基于 EFSM 模型的等价类测试*)

易国洪　卢炎生
(华中科技大学计算机科学与技术学院　武汉 430074)

摘　要　等价类测试是有效减少测试用例而又较小地影响测试效果的一种高效的测试方法,但是对于测试者来说,寻找一种等价类测试的划分方法十分重要,本文提出了一种基于 EFSM(Extended Finite State Machine)模型的等价类测试划分的方法,通过对 EFSM 模型数据依赖和控制依赖分析,给出了等价类测试划分的具体方法和算法,并给出了等价类划分方法的完备性和无冗余性证明。

关键词　EFSM,等价类,软件测试,数据依赖分析,控制依赖分析

Equivalence Testing Based on EFSM

YI Guo-Hong　LU Yan-Sheng
(Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract　This paper presents a method of equivalence testing based on EFSM, and offers an algorithmic way of partition of Equivalence. The paper describes and defines the data dependence and control dependence based on EFSM. Different types of dependencies are identified between elements of EFSM system model. Equivalence testing sequence identifies by dependencies analysis. The paper proves redundancy and Completeness about partition of Equivalence.

Keywords　EFSM, Equivalence testing, Software testing, Data dependence analysis, Control dependence

1　引言

使用等价类测试有两个动机[1]:其一是希望进行完备的测试;其二希望避免冗余,使每个测试都是有效的。等价类测试是有效减少测试用例而避免冗余测试用例的一种高效测试方法,但是对于测试者来说,寻找一种等价类测试的划分方法十分重要,等价类测试的关键问题是等价类测试用例构成集合的划分,划分后的等价类测试用例是一组互不相交的子集。且这些子集是整个集合,这说明了测试的完备性和无冗余性,当然找到一种精确的等价类测试划分方法十分困难。本文提出了一种近似的等价类测试的划分的方法,能有效提高测试效率。

2　扩展的有限状态机模型

有限状态机模型是描述同步序列机很好的研究模型[2],扩展的有限状态机模型 EFSM(Extended Finite State Machine)[3~5]是在传统的有限状态机模型(FSM, Finite State Machine)基础上对其状态转换加以扩展,EFSM 与 FSM 一样是由两个部分组成:(1)状态集合(在状态图中用结点表示,含有起始状态和结束退出状态),(2)状态转换集合(在状态图中用有向边表示)。EFSM 与 FSM 不同的是对状态转换的定义不同,EFSM 中的状态转换包括 3 个方面的内容:(1)事件(或方法);(2)事件(或方法)触发的条件;(3)满足条件后执行的动作序列。图 1 给出了一个包含状态和状态转换的 EFSM 简单模型。

一个动作序列的规范就是一个描述,它说明了该动作序列所造成的状态改变必须满足的所有特定的相关特性。在 EFSM 模型中状态转换发生的必要条件是其进行状态转换的条件必须为真(true),否则就不可能发生状态转换。当满足条件后,就会发生状态转换,并且执行一个动作序列。

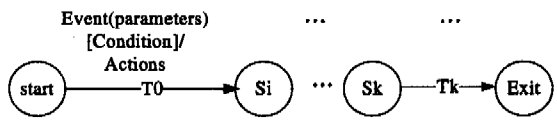

图 1　EFSM 模型

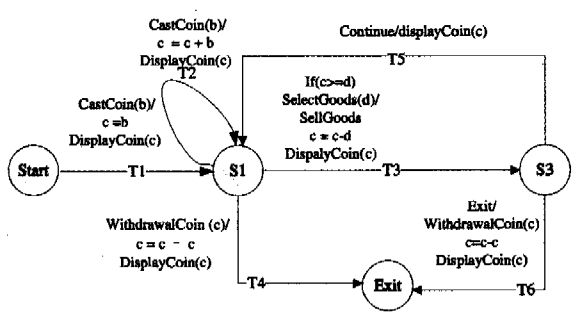

图 2　自动售货机系统的 EFSM 模型

以自动售货机为例来说明 EFSM 模型,根据自动售货机的处理过程,假定自动售货机可以处理如下几个事务:投币(Cast)、退币(Withdrawal)、找零(Change),售货(Sell)、显示金额(Display),而这些事务在 EFSM 中通过用状态转换来表示和实现,如图 2。状态转换 T1,T2 完成一次投币或多次投币可能的情况;状态转换 T4 表示退币的情况;状态转换 T3 表示用户选择商品自动售货机售货的情况;状态转换 T6 表示自动售货机将自动找出零币;状态转换 T5 表示用户欲继

