

基于 TLA 的正确性验证方法

庞士焕¹, 朱相冰², 张琦³, 汤萍萍⁴

(1. 安徽师范大学 教育科学学院, 安徽 芜湖 241000;

2. 安徽师范大学 物理与电子信息工程学院, 安徽 芜湖 241000;

3. 西北大学 软件学院, 陕西 西安 710000;

4. 东南大学 计算机学院, 江苏 南京 211189)

摘要:随着面向服务的体系结构的发展,有效地组合单个分布的 web 服务以提供更有价值的服务成为新的热点问题。然而,在这一研究领域还存在诸多问题,比如 web 服务用哪种方式组合,能否实现自动组合,对组合服务进行正确性验证等等。文中主要是针对组合服务的正确性验证问题,引入时序逻辑 TLA。通过把组合服务的 BPEL 描述转换为 TLA 可以理解的自动机的形式,这种方法可以很好地验证组合逻辑的正确性以及快速发现死锁等问题。

关键词:Web 服务; TLA; BPEL; Conversation 模型; 正确性验证

中图分类号: TP393

文献标识码: A

文章编号: 1673-629X(2009)03-0055-04

A Method of Verification Based on TLA

PANG Shi-huan¹, ZHU Xiang-bing², ZHANG Qi³, TANG Ping-ping⁴

(1. Department of Educational Science, Anhui Normal University, Wuhu 241000, China;

2. Department of Physics and Electronic Information, Anhui Normal University, Wuhu 241000, China;

3. Department of Software Engineering, Northwest University, Xi'an 710000, China;

4. Department of Computer Science, Southeast University, Nanjing 211189, China)

Abstract: With the evolution of service-oriented architecture, composing distributed web services to provide more valuable one has become a key area in the software engineering research. However, the research on these emerging technologies is far from mature, such as how to compose web service, to realize automatic composition, composition verification and so on. In this paper, a temporal logic named TLA is introduced to solve the problem of composition verification. The BPEL description of composed web services will be transformed to automation which can be used by TLA, and this method is effective to verify composed services and find dead lock.

Key words: Web service; TLA; BPEL; conversation model; verification

0 引言

Web 服务(Web services)当前正引起工业界和学术界的广泛关注。它是自包含的、模块化的商业应用,基于 XML, SOAP, WSDL 和 UDDI 等标准,通过 Internet 被发布、定位和调用。正是由于 Web 服务的上述特性,使得它非常适合于当今商务应用的集成。Web 服务组合就是将已有的单个服务按照一定的逻辑组织起来以构建新服务,从而节约成本并实现更为强大的功能。然而,在这一研究领域还存在诸多问题^[1],比

如 web 服务用哪种方式组合,能否实现自动组合,对组合服务进行正确性验证等等。

文中主要针对组合的正确性验证方面展开研究。大量的文献资料表明,服务组合有很多方法,这些方法被归并为两大类,即基于流程组合的思想,其中最典型的是 BPEL 组合方法,以及借助于人工智能的思想,如语义 Web。而文中的研究前提是采用典型的 BPEL 组合方法,即基于已有的组合服务的 BPEL 描述文档来研究组合的正确性与否。关于组合的正确性验证目前已经有若干种形式化方法,如 Petri 网、进程代数、时序逻辑等。在这些验证解决方案中,时序逻辑的方法由于引入了时间因素,更容易表示同步和异步的概念,对于描述一个并发系统有更多的优势。因此,基于时序逻辑研究 Web 服务组合是很有意义的。

收稿日期:2008-07-19

基金项目:安徽省自然科学基金项目(KJ2007B061);安徽省教育厅重点项目(KJ2008A142C)

作者简介:庞士焕(1955-),女,安徽芜湖人,实验师,研究方向为计算机科学。

TLA(Temporal Logic of Actions)^[2]是 Leslie Lamport 提出的一种逻辑,它将时序逻辑与传统逻辑有机地结合起来,能够很好地描述系统的行为,有较强的描述能力和推理能力。将基于 TLA,把组合服务的 BPEL 描述文档转化成 TLA,然后用 TLA 自带的检验工具 TLA+ 来验证^[3]。最后还需强调一点,文中的重点工作是如何完成从服务组合的 BPEL 描述到 TLA 的转换,至于如何对组合服务进行 BPEL 描述,以及如何使用 TLA 的检验工具 TLA+ 进行验证均不在讨论范围之内。

