

基于 CHAM 的 SA 级全路径覆盖方法

赵 磊¹, 徐士华², 赵国生¹, 伦立军³

(1 哈尔滨师范大学 现代教育技术中心, 黑龙江 哈尔滨 150025; 2 黑龙江司法警官职业学院, 黑龙江 哈尔滨 150060;
3 哈尔滨师范大学 计算机科学与信息工程学院, 黑龙江 哈尔滨 150025)

摘 要: 文中提出了通过 CHAM 描述的 SA 规格说明生成 LTS, 并根据测试需求进行测试覆盖准则的选取, 然后利用全路径测试方法, 生成基于此覆盖准则的测试路径. 最后以 B/S 结构为例, 验证了该方法在生成 SA 级的测试路径上是可行的.

关键词: 软件体系结构; 测试路径; 标号迁移系统; 测试覆盖准则; 化学抽象机

中图分类号: TP311.5

文献标识码: A

文章编号: 1000-7180(2010)04-0140-05

Method of All Path Coverage at SA Level Based on CHAM

ZHAO Lei¹, XU Shi-hua², ZHAO Guo-sheng¹, LUN Li-jun³

(1 Modern Education and Technology Center, Harbin Normal University, Harbin 150025, China;

2 The Professional Judicial Police College of Heilongjiang, Harbin 150060, China;

3 Institute of Computer Science and Information Engineering, Harbin Normal University, Harbin 150025, China)

Abstract: This paper proposes an approach to generate LTS of SA specification described by CHAM. According to the testing requirement we select testing criteria. By using all paths testing method, we generate testing paths based on the testing criteria. Finally, Regarding Browser/Server architecture as an example, the method on generating testing paths of SA is shown feasible.

Key words: software architecture; testing path; LTS; testing criteria; chemical abstract machine

1 引言

利用 SA 模型来指导测试是目前软件测试的一个重要研究领域. 软件体系结构描述语言 ADLs 从不同方面对软件体系结构进行规约和建模分析, 是对 SA 的形式化描述^[1]. 这些 ADLs 虽然都能够对 SA 进行建模, 但在描述 SA 的动态性方面表现不足. 为描述 SA 的动态性, 文各引入 CHAM.

目前针对 SA 的测试技术的研究现状主要表现在: A. Bertolino 等人^[2-3]提出利用软件系统的有限状态过程 FSP 来对软件体系结构进行建模, 同时引入 LTSA 工具和 FC2tools 工具, 以实现软件体系结构测试的自动化. N. Medvidovic 和 D. S. Rosenblum 等人^[4]提出把 UML 应用到软件体系结构的一致性

测试中, 用标准的 UML 表示法来模拟软件体系结构的行为, 通过 UML 的状态图来指定系统的行为信息, 为软件体系结构建模. M. Wermelinger^[5]指出化学抽象机是一种对动态软件体系结构的分析和测试非常有用的形式化描述技术. Henry Muccini^[6]提出了如何将体系结构级的测试转化为代码级的测试.

文中提出一种基于 CHAM 的新技术: 利用化学抽象机 CHAM 能够描述 SA 的动态规格说明, 且这一动态性能够通过 LTS 表现出来这一特点, 可以根据 CHAM 的 SA 描述来建立 LTS 模型, 再根据测试需求对 LTS 图进行测试准则的选取, 然后根据测试覆盖准则生成 LTS 中的测试路径, 从而完成对整个 SA 的测试工作.

收稿日期: 2009-03-12; 修回日期: 2009-06-18

基金项目: 黑龙江省教育厅科技项目(11531237)

2 基本概念

SA通过建模构件和它们之间的交互来表示系统的结构,基于SA的描述通过结构的相关抽象提供了一个完整的系统模型,SA描述的一个重要特征是能够有效地表示系统的动态性,FSMs(Finite State Machines)、Petri网、LTS都能够有效地建模SA的所有动态行为.在此,我们选用CHAM来描述SA,然后生成基于此规格说明的LTS,以表示整个SA系统的动态行为.

