基于UML状态图的C/S模式软件系统的一致性测试例生成

叶新铭 王谱新 白翔宇 谢 辉
(内蒙古大学计算机学院 呼和浩特 010021)

摘 要 C/S模式的软件系统具有多层次结构、采用面向对象编程技术等特点,为其生成一致性测试变得十分困难。使用UML状态图,为一个C/S模式的软件系统建模,将建立好的模型转换为扩展的有限状态机,并在该有限状态机上使用UIO序列与中国邮递员算法相结合的方法进行测试例的生成,最后应用数据流分析技术对生成的测试例进行分析,排除了其中不可执行的测试例。该方法利用了UML状态图易于建模的优点,降低了测试生成的难度,缩短了测试例的长度并节省了软件开发的成本。

关键词 软件测试,C/S模式,UML状态图,有限状态机,UIO,中国邮递员算法
中图法分类号 TP306.2 **文献标识码** A

Method of Conformance Test Case Generation for C/S Model System Based on UML State Chart

YE Xin-min WANG Pu-xin BAI Xiang-yu XIE Hui
(Computer College of Inner Mongolia University, Hohhot 010021, China)

Abstract C/S model based system has characters of multi-layers architecture and Object oriented based programming technology etc. It is difficult to carry out conformance testing for this kind of system. This paper presented a model for a topic C/S system based on UML state chart, and then transformed it into EFSM. A method combining UIO and Chinese post man algorithm was applied on the EFSM to generate conformance test cases. Finally, we adopted data flow based analyzing technology to exclude the test cases that are not executable. The UML state chart is good at modeling software system, so we utilized such advantage to reduce the difficulty of test case generation, and decrease the length of test cases and cost of software development.

Keywords Software test, C/S model, UML statechart, FSM, UIO, Chinese postman algorithm

随着面向对象和网络技术的快速发展,软件系统已不仅仅局限于在单机上运行,C/S(客户机/服务器,Client/Server)模式的软件系统应运而生。基于C/S模式的系统具有如下特征:多层次体系结构、图形用户界面、面向对象程序设计技术、数据分布存储、并发控制以及平台异构等[1]。采用传统的有限状态机对其动态行为建模时,常常会出现并发描述困难、状态爆炸等问题,因此使得建模并生成一致性测试例变得非常困难。UML状态图是一种表达能力丰富、功能强大的建模语言,适用于许多领域并可以应用在软件开发的各个阶段。使用UML状态图来描述软件的需求分析、总体设计和详细设计,能够方便行业专家、用户、软件系统分析和设计人员之间的沟通与合作,减少软件开发中出现的错误。但UML状态图具有层次化、并发和广播消息等特征,所以在其基础上生成测试例也存在极大的困难。目前基于有限状态机的测试例生成技术已十分成熟,本文使用将UML状态图通过扁平化处理转换为扩展有限状态机的策略。在建模时,利用了UML状态图的优点,同时在测试例生成阶段又能以有效的手段来处理。在转换后的扩展有限状态机的基础上,本文使用控制流与数据流相结合的方法获得测试例,避免了仅基于控制流技术生成测试例时所产生的部分测试例不可执行的情形,同时解决了仅基于数据流技术生成的测试例覆盖度不高的问题。最后,在科技部科技支撑计划重点项目数字音像内容集成分发平台的数字音像集成管理系统中应用此方法,进行了一致性测试。

1 UML状态图到扩展的有限状态机的转换技术

UML状态图(state chart diagram)展现了一个状态机[2],它由状态、转换、事件和活动组成,描述了对象、系统、子系统之间的状态变迁关系。通过状态图,可以了解到对象的状态关系以及对象受到事件的激励,经过变迁关系,到达另外的状态。它描述的是系统的动态功能,属于动态模型。图1显示了数字音像集成管理系统中节目查询模块的UML状态图。

UML状态图与扩展的有限状态机的主要区别是层次化、并发和广播消息,因此将UML状态图转换为扩展的有限状态机可以看成是UML状态图扁平化的过程,也就是将

到稿日期:2008-08-05 返修日期:2008-10-24 本文受科技部科技支撑计划重点项目(No. 2006BAH02A11)和国家自然科学基金项目(60563004)资助。

叶新铭(1943-),男,教授,博士生导师,主要研究方向为计算机网络,E-mail:xmy@imu.edu.cn;王谱新(1979-),男,硕士生,主要研究方向为计算机网络;白翔宇(1976-),男,讲师,主要研究方向为无线网络与移动通信;谢 辉(1980-),男,讲师,主要研究方向为网络管理等。

