应用 Petri 网改进 BPEL 程序的正确性*

余 波[1,2]
(1. 国防科学技术大学 计算机学院, 长沙 410073; 2. 中南林业科技大学 涉外学院, 长沙 410042)

摘 要: 为了解决 BPEL 组合服务的正确性保证问题,提出一种基于 Petri 网改善组合服务的正确性的集成方法。提出将 BPEL 程序转换成由 PNML 表示的 Petri 网的规则,实现由 BPEL 程序转换成 Petri 网和基于 Petri 网导出变迁触发序列的算法,借助一个开源工具 PIPE2 可分析与评价组合服务的正确性、活性等属性,所导出的触发序列可生成 BPEL 程序的单元测试的测试序列。案例研究表明,此方法有助于保证 BPEL 程序的正确性。

关键词: 组合 Web 服务; 可执行流程语言; Petri 网; 变迁触发序列
中图分类号: TP311 **文献标志码**: A **文章编号**: 1001-3695(2011)09-3348-03
doi:10.3969/j.issn.1001-3695.2011.09.041

Improving correctness of BPEL program with Petri net

YU Bo[1,2]
(1. *College of Computer Science & Technology*, *National University of Defense Technology*, *Changsha* 410073, *China*; 2. *College of Swan*, *Central South University of Forestry & Technology*, *Changsha* 410042, *China*)

Abstract: To solve the problem of correctness of composite Web service defined by the BPEL, this paper presented an integrated method for improving the correctness of composite Web service defined by BPEL against Petri net. The rules were firstly presented for translating the composite Web service defined by BPEL into a Petri net described by PNML. Then, the algorithms for deriving Petri net presented by PNML from BPEL program and generating transition fired sequences from Petri net were presented. With an open source tool PIPE2, the correctness, liveness and other properties of BPEL program could be verified and analyzed. Generated the test sequence for testing BPEL program from the fired sequences. The case study shows that the presented method is feasible to improve the correctness of BPEL program.

Key words: composite Web service; business process executable language; Petri net; transition fired sequence

0 引言

业务流程执行语言(BPEL4WS 或 BPEL)已经成为构建组合 Web 服务的事实标准。构建组合 Web 服务时,需要保证组合服务满足连通性、非功能质量属性、正确性以及可扩展性等属性[1]。BPEL 可保证所表示的组合服务满足服务连通性,但是不能保证其正确性,且这个问题至今没有一个完整的解决方案。BPEL 程序的形式语义的主要表示方法有进程代数、Petri 网和自动机等。Petri 网是一种具有严格数学基础的可视化建模流程的形式化方法,适合于描述并发系统的顺序、并发、冲突和同步等关系,以及证明是否存在死锁或者活锁。若将 Web 服务的方法和状态分别映射为 Petri 网的迁移和库,由此,可将 Web 服务转换成 Petri 网模型[2]。

目前,将 BPEL 程序转换成 Petri 网的工作主要分为三类:

a)基于 Petri 网检查 BPEL 的形式语义,即先将 BPEL 形式语义映射成 Petri 网,借助 Petri 网的自动验证工具检查有界性、无死锁、活性或公平性等属性。例如,针对基本和结构化的活动以及异常行为等 BPEL 流程的标准行为导出 Petri 网,对其进行分析和检查的工具[3,4];Aalst 等人[5]提出报务行为的一致性检查方法,并提供一个将 BPEL 程序转换成 Petri 网标记语言 PNML[6]表示的工具 BPEL2PNML[5]。

b)围绕业务流程的可用性、正确性或者兼容性等相关属性进行研究。例如,针对 Web 服务组合流程资源的表达及组合流程正确性验证问题,提出基于层次颜色 Petri 网的 Web 服务组合描述模型[7]。这两类工作主要应用 Petri 网的分析技术对 BPEL 服务进行形式化分析与验证。而且多数工作将 BPEL 程序转换成自定义的 Petri 网表示形式,使得转换得到的 Petri 网只能够应用特定的 Petri 网分析工具进行分析与验证。

c)针对已有工作主要关注形式分析的缺陷,Fabra 等人[8]提出解决转换生成的 Petri 网的可执行问题的解决方案,通过将 BPEL 服务转换成工作流网以仿真服务行为。