*)本课题研究得到"十五"国防科技预研项目(编号:41315.9.2)资助。易国洪　博士生,讲师,研究方向为软件工程;卢炎生　教授,博士生导师,主要研究方向为软件工程、特种数据库。

续购买。

在EFSM模型中，因为各状态转换分别表示不同的操作处理事务，故可把测试的一个操作处理序列看成一个状态转换序列，如测试这样一个操作序列：投币、售货、找零后退出。对于图2来说，一个具体的测试数据如：Cast(10)，SelectGoods(5)，Change(5)。这样的一个测试数据可用状态转换序列表示出来，用状态转换序列表示为：T1，T3，T6。因为测试数据可以转换为相应的状态转换序列，所以可以通过用状态转换序列来表示相应的测试数据。那么就可根据EFSM来尽可能找到所有可能的状态转换序列，因为许多EFSM模型中都会出现环而使得其状态转换序列是不可数的(如图2)，所以其测试用例也是不可数的，要穷尽所有的测试是不可能的。故本文提出了一种等价类测试用例划分方法，它能有效的减少测试用例，且减少的冗余测试用例对整个测试起到的作用和效果影响是可以忽略的，从而达到穷尽所有测试数据而进行完备测试的相同效果。

各状态转换分别表示不同的操作处理事务，操作处理事务有先后次序关系，从而发生的状态转换也就有一定的先后次序，那么以有向边表示状态转换之间的先后次序，以椭圆结点表示状态转换，其中又以粗线椭圆结点表示起始状态转换，以双线椭圆结点表示结束状态转换，就可得到EFSM的状态转换次序图，如图3。

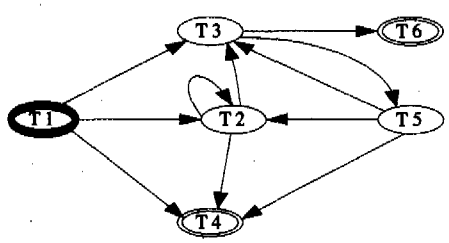

图3 自动售货机系统的EFSM状态转换次序图

3 EFSM模型的依赖分析

对EFSM模型进行依赖分析主要是分析其状态转换之间的数据依赖和控制依赖。

定义1 状态转换Ti的定义变量集 $D[Ti]=\{Var \mid Var \text{ is defined in } Ti\}$

定义2 状态转换Ti的使用变量集 $U[Ti]=\{Var \mid Var \text{ is used in } Ti\}$

定义3 状态转换Ti的始发状态 $Sb[Ti]=\{S \mid S \text{ is the State where Ti Started from}\}$

定义4 状态转换Ti至状态转换Tk的路径集 $R[Ti, Tk]=\{R_{ikj} \mid R_{ikj} \text{ is a route from Ti to Tk}\}$

定义5 状态转换Ti至状态转换Tk基于路径R_{ikj}的定义变量集 $D(Ti, Tk)=\{S \mid S \text{ is the State in the } R_{ikj}\}$

3.1 数据依赖分析

EFSM的数据依赖分析概念来源于传统的程序中的数据依赖分析[4]，数据依赖简单的说就是如果在状态转换Tk中使用了在状态转换Ti中定义的变量，且这一变量从状态转换Ti到状态转换Tk之间没有被其它的状态转换重新定义或加以改变，那么状态转换Ti与状态转换Tk则存在着数据依赖，这时我们称状态转换Tk数据依赖于状态转换Ti，或称为状态转换Ti数据决定状态转换Tk。

定义6 状态转换Tk数据依赖于状态转换Ti：

$Tk \xleftarrow{D} Ti$，iff $\exists \mu \in D[Ti], \mu \in U[Tk]$，且 $\mu \neq D(Ti, Tk)$