UML 状态图中层次化的状态和变迁关系转换为全局的状态和变迁关系。由 UML 状态图操作语义[3]知：UML 状态图中的格局是当前所有活动状态的集合，因此可以将一个格局看作是转换后状态机的一个全局状态。而 UML 状态图中满足能够将一个格局变迁到另一个稳定格局的变迁集合就为这两个状态中的一条全局变迁。转换的算法如下。

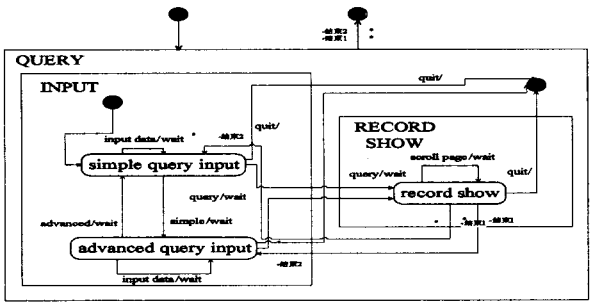


图 1　节目查询 UML 状态图

构建全局状态：

(1) 搜索 UML 状态图中所有状态找到整个状态图的根状态。

(2) 从根状态递归搜索每个状态的子状态。若当前状态为 OR 复合状态，则将它的一个子状态复制到当前格局中，并以该子状态为输入递归搜索。若当前状态为 AND 复合状态，则将其全部子状态复制到当前格局中，并轮流搜索该状态的子状态。若当前状态为简单状态，则返回。

构建全局变迁：

(1) 任取两个全局状态：$S1$ 和 $S2$。

(2) 搜索 UML 状态图中的每个状态，求解状态集 S，其中 S 的元素为 $S1$ 和 $S2$ 的交集。求得状态集 $S1'$和 $S2'$，其中 $S1'=S1-S$；$S2'=S2-S$。

(3) 搜索 UML 状态图中所有变迁 t，求解离开状态为 $S1'$并且到达状态为 $S2'$的变迁，构成变迁集合 $T1$。求解源状态为 S 且目的状态也为 S 的变迁，构成变迁集合 $T2$。求解属于变迁集合 $T1$、不属于变迁集合 $T2$ 的变迁，构成变迁集合 $T3$，即所求 $S1$ 到 $S2$ 的全局转换路径。

通过应用上述算法到图 1 所示的 UML 状态图中，我们可以得到如图 2 所示的有限状态机。

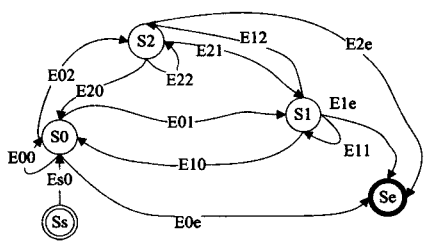


图 2　节目查询的 EFSM

其中，$S0$ 为简单查询输入状态，$S1$ 为高级查询输入状态，$S2$ 为查询结果显示状态，Ss 为起始状态，Se 为结束状态。各边的输入输出如表 1 所列。

表 1　节目查询的变迁矩阵

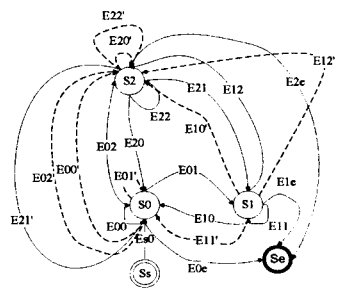
	Ss	S0	S1	S2	Se
Ss	—	Es0 ept/ept	—	—	—
S0	—	E00 sdi/sdo	E01 aqi/its	E02 sq/gsh	E0e c/q
S1	—	E10 sqi/ith	E11 adi/ado	E12 aq/gsh	E1e c/q
S2	—	E20 sdi/sdo	E21 adi/ado	E22 ps/gsh	E2e c/q
Se	—	—	—	—	—

2　一致性测试例的生成

将 UML 状态图转换为扩展的有限状态机后，就可以应用成熟的基于有限状态机的测试例生成方法来生成一致性测试例[4]。目前常见的基于有限状态机的测试例生成方法有：T 方法、D 方法、W 方法、Wp 方法、UIO 方法。其中 T 方法的随机性太强，容易生成大量的冗余用例；D 序列能发现故障点但不能保证总是存在；W 以及 Wp 方法能发现比较复杂的错误，但它们的测试序列长度在实际的复杂系统中很难使用；而 UIO 序列的长度相对较短，生成算法也简单。只要 FSM 不具有等价状态，几乎都存在 UIO 序列，因而在实际中得到普遍使用。

