时间区间的绝对距离).

给定两个时间区间 $v=[t_{\min},t_{\max}]$ 和 $v'=[t'_{\min},t'_{\max}]$,其中 v 表示候选关系中的时间区间(简称候选区间),v' 表示查询关系中的时间区间(简称查询区间),那么,时间区间 v 与 v' 的绝对距离可表示为 $ADist(v,v')$,其具体计算公式如式(5)所示.

$$ADist(v,v')=\begin{cases}0, & \text{若 } t_{\min}=t'_{\min} \text{ 且 } t_{\max}=t'_{\max}\\ \dfrac{|t_{\min}-t'_{\min}|+|t_{\max}-t'_{\max}|}{|t'_{\max}-t'_{\min}|}, & \text{若 } t'_{\max}\neq t'_{\min} \text{ 且 } t'_{\min}\leq t_{\min} \text{ and } t_{\max}\leq t'_{\max}\\ 1, & \text{否则}\end{cases} \tag{5}$$

根据式(5),在计算候选区间与查询区间的绝对距离时,总的可以分为 3 种情形:候选区间与查询区间重合、查询区间包含候选区间、其他情形.下面分别给出这 3 种情形的计算示例.

(1)查询区间与候选区间重合

候选区间与查询区间重合的示例如图 3 所示,当 $v=[1,1]$,$v'=[1,1]$或 $v=[1,3]$,$v'=[1,3]$时,根据式(5),可以得到:$ADist(v,v')=0$.

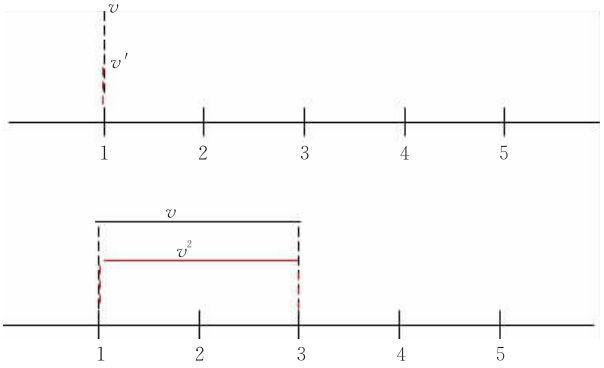


图 3 时间区间重合的绝对距离计算示例

(2)查询区间包含候选区间

查询区间包含候选区间的示例如图 4 所示,当 $v=[2,3]$,$v'=[1,4]$时,根据式(5),可以得到:$ADist(v,v')=(|4-3|+|2-1|)/|4-1|=2/3\approx 0.67$.

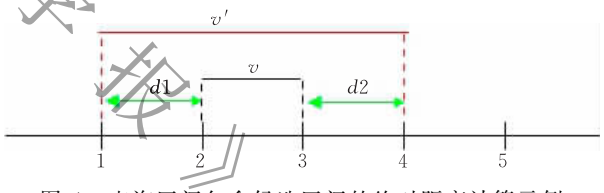


图 4 查询区间包含候选区间的绝对距离计算示例

(3)其他情形

除了上述候选区间与查询区间重合、查询区间包含候选区间之外的其他情形,例如,如图 5 所示的,候选区间包含查询区间($v=[1,2]$,$v'=[2,3]$),候选区间与查询区间相交($v=[1,3]$,$v'=[2,4]$或 $v=[2,4]$,$v'=[1,3]$)或相离($v=[1,1]$,$v'=[2,2]$或 $v=[1,3]$,$v'=[3,5]$)时,在进行带时间约束的流程行为匹配中,为了排除存在可能不满足查询区间的候选区间的情况,文中将这些情况下的候选区间与查询区间的绝对距离都定为 1,也即 $ADist(v,v')=1$.

定义 13(候选关系与查询关系匹配值).

假定候选模型与查询模型分别为 $TN_1=(P_1,T_1,F_1,L_1,lf_1,V_1,tf_1)$和 $TN_2=(P_2,T_2,F_2,L_2,lf_2,V_2,tf_2)$,用 $or_1=ortc(x_1,y_1,rt_1,v_1)$,$or_2=ortc(x_2,y_2,rt_2,v_2)$分别表示 TN_1 中 x_1,y_1 间带时间

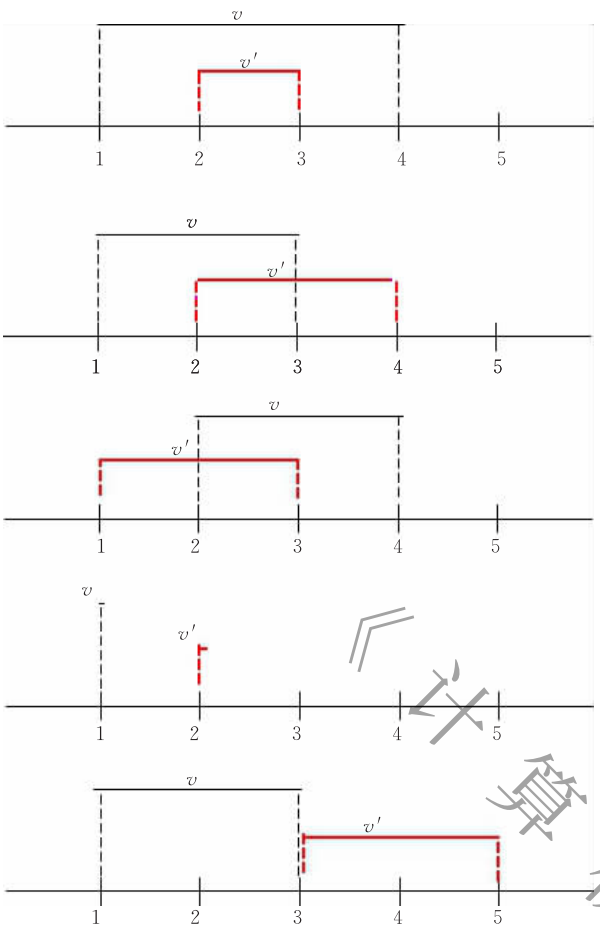


图 5　其他情形下绝对距离的计算示例

约束的执行次序关系(候选关系)及 TN_2 中 x_2，y_2 间带时间约束的执行次序关系(查询关系)，那么，关系 or_1，or_2 的匹配值可表示为 $MV(or_1,or_2)$，其计算公式如式(6)所示.