由于形式化验证技术关注程序设计阶段的缺陷,而软件的动态测试则关注程序在执行过程中是否存在缺陷,对组合服务行为的 Petri 网模型的仿真执行并不等同于对其进行动态测试,且这些 Petri 网模型生成的变迁触发序列不能够直接用于生成测试用例。因此,即使 BPEL 程序通过形式验证,对其进行动态测试有助于发现组合服务运行时的错误。

BPEL 程序的动态测试主要分为单元测试和集成测试两种类型。BPEL 程序的单元测试即其结构测试,通常假定在 BPEL 程序的单元测试中,参与构造 BPEL 程序的单个 Web 服务已经通过测试。本文所提及的测试即指 BPEL 程序的单元测试。

收稿日期: 2011-01-26; **修回日期**: 2011-03-14 **基金项目**: 国家重点基础研究发展计划资助项目(2005CB321802);湖南省教育厅高校教学改革资助项目(2010-428)

作者简介:余波(1969-),男,湖南长沙人,高级工程师,副教授,博士研究生,主要研究方向为软件测试、Web 工程(hnaxdsjk@126.com).

适合于验证目的的 BPEL 程序建模必须正确表示参与 BPEL 服务组合的各个服务的执行顺序和关注 BPEL 服务的行为状态变化;而测试目的的系统建模则要求模型能够正确表示参与 BPEL 服务组合的各个服务的执行顺序及其在执行过程中必须满足的约束或条件。因此,需要建立 Web 服务的操作到变迁和条件到库所的映射。

基于 Petri 网模型的测试用例和路径生成需解决如下问题:a)定义测试的用例和路径生成的测试充分性准则,设计满足选定的测试充分性准则的测试用例;b)P/T 网的测试中,基于路径覆盖测试设计测试用例时,需要区分可行与不可行路径。对于一条测试路径,如果存在输入使得路径是可执行的,则称此路径是可达的;如果存在一组输入数据,使得可达路径上的各个条件成立且各个操作均被执行,则称此可达路径是可行路径。目前,不存在一个可行的算法来判断路径是否是可行的。

本文提出一种集成的保证 BPEL 程序正确性的方法。先将 BPEL 程序直接转换成 Petri 网,然后应用开源工具 PIPE2 分析死锁、活性和有界性等特性,遍历 Petri 网,导出变迁触发序列,以适当的值替换变迁触发序列中的形式参数,从而得到可应用于 BPEL 程序测试的执行序列。

1　BPEL 规范和 Petri 网

1.1　BPEL 规范

BPEL 规范包含若干属性和元素定义。BPEL 活动分为不包含其他活动的基本活动和由活动集合构成的结构活动。前者包括 empty、invoke、receive、reply、assign、wait、throw 和 terminate 等;后者包括 flow、switch、while、sequence、pick、compensate 和 scope。

已有许多商用和开源的 BPEL 引擎,如 JBPEL 是 JBOSS 平台上支持 BPEL4WS 规范的引擎(http://labs.jboss.com/jbossjbpm/downloads/)。

1.2　Petri 网及表示语言

将组合服务建模成 Petri 网时,一个服务关联一个 Petri 网通过输入/输出两个端口来描述 Web 服务的行为。Petri 网可仿真顺序、选择、无序、迭代、并行、交替等组合操作[1]。

Petri 网标记语言(PNML)是由 ISO/IEC 建议的、以 XML 交换格式表示 Petri 网的一种规范[5]。PNML 定义的文档以〈pnml〉为其根元素,包含〈place〉〈arc〉和〈transition〉三类元素;每个元素又包含一些子元素或者属性,如〈arc〉元素有三个属性:id、source 和 target。

应用 PNML 标准规范表示从 BPEL 程序转换生成的 Petri 网,可实现众多支持 PNML 表示的工具上分析与验证 Petri 网的正确性等属性。目前,有许多基于 Petri 网的验证工具,如伦敦帝国学院提供的开源工具 PIPE2 可用于验证分布式系统的正确性等性质以及评价系统的性能,提供包括绘图、仿真以及定量与定性分析以 PNML 表示的 Petri 网的功能模块集。

