基于因果图的软件测试方法①

王立荣[1]　何　炜[2] (1.江苏自动化研究所 江苏 连云港 222006;
2.连云港杰瑞深软有限公司 江苏 连云港 222006)

摘　要： 因果图方法是一种有效的软件测试方法，适合于描述对于多种条件的组合、相应产生多个动作形式的测试用例设计。因果图最终被转换为判定表，但是由因果图到判定表的转换是很困难的。描述了因果图方法的原理并详细说明了因果图到判定表的转换算法，介绍了因果图方法测试用例的生成。

关键词： 软件测试；因果图；判定表；测试用例

Causal Map: A Method of Software Testing

WANG Li-Rong[1], HE Wei[2]
(1.Jiangsu Automation Research Institute, Lianyungang 222006, China; 2.Jerry Deep Soft Co., Ltd. Lianyungang, Lianyungang 222006, China)

Abstract: Causal map is an effective method of software testing, suitable for a wide range of conditions described in the portfolio, with a number of test cases produced from the corresponding forms of action. Causal map is finally converted to the determine table, but the conversion from the causal graph to the determine table is very difficult. This paper describes the principle of Causal mapping, presents a detailed description of the Causal map to determine meter conversion algorithm, and introduces the causal mapping methods to generate test cases.

Keywords: software testing; causality diagram; determine table; test cases

原因结果图法是软件测试中的一种重要方法。原因结果图法(俗称因果图法)是由美国 IBM 公司的 Elemendorf 在吸收硬件测试中自动生成逻辑组合电路测试等技术的基础上于 1973 年提出的，它是作为进行功能测试把功能说明书形式化的一种记述方法[1]。因果图法是用逻辑式描述程序的输入条件(原因)和输出条件(结果)，同时，用制约条件描述输入条件间的依赖关系的一种方法，其特征在于图式记述。

1　因果图

因果图是一种利用图形直观表示事物因果关系的知识表示方式，因果图可以形式化表示为：C=<S,A,R>，且 S=<X,B,G,P>其中符号含义如下：

C 一因果图模型；S 一因果图结构；

X 一中间事件，用来表示任何有原因的事件。在图形上以至少含有一条输入边，可以不含或含有 1 至多条输出边的圆圈节点表示。

B 一基本事件，用来表示任何没有原因或不追究其原因的事件，并且它至少为一个中间事件的原因。显然由于任何一个基本事件都不可能是另一基本事件的原因，基本事件之间相互独立。在图形上以不含有任何输入边但至少含有一条输出边的方框节点表示。

G 一逻辑门，它把输入变量通过逻辑运算组合成输出变量，输入变量到输出变量的映射间既可以是简单的与、或关系，也可以是复杂的逻辑表达式。图形上以至少含有两条输入边和一条及以上输出边的门节点表示。

P 一连接事件，它表示父节点事件(原因)导致子节

① 收稿时间:2009-08-11;收到修改稿时间:2009-09-08

点事件(结果)发生的事件，当父节点事件发生并且该连接事件发生时，子节点事件必定发生。从数值上其概率表示父节点与子节点间的因果强度，但作为一个事件，它与父节点事件相互独立。可见连接事件之间相互独立，而且连接事件与基本事件之间也相互独立。图形上表示为从基本事件、中间事件或逻辑门出发，始终指向中间事件的一条有向弧，指向同一个中间事件的所有连接事件是“或”关系。