$$MV(or_1,or_2)=\begin{cases}1-ADist(v_1,v_2)，若(rt_1=rt_2=\text{‘}\rightarrow\text{’})\\ \quad 且(lf_1(x_1)=lf_2(x_2),lf_1(y_1)=lf_2(y_2))\\ 1，\quad 若(rt_1=rt_2=\text{‘}\|\text{’})\\ \quad 且((lf_1(x_1)=lf_2(x_2),lf_1(y_1)=lf_2(y_2))\\ \quad 或(lf_1(x_1)=lf_2(y_2),lf_1(y_1)=lf_2(x_2)))\\ 1，\quad 若\ (rt_1=rt_2=\text{‘}\#\text{’})\\ \quad 且((lf_1(x_1)=lf_2(x_2),lf_1(y_1)=lf_2(y_2))\\ \quad 或(lf_1(x_1)=lf_2(y_2),lf_1(y_1)=lf_2(x_2)))\\ 0，\quad 否则\end{cases}\tag{6}$$

定义 14(候选关系集与查询关系集的匹配集值).

假定候选模型与查询模型分别为 $TN_1=(P_1,T_1,F_1,L_1,lf_1,V_1,tf_1)$ 和 $TN_2=(P_2,T_2,F_2,L_2,lf_2,V_2,tf_2)$，用 ORS_1 和 ORS_2 分别表示 TN_1 的候选关系集及 TN_2 的查询关系集，那么，关系集 ORS_1 及 ORS_2 的匹配集值表示为 $MSV(ORS_1,ORS_2)$，其计算公式如式(7)所示.

$$MSV(ORS_1,ORS_2)=\sum_{or_1\in ORS_2}\sum_{or_2\in ORS_1}MV(or_1,or_2)\tag{7}$$

在式(7)中，or_1，or_2 分别表示 TN_1 的一个候选关系及 TN_2 中的一个查询关系，$MV(or_1,or_2)$ 表示关系 or_1，or_2 的匹配值.

根据定义 12 到定义 14，下面给出了候选模型与查询模型间带时间约束的流程行为匹配度的定义及其计算公式，如定义 15 所示.

定义 15(带时间约束的流程行为匹配度).

假定候选模型与查询模型分别为：$TN_1=(P_1,T_1,F_1,L_1,lf_1,V_1,tf_1)$ 和 $TN_2=(P_2,T_2,F_2,L_2,lf_2,V_2,tf_2)$，用 ORS_1 和 ORS_2 分别表示 TN_1 的候选关系集及 TN_2 的查询关系集，那么，模型 TN_1 和 TN_2 间带时间约束的流程行为匹配度可表示为 $BMD(TN_1,TN_2)$，其具体计算公式如式(8)所示.

$$BMD(TN_1,TN_2)=\frac{MSV(ORS_1,ORS_2)}{|ORS_2|}\tag{8}$$

在式(8)中，$MSV(ORS_1,ORS_2)$ 分别表示 TN_1 的候选关系集及 TN_2 的查询关系集的匹配集值，$|ORS_2|$ 表示 TN_2 的查询关系集 ORS_2 的大小. 根据式(8)，$MSV(ORS_1,ORS_2)$ 与 $|ORS_2|$ 的比值(取值范围是 0 到 1)越大，候选模型 TN_1 与查询模型 TN_2 的行为匹配度就越大；当 TN_1 的候选关系集及 TN_2 的查询关系集的匹配集为空时，它们的流程行为匹配度为 0. 但当 TN_1 的候选关系集中某一子集能够完全匹配 TN_2 的查询关系集时，它们的流程行为匹配度为 1.

为了更好地理解候选模型与查询模型间带时间约束的流程行为匹配度的相关概念及定义描述，下面给出一个具体的示例，如例 6 所示.

例 6.　假定一个模型库 R 中包含图 6 给出的 3 个用 Timed WF-net 表示业务流程模型：TN_1，TN_2，TN_3，需要检索的查询模型为 q，在 q 中包含标签分别为“B”、“C”、“D”的 3 个任务节点，且它们之间的关系满足：(“B”，“C”，→，[2,3])，(“B”，“D”，→，[2,4])，(“C”，“D”，‖). 根据定义 12 到定义 15，可以得到：$BMD(TN_1,q)\approx 0.51$，$BMD(TN_2,q)\approx 0.33$，$BMD(TN_3,q)=0$. 显然，TN_1 与 q 的流程行为匹配度最大(大于给定的行为匹配度阈值 0.4). 此时，可认为 TN_1 与 q 的行为最匹配，TN_1 是满足查询模型 q 的检索结果.