化学抽象机CHAM这一形式化系统是由Berry和Boudol共同开发出来的^[7].CHAM主要使用了化学隐喻,即化学抽象机由一组分子 $M_1, M_2, \dots$ 、溶液 $S_1, S_2, \dots$ 和迁移规则 $T_1, T_2, \dots$ 组成.分子构成了CHAM的基本构件;溶液是有限多个分子的集合,它解释成一个CHAM状态;迁移规则指出了在CHAM中溶液状态改变的依据.这里用 T 来标识迁移规则集合和相应的标号集合.

定义1 化学抽象机描述的SA规格说明是一个5元组 (M, S, T, S_0, S_f) ,其中 M 是体系结构组成构件的语法描述,构件被分为3类:数据元素、处理元素、连接元素; S 是溶液的结合,即系统中构件的集合; $T \subseteq S \times S$ 是一组描述构件间交互活动的迁移规则的集合,其中迁移关系 $S_i \rightarrow S_j$ 提供了溶液变化的依据; $S_0 \in S$ 表示系统初始状态的溶液; $S_f \subseteq S$ 表示系统终止状态的溶液集.

通过CHAM描述的SA规格说明,就可以导出基于SA的测试计划和测试数据,由于CHAM能够有效的描述SA的动态特性,并且其动态特性能够用LTS表示出来,因此生成LTS是SA测试的关键.

定义2 标号迁移系统LTS描述的SA是一个五元组,即 (S, L, S_0, S_f, T) , S 是SA中状态的集合, L 是标号的集合, $S_0 \in S$ 是系统的初始状态, $S_f \subseteq S$ 是系统的终止状态的集合.而 $T = \{\xrightarrow{l} \subseteq S \times S \mid l \in L\}$ 是标有元素 l 的构件间的迁移关系.

LTS中的每个状态对应于CHAM中的一种溶液,因此,可以用分子的集合来描述构件的状态,LTS图中弧上的标号标明了使系统从弧尾节点状态转换到弧头节点状态的转换规则.在得到LTS后,需要定义一条LTS中的路径,以指导测试.

定义3 在LTS中, $p = S_0 \xrightarrow{l_1} S_1 \xrightarrow{l_2} S_2 \xrightarrow{l_3} \dots \xrightarrow{l_n} S_n$ 就是一条路径,其中 $S_i (i = 0, \dots, n)$

表示状态, $l_i (i = 1, \dots, n)$ 表示标号, $\xrightarrow{l_1}, \xrightarrow{l_2}, \dots, \xrightarrow{l_n}$ 表示标有标号 l 的迁移关系.如果 S_0 是初始状态, S_n 是终止状态,则 p 就是一条完全路径.为了简洁,路径也可用标号的序列表示: $p = l_1, l_2, \dots, l_n$.

3 CHAM描述导出LTS

以下定义给出了CHAM向LTS的导出机制:

定义4 由迁移规则集合 T 推导出的操作语义:假设 T 为一个CHAM的迁移规则集合,那么 T 定义了一个关系 $D \subseteq M \times M$ (M 为分子),则关系 D 是能满足迁移规则的最小关系.

定义5 给定一个迁移规则集 T ,从溶液 S_0 到溶液 S_n 的 T 派生是一个序列 $\{S_i, 0 \leq i \leq n\}$, $n > 0$,所以,对于任何 $0 \leq i \leq n-1$, $S_i \rightarrow_R S_{i+1}$.如果从 S_i 到 S_j 存在一个 T 派生,则溶液 S_j 被称为 S_i 的一个 T 派生, S_0 的派生集可以表示为 $D_R(S_0)$.

根据定义2~5,可以得到CHAM到LTS的导出机制.文中对叶俊民等人在文献[8]中给出了CHAM模型到LTS的状态树生成算法进行了改进,用图来表示LTS,同时对LTS中的重复状态进行了合并.基于此,文中提出了改进的LTS状态图生成算法.根据文献[7],广义表(Lists)放松对表元素的原子限制,允许它们具有其自身结构(即表中的元素或原子或广义表).因此,规则和溶液的数据结构可以用广义表来描述.同时算法设有SET1、SET2数组和Match函数.SET1和SET2数组分量为广义表,分别用来存储生成的新溶液和存储每个生成的LTS状态,且该数组是上下文无关的.Match函数用以检查每一对分子 C_i 和 C_j 是否可以发生反应,如果可以发生反应,则返回将要使用的规则编号,否则返回0编号.如果生成的状态没有出现过,则将该状态存入SET2数组中,同时构造LTS图.如果生成的状态出现过,则构造LTS状态图.