传统的 UIO 方法通过遍历有限状态机，以求得满足覆盖准则的测试序列。然而，由于软件系统说明的有限状态机往往不是对称的，因此遍历过程中经常需要重复地遍历某些路径。中国邮递员算法[5]是求解有向或无向图中覆盖所有变迁至少一次的方法。本文在遍历过程中使用将 UIO 序列与中国邮递员算法结合的方法，这样可以在保证相同覆盖准则的条件生成测试子序列的优化组合。当我们得到了一个有限状态机的 UIO 序列后，要通过中国邮递员算法得到一个开销尽可能小的测试序列，所要做的工作是寻找一个如下形式的最小成本(其中每边的成本设为 1)输入序列(其中成本表示测试序列的长度)：(ri)\(TEST(Vi1,Vj1;L1)) \…(TEST(Vi|E|,Vj|E|;L|E|))。其中 ri 是 Reset 变迁，(TEST(Vi|E|,Vj|E|;L|E|)是机 M 中任意变迁加上该变迁目的状态的 UIO 变迁的路径。换句话说，最小成本序列对每个(Vi,Vj;Lk)∈E 都包含一个 TEST(Vi,Vj;Lk)子序列，且没有任意两个子序列彼此重复。此时将原有限状态机中的每一个变迁 Ei∈E 与该变迁尾状态的 UIO 序列相连，构成新的变迁 Eci，并将此新的变迁添加到原有限状态机中，使得构成的新有限状态机 G′的状态集为原状态集 G 的状态集 V，新有限状态机 G′的边集 E′为原有限状态机边集与新增加边集的并，即 G′=(V′,E′)，且 V′≡V，E′≡E∪Ec，其中 Ec={(vi,vk;Ll·UIOj):(vi,vj;Ll)(E,TAIL(UIOj)=vk}。

图 3 就是图 1 的扩展图 G′，其中各状态的 UIO 序列为，S0:E01 aqi/its、E02 sq/gsh，S1:E10 sqi/ith、E12 aq/gsh，S2:E22 ps/gsh。这样将 UIO 序列添加到每边的尾状态上构成新的边集 Ec={E00′,E01′,E02′,E11′,E10′,E12′,E22′,E20′,E21′}。这样求解最小成本输入序列的问题就转化为在新的有限状态机 G′的边导出子图 Gc 中求解一条欧拉游历的问题。如果 Gc 是一个有向欧拉图，则其必然存在一条欧拉游历(对于有向图是否为一个欧拉图的判定条件是每个图中



S0:E01 aqi/its、E02 sq/gsh，S1:E10 sqi/ith、E12 aq/gsh，S2:E22 ps/gsh

图 3　节目查询模块的基于 UIO 序列的增广图

的顶点的出度和入度相等)。如果 Gc 不是有向欧拉图,则需要添加 G 中的边,使 Gc 成为有向欧拉图。可以看出,这个问题模型就是在一个强连通的有向图上添加一些边,使其变为欧拉图,同时要求添加的边的成本 C=(本文中以边的长度为成本开销)最小。此类问题的解法可以使用最小费用最大流问题的模型进行求解[5]。

整个测试例生成算法描述如下:

(1) 在图 G 的基础上把所有测试子序列的头尾两个端点(状态)相连,构成有向边集 Ec。将 Ec 添加到原来的有向图 G,构成一个新图 G′。寻找最短测试序列的问题等价于一个赋权有向图(在本文中权值等于路径长度)上的中国乡村邮递员问题,即寻找赋权有向图的最优有向邮路。为 G′的每条边附加一个数值,表示在最终的测试序列中边的重复次数,E 中的边初始化为零次,Ec 中的边初始化为一次,转(2)。

(2) 如果 G′是有向欧拉图(即每个状态的出度等于入度),则 G′的有向欧拉回路就是最优有向回路,即最优测试序列(求解有向欧拉图中的欧拉游历的算法在下面将有描述),终止。否则转(3)。

(3) 添加重复的弧,使 G′成为有向欧拉图。此时寻找最短测试序列的问题转化为寻找使添加弧的权和最小的方法。这个问题可转化为最小成本最大流的问题。计算出需要添加的弧及重复次数后将其添加到 G′,转(2)。

通过应用最小费用最大流算法我们得出需要在 Gc 中复制边(E21)3 次即可以使其成为一个欧拉图。将图 Gc 变换为一个有向欧拉图后,根据有向欧拉图的性质我们可以用以下算法求得此图的欧拉游历路径:

(1) 在图 G 中任意找一个回路 C;

(2) 将图 G 中属于回路 C 的边删除;

(3) 在残留图的各极大连通子图中分别寻找欧拉回路;

(4) 将各极大连通子图的欧拉回路合并到 C 中,得到图 G 的欧拉回路。

至此我们便得到了一个该 C/S 系统查询模块的测试序列:

E01\E00\E20\E22\E21\E10\E21\E21\E12\E21\E11\E02

其实际的执行路径为

E01\E10\E00\E02\E20\E02\E22\E22\E21\E10\E02\E21\E12\E21\E12\E22\E21\E11\E10\E02\E22。

得到测试路径后,我们将在其基础上选择测试输入数据,以便于后期的测试。在实际的测试过程中,每一个输入控件都有可能有很多的输入值,但并不是对所有的值都要进行测试。为了避免出现无穷的输入数据组合,实现选用少量、高效的测试数据进行尽可能完备的测试的目标,我们对测试数据的选择使用等价类划分和边界值分析的方法对测试路径进行扩充。以上的测试例生成过程是从控制流角度出发,通过状态覆盖和变迁覆盖的准则生成测试例。仅考虑控制流的测试序列生成方法所生成的测试路径,往往会包含部分不可执行的错误路径,如某语句使用了一个变量,而在此之前并未对其定值,或者某语句某变量进行赋值,但却从未使用过该变量,以及谓词变量值的判断错误等。因此对仅考虑控制流技术生成的测试序列应用数据流分析技术[6]便可以排除序列中不可执行的测试例,从而进一步缩短测试序列的长度。在选择数据流标准时需要进行权衡,较强的标准能够找到较多的错误,而较弱的标准一般使用较少的测试例就能够满足。虽然 all-paths 标准的差错覆盖能力最强,但是因为 NNM 管理系统的形式化模型中包含环,选择 all-paths 标准会产生无数完全路径,这里我们选择了 all-du-paths 标准。对于前面生成的测试路径,可能会存在部分非完全路径,无法应用 all-du-paths 标准。对于此类路径,我们通过 Dijkstra 算法求出其尾状态到起始状态的最短路径,然后将测试路径与其拼接,以形成一条完全路径如此便可以应用 all-du-paths 标准。

3 测试过程

数字音像集成管理系统是基于 C/S 模式的、在内容运营商系统中实现对音像内容管理、查询和维护的平台。对于运营商来说,它是对分布式资源网络中所拥有的音像资源、视听节目、广告策略进行集中管理的重要单元。我们通过在该系统中应用前面讨论的测试例生成方法,共产生了 95 个测试例。通过在该系统中实施这些测试例,共发现 7 个功能不一致的 BUG。与手工测试相比,自动生成的测试例长度相对较短,发现的错误数与测试例长度之比要高于手工测试的情况。测试过程中存在部分自动生成测试例未发现的 BUG,主要是用户易用性方面的错误,如查询一次后上次的查询输入应予以保留等问题,此类问题仍需手工测试检测。

结束语 C/S 模式的软件系统已经得到了广泛的应用,但其具有的多层次体系结构,图形用户界面、面向对象程序设计技术等特征使得为其建模变得十分困难。UML 目前已成为事实上的软件建模标准,它表达能力丰富、功能强大,适用于 C/S 模式软件开发的各个阶段。虽然 UML 具有诸多优点,但基于 UML 模型的测试例生成技术还不是十分完善。有限状态机理论已发展多年,基于有限状态机的测试生成理论已很成熟,但应用有限状态机到软件建模过程中会存在诸如状态爆炸等问题,使得测试生成难以实现。本文提出一种测试例生成方法,该方法使用 UML 状态图对 C/S 模式软件系统的动态行为进行建模,再将 UML 状态图转换为扩展的有限状态机,之后在扩展的有限状态机的基础上使用结合 UIO 序列的中国邮递员算法生成一致性测试例,最后对生成的测试例进行数据流分析,排除其中不可执行的路径。通过将 UML 状态图转换为扩展的有限状态机,使得软件测试过程中的建模和测试生成利用到了 UML 状态图和扩展的有限状态机各自的优点,提高了软件开发初期的模型重用率。在测试生成阶段,通过使用 UIO 序列与中国邮递员算法相结合的方法对测试例进行全局的优化,缩短了测试例的长度。最后,通过使用数据流分析技术对生成的测试例进行分析,排除不可执行的路径,进一步缩短了测试例长度。

参考文献

[1] 李小将,樊天晴,胡正国. Client/Server 系统的测试策略[J]. 计算机工程,2002,28(8):38-58

[2] UML specification 1.5 [EB/OL]. http://www.omg.org/uml

[3] 李留英,王戟,齐治昌. UML Statechart 图的操作语义[J]. 软件学报,2001,12(12):1864-1873

[4] Bertolino A, Marre M. Automatic Generation of Path Covers Based on the Control Flow Analysis of Computer Programs [J]. IEEE Trans. Software Eng, 1994, 20(12): 855-899

[5] 胡运权. 运筹学教程(第 3 版)[M]. 北京:清华大学出版社,2007

[6] 李华,叶新铭. 基于 Petri 网的数据流与控制流相结合的协议测试[J]. 内蒙古大学学报,1998,29(5):702-709