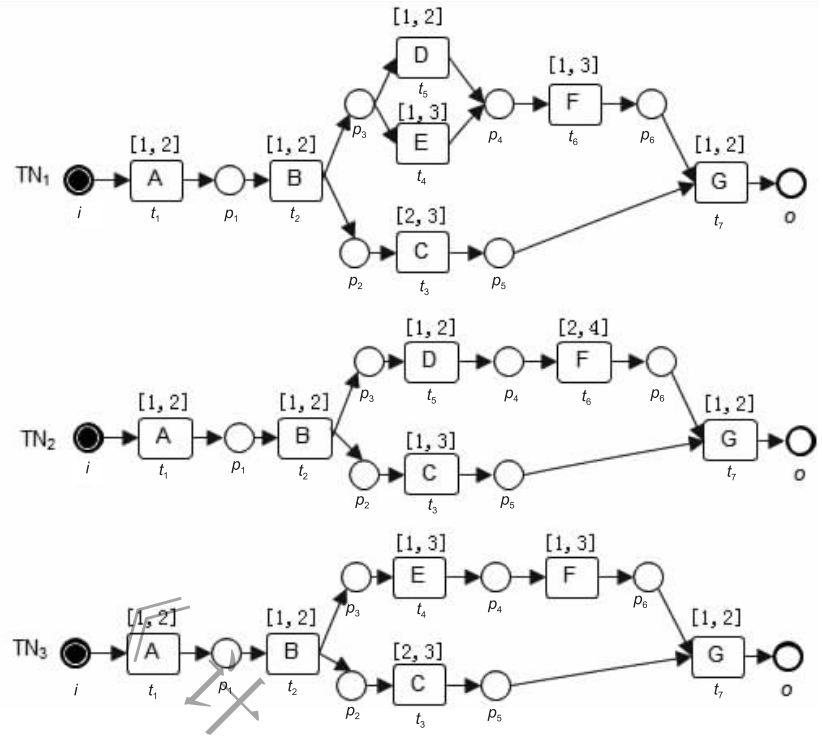


图 6 流程行为配度计算示例

为了计算候选模型与查询模型的带时间约束的流程行为匹配度,可以先用算法 1 计算得到两个模型中各任务节点间带时间约束的执行次序关系集,然后,根据定义 12～15,就可以得到候选模型与查询模型的带时间约束的流程行为匹配度,其具体实现过程如算法 2 所示.

算法 2. 带时间约束的流程行为匹配算法(用 AMPB_ORTC 表示).

输入:TN=(P_1,T_1,F_1,L_1,lf_1,V_1,tf_1) //候选模型
q=(P_2,T_2,F_2,L_2, lf_2,V_2,tf_2) //查询模型
ORS//候选模型 TN 的带时间约束的任务执行次序关系集
ORS′//查询模型 q 的带时间约束的任务执行次序关系集
输出:*BMD*(TN,q) //候选模型 TN 与查询模型 q 间带时间约束的流程行为匹配度

1. *MSV*=0;
2. FOR EACH *or*′∈*ORS*′ DO
3. { FOR EACH *or*∈*ORS* DO
4. *MSV*+=*MV*(*or*,*or*′);
5. }
//根据定义 12～15 得到候选模型 TN 与查询模型 q 间带时间约束的流程行为匹配度
6. *BMD*(TN,q)=*MSV*/|*ORS*′|;
7. return *BMD* (TN,q);

在算法 2 中,第 1 行将候选关系集与查询关系集的匹配集值初始为空,第 2～5 行用于根据定义 12～14 来计算候选关系集 *ORS* 和查询关系集 *ORS*′的匹配集值;第 6 行用于根据定义 15 来计算候选模型 TN 与查询模型 q 的带时间约束的流程行为匹配度.

5.3 改进后的基于行为的两阶段流程查询算法

基于 5.1 节及 5.2 节的描述,我们提出了一个改进的基于行为的两阶段流程查询算法,该算法将整个查询过程分成两个部分:过滤阶段及精化阶段.在第一个阶段,用带时间约束的任务执行次序关系索引 ORTCIndex 去过滤大规模云工作流模型库,得到小规模的候选流程集;然后,在第二个阶段应用带时间约束的流程行为精确匹配算法去验证候选模型与查询模型行为匹配度是否大于等于给定的阈值.下面给出了在不考虑标签语义相似性情况下基于行为的两阶段流程查询过程,其具体实现如算法 3 所示.

算法 3. 改进后的基于行为的两阶段流程查询算法.

输入:q //用 Timed WF-nets 表示的查询模型
R //Timed WF-nets 模型库
输出:Rq //查询结果模型集
//过滤阶段

```
1. S′=GetCFS(q);//获取查询模型q的完全触发序列集
2. ORS′=ORTC_CFS(q,S′); //获取查询模型q的带
                        时间约束的任务执行次序关系集
3. FOR EACH or′∈ORS′ DO
4.   { model_set=getModelListByORTCIndex(or′,R);
5.     add model_set to model_set_lists;
6.   }
7.   Rq=GetIntersection(model_set_lists);
//精化阶段
8.   FOR EACH c∈Rq DO
9.   { S=GetCFS(c);
10.    ORS=ORTC_CFS(c,S );
11.    IF AMPB_ORTC(c,q,ORS,ORS′)<θ THEN
12.      delete c from Rq;
13.  }
14.  return Rq;
```

在算法 3 中，第 1～7 行表示流程模型查询的过滤阶段；第 8～13 行表示流程模型查询的精化验证阶段. 在过滤阶段，首先，得到查询模型 q 的完全触发序列集 S'（第 1 行），并根据 S' 应用算法 1 得到查询模型 q 的带时间约束的任务执行次序关系集 ORS'（第 2 行）；然后，对于 ORS' 中的每一关系 or'，应用函数 getModelListByORTCIndex(or',R) 根据式(6)可以得到包含当前关系 or' 的所有候选模型，存于一个临时候选模型集中（第 3～6 行）；最后，对这些临时候选模型集求交集可得到包含查询模型 q 所有任务执行次序关系的候选模型集（第 7 行）. 在精化验证阶段，对于候选模型集的每一候选模型 c，第 9 行用于获取候选模型 c 的完全触发序列集 S，第 10 行用算法 1 来获取候选模型 c 的带时间约束的任务执行次序关系集 ORS，第 11 行用算法 2：AMPB_ ORTC(c,q,ORS,ORS') 去验证候选模型 c 与查询模型 q 的行为匹配度是否大于等于给定的阈值 θ. 如果不是，模候选模型 c 将从候选模型集 Rq 中删除（第 12 行）. 最后，留在模型集 Rq 中流程模型就是与查询模型 q 精确匹配的结果模型.