2　BPEL4WS 转换至 Petri 网

2.1　转换规则

BPEL 程序转换成 P/T 的规则定义如下:

a)对每个基本活动,将其转换成 Petri 网的迁移。

每个原子活动 invoke、receive、reply、terminate、wait、assign、throw 和 empty 均映射成一个迁移。

b)活动的每个条件,对应 Petri 网的一个库所。

如果一个基本活动的前置条件为空,则这个库所的托肯值设置为 1。

PNML 可以表示变迁与变迁、库所与库所直接相连的情形。因此,在 BPEL 程序中,如果顺序包含多个条件或者操作,转换得到的 Petri 网也将存在多个同类节点直接相连。如果此 Petri 网应用于形式化验证,则需要合并成此类型的一个节点;如果应用于生成变迁触发序列,则不需要合并。

c)对于每个结构活动,将包含在其中的基本活动转换成迁移,将条件转换为库所。

$s_0, s_1, s_2, \cdots$ 表示库所,$T_0, T_1, T_2, \cdots$ 表示迁移,结构活动的控制结构如下所示。Sequence 结构活动包含两个可执行的活动 T_0 和 T_1,则转换生成的 Petri 网表示如图 1 所示。

〈sequence〉
　〈activity T_0 …〉
　〈activity T_1 …〉
〈/sequence〉

Switch 结构活动包含两个可执行的活动 T_1 和 T_2,则其 Petri 网表示如图 2 所示。

〈activity T_0 …〉
〈switch …〉
　〈case P_1 = true …〉〈activity T_1 …〉
　〈case P_1 = false〉〈activity T_2 …〉
〈/switch〉

若图 14 中 T_0、T_1 和 T_2 是结构活动,则 T_0、T_1 和 T_2 均对应一个 Petri 网。

图 1 表示流程顺序执行,变迁触发序列是:$\{S_0, T_0, S_1, T_1, S_2, \cdots\}$。图 2 表示根据 P_1 的取值不同,T_1 或者 T_2 被选择执行,变迁触发序列是:$\{P_0, T_0, P_1, T_1, P_2, \cdots\}$ 或 $\{P_0, T_0, P_1, T_2, P_3, \cdots\}$。在图 3 中,根据 P_1 的取值,分别执行 T_1 或 T_3,操作序列是:$P_0, T_0, P_1, \{T_1, P_2, T_2, P_1\}*, T_3, P_3$,而 $\{T_1, P_2, T_2, P_1\}*$ 可以执行零次或者多次。图 4 表示根据 P_1 的取值,分别执行 T_1 或 T_2,操作序列是:$P_0, T_0, P_1, (T_1, P_2, \cdots, T_2, P_3, \cdots)\cdots$。

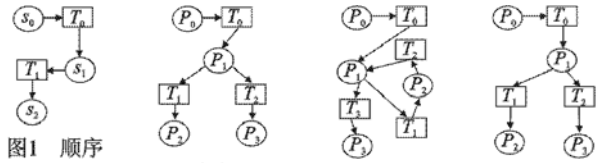

图1　顺序 sequences　图2　分支switch　图3　循环while　图4　选择pick

对于并行活动,只考虑一种简化方案,即只将并行活动转换为顺序活动序列。显然,这种方案只能够表示并行执行的一种可能情形,并不能够真正表达并行活动的执行情况。While、pick 和 scope 等结构活动的描述略。

2.2　从 BPEL 程序导出 Petri 网

算法 1　由 BPEL 程序导出 Petri 网

输入:BPEL 程序文档。

输出:以 PNML 表示的 Petri 网文档。

(1)初始化表示 Petri 网的变量 rootPetri 及其他变量;

(2)打开 BPEL 程序文档,建立文档树 rootDOC;

(3)依次遍历文档树 rootDOC 的各节点,根据元素名区分活动与条件节点,记入节点的类型域;

(4)从根节点 rootDOC 开始依次取每个节点;

(5)如果当前节点 currentNode 的类型是 sequence;

(6)按 2.1 节规则 c)和图 1 表示的规则执行(8)(10);

(7)如果是基本活动,则先创建条件节点,且 token 值置为 1,将此节点插入 Petri 网的相应位置,然后按规则 1 处理之;