下面给出LTS的生成算法,当溶液反应终止时,对应的LTS就已经构造成功:

输入:初始溶液 S_0 ,终止溶液集 S_f ,迁移规则 $T_k (k = 0, 1, \dots)$.

输出:LTS.

Step1: initialize SET1, initialize SET2,使之为空;并将 S_0 存入SET1中,终止溶液集 S_f 中的每种溶液分别存入SET2中,构造LTS,生成初始溶液 S_0 和终止溶液集 S_f ;

Step2: while (数组SET1不为空)

{从 SET1 中取出一个溶液 S_i , 并将 S_i 从 SET1 中删除;

for(S_i 中的每一个分子 C_i)

{If (C_i 属于自反应型分子(自反应型分子就是规则中只有一个分子, 且在条件满足时可以自己反应, 即该规则自己用其右部重写其左部))

{根据迁移规则 T_m , 对 C_i 进行重写操作, 同时生成溶液 S_p ;

If (S_p 不包含在溶液集 S_f 中) {将 S_p 存入 SET1 中;}

If (SET2 中存在溶液 S_p 并且在 LTS 图中从 S_i 到溶液 S_p 不存在迁移)

{则构造 LTS, 在 LTS 中生成一个从 S_i 到 S_p 的迁移 T_m ;

Else if (SET2 中不存在溶液 S_p)

{将 S_p 存入 SET2 中, 构造 LTS, 在 LTS 中生成一个新溶液, 同时生成从 S_i 到新生成溶液的一个迁移 T_m ;

For(S_i 中的每一对分子 C_i 和 C_j)

{If (Match(C_i, C_j, T_k)) 的匹配计算结果中的 T_k 不为空)

{则将 C_i, C_j 根据迁移规则 T_k 进行重写操作, 生成溶液 S_q ;

If (S_q 不包含在溶液集 S_f 中) {将 S_q 存入 SET1 中;}

If (SET2 中存在溶液 S_q 并且在 LTS 图中从 S_i 到溶液 S_q 不存在迁移)

{则构造 LTS, 在 LTS 中生成一个从 S_i 到 S_q 的迁移 T_k ;

Else if (SET2 中不存在溶液 S_q)

{将 S_q 存入 SET2 中, 构造 LTS, 在 LTS 中生成一个新溶液, 同时构造从 S_i 到新生成溶液的一个迁移 T_k ;

Step3: 算法结束.

根据以上的 LTS 生成算法, 可以得到最终 LTS 图.

4 基于 LTS 的测试路径生成

4.1 基于 LTS 的测试覆盖准则

得到 LTS 后, 就要考虑在此 LTS 上应用什么覆盖准则来生成测试路径. 由于生成的 LTS 属于状态图, 图中只有迁移和溶液的状态, 且根据 LTS 的生成过程, 可以定义基于 LTS 的测试覆盖准则, 且以下覆盖准则的能力依次减弱:

(1) 全路径覆盖: 覆盖 LTS 中所有路径至少一次.

(2) 基本路径覆盖: 覆盖 LTS 中所有的基本路径至少一次.

(3) 迁移覆盖: 覆盖 LTS 的所有迁移至少一次.

(4) 状态覆盖: 覆盖 LTS 的所有状态至少一次.

4.2 基于 LTS 的测试路径生成

由于基本路径覆盖、迁移覆盖、状态覆盖都不能做到对 LTS 进行完全测试, 生成的测试路径集都不完整, 因此, 要想对 LTS 进行完全测试, 生成 LTS 中的所有测试路径, 就需要采用基于 LTS 的全路径覆盖准则, 来生成 LTS 中的所有测试路径. 基于此, 文中提出了基于 LTS 的全路径生成算法的基本思想.

首先定义两个数组 PATH1 和 PATH2, 这两个数组中的每个元素用来存储每条路径, 最后得到的数组 PATH2 中包含了 LTS 中的所有路径.

输入: LTS.

输出: LTS 中的所有路径集.