在实际的业务流程检索中，需要考虑节点标签的语义相似度. 因此，为了使算法 3 更适合实际应用，我们根据文献[21]提到的标签相似度计算函数 *labelsim* 对算法 3 中的式(6)进行修改，其修改后的公式如式(9)所示.

$$MV'(or_1,or_2)=\begin{cases}1-ADist(v_1,v_2), & \text{若}(rt_1=rt_2=\text{`}\rightarrow\text{'})\\ & \text{且 } labelsim(lf_1(x_1))\geq\theta'\\ & \text{且 } labelsim(lf_1(y_1),lf_2(y_2))\geq\theta'\\ 1, & \text{若}(rt_1=rt_2=\text{`}\|\text{'})\\ & \text{且}(labelsim(lf_1(x_1),lf_2(x_2))\geq\theta'\\ & \text{且 } labelsim(lf_1(y_1),lf_2(y_2))\geq\theta')\\ & \text{或}((labelsim(lf_1(x_1),lf_2(y_2))\geq\theta'\\ & \text{且 } labelsim(lf_1(y_1),lf_2(x_2))\geq\theta'))\\ 1, & \text{若}(rt_1=rt_2=\text{`}\#\text{'})\\ & \text{且}(labelsim(lf_1(x_1),lf_2(x_2))\geq\theta'\\ & \text{且 } labelsim(lf_1(y_1),lf_2(y_2))\geq\theta')\\ & \text{或}((labelsim(lf_1(x_1),lf_2(y_2))\geq\theta'\\ & \text{且 } labelsim(lf_1(y_1),lf_2(x_2))\geq\theta'))\\ 0, & \text{否则}\end{cases}\tag{9}$$

在式(9)中，函数 *labelsim* 用于计算两个标签的语义相似度，θ' 表示标签语义相似度阈值.

6　实验验证

为了对上述算法进行实验验证，我们开发了一个实验支撑平台——云工作流管理平台（http://platform.pasp.cn）. 该平台已开发上线的功能包括：云工作流可视化设计器（http://cbpm.pasp.cn/），流程模型模拟生成工具，模型检索工具，实验结果图形化显示工具. 基于模拟生成的大规模流程模型库及真实的云工作流模型库做了大量验证实验. 在这些实验中，所有索引（如 ORTCIndex 索引、TARIndex 索引及标签索引）都是通过 Apache Lucene（http://lucene.apache.org）进行管理和维护.

由于 ORTCIndex 索引的构建及改进后的基于行为的两阶段流程查询算法（算法 3）的精化验证过程都需要应用算法 1 获取模型中各任务节点间带时间约束的执行次序关系集，且候选模型验证的主要时间花费在获取候选模型中各任务节点间带时间约束的执行次序关系集，这就意味着算法 3 的时间消耗主要由过滤后的候选模型数及算法 1 的时间消耗来决定. 因此，评估实验主要需要验证算法 1 及算法 3 的效率.

6.1 合成数据集上的实验验证

(1)算法 1 在合成数据集上的评估实验

为了分析算法 1(用 ORTC-CFS 表示)的计算成本(时间消耗及存储空间)与模型大小(模型中变迁个数)之间的关系,并比较算法 ORTC-CFS 与文献[19]算法(用 ORU-CPU 表示)的计算成本,我们用云工作流管理平台中的模型生成工具按文献[10]中给定的控制流生成规则自动生成了 600 个 Timed WF-net 模型,其中每一任务节点的时间约束,也是根据设定的规则(如简单任务节点其时间约束都设定为[1,2],一般任务节点其时间约束都设定为[2,4],复杂任务节点其时间约束都设定为[3,6])自动生成.这 600 个流程模型要根据模型大小(实验中生成模型的变迁数分别为 10,20,30,40,50,60,并且冲突变迁对数一样都设定为 4)被均分成 6 个模型子集(每一模型子集中有 100 个模型).然后,分别应用算法 ORTC-CFS 及算法 ORU-CPU 在 6 个模型子集中计算每一模型中各任务节点间带时间约束的执行次序关系,每次计算都将时间消耗及存储空间记录在结果数据表中.最后,通过实验结果显示工具去查看实验结果的图形报表,如图 7 及图 8 所示.