A 一参数，包括基本事件的先验概率、连接事件的连接概率等。

下图为一个典型因果图示例[2]。

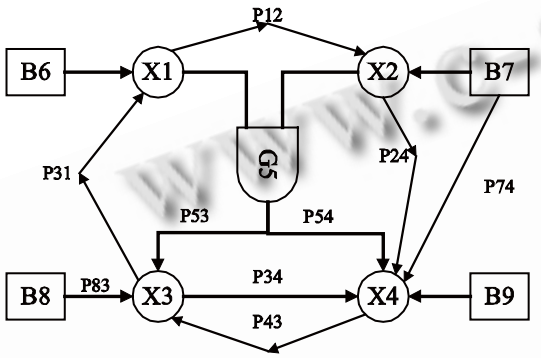

图 1 典型因果图

模型在推理前要求因果结构及所有参数已知。在图中未有任何标记的连接事件，表示当源事件发生的时候，目的事件一定发生，即该事件发生的概率为 1。

因果图在表达上没有要求拓扑结构必须为有向无环图，而允许出现有向环(如图 l 所示的回路(X3—P34—X4—P43—X3)，因此能够表达反馈等问题。因果图在给定事件之间的因果关系时，一方面采用了连接事件的概念，连接事件仅与其相连的父节点事件和子节点事件相关，其数量只随原因的增加而线性增加，每增加一个原因只需增加一个连接事件；另一方面连接概率表示的是因果强度而不是条件概率，能与专家知识相吻合；再一方面引入了逻辑门的概念，可以图形化表示原因之间的逻辑关系，因此能较为方便地给定事件之间的因果关系。

因果图具有以下特点：

① 完全基于概率论，有良好的理论基础。

② 能够处理因果环路结构。因为因果图表达的是用事件概率描述的领域随机变量间的因果关系，在其中蕴涵了一种联合概率分布，这样它对图形的拓扑结构没有限制。

③ 采用直接因果强度而不是条件概率，避免了在给定知识时知识间的相关性问题。这与领域专家头脑中的知识结构相对应，便于专家知识获取。

④ 引入了动态特性，能根据在线收到的信息动态变换因果图形结构，使之更符合当前时刻的客观实际。

⑤ 具有灵活的推理方式，既能由因到果，也可由果到因，还可因果混合。

2 因果图到判定表的转换

因果图法不仅帮助测试人员系统地产生高效测试情况，而且还能指出原规范中的不完全性和二义性。因果图法最重要、最困难的一步是将画好的因果图转换成有限项的判定表[3]。

因果图算法设计中存在下列几个问题：

(1) 历来的功能测试，都是测试人员阅读功能说明书，根据个人的经验和直观挑选出测试条件数据。因此，造成了测试条件数据的遗漏及重复很多，是发生错误的重要原因。为了解决此问题，可把对测试条件数据的挑选分为两个阶段进行。第一阶段，由用自然语言等描述功能说明书生成用逻辑表达式表示的形式功能说明书；第二阶段，把这一逻辑表达式输入到计算机，自动生成测试条件数据。

(2) 因果图中结果结点、中间结点和原因结点之间的关系很复杂，在设计算法时，必须分开考虑。设计分为静态部分和动态部分两部分分别进行。由于原因结点之间的约束关系是给定的，而中间结点和原因结点的赋值是根据结果结点的赋值来确定的，故将前一部分分为静态部分(约束关系)；后一部分分为动态部分(逻辑关系)。在设计算法时，先解决动态部分，然后再考虑静态部分。

(3) 对因果图的回溯，是从各结果结点开始分别进行的，当不考虑结点之间约束关系时，回溯所涉及的图可以简单看作一个具有多输入，单输出的电路，此时很容易进行回溯，依据因果图可以找到结果和原因的关系(用逻辑表达式表达)，再将此逻辑表达式转化为合取范式即可。但是这种方法显然将产生许多低效益的测试情况，如果测试情况的数目太大而不能实现，测试人员就会选择一些测试情况子集合，但是不能保证会把低效益的测试情况删去[4]。

因此，在设计因果图算法时，必须考虑以下几点：

① 当回溯到一个输出应为 0 的“ 或 ”结点时，不要给多于一个的 “ 或 ” 输入同时置 1。

② 回溯到一个输出应为 0 的“ 与 ”结点时，当然要列举出所有使得输出为 0 的输入组织情况。然而，如果人们测试了一个输入为 0 但有一个或多个其它输入为 1 的情况，就不必列举所有其它的输入可能为 1 的条件了。