Step1: 初始化数组 PATH1, 使之为空, 初始化节点 S_0 使之成为初始节点, 初始化节点 S_f 使之成为终止节点.

Step2: 从初始节点 S_0 向下搜索, 找到所有与 S_0 直接相连的边($(S_0 \rightarrow S_i), (S_0 \rightarrow S_j) \dots$), 同时修改数组 PATH1, 将每条路径分别存入数组 PATH1 中(即将路径分别存入数组中的每个元素). 当数组 PATH1 不为空时, 作:

Step2.1: 对数组 PATH1 中的每条路径 $C_i (i = 0, 1, \dots)$ 进行检查, 如果路径 C_i 的最后一个节点为 S_f 则将 C_i 存入数组 PATH2 中, 同时将其从数组 PATH1 中删除. 否则作:

Step2.2: 沿路径 C_i 中的最后一个节点 S_k 继续向下搜索, 找出与 S_k 节点直接相连的路径($(S_k \rightarrow S_p), (S_k \rightarrow S_q) \dots$), 并将其与路径 C_i 进行连接, 即将与 S_k 节点直接相连的路径连接到 C_i 的后面($(S_0 \dots \rightarrow S_k \rightarrow S_p)(S_0 \dots \rightarrow S_k \rightarrow S_q) \dots$), 并分别存入数组 PATH1 中, 同时将 C_i 从 PATH1 中删除.

Step3: 算法结束.

生成的数组 PATH2 中的所有路径就是 LTS 的所有测试路径. 根据 PATH2 中的测试路径, 设计测试数据的输入和预期输出.

5 实例

5.1 浏览器/服务器(B/S)实例

B/S体系结构风格具体结构为:浏览器/Web服务器/数据库服务器.图1给出了基于构件和连接件的静态B/S的SA描述,方框代表构件,箭头代表连接件,箭头上的标签表示数据元素通过连接件进行交互.

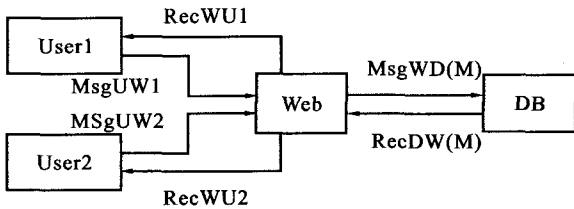


图1 静态B/S的SA描述

5.2 CHAM描述B/S结构

CHAM描述的B/S结构的规格说明包括分子的语法描述、迁移规则的集合和初始溶液 S_0 , 其中分子的语法描述包括:

$$M := P | C | D | M.M$$

$$P := \text{User} | \text{Web} | \text{DB}$$

$$D := \text{MsgUW} | \text{MsgWD} | \text{RecDW} | \text{RecWU}$$

$$C := i(D) | o(D)$$

在此, M 表示 B/S 结构的构件集合, P 为处理元素的集合, 包含 User、Web 和 DB 三个构件. C 为连接元素的集合, 指出系统构件之间可能的数据通信, 包含 i 和 o 两个操作, 表示数据元素的交互端口, 且 i 为输入端口, o 为输出端口. D 为数据元素的集合, 表示在不同系统构件之间进行的信息交换, 包含 MsgUW、MsgWD、RecDW、RecWU 四个分子, 分别表示用户向 Web 服务器发送的信息, Web 服务器向 DB 服务器发送的信息, Web 服务器接收到 DB 服务器的反馈信息, 用户接收到 Web 的反馈信息. 而符号“.”表示了处理元素输入输出行为的状态.

在用 CHAM 定义系统体系结构时, 必须定义 CHAM 的初始溶液, 即明确系统的最初状态. 对于 B/S 结构, 其初始溶液 S_0 为 $\text{User}, i(\text{MsgUW}). o(\text{MsgWD}). i(\text{RecDW}). o(\text{RecWU}). \text{Web}, i(\text{MsgWD}). o(\text{RecDW}). \text{DB}$.