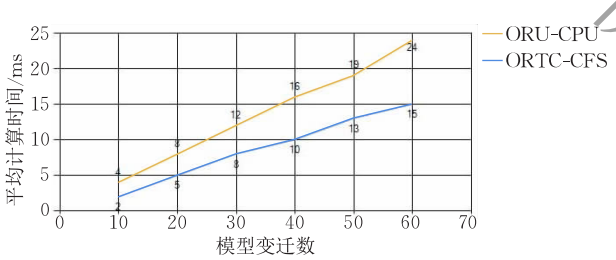


图 7 算法 ORTC-CFS 及算法 ORU-CPU 的时间消耗与模型大小之间的关系

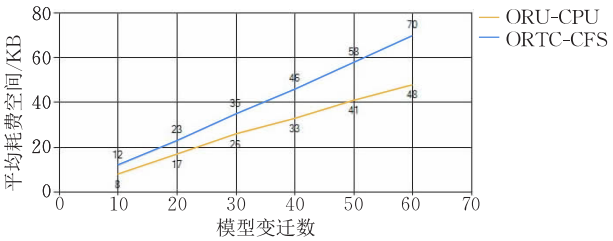


图 8 算法 ORTC-CFS 及算法 ORU-CPU 的存储空间与模型大小之间的关系

由于每次计算的时间消耗及存储空间变化趋势是相似的,因此,可以用算法的平均计算时间消耗及平均存储空间作为指标来比较算法的优劣.在图 7 中,随模型中变迁数的增多,算法 ORTC-CFS 及算法 ORU-CPU 花费的时间都相应增加;对于同一大小的模型,算法 ORTC-CFS 与算法 ORU-CPU 花费的时间更少,并且随模型大小的增大,算法 ORTC-CFS 将节省更多计算时间,这与第 4.2 节最后的算法分析是一致的.然而,如图 8 所示,随模型变迁数的增多,算法 ORTC-CFS 比算法 ORU-CPU 耗费更多存储空间,这是因为算法 ORTC-CFS 需要更多空间去计算和存储每一任务的时间约束.

此外,为了分析算法 ORTC-CFS 的计算成本与模型中冲突变迁对数之间的关系,本文用云工作流管理平台中的模型生成工具按设定的规则自动生成了另外 600 个大小一样的 Timed WF-net 模型(每一模型中变迁个数都设定为 50).这 600 个流程模型要根据模型中冲突变迁对数(实验中生成模型的冲突变迁对数分别为 2,4,6,8,10,12)被均分成 6 个模型子集(每一模型子集中有 100 个模型).然后,分别应用算法 ORTC-CFS 及算法 ORU-CPU 在 6 个模型子集中计算每一模型中各任务节点间带时间约束的执行次序关系集,每次计算的时间消耗及存储空间都是记录在结果数据表中.最后,通过实验结果显示工具去查看实验结果的图形报表,如图 9 及图 10 所示.

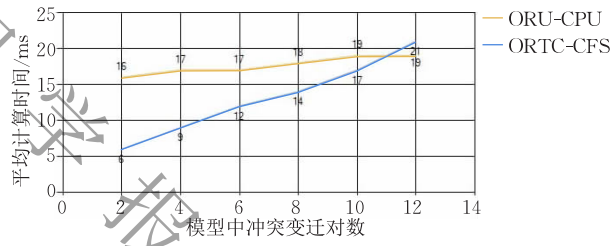


图 9 算法 ORTC-CFS 及算法 ORU-CPU 的时间消耗与模型中冲突变迁对数间的关系

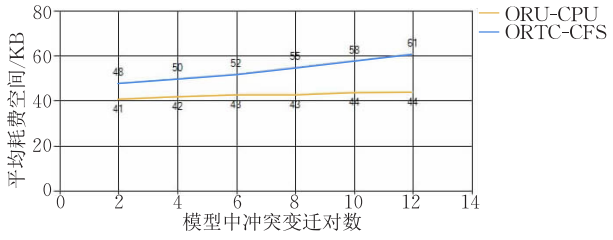


图 10 算法 ORTC-CFS 及算法 ORU-CPU 的存储空间与模型中冲突变迁对数间的关系

在图 9 中,随模型中冲突变迁对数的增多,算法 ORTC-CFS 将花费越来越多的时间,而算法 ORU-CPU 的时间消耗则变化不大.这表明,算法 ORTC-CFS 比算法 ORU-CPU 更依赖于模型中冲突变迁对数,且当模型中冲突变迁对数小于 $2\times\log_2(n)=2\times\log_2(50)\approx11$ 时(n 表示模型中变迁个数),对于具有相同冲突变迁对数的模型,算法 ORTC-CFS 比算法 ORU-CPU 花费的计算时间更少.此外,从图 10

可得出模型中冲突变迁对数对于算法所需存储空间的影响并不明显.

综上所述，与算法 ORU-CPU 相比，虽然算法 ORTC-CFS 的计算时间消耗更依赖于模型中冲突变迁对数并且需更多计算存储空间，但对于具有相同变迁数(用 n 表示)及相同冲突变迁数(用 k 表示)的模型来说，且满足 $k<2\times\log_2(n)$时，算法 ORTC-CFS 将花费更少的计算时间.

(2)算法 3 的在合成数据集上的评估实验

为了比较算法 3(不考虑标签的语义相似度时用 PS-ORTC 表示)和文献[4]提出的基于行为的两阶段流程检索算法(不考虑标签的语义相似度时用 PS-TAR 表示)及应用文献[23]中带时间约束的流程图距离度量算法进行流程检索(在检索时通过流程任务节点标签进行过滤，不考虑标签的语义相似度时用 PS-TDG 表示)的查询效率，我们用云工作流管理平台中的模型生成工具按文献[10]提出的控制流生成规则模拟生成 6 个大规模流程模型集(它们的模型数分别为 100 000，200 000，300 000，400 000，500 000 和 600 000)，基于这 6 个模型集做了一些评估实验. 在每一模型集中，所有 Timed WF-net 模型都是根据云工作流的特征(例如，存在大量相同或相似行为的流程片段)，通过模型生成工具自动生成的(每一变迁的时间约束都是根据设定的规则自动生成的). 此外，还用模型生成工具生成了 10 个 Timed WF-net 流程模型片段作为查询模型 q_i($1\leq i\leq 10$)；然后，用算法 PS-ORTC、算法 PS-TAR 及算法 PS-TDG 在每一个流程模型集中对 10 个查询模型各做一次查询，每一次查询响应时间都是保存在查询结果记录表中. 最后的实验结果如图 11 所示.