② 当回溯的一个输出应为 0 的‘ 与 ’结点时，只需要列举所有的输入为 0 的一个条件(如果此“ 与 ”结点处于圈的中间，它的输入来自其它中间结点，则可能有极多的所有输入为 0 的情况)。

④ 因果图的原因结点之间存在各种约束条件，因此当回溯到原因结点时，应对当前产生的测试情况集的合法性进行检查并及时调整。

综合以上几点考虑，可以得到如下的算法框图。

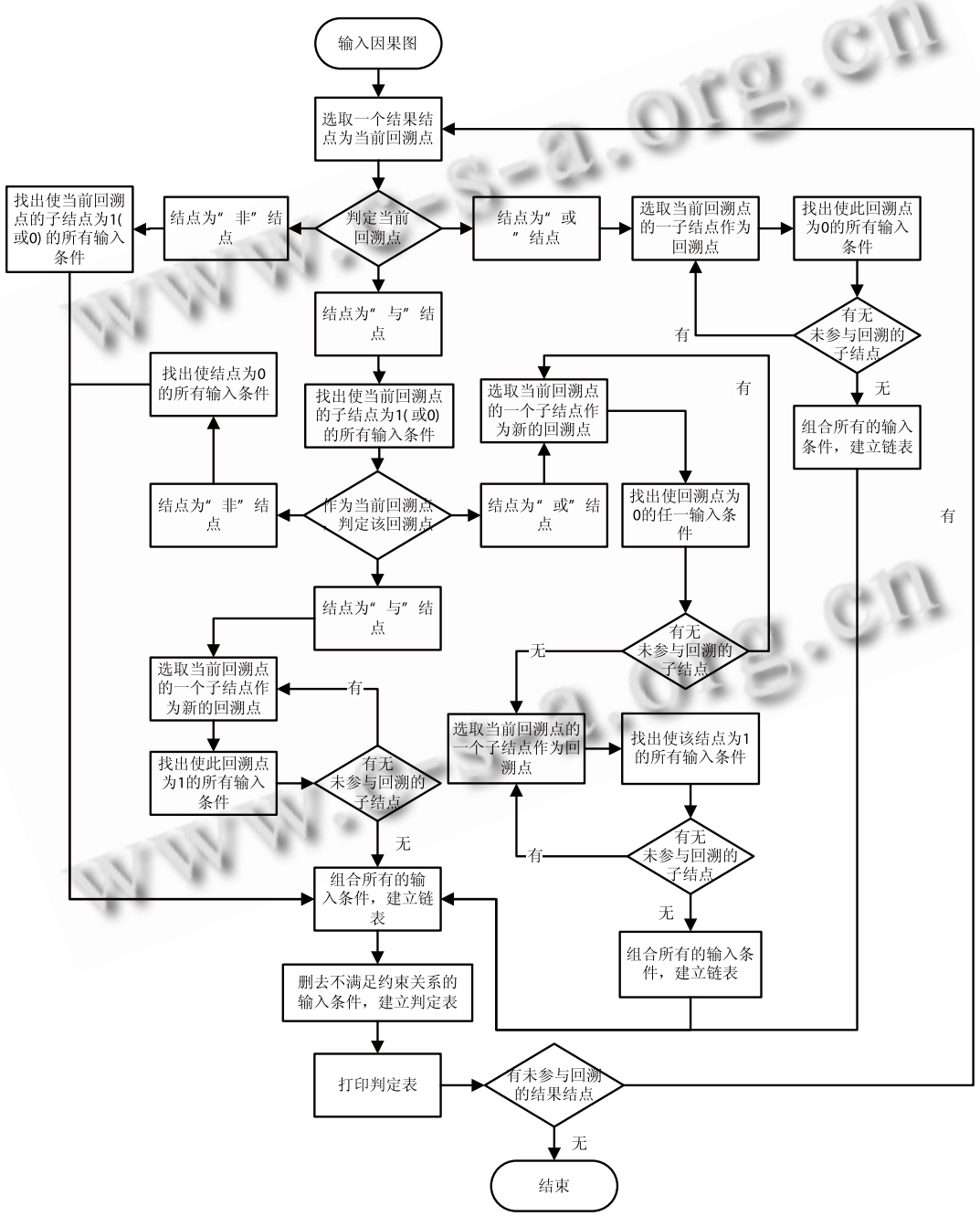

图 2　因果图到判定表的转换算法框图

运用因果图到判定表的回溯算法可望得到如下效果：

(1) 可靠性：目前大多认为发生不合格的原因大多是由于遗漏测试条件数据。上述算法由于是从逻辑表达式自动生成没有遗漏的测试条件数据，故可减少未被发现的不合格点，另外，在把功能说明书形式化阶段，还可发现功能说明书的不明确及不完全等情况。

(2) 生产率：由于在以往的经验方法中，不清楚测试条件数据的质量，所以倾向于编制超过需要量的测试条件数据。上述算法针对这一点，从逻辑表达式自动生成没有重复的测试条件数据，故可减少测试条件数据，另外因自动编制测试用单据，还可减少测试文档编制工作。

(3) 可管理性：在历来的测试方法中，因开发人员的经验和工作年限不同，测试条件数据的质量大多会大幅度地变动。针对这一点，本方法由逻辑表达式机械地生成测试条件数据，故可防止因人而异造成的测试质量的偏差，还使得对测试进度的管理变得更容易。

3 因果图法测试用例的生成

在软件测试用例生成时考虑输入条件之间的相互组合，可能会产生一些新的情况，但要检查输入条件的组合不是一件容易的事情，即使把所有输入条件划分成等价类，它们之间的组合情况也相当多，因此必须考虑采用一种适合于描述对于多种条件的组合，相应产生多个动作的形式来考虑设计测试用例，这就需要利用因果图。通过画因果图，在图上标明约束和限制，转换成判定表，然后设计测试用例[5]。适合于检查程序输入条件的各种组合情况。