根据数据依赖的定义由图2可得到自动售货机系统的EFSM模型的数据依赖图如图4（其中椭圆结点表示状态转换）所示。

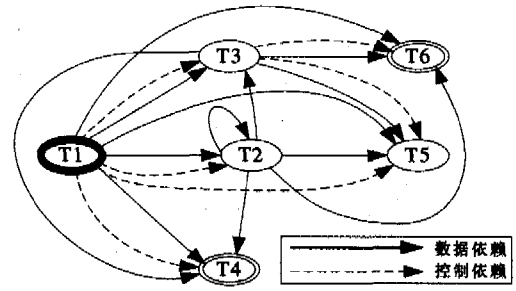

图4 自动售货机系统的EFSM数据依赖和控制依赖图

3.2 控制依赖分析

EFSM的控制依赖分析概念也是来源于传统的程序中的控制依赖分析[4]，只是对其重新进行了扩展定义，控制依赖是描述状态转换之间存在的控制依赖关系。

定义7 状态转换 $Sb[Tk]$后支配状态转换 $Sb[Ti]$：

$S_b[Tk] \propto S_b[Ti]$，iff $Rb_{ikj} \in R(Sb[Ti], Sb[Tk])$，$\forall Reb_i \in R(Sb[Ti], TE)$，$Sb[Tk] \in Reb_i$，其中TE是结束退出状态。

定义8 状态转换Tk控制依赖于状态转换Ti：

$Tk \xleftarrow{c} Ti$，iff $S_b[Tk] \propto Ti$，& $\neg S_b[Tk] \# S_b[Ti]$

由图2可得到自动售货机系统的EFSM模型的控制依赖图如图4。

4 等价类测试的划分

4.1 依赖子图

定义9 在EFSM中，所有的状态转换构成的集合，称为状态转换集，记为TV；由起始状态结点发出的所有状态转换，称为起始状态转换集，记为TS；到达结束状态结点的所有的状态转换，称为结束状态转换集，记为TE。

定义10 合法状态转换集 $VS=\{S \mid S \subset TV, \exists ts \in S, ts \in TS, \& \exists te \in S, te \in TE\}$。

定义11 最小状态转换集 $VSmin=\{S \mid \exists Rsei \in R[TS, TE], S \in Rsei\}$。

定义12 一个测试序列对应着EFSM的数据依赖和控制依赖图中的一部分，这一部分图称为依赖子图；最小状态转换集中状态转换结点之间的所有数据依赖和控制依赖形成的子图，称为最小状态转换集的最大依赖子图。

在EFSM的数据依赖和控制依赖图中删除不在最小状态集中所有结点及其数据依赖和控制依赖，即可得到最小状态转换集的最大依赖子图。

定义13 最小状态转换集是其所包含状态转换结点的最小的测试序列，根据这个测试序列对应的数据依赖和控制依赖子图，称为最小状态转换集的最小依赖子图。

如：3个测试序列Test1(T1，T3，T6)、Test2(T1，T2，T3，T6)、Test3(T1，T2，T2，T3，T6)分别可标识为不同的依赖子图，同时也可根据依赖子图确定包含这三个测试序列的三个测试序列集。每个测试序列集划分为一个等价测试用例集。依赖子图如图5、图6、图7所示。

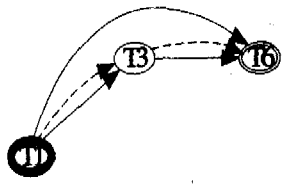

图 5　Test1 的依赖子图

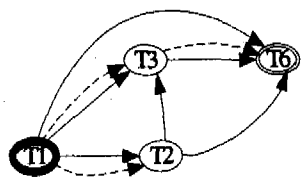

图 6　Test2 的依赖子图

图 5 为{T1,T3,T6}最小状态转换集对应的依赖子图，其最大依赖子图与最小依赖子图相同；图 6 为最小状态转换{T1,T2,T3,T6}的最小依赖子图；图 7 则为最小状态转换{T1,T2,T3,T6}的最大依赖子图。