1 BPEL 规范简介

BPEL 的全称是 BPEL 4WS(Business Process Execution Language for Web Services, Web 服务的业务流程执行语言),主要是负责将一组现有的单个 Web 服务整合起来,从而定义一个新的组合服务。BPEL 是 WSFL 和 XLANG 融合的产物,已成为学术界和工业 Web 服务组合的主要描述语言之一。下面介绍几个 BPEL 的关键概念:

BPEL 流程类似于算法的流程图,流程的每一步称为一个活动。BPEL 的活动分为基本活动和结构化活动:

(1)基本活动,是与外界进行交互的最简单的形式,与服务进行交互、传输数据或者处理异常,存在以下几个元素:<invoke>(调用某个 Web 服务上的操作),<receive>(等待一条消息来响应由某人从外部进行调用的服务接口的操作),<reply>(生成输入/输出操作的响应),<wait>(等待一段时间),<assign>(把数据从一个地方复制到另一个地方),<throw>(指明某个地方出错了),<terminate>(终止整个服务实例),<empty>(什么也不做)。

(2)结构化活动,规定了一组活动发生的顺序。结构化活动通过将业务流程执行的基本活动整合到结构中来描述业务流程是如何被创建的,它包括:<sequence>(顺序执行),<switch>(产生分支),<while>(循环),<pick>(执行几条可选路径中的一条),<flow>(并行执行)。

2 TLA 简介

TLA^[1](Temporal Logic of Actions,简称 TLA)是由 Lamport 提出的,用来描述和推理并发系统的一种时序逻辑。使用 TLA 描述一个并发系统时,需要将系统的动作(行为)分解为有序的状态集合,再将状态的变化描述为相应的逻辑公式。TLA 公式包含以下的

形式^[3]: $P \Box [A]_f \Box F \exists x: F \rightarrow F \wedge G F \vee G F \rightarrow G F \equiv G$ 等等。

下面将对部分表达式的含义做出解释:

σ 表示一个行为, σ_i 表示该行为的第 $(i+1)$ 个状态。

设 P 是一个 predicate, $\sigma \models P$ 为真当且仅当行为 σ 的初始状态满足 P ,即 σ_0 满足 P 。

设 A 是一个 action, $\sigma \models A$ 为真当且仅当行为 σ 的开始两个状态满足 A ,即 $\sigma_0 \rightarrow \sigma_1$ 是一个满足 A 的 step。

$\Diamond F$ 等于 $\neg \Box \neg F$,表示 F 将在某个时刻为真。

$F \vee G$ 等于 $\Box (F \Rightarrow \Diamond G)$,表示只要 F 为真, G 最终也会为真。同理,可将 $F \wedge G$ 展开。

$\Box [A]_f$: 一个行为 (behavior) 满足 $\Box [A]_f$ 当且仅当该行为的任意 step 满足 A 或 f 不发生变化。

$\exists x: F$: 一个行为 (behavior) 满足 $\exists x: F$ 当且仅当存在 x 的某一赋值后产生的行为满足 F 。

标准的 TLA 的描述形式形如 $\text{Init} \wedge \Box [\text{Next}]_v \wedge \text{Liveness}$ 。其中, Init 指初始状态; Next 指下一状态关系,规定了系统所有可能的动作, v 是所有变量组成的元组,且 $[\text{Next}]_v = \text{Next} \vee (v' = v)$; Liveness 则是一个时序公式,通过合取各动作的公平性条件来规定系统的活性属性。一个 TLA 的表达式,本质上描述的的是一个状态机。

3 BPEL 转化为 TLA 的研究

组合服务的 BPEL 描述实际上是一个文档,本身不能直接转换为 TLA 逻辑,它们之间必须有一个可用的桥梁,那就是服务组合的 Conversation 模型^[4]。Conversation 模型在形式化上是各种自动机的组合,而 TLA 本质上描述的也是一个状态自动机,正好衔接起来。于是我们的工作首先是将 BPEL 文档转化为 Conversation 模型的形式化描述语言,再将 Conversation 模型转换为 TLA 逻辑,从而最终完成了从 BPEL 到 TLA 的转换。