利用因果图生成测试用例的基本步骤如下：

① 分析软件规格说明描述中，哪些是原因(即输入条件或输入条件的等价类)，哪些是结果(即输出条件)，把输入条件或输入条件的等价类作为原因，把输出条件作为结果，一一列出原因和结果，并分别赋予一个标识符[6]。

② 分析软件规格说明描述中的语义找出原因与结果之间、原因与原因之间对应的关系，根据这些关系，画出因果图。

③ 根据语法或测试场景限制，有些原因与原因之间，原因与结果之间的组合情况不可能出现。在因果图上加入一些记号表明这些约束或限制条件。

④ 把因果图转换为判定表，对于有限条目判定表，有 n 个条件，则必须有 2n 条规则。

⑤ 把判定表的每一列拿出来作为依据，设计测试用例，测试用例数目随输入数据数目的增加而线性地增加。

4 结语

因果图方法是一种有效的软件测试方法，它能发现程序与外部说明书之间的差异，指出功能说明书的不完整性和二义性，因果图最终被转换为判定表，可以帮助测试人员系统地生成一组高效的测试用例。在软件测试中利用因果图方法可以有效地生成无遗漏及重复的测试数据条件，得到准确的测试结果。因此基于因果图的软件测试方法值得进一步进行研究。

参考文献

1 樊兴华,仲昕,张勤,等.因果图推理的一种新方法.计算机科学, 2001,28(11):48－52,43.
2 王洪春,石庆喜,张勤.基于因果图的一种推理算法.微电子学与计算机, 2005,22(5):1－3,7.
3 Patton R.周予滨,姚静译.软件测试.北京:机械工业出版社, 2002.
4 王洪春.基于因果图的一种知识获取方法.计算机仿真, 2006,23(3):126－128.
5 Jorgensen PC.韩柯,杜旭涛译.软件测试(原书第二版).北京:机械工业出版社, 2003.
6 金凌紫.面向对象软件测试技术进展.计算机研究与发展, 1998,35(1):6－13.