对 EFSM 的数据依赖和控制依赖分析可以得到 EFSM 的依赖图，一个测试序列对应着唯一的数据依赖和控制依赖子图。

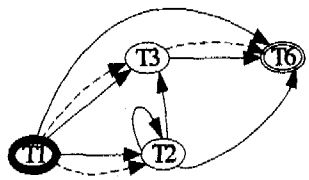

图 7　Test3 的依赖子图

4.2　等价类测试划分的标准

假设 1　如果两个测试序列得到相同的依赖子图，则认为这两个测试序列是等价的，其相应的测试用例也是等价的[4]；反之，根据相同的依赖子图得到的多个测试序列和其相应的测试用例则也是等价的。

假设 1 是本文提出等价类测试划分的基础和标准。因为可以根据一个测试序列确定一个依赖子图，又可根据依赖子图确定其对应的测试序列，所以我们可以通过找出所有的依赖子图对测试序列进行等价类的划分，如果一个依赖子图确定了则等价的测试序列也就确定了。那么对测试用例的确定就可转化为对依赖子图的确定。

等价类测试划分即在 EFSM 数据依赖和控制依赖图中寻找所有的最小状态转换集对应的所有依赖子图。等价类测试的划分步骤：1)由 EFSM 状态转换次序图生成最小状态转换集；2)根据 EFSM 的依赖分析图和最小状态转换集生成最大依赖子图和最小依赖子图；3)通过最大依赖子图和最小依赖子图求得所有的依赖子图；4)根据依赖子图确定等价类的测试用例。

4.3　等价类测试划分的相关算法

算法 4.1　由 EFSM 状态转换次序图生成最小状态转换集算法。首先将 EFSM 状态转换次序图按广度优先搜索算法生成森林。若 TS 集基数为 1 时，生成的是一棵树，其具体算法如下：

```
BFS(G,TS)
//实现对给定图的边进行广度优先查找遍历
//输入图 G=(V,E),TS
//输出森林,图 G 顶点可重复,图 G 的边不可重复也不可遗漏。按照
    被 BFS 遍历访问到的
/后次序用连续的整数标记,将 E 中的每条边标记为 0,表示还未访问
count←0;
for each vertex v in TS do
      let v be each tree's root node
      bfs(v)
endfor
bfs(v)
//访问所有以 v 起始结点未访问的边,然后按照全局变量 Count 的
    值,根据访问的先后次序加以标记
initialize a queue with v;
while the queue is not empty do
  for each edged e which first node is v
    if e is marked with 0
      count←count + 1;
      mark e with count and second node of edged e to the queue;
      let second node of edged e be child node of it's first node
    endif
  endfor
  remove vertex v from the front of the queue
endwhile
```

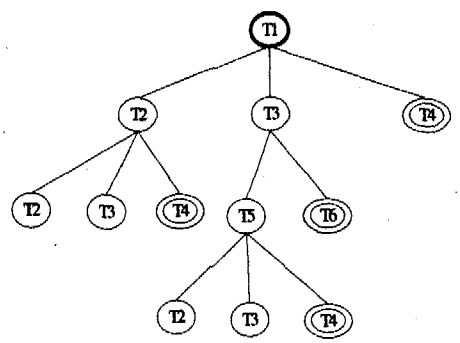

图 8　根据 EFSM 状态转换次序图生成的树

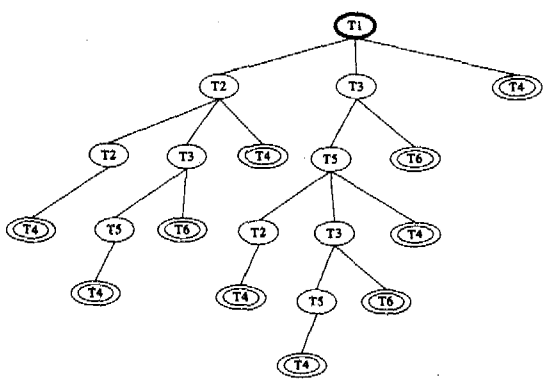

图 9　由图 8 生成的最小状态转换集树

算法 4.2　根据算法 BFS(G,TS)生成的森林生成最小状态转换集树。

```
MinSetTree(G1,TE)
//输入:G1={根据算法 BFS(G,TS)生成的森林(如图 8)},TE
//输出叶子节点全是结束状态转换的森林(如图 9),
for each tree do
  for each leaf-node lv do
    if lv not in TE
      search all routes which leaf-node in TE and contain lv in the
        forest;
      generate childtree by routes which root is lv
      and let childtree replace leaf-node lv
    endif
  endfor
endfor
```