(8)如果是条件,则按 2.1 节规则 b)处理之,将生成的节点插入到 Petri 网;

(9)如果当前节点是结构活动,则转(6);

(10)如果当前节点 currentNode 的类型是 switch;

(11)按 2.1 节规则 c)和图 2 等表示的规则处理之,依次执行(8)(10);

(12)如果节点类型是 pcik、while、flow、compensate 或者 scope,采取类似 switch 的处理步骤;

(13)如果是基本活动或者是条件,分别按规则 1 或规则 2 生成一个节点,在此节点的相应域记录基本活动的信息,插入到 Petri 网中;

(14)遍历 rootPetri,按照 PNML 规范写入文件;

(15)结束。

算法 1 由一个 BPEL 文档生成一个 Petri 网表示。如果文档树节点数是 n, 算法 1 的执行时间复杂度为 $O(n^2)$。

2.3　测试充分性准则

Zhu 和 He 归纳四种测试 predicate/transition (P/T)网的方法,特别提出基于代数规约测试 P/T 网的测试充分性准则,以观察等价理论为基础,定义测试 Pr/T 的面向变迁、面向状态、面向流和面向代数规约四类充分性准则。

2.4　从 Petri 网生成变迁触发序列

在 BPEL 程序的单元测试时,可以求执行根据 BPEL 程序结构构造的操作序列以检查 BPEL 程序的逻辑结构是否存在缺陷。当建立 BPEL 程序到 Petri 网的映射后,这些序列即遍历 BPEL 程序导出的 Petri 网的变迁触发序列[6]。遍历 Petri 网时,如果存在 while 结构,则有可能陷入死循环,对此,需要事先指定一个参数 degree,限定循环执行次数不超过这个值。

算法 2 使用十字链表来表示预处理过的 Petri 网,数组 tran[ ]、pred[ ]的每个元素包含 name 和 count 域。

算法 2　从 Petri 网导出全部变迁触发序列

输入:Petri 网的 PNML 表示的文档。

输出:Petri 网的变迁触发序列。

(0) $i=j=0$; 初始化 tran[ ],pred[ ];

(1)读入 Petri 网,建立文档树 rootPetri;

(2)自根 rootPetri 层次遍历文档树,依次取节点 p;

(2.1)如果 p 是谓词节点且未处理过;

(2.1.1) pred[$i++$]. name = p. name;

(2.1.2) pred[$i++$]. count = 0;

(2.2)如果 p 是迁移节点且未处理过;

(2.2.1) tran[$i++$]. name = p. name;

(2.2.2) tran[$i++$]. count = 0;

(3)预处理文档树 rootPetri;//十字链表表示 Petri 网

(3.1)对于库所,创建一个节点以表示此库所,son 指针指向迁移的第一个后继节点,cousin 指针指向第一个后继节点的第一个兄弟节点,标记域指示此节点是迁移;

(3.2)对于迁移,创建一个节点表示此迁移,son 指针指向此库所第一个后继节点,cousin 指针指向兄弟库所的第一个节点,标记域指示此节点是库所;

(3.3)将库所和迁移节点添加到十字链表 pList;

(3.4)如果 rootPetri 还有未处理节点,则转(2.1);

(4) p = pList; 初始化顺序存储结构堆栈 stack;

(4.1)全部 pList 节点依次存入 processedNode 数组;

(4.2)设置每个节点循环初值;nodeCount[$0\cdots n-1$] = 0;

(5)如果 p 为非终端节点;

(5.1)如果 p 的孩子节点非空;

(5.2)孩子节点在 nodeCount 中的标记值增 1;

(5.3)如果孩子节点的循环次数小于指定值 degree;

(5.4) p 的孩子节点进栈;修改 p 指向其孩子节点;

(5.5)从栈底至栈顶依次输出各个节点;

　　//输出一条变迁触发序列

(5.6)如果 p 是终端节点,确定 p 在 processedNode[ ]的位置 i;

(5.7) nodeCount[$i+1,\cdots,n-1$] = 0;

(5.8)当前终端节点退栈;

(5.9)如果 p 有堂兄弟节点, $p=p->$ cousin,转(5);

(5.10)如果全部 tran[ ]. count > 0,则算法结束;