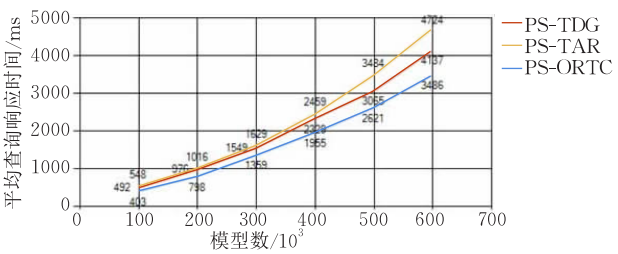


图 11　在合成数据集中查询效率的比较

在图 11 中，随流程模型集规模的增大，算法 PS-ORTC、算法 PS-TAR 及算法 PS-TDG 都要花费更多查询响应时间. 同时，对于同一大小的模型库，算法 PS-ORTC 的平均查询响应时间明显少于算法 PS-TAR 和算法 PS-TDG 的平均查询响应时间，并且随流程模型数量的增多，算法 PS-ORTC 将节省更多的查询响应时间.

为了量化两个算法(例如算法 PS-ORTC 及算法 PS-TAR)查询响应时间的差异，本文用平均加速比去比较它们之间的查询效率. 下面给出了平均加速比的定义及其计算公式.

定义 16(平均加速比 Average Speedup Rate).

应用给定的两个算法 A_1 和 A_2，应用算法 A_1 和 A_2 分别做了 n 次流程查询实验，每次实验的平均查询响应时间是分别为 t_{11}，t_{12}，…，t_{1n} 和 t_{21}，t_{22}，…，t_{2n}. 与算法 A_1 相比，算法 A_2 的平均加速比用 $asr(A_1,A_2)$ 表示，其中，$asr(A_1,A_2)$ 的计算公式如式(10)所示.

$$asr(A_1,A_2)=\left(\sum_{i=1}^{n}t_{1i}-\sum_{i=1}^{n}t_{2i}\right)\Big/\sum_{i=1}^{n}t_{1i} \quad (10)$$

与文献[4]中相应的算法 PS-TAR 及算法 PS-TDG 相比，根据定义 16 及图 11，本文所提算法 3(PS-ORTC)的平均加速比分别是 23% 和 15%. 也就是说，在不考虑标签的语义相似度时，算法 3 的查询效率明显高于文献[4]、文献[23]所提流程检索算法的查询效率.

6.2　真实数据集上的实验验证

(1)算法 1 在真实数据集上的评估实验

为了比较算法 ORTC-CFS 与算法 ORU-CPU 在实际应用中的计算成本，我们基于真实的流程模型集(例如，SAP 参考模型库)做了比较实验. 在实验中，应用云工作流管理平台中的模型导入与转换工具将 600 个 SAP 参考模型转成 Timed WF-nets 模型)转换过程中，每一 SAP 参考模型中任务节点的时间约束也是按预设规则进行自动设置). 转换后的 SAP 参考模型库的特征如表 1 所示.

表 1　SAP 参考模型库的特征

	变迁数	冲突变迁对数
Min	1.0	0
Max	53.0	10.0
Avg	7.5	3.4
Std	7.3	2.9

在真实数据的实验中，构建了 6 个模型集，每一模型集中流程个数分别为 50，100，150，200，250，300，且模型集中所有流程都是从已转换后的 600 多个 SAP 参考模型随机抽取得到. 然后，分别应用算法 ORTC-CFS 及算法 ORU-CPU 在 6 个模型集中计算每一模型中各任务节点间带时间约束的执行次序关系集. 由于每一模型集的模型都是随机从 SAP 参考模型库抽取，这就保证实验结果的结论效度是可靠的. 最终的实验结果如图 12 及图 13 所示.

在图 12 中，随模型集中模型数量的增多，算法

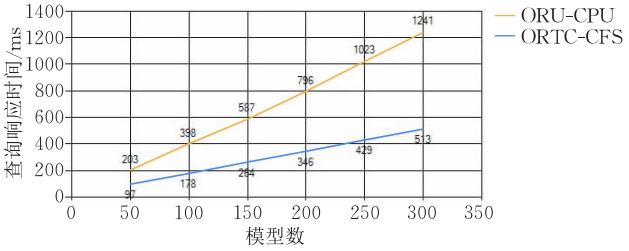


图 12 两个算法在真实数据集中的计算时间比较

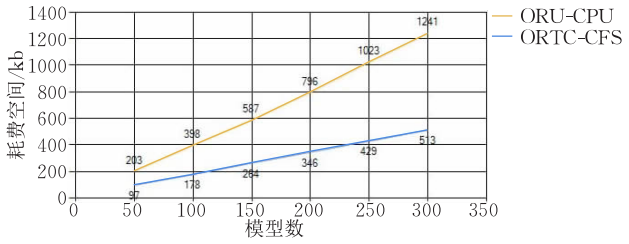


图 13 两个算法在真实数据集中的存储空间比较

ORTC-CFS 及算法 ORU-CPU 花费的时间都在增多;对于同一大小的模型集,算法 ORTC-CFS 与算法 ORU-CPU 消耗更少的计算时间,且随模型集大小增大,算法 ORTC-CFS 将节省更多的时间.此外,如图 13 所示,随模型集大小增大,算法 ORTC-CFS 比算法 ORU-CPU 却将花费更多的存储空间,这是因为,算法 ORTC-CFS 需要更多的空间去存储关系的时间约束属性.尽管如此,算法 ORTC-CFS 所需的总存储空间仍是可接受的.