定理　算法 4.1 和算法 4.2 求出的最小状态转换集集合，为状态转换次序图包含的所有最小转态转换集。

证明：反证法，假设存在一个最小状态转换集 S_i，不在算法 4.1 和算法 4.2 求出的最小状态转换集集合中，那么因为 S_i 是一个最小状态转换集，则 S_i 中各状态转换在状态转换次序图中必定存在一条路径(t_{i0}，…，t_{ik}，…，t_{in})且 $t_{i0}\in$TS，$t_{in}\in$TE，$t_{ik}\in$TV−TS−TE，那么根据算法 4.1 由于 t_{ik} 在状态

转换次序图，那么根据算法 4.1 生成的森林中必定包含 t_{ik} 结点，那么 t_{ik} 结点要么是叶子结点，要么是非叶子结点。如果 t_{ik} 结点是叶子结点，则由于 $(t_{ik},\cdots,t_{in})$ 存在一条路径，则根据算法 4.2 可知，将以 $(t_{ik},\cdots,t_{in})$ 分枝代替 t_{ik} 结点，则在算法 4.2生成的最小状态转换集树中存在一条分枝构成的路径 $(t_{i0},\cdots,t_{ik},\cdots,t_{in})$，即由 $(t_{i0},\cdots,t_{ik},\cdots,t_{in})$ 各状态转换结点构成最小状态转换集 S_i 可由算法 4.1 和算法 4.2 求出。也即最小状态转换集 S_i 在算法 4.1 和算法 4.2 求出的最小状态转换集集合中，这与假设矛盾。如果 t_{ik} 结点是非叶子结点，又由于存在一条从 $(t_{i0},\cdots,t_{ik})$ 的路径同时也存在一条 $(t_{ik},\cdots,t_{in})$ 路径。故由算法 4.1 生成的森林中必定包含 $(t_{i0},\cdots,t_{ik},\cdots,t_{in})$ 分枝，并且这条分枝的叶子结点 $t_{in}\in TE$。那么这条分枝上的所有不同的结点即可构成最小状态转换集 S_i，也即最小状态转换集 S_i 在算法 4.1 和算法 4.2 求出的最小状态转换集集合中，这也与假设矛盾，证毕。

算法 4.3 根据 EFSM 的依赖分析图和最小状态转换集生成最大依赖子图和最小依赖子图。

(1)根据 EFSM 的数据依赖图和控制依赖图和最小状态转换集，删除 EFSM 的数据依赖图和控制依赖图不在最小状态转换集中的状态转换结点的所有依赖与被依赖的关系，即可得到最小状态转换集的最大依赖子图。

(2)通过最小状态转换集确定最小的依赖子图，由于最小状态转换集通过变换状态转换的次序可以生成一个或多个最基本的测试序列(如(T1,T2,T3,T5,T4)和(T1,T3,T5,T2,T4)最小状态转换集相同，但是通过变换状态转换的次序生成了两个最基本的测试序列，其依赖子图也是不同的)，通过生成测试序列可以确定其状态转换集的最小数据依赖和控制依赖子图；首先根据最小状态转换集状态转换的次序标识出该测试序列的状态转换之间的依赖。接着将没有作标识的依赖(有向边)从最大依赖子图中删除。余下就是这一测试序列最小依赖子图。

算法 4.4 通过最大依赖子图和最小依赖子图求得所有的依赖子图。

根据最大的依赖子图，找出最小状态转换集的所有的依赖子图，不同的依赖子图即为独立的等价测试类。下面给出通过最小状态转换集的最大依赖子图和最小依赖子图确定所有依赖子图的相应的算法：

(1)如果最小依赖子图与最大依赖子图相同(即它们的有向边集相等)，则最小依赖子图与最大依赖子图相同，并对最小依赖子图加以标识。

(2)有标识的最小依赖子图中逐一添加有向边(该有向边属于最大依赖子图而不在最小依赖子图中)，生成依赖子图。将添加有向边的每一种组合作为一个依赖子图。

(3)重复(2)，直到对所有的最小状态转换集对应的没有标识的最小依赖子图处理完。

(4)通过验证所有的依赖子图对应的测试序列，验证这样的测试序列存不存在，将不存在测试序列对应的依赖子图加以标识，将所有的不存在测试序列的依赖子图作为一个单独的等价类。将存在的测试序列加以保留。