(5.11)如果全部 pred[ ]. count > 0,则算法结束。

步骤(5.1)(5.9)表示循环至多遍历 degree 次。假定 Petri 网的节点数等于 n,则算法 2 的执行时间复杂度是 $O(n^2)$。若 BPEL 程序不存在多重循环,且 degree >> 1,算法 2 所需要的最大存储空间是 $O(n*\text{degree})$。如果算法 2 从(5.10)或(5.11)退出,则表明分别生成满足变迁覆盖或者谓词覆盖的触发序列。

对变迁触发序列的参数赋值可得到执行序列,这些值随机生成或者人工选择。

3　案例研究

基于上述规则与算法,实现一个原型工具。以 JBPEL 引擎发布的 Travel. bpel 为例,演示所提出的方法。

当生成的 Petri 网的某些表示迁移与库所的节点交织在一起时,可人工调整节点位置,这不会改变节点间原有的连接关系。图 5 表示生成的 Petri 网在 PIPE2 工具中的运行截图,可以运行 PIPE 工具验证所生成的 Petri 网的性质。

图5　借助PIPE2验证Petri网

下面给出由算法 2 生成的一条变迁触发序列。

{travel} TravelApproval( );

EmployeeTravelStatusRequest. employee = TravelRequest. employee ;

{employee} EmployeeTravelStatusResponse = EmployeeTravelStatus( );

FlightDetails. flightData = TravelRequest. flightData ;

FlightDetails. travelClass = EmployeeTravelStatusResponse. travelClass;

{airline} FlightAvailability( );

{airline} FlightTicketCallback( );

{airline} FlightAvailability( );

{airline} FlightTicketCallback( );

if(bpws:getVariableData('FlightResponseAA','confirmationData','/confirmationData/Price')

(下转第 3367 页)

a)本文所提质量计算式(1)(3)中每个变量的值,可通过直接在网站中手工分析的方法获取,因而可计算出准确的质量指标值,对各个数据源的排名将是实际的排名。另外,通过数据源质量估计模型得到上述公式每个变量的值,计算出的质量指标值将存在一定的误差,对各个数据源的排名将与实际排名存在一定的距离,用 Kendall 系数 K_1 来表示。

b)对文献[2]所提质量计算公式中每个变量的值,通过在网站中手工分析的方法获取,计算出准确的质量指标值,获得各数据源的实际排名;再通过文献[2]估算方法获得具有一定误差的质量指标值,用 Kendall 系数 K_2 来表示各数据源排名与实际排名之间的距离。

最后计算结果为 $K_1 = 0.628$, $K_2 = 0.573$。本文方法所得到的 Kendall 系数明显高于已有的方法,表明本文所提方法具有较好的优越性。

4　结束语

Deep Web 数据源的选择在整个集成系统中占很重要的位置,选择质量好的数据源可以给整个集成系统以更快的速度获得质量更高的数据。本文提出了一种获取查询相关性记录进而求得数据源的查准率和查全率的 FR 方法,利用查准率和查全率来建立质量估计模型,然后获取数据源的质量分数,最后选择质量最高的 N 个数据源。通过与同类方法的比较,验证了本文所提方法的优越性。

参考文献:

[1] GRAVANO L, IPEIROTIS P G, SAHAMI M. QProber: a system for automatic classification of hidden-Web databases[J]. ACM Trans on Information Systems, 2003,21(1): 1-41.

[2] LI Wei-jing, YUAN Fang, ZHANG Ming. Selection of Deep Web database based on retrieval performance[C]//Proc of the 2nd International Workshop on Education Technology and Computer Science. 2010:72-75.

[3] QU Zhen-dong, SHEN De-rong, YU Ge, *et al*. DSSM: a data sources selection model for Deep Web[C]//Proc of the 6th Web Information Systems and Applications Conference. 2009:163-168.

[4] YUAN Fang, FAN Jing-chuan, LIU Shuang. Selection of Deep Web database based on query relevance[J]. Journal of Information and Computational Science, 2009,6(11):1113-1120.

[5] LIU V Z, LUO R C, CHO J, *et al*. DPro: a probabilistic framework for hidden Web database selection using dynamic probing[R]. [S. l.]: Computer Science Department, University of California, Los Angeles, 2003.