(2)算法 3 在真实数据集上的评估实验

为了去观察查询响应时间与标签相似度阈值之间的关系(根据算法 3,由于流程行为匹配度阈值 θ 只影响查询精度并不影响查询响应时间,因此,在实验中设定流程行为匹配度阈值 $\theta=0.8$)并评估算法 3(考虑标签的语义相似度时用 PS-ORTC′表示)和文献[4]提出的两阶段流程检索算法(考虑标签的语义相似度时用 PS-TAR′表示)及文献[23]中相应算法(考虑标签的语义相似度时用 PS-TDG′表示)在真实的大规模云工作流模型中的检索效率.基于陶瓷云服务平台中的云工作流模型库 R'(具有 10 万多个租户业务流程模型,每一模型中无时间约束的任务将统一设置时间约束值)做了相关比较实验.首先,从 R' 中抽取 10 个真实的流程片段(这些流程片段的任务节点及流关系都是从真实的租户业务流程模型得到)作为查询模型 q_i($1\leq i\leq 10$).对于每一查询模型 q_i,都设置不同的标签相似度阈值 θ_1(θ_1 的取值范围是 0.5 到 1,增长步长是 0.1).最后的实验结果如图 14 所示.

在图 14 中,随标签相似度阈值的增大,算法

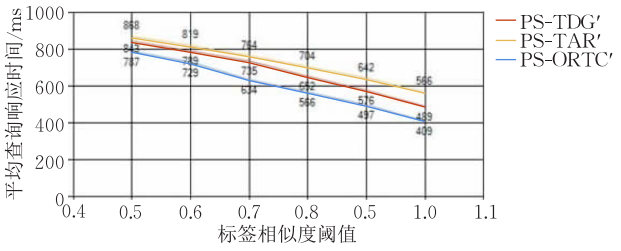


图 14 不同标签相似度阈值下查询响应时间的比较

PS-ORTC′、算法 PS-TAR′及算法 PS-TDG′的平均查询响应时间都在减少,这是因为标签相似度阈值越大,具有相似 ORTCIndex 越少,进而导致候选模型数减少,因此,节省了查询时间.此外,图 14 也表明对于同一标签相似度阈值,算法 PS-ORTC′的平均查询响应时间也少于算法 PS-TAR′及算法 PS-TDG′的平均查询响应时间,并且随标签相似度阈值的增大,算法 PS-ORTC′将节省更多的查询响应时间.

因此,在考虑标签语义相似性的情况下,与算法 PS-TAR′及算法 PS-TDG′相比,由图 14 所示的数据可以得到算法 PS-ORTC′的平均加速比分别是 17%和 11%.换言之,算法 PS-ORTC′的查询效率明显高于算法 PS-TAR′及算法 PS-TDG′的查询效率.

基于上述所有的实验结果,可以得出结论:不管是否考虑标签语义相似性,本文所提的方法能显著提高基于行为的大规模流程模型库的检索效率.

7 结论与展望

本文针对如何提高云工作流中大规模流程模型库的检索效率问题,提出了一个改进的基于行为的两阶段流程检索方法.该方法在过滤阶段通过考虑任务的执行时间约束,构建过滤能力更强的带时间约束的任务执行次序关系索引 ORTCIndex,该索引可大大减少云工作流模型库中候选模型的数量;在精化阶段,基于带时间约束的任务执行次序关系集,提出一种更精确有效的基于行为的流程行为匹配算法,并应用该算法对候选模型集进行精化验证.基于模拟生成的大规模流程模型库及真实的云工作流模型库上做了大量评估与验证实验.实验结果表明,与现有方法相比,本文所提的方法具有更高的检索效率.因此,改进后的基于行为的两阶段流程检索方法可以显著提高基于行为的大规模云工作流模型库的检索效率.

综上所述，本文提出的流程检索算法对于如何提高云工作流中大规模流程库的检索效率具有一定的指导与实践意义.但本文的研究工作还有许多需要进一步完善的地方.例如，如何支持具有环路的流程任务执行次序关系的有效计算，并结合流程任务执行的时间与概率两方面约束，去增强定量化任务执行次序关系索引的过滤能力，以进一步提高基于行为的流程模型检索的效率，这将是我们下一步需要完善和研究的工作.

致　谢　感谢清华大学软件学院的王建民老师、闻立杰老师、金涛老师与我们分享他们所掌握的实验数据资源！

参　考　文　献

[1] Baeyens T. BPM in the cloud//Proceedings of the International Conference on Business Process Management. LNCS 8094. Berlin, Germany, 2013: 10-16

[2] Huang H, Peng R, Feng Z. Efficient and exact query of large process model repositories in cloud workflow systems. IEEE Transactions on Services Computing. DOI: org /10.1109/TSC.2015.2481409

[3] Chai Xue-Zhi, Cao Jian. Cloud computing oriented workflow technology. Journal of Chinese Computer Systems, 2012, 33(1): 90-95(in Chinese)
(柴学智，曹健. 面向云计算的工作流技术. 小型微型计算机系统，2012，33(1)：90-95)

[4] Jin T, Wang J, Wen L. Efficient retrieval of similar workflow models based on behavior. Web Technologies and Applications. Berlin Heidelberg, Germany: Springer, 2012: 677-684

[5] Weidlich M, Mendling J, Weske M. Efficient consistency measurement based on behavioural profiles of process models. IEEE Transactions on Software Engineering, 2010, 37(3): 410-429

[6] Jin T, Wang J, Wong R K, et al. Querying business process model repositories. World Wide Web-Internet & Web Information Systems, 2014, 17(3): 427-454