3.1 Web 服务组合的 Conversation 模型

在 Web 服务组合的 Conversation 模型中,各个原子服务间通过发送和传递消息进行交互。由于是异步传送,每个服务配备一个 FIFO 队列,用于存放接收的消息(如图 1 所示)。单个服务的内部流程使用 mealy 自动机描述。

下面给出 Conversation 模型^[4]的形式化描述。在该模型中,一个组合的 Web 服务可以表示成元组 $S = ((P, M), A_1, A_2, \dots, A_n)$ 。 (P, M) 中 P 是单个原子服务的集合, M 是所有消息类型的集合; A_1, A_2, \dots ,

A_n 对应着 n 个原子服务,对于第 i 个服务 A_i 是一个 mealy 自动机 $(M_i^{\text{in}}, M_i^{\text{out}}, T_i, s_i, F_i, \Delta_i)$ 。这里, M_i^{in} , M_i^{out} 是该服务输入或输出的消息类型; T_i, s_i, F_i 分别表示所有状态的集合、初始状态的集合与最终状态的集合; Δ_i 为状态转换函数。

对于上述的一个合成的 Web 服务 $S = ((P, M), A_1, A_2, \dots, A_n)$, 从全局的角度可以表示成以下的形式: $\gamma = (Q_1, t_1, \dots, Q_n, t_n, w)$, 对于第 i 个 Web 原子服务 Q_i, t_i 分别表示服务的消息队列和服务的状态; w 表示全局角度记录的 Web 服务交互的消息序列。

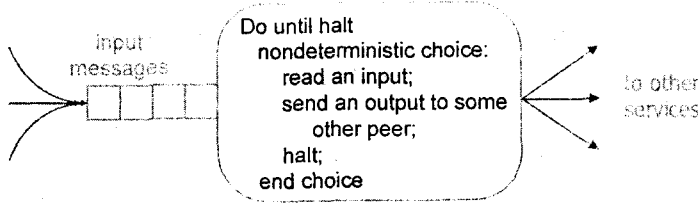


图1 Conversation模型中的单个服务

把组合服务 S 一次运行后 w 消息序列称为 S 的一个会话(Conversation),使用 $L(S)$ 表示 S 所有会话的集合。

3.2 BPEL 描述转化为 Conversation 模型

组合服务的 Conversation 模型中每一个原子服务都是含有 FIFO 队列的 mealy 自动机,因此 BPEL 转化为 Conversation 模型的问题也就变为如何将 BPEL 文档转化为一个这样的自动机。问题的解决分为两步:

1) 按一定的规则生成 Conversation 模型中服务之间交互所传递的消息。针对 BPEL 的每个活动以及其中包含的信息可以生成相应消息,消息生成的算法如下(以 $\langle \text{invoke} \rangle$ 、 $\langle \text{receive} \rangle$ 为例):

①关于 $\langle \text{invoke} \rangle$ 活动:如果该活动的 `outputVariable` 属性值为空(表示异步操作,调用后没有返回值),生成消息 `sOwner.Operation.IN`。这里 `sOwner` 是被调用 Web 服务的进程名字, `Operation` 是该活动的 `operation` 属性值。如果该活动的 `outputVariable` 属性值不为空(表示同步操作,有返回值),在生成消息 `sOwner.Operation.IN` 之外,再生成消息 `sOwner.Operation.OUT`, `sOwner` 与 `Operation` 含义同上。

②关于 $\langle \text{receive} \rangle$ 活动:生成消息 `Owner.Operation.IN`。这里, `Owner` 是 $\langle \text{receive} \rangle$ 活动所在 Web 服务的进程名字, `Operation` 是该活动的 `operation` 属性值。

2) 根据每个 BPEL 流程的结构生成相应的 mealy 自动机。将基本活动转化为一个自动机^[5]。以 $\langle \text{receive} \rangle$ 、 $\langle \text{reply} \rangle$ 为例,其他的基本活动可以使用类似的方式处理:

① $\langle \text{receive} \rangle$ 活动如图 2 所示。



图2 $\langle \text{receive} \rangle$ 活动转化为自动机

② $\langle \text{reply} \rangle$ 活动如图 3 所示。

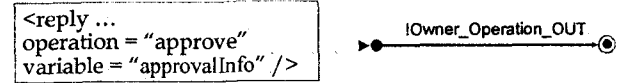


图3 $\langle \text{reply} \rangle$ 活动转化为自动机

这里,“?”表示接收一个消息,“!”表示发送消息。

结构化活动中可能包含基本活动和结构化活动。算法基本思想是按一定的规则,将其所包含活动对应的自动机组组合一个大的自动机^[5]。这里仅讨论 $\langle \text{sequence} \rangle$ 和 $\langle \text{flow} \rangle$ 活动,其他的结构化活动可以使用类似的方式处理。

$\langle \text{sequence} \rangle$ 活动:只要将包含的所有活动首尾相连即可(如图 4 所示)。

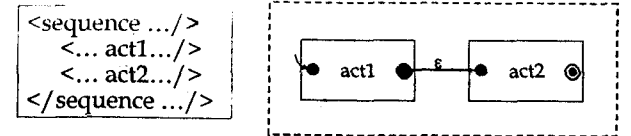


图4 $\langle \text{sequence} \rangle$ 活动转化为自动机

$\langle \text{flow} \rangle$ 活动:在 $\langle \text{flow} \rangle$ 中,首先找出满足 `link` 条件的一系列活动(如图 5 所示),再将这些活动对应的自动机做笛卡儿乘积。

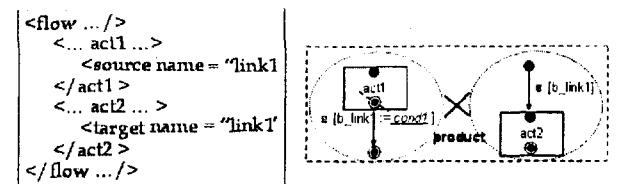


图5 $\langle \text{flow} \rangle$ 活动转化为自动机

3.3 Conversation 模型转化为 TLA

利用 Conversation 模型将系统描述为 TLA 表达式,基于 Conversation 模型的组合服务 $S = ((P, M), A_1, A_2, \dots, A_n)$, 算法如下:

步骤 1, 对每个服务 A_i 引入变量 `Curstatei`, Q_i ; 这里, `Curstatei` $\in T_i$; Q_i 表示该服务的消息队列。

步骤 2, 初始化。 $\forall A_i \in S$ 有 `Curstatei = si`, $Q_i = \langle \rangle$, 将它们合取, 得到 `Init` 表达式。

步骤 3, 对于每个服务 A_i , $\forall \tau_j \in \Delta_i$,

a) $\tau_j = (q_1, !\text{msg}, q_2)$, 设消息是发送给服务 A_i 的, 则添加 `action`:

$$A_{i_step_} \tau_j \equiv \wedge \text{Curstate}_i = q_1 \wedge \text{Append}(Q_i,$$

$\text{msg}) \wedge \text{Curstate}_i' = q_2 \wedge \text{UNCHANGED} < \text{other variables} >$

这个表达式将 Curstate_i 状态由 q_1 变为 q_2 , 同时将消息 msg 存放到目的服务 A_i 的队列中, 并保持其他变量的值不发生改变。

b) $\tau_j = (q_1, ?\text{msg}, q_2)$, 添加 action:

$A_i\text{-step-}\tau_j \equiv \wedge \text{Curstate}_i = q_1 \wedge \text{Head}(Q_i) = \text{msg} \wedge \text{Tail}(Q_i) \wedge \text{Curstate}_i' = q_2 \wedge \text{UNCHANGED} < \text{other variables} >$

这个表达式将 Curstate_i 状态由 q_1 变为 q_2 , 检测服务 A_i 的队列头是否为 msg 消息, 如果是从队列中删除该消息, 并保持其他变量的值不发生改变。

步骤 4, 针对组合服务一次执行终止时的状态, 添加 action:

$S\text{-step-}\text{reset} \equiv \wedge \forall A_i \in S: \text{Curstate}_i \in F_i, Q_i = < >$

$\wedge \forall A_i \in S: \text{Curstate}_i' \in s_i, Q_i = < >$

这个表达式表示当所有服务都进入自己的终态, 且各自队列为空时; 所有变量恢复到初始状态, 开始新一轮的运行周期。

步骤 5, 将步骤 3 和 4 中得到表达式析取, 得到 Next 表达式。

步骤 6, 得出整个系统的 TLA 表达式, $\text{Init} \wedge \square[\text{Next}]_v$ 。

4 结束语

TLA 作为一种强大的并发系统的描述工具, 可以

从不同的角度、不同的层次对一个并发系统进行描述和验证。文中是从组合服务的 Conversation 模型导出 TLA 的描述文档, 而其他的 Web 服务的组合模型 (如 Roman 模型, Owl-S 模型) 都可以进行类似的转换, 它们将从不同的角度描述组合服务的相关属性。除了 BPEL, 工业界和学术界还有很多其他组合服务描述语言 (如 WS-CDL, OWL-S), 研究如何将它们建模, 进而转化为 TLA 描述文档是很有意义的工作。

讨论了如何将 BPEL 描述转化为 Conversation 模型, 仅仅给出了部分具有代表意义的基本活动 ($<\text{invoke}>$, $<\text{receive}>$, $<\text{reply}>$, $<\text{assign}>$) 和结构化活动 ($<\text{sequence}>$, $<\text{flow}>$) 的转化算法, 而其他活动的转化方法, 需要继续研究。

参考文献:

- [1] Milanovic N, Malek M. Current Solutions for Web Service Composition[D]. New York, USA: IEEE Internet Computing, IEEE Educational Activities Department, 2004: 51-59.
- [2] Lamport L. Introduction to TLA[R]. Canada: University of Waterloo, 1994.
- [3] Lamport L. Specify System: The TLA+ Language and Tools for Hardware and Software Engineers[M]. [s. l.]: Addison-Wesley, 2002.
- [4] Fu X, Bultan T, Su J. Conversation protocols: A formalism for specification and verification of reactive electronic services [C]// Essex, UK: Implementation and application of automata, Elsevier Science Publishers Ltd., 2004: 19-37.
- [5] Fu X, Bultan T, Su J. Analysis of interacting BPEL Web Services[C]// In the Proc. of 13th Int. World Wide Web Conf(WWW). New York, USA: ACM, 2004: 621-630.

(上接第 54 页)

于节约法所得到的结果 79.5km。而且第 4 次还得到了最优解 67.5km, 其对应的配送路径为: 0-4-7-6-0; 0-1-3-5-8-2-0。可见, 利用遗传算法可以方便有效地求得物流配送路径优化问题的最优解或近似最优解 (或称满意解)。

4 结束语

从上述的实验结果可以看出, 遗传算法在物流配送路径优化中获得了良好的效果。证明了用遗传算法在解决诸如车辆路径问题确实具有优良的性能。利用其可以方便有效地求得物流配送路径优化问题的最优解或满意解。遗传算法的最大优点在于能够快速地找到较优解, 不必在所有的可行解中去搜索; 不足之处在于求解过程中有可能陷入局部的最优解中, 无法找到

全局的最优解。

参考文献:

- [1] 周明, 孙树栋. 物流合理化的数量方法遗传算法及应用 [M]. 北京: 国防工业出版社, 1999.
- [2] 朗茂祥. 基于遗传算法的物流配送路径优化问题研究[J]. 中国公路学报, 2002, 15(3): 77-78.
- [3] 陈国良, 王煦法, 庄镇泉, 等. 遗传算法及其应用 [M]. 北京: 人民邮电出版社, 1996.
- [4] 刘刚. 用改进的遗传算法优化物流配送中心的选址[J]. 上海商学院学报, 2007, 8(3): 55-57.
- [5] 玄光男, 程润伟. 遗传算法与工程优化 [M]. 北京: 清华大学出版社, 2004.
- [6] 王家文, 王皓, 刘海, 等. MATLAB7.0 编程基础 [M]. 北京: 机械工业出版社, 2005.