[6] WEIS M, NAUMANN F. Detecting duplicate objects in XML documents[C]//Proc of International Workshop on Information Quality in Information Systems. New York: ACM Press, 2004:10-19.

[7] YU C, PHILIP G, MENT Wei-yi. Distributed top-*n* query processing with possibly uncooperative local systems[C]//Proc of the 29th International Conference on Very Large Data Bases. 2003:117-128.

[8] PIPINO L L, LEE Y W, WANG R Y. Data quality assessment[J]. Communication of the ACM, 2002,45(4): 211-218.

[9] KNUTH D E. 计算机程序设计艺术[M]. 北京:国防工业出版社, 2002.

[10] Open directory[EB/OL]. [2011-02-21]. http://www.dmoz.org.

[11] VULTURE. Kendall 等级相关系数[EB/OL]. [2011-03-29]. http://wiki.mbalib.com/wiki/Kendall%E7%AD%89%E7%BA%A7%E7%9B%B8%E5%85%B3%E7%B3%BB%E6%95%B0.

(上接第 3350 页)

```
⇐ bpws:getVariableData('FlightResponseDA','confirmationData','
    /confirmationData/Price'));
    TravelResponse = FlightResponseAA ;
    {travel}ClientCallback();
```

上面的示例中,“{}”包含服务名,“()”表示前面的标志名是操作名,“()”包含操作参数,“=”表示赋值,“if”后附带条件。这个变迁触发序列即测试序列。

4　结束语

针对组合 Web 服务的正确性的保证问题,提出一种结合形式化验证和软件测试的保证 BPEL 程序正确性的集成方法。与已有转换工作相比,该方法从测试的角度对 BPEL 程序建立由 PNML 表示 Petri 网模型,此模型可在支持以 PNML 表示的 Petri 网为输入的验证工具环境验证与分析 BPEL 服务的性质;提出由 Petri 网导出变迁触发序列的算法,并实现一个原型工具。案例研究表明,转换得到的 Petri 网既可用于 Petri 网验证,又可生成测试序列,有助于改善 BPEL 程序的正确性。将来的工作是结合 Petri 网的数据流等其他测试充分性准则,研究并解决生成满足所选定的测试充分性准则的变迁触发序列的问题。此外,还需研究对变迁触发序列所包含的参数自动赋值的方法,以得到可执行序列。

致谢　中南林业科技大学涉外学院庞泽亮参与原型工具的部分实现工作。

参考文献:

[1] MILANOVIC N, MALEK M. Current solutions for Web service composition[J]. IEEE Internet Computing, 2004,8(6):51-59.

[2] DONG Wen-li, YU Hang, ZHANG Yu-bing. Testing BPEL-based Web service composition using high-level Petri nets[C]//Proc of the 10th IEEE International Enterprise Distributed Object Computing Conference. Washington DC:IEEE Computer Society,2006:441-444.

[3] HINZ S, SCHMIDT K, STAHL C. Transforming BPEL to Petri nets [C]//Proc of International Conference on Business Process Management. Berlin Heidelberg:Springer-Verlag, 2005:220-235.

[4] OUYANG Chun, VERBEEK H M W, Van der AALST W M P, *et al*. Formal semantics and analysis of control flow in WS-BPEL[J]. Science of Computer Programming, 2007,67(2-3):162-198.

[5] Van der AALST W M P, DUMAS M, OUYANG Chun, *et al*. Conformance checking of service behavior[J]. ACM Trans on Internet Technology, 2008,8(3):1-30.

[6] KINDLER E. Concepts, status, and future directions[R]. Fachtagung, Braunschweig:EKA, 2006:35-55.

[7] 李景霞,程久军. Web 服务组合的层次颜色 Petri 网描述模型[J]. 计算机工程, 2009,35(24):39-40,44.

[8] FABRA J, FABRA P A. BPEL2DENEB: translation of BPEL processes to executable high-level Petri nets[C]//Proc of the 4th International Conference on Internet and Web Applications and Services. Washington DC:IEEE Computer Society,2010:496-505.

[9] ZHU Hong. A methodology of testing high-level Petri nets[J]. Information and Software Technology, 2002,44(8):473-489.