[7] Zhang Yiwen, Cui Guangming, Zhao Shu. IFOA4WSC: A quick and effective algorithm for QoS-aware service composition. International Journal of Web and Grid Services, 2016, 12(1): 81-108

[8] Murata T. Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 1989, 77(4): 541-580

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

[10] Wang J, He T, Wen L, et al. A behavioral similarity measure between labeled Petri nets based on principal transition sequences//Proceedings of the International Conference on the Move to Meaningful Internet Systems. Graz, Austria, 2010: 394-401

[11] Dong Z, Wen L, Huang H, et al. CFS: A behavioral similarity algorithm for process models based on complete firing sequences//Proceedings of the International Conference on the Move to Meaningful Internet Systems. Amantea, Italy, 2014: 202-219

[12] Zha H, Wang J, Wen L, et al. A workflow net similarity measure based on transition adjacency relations. Computers in Industry, 2010, 61(5): 463-471

[13] Wand Wen-Xing, Wand J, et al. TAR: An improved process similarity measure based on unfolding of Petri nets. Computer Integrated Manufacturing Systems, 2012, 18(8): 1774-1784

[14] Liu H, Liu G, Wang Y, et al. A novel behavioral similarity measure for artifact-oriented business processes. Technology for Education and Learning. Berlin Heidelberg, Germany: Springer, 2012: 81-88

[15] Bergmann R, Gil Y. Similarity assessment and efficient retrieval of semantic workflows. Information Systems, 2014, 40(1): 115-127

[16] Song W, Jacobsen H A, Ye C, et al. Process discovery from dependence-complete event logs. IEEE Transactions on Services Computing, 2016, 9(5): 714-727

[17] Jin T, Wang J, Wen L. Querying business process models based on semantics//Proceedings of the International Conference on Database Systems for Advanced Applications. Berlin, Germany, 2011: 164-178

[18] Wang S H, Wen L J, Wei D S, et al. SSDT matrix-based behavioral similarity algorithm for process models. Computer Integrated Manufacturing Systems, 2013, 19(8): 1822-1831

[19] Jin T, Wang J, Wen L, et al. Computing refined ordering relations with uncertainty for acyclic process models. IEEE Transactions on Services Computing, 2014, 7(3): 415-426

[20] Song W, Jacobsen H A, Ye C, et al. Process discovery from dependence-complete event logs. IEEE Transactions on Services Computing, 2016, 9(5): 714-727

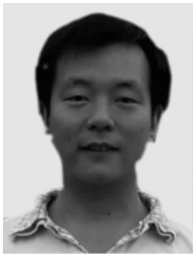
[21] Jin T, Wang J, Rosa M L, et al. Efficient querying of large process model repositories. Computers in Industry, 2013, 64(1): 41-49

[22] Zhang Yiwen, Cui Guangming, Deng Shuiguang, He Qiang. Alliance-aware service composition based on quotient space//Proceedings of the IEEE International Conference on Web Services (ICWS). San Francisco, USA, 2016: 340-347

[23] Ma Yinglong, Zhang Xiaolan, Lu Ke. A graph distance based metric for data oriented workflow retrieval with variable time constraints. Expert Systems with Applications, 2014, 41(4): 1377-1388

[24] Song W, Xia X, Jacobsen H A, et al. Efficient alignment between event logs and process models. IEEE Transactions on Services Computing, 2017, PP(99): 1-1

[25] Song W, Jacobsen H A. Static and dynamic process changes. IEEE Transactions on Services Computing. DOI: 10.1109/TSC.2016.2536025



HUANG Hua, born in 1981, Ph. D., associate professor. His research interests include service computing and business process model management.

PENG Rong, born in 1975, Ph. D., professor. Her research interests include requirements engineering, software engineering and service computing.

FENG Zai-Wen, born in 1980, Ph. D., lecturer. His research interests focus on business process management.

Background

Cloud workflow is a workflow management system deployed in cloud computing environment. A large number of tenants can design, configure and run their business processes on it. With the increase of tenants' applications, the underlying cloud workflow systems accumulate huge amount of process descriptions (i. e., business process models). For example, there are more than 2000 enterprise users (namely tenants) and 100000 business processes (roughly 50 business processes per tenant) in the cloud workflow management system of Ceramic Cloud Service Platform (CCSP for short, http://www.pasp.cn). So how to efficiently query large process model repositories in cloud computing environment is challenging[2].

To improve the efficiency of the process model retrieval, the existing process model retrieval approaches based on behavior adopt the index based on process execution path/task executing order relation to filter large-scale process repositories to reduce the number of candidate models. Due to that the construction and maintenance of the index based on process execution path is difficult, and the filtering ability of the index based on task executing order relation that does not consider the constraint attributes (e. g., time constraints) is poor in the cloud workflow model repositories including many process fragments with the same or similar behavior, these approaches are not suitable for the efficient retrieval based on behavior of large-scale cloud workflow model repositories. Thus, this paper proposes an improved two-phase process retrieval approach based on behavior to improve the retrieval efficiency based on behavior of large-scale cloud workflow model repositories.

This work is supported by the National Key Research and Development Plan of China under Grant Nos. 2017YFB0503700, 2016YFB0501801, the National Natural Science Foundation of China under Grant Nos. 61100017, 61170026, the National Standard Research Project under Grant No. 2016BZYJ-WG7-001, the Fundamental Research Funds for the Central Universities of China under Grant Nos. 2012211020203, 2042014kf0237, the Key Research and Development Plan of Jiangxi Province under Grant No. 20171ACE50022, the Natural Science Foundation of Jiangxi Province under Grant No. 20171BAB202011 and the Science and Technology Research Project of Jiangxi Education Department under Grant Nos. GJJ160906, GJJ171524.