算法 4.5 根据依赖子图确定等价类的测试用例。

(1)将没有标识的的最小依赖子图和最大依赖子图和没有标识的依赖子图作为等价类的划分。

(2)保留的测试序列可以直接作为相应依赖子图等价类的测试用例，最小依赖子图和最大依赖子图对应的测试序列也分别作为不同等价类的测试用例，将所有的不存在测试序列的依赖子图作为一个单独的等价类。

定理 一个依赖子图当且仅当对应着一个等价类的测试序列集。

(1)充分性证明

证明：反证法，假定存在同一等价类测试序列对应着 n 个依赖子图($n>1$)，因为一个等价类测试序列必属于某一最小状态转换集，所以其状态转换是唯一确定的，由假定则说明这个等价类测试序列至少 n 次测试时会对相同的状态转换产生不同的数据依赖和控制依赖关系图，这与同一等价类测试序列在 EFSM 模型中产生相同的数据依赖和控制依赖图矛盾。

(2)必要性证明

证明：反证法，假定存在同一个依赖子图对应着多个等价类测试序列，因为不同的等价类测试序列其对应的最小状态转换集分两种情况，1、其最小状态转换集不同，2、其最小状态转换集相同，对于前者最小状态转换集不同，由上述算法可知因为其状态转换不同，故其依赖子图必不相同，显然与假定矛盾，对于后者其最小状态转换集相同，即具有相同的状态转换，根据最小状态转换集，由算法可确定其最大依赖子图和最小依赖子图及其所有的依赖子图，如果最大依赖子图与最小依赖子图相同，由算法知其该最小状态转换集的所有依赖子图都相同并把它们划分为一个等价类，这与假定矛盾，如果最大依赖子图与最小依赖子图不同，这也与假定矛盾。

(3)等价类测试划分的完备性和无冗余性证明

证明：由定理 1，可知一个测试序列集属于一个依赖子图，且这个测试序列集中的测试序列不再属于另一个依赖子图。故其等价类测试的划分是不相交，所以等价类测试的划分是无冗余的。因为对 EFSM 所有依赖子图都进行了划分，特别是那些依赖子图存在而实际的测试序列不存在，其也划分为一类作为不合法的测试序列。根据算法求出了 EFSM 的所有依赖子图，所有的测试序列都对应着某一依赖子图，故其划分是完备的。所以我们给出的等价类测试的划分是完备和无冗余的。

虽然根据依赖子图划分的等价类测试序列是无冗余的，但是在实际的测试中等价类测试还往往和边界条件相结合来决定一个等价类的具体测试用例及其用例的个数，以达到更好的测试效果。

结束语 本文提出了一种等价类测试的划分方法，并给出了确定等价类测试用例的具体算法，通过进行等价类测试划分，可以大大减少测试用例，具有较强的实践指导作用，能快速确定测试的最小集，降低测试成本，提高软件的可靠性等方面有着重大的意义。今后的主要工作将进一步完善等价类测试用例的自动生成和完成通过需求规格说明自动生成 EFSM。

参 考 文 献

1 Jorgensen P C. 韩柯等译. 软件测试. 北京：机械工业出版社，2003
2 Moore E F. Sequential Machines: Selected Papers. Addison Wesley, Reading, 1964. 12～20
3 Chow T S. Computer Software and Applications Conference, In: COMPSAC '78. The IEEE Computer Society's Second International , Nov 13-16, 1978, 1978. 169～174
4 Vaysburg B, Tahat L H, Korel B . Dependence Analysis in Reduction of Requirement Based Test Suites. In: Proceedings of the 2002 ACM SIGSOFT, 2002. 107～111
5 Kwang-Ting Cheng, Krishnakumar A S. Automatic Generation of Functionial Vectors Using the Extended Finite StateMachine Model. ACM Transactions on Design Automation of Electronic Systems, 1996, 1(1): 57～79