以 CHAM 规格说明为基础, 并根据 B/S 结构的工作原理, 给出了基于 B/S 结构的迁移规则. 由于终止溶液集 S_f 中的终止溶液不唯一, 且存在循环, 作如下规定: User1 和 User2 每次只发送一次数据请求, 当 User1 和 User2 都发送数据请求时, 则应用

反应规则 T7 和 T8 后, 反应终止. B/S 结构的迁移规则:

$$T_0: \text{User} = o(\text{MsgUW1}). i(\text{RecWU1}). \text{User1}, o(\text{MsgUW2}). i(\text{RecWU2}). \text{User2}$$

$$T_1: o(\text{MsgUW1}). i(\text{RecWU1}). \text{User1}, i(\text{MsgUW}). o(\text{MsgWD}). i(\text{RecDW}). o(\text{RecWU}). \text{Web} = i(\text{MsgUW}). o(\text{MsgWD}). i(\text{RecDW}). o(\text{RecWU}). \text{Web}, o(\text{MsgWD1}). i(\text{RecDW1}). o(\text{RecWU1}). \text{Web}, i(\text{RecWU1}). \text{User1}, o(\text{MsgUW1})$$

$$T_2: o(\text{MsgUW2}). i(\text{RecWU2}). \text{User2}, i(\text{MsgUW}). o(\text{MsgWD}). i(\text{RecDW}). o(\text{RecWU}). \text{Web} = i(\text{MsgUW}). o(\text{MsgWD}). i(\text{RecDW}). o(\text{RecWU}). \text{Web}, o(\text{MsgWD2}). i(\text{RecDW2}). o(\text{RecWU2}). \text{Web}, i(\text{RecWU2}). \text{User2}, o(\text{MsgUW2})$$

$$T_3: o(\text{MsgWD1}). i(\text{RecDW1}). o(\text{RecWU1}). \text{Web}, i(\text{MsgWD}). o(\text{RecDW}). \text{DB} = i(\text{RecDW1}). o(\text{RecWU1}). \text{Web}, i(\text{MsgWD}). o(\text{RecDW}). \text{DB}, o(\text{RecDW1}). \text{DB}$$

$$T_4: o(\text{MsgWD2}). i(\text{RecDW2}). o(\text{RecWU2}). \text{Web}, i(\text{MsgWD}). o(\text{RecDW}). \text{DB} = i(\text{RecDW2}). o(\text{RecWU2}). \text{Web}, i(\text{MsgWD}). o(\text{RecDW}). \text{DB}, o(\text{RecDW2}). \text{DB}$$

$$T_5: o(\text{RecDW1}). \text{DB}, i(\text{RecDW1}). o(\text{RecWU1}). \text{Web} = o(\text{RecWU1}). \text{Web}$$

$$T_6: o(\text{RecDW2}). \text{DB}, i(\text{RecDW2}). o(\text{RecWU2}). \text{Web} = o(\text{RecWU2}). \text{Web}$$

$$T_7: o(\text{RecWU1}). \text{Web}, i(\text{RecWU1}). \text{User1}, o(\text{MsgUW1}) = \text{User1}, o(\text{MsgUW1}). i(\text{RecWU1})$$

$$T_8: o(\text{RecWU2}). \text{Web}, i(\text{RecWU2}). \text{User2}, o(\text{MsgUW2}) = \text{User2}, o(\text{MsgUW2}). i(\text{RecWU2})$$

5.3 B/S 结构的溶液集及其测试路径

根据 B/S 结构的 CHAM 描述和 CHAM 到 LTS 的导出算法, 可以得到 B/S 结构的 LTS 图, 通过实验, 一共生成了 25 个溶液状态, 如图 2 所示. 然后采用基于 LTS 的全路径覆盖准则及相应的全路径生成算法, 一共生成了 70 条 LTS 中的测试路径, 如图 3 所示.

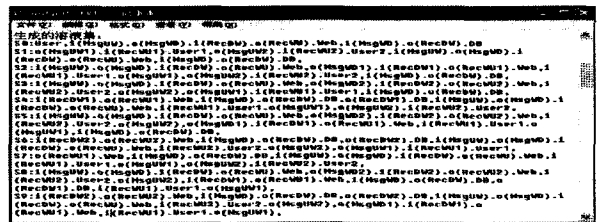


图2 B/S结构的溶液集

6 结束语

文中提出一种基于 CHAM 的 SA 级全路径覆

盖方法.该方法通过 CHAM 形式化描述整个软件系统,用标号迁移系统 LTS 来对系统的动态行为进行建模,建立了软件体系结构的测试模型,并利用此模型进行软件体系结构级测试的选取,并最终生成了 LTS 中的所有测试路径.并以 B/S 体系结构为例,通过实验验证了该方法在生成 B/S 这一结构的所有测试路径上是可行的.下一步的工作将主要集中在如何根据生成的所有测试路径进行测试数据的自动生成上.同时,SA 级测试向代码级测试的转换也是下一步研究的重点.

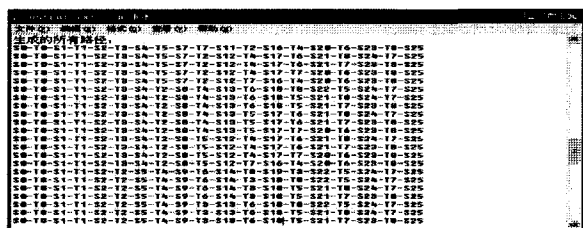


图 3 B/S 结构的测试路径

参考文献:

- [1] 赵磊,伦立军,徐士华.基于软件体系结构的测试路径生成方法[J].微电子学与计算机,2008,25(1):177-180.
- [2] Bertolino A, Inverardi P, Muccini H. Formal methods in testing software architectures[M]. SA Lectures, SFM-03: LNCS 2804, Springer, 2003:124-149.
- [3] Bertolino A, Inverardi P, Muccini H. An explorative journey from architectural tests definition down to code test execution[C]// Proceedings of the 23rd International Conference on Software Engineering. Canada: IEEE Computer Society, 2001:211-220.
- [4] Medvidovic N, Rosenblum D S, Redmiles D F. Modeling software architectures in the unified modeling language[J]. ACM Transactions on Software Engineering and Methodology, 2002,11(1):13-15.
- [5] Wermelinger M. Towards a chemical model for software architecture reconfiguration[J]. IEEE Proceeding Software, 1998(5):130-136.
- [6] Muccini H, Bertolino A, Inverardi P. Using software architecture for code testing[J]. IEEE Trans. Softw. Eng, 2004,30(3):160-171.
- [7] Berry G, Boudol G. The chemical abstract machine[J]. Theoretical Computer Science, 1992(96):217-248.
- [8] 叶俊民,王振宇,曹瀚,等.基于 CHAM 模型的 LTS 状态树生成算法[J].哈尔滨工程大学学报,2003,24(3):287-291.

作者简介:

赵磊男,(1979-),硕士,工程师.研究方向为软件测试、软件体系结构.
徐士华女,(1978-),硕士,讲师.研究方向为软件测试、构件.
赵国生男,(1977-),博士,工程师.

(上接第 139 页)

性较好.该系统可进一步推广到消防、码头、车站等应用场合,具有较高的实用价值.不足之处是为了保证系统野外工作的可靠性,系统采用了低波特率来传输数据,导致传输速率低,因此不适合大数据量的传输.

参考文献:

- [1] 中华铁路网.铁路平面无线调车系统[EB/OL]. (2007-12-06)[2009-04-01]. <http://www.tieliu.com.cn/tlbk2/print.asp?articleid=96787>.
- [2] Motorola. Professional radio option interface specification [M]. New York, Ireland: Motorola, Inc, 2004.
- [3] Motorola. Professional radio option interface specification electrical[M]. New York, Ireland: Motorola, Inc, 2003.
- [4] 陈永浦.现代通信系统和信息网[M].北京:电子工业出版社,1996.

版社,1996.

- [5] 樊昌信,詹道庸.通信原理[M].北京:国防工业出版社,1995.
- [6] 王新梅,肖国镇.纠错码—原理与方法[M].西安:西安电子科技大学出版社,1991.
- [7] 王薇,刘文予,金欣,等.无线视频通讯中的自适应差错非均匀保护方案[J].微电子学与计算机,2004,21(7):53-57.
- [8] 肖洁,王志功,胡庆生,等.10Gb/s Reed-Solomon(255,239)解码器的设计[J].微电子学与计算机,2005,22(8):39-43.

作者简介:

陈志贤男,(1979-),博士后.研究方向为网络安全、数据挖掘、可信计算及计算机在通信中的应用等.