测试用例设计之因果图方法

by:授客 QQ：1033553122

一. 方法简介

1. 定义

是一种利用图解法分析输入的各种组合情况，从而设计测试用例的方法，它适合于检查程序输入条件的各种组合情况。

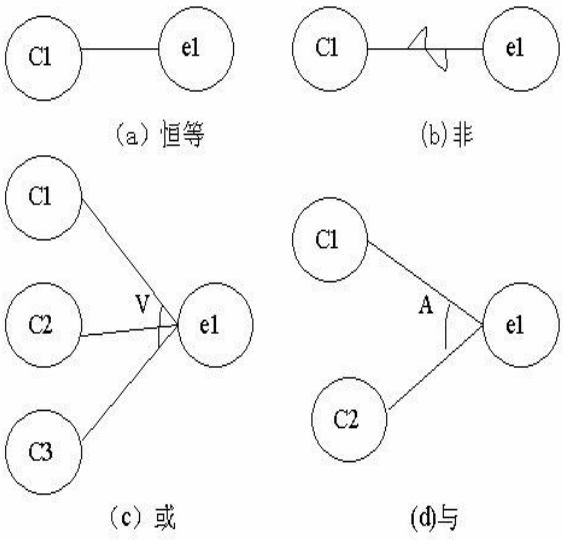
2. 因果图法产生的背景：

等价类划分法和边界值分析方法都是着重考虑输入条件，但没有考虑输入条件的各种组合、输入条件之间的相互制约关系。这样虽然各种输入条件可能出错的情况已经测试到了，但多个输入条件组合起来可能出错的情况却被忽视了。

如果在测试时必须考虑输入条件的各种组合，则可能的组合数目将是天文数字，因此必须考虑采用一种适合于描述多种条件的组合、相应产生多个动作的形式来进行测试用例的设计，这就需要利用因果图（逻辑模型）。

3. 因果图介绍

1） 4 种符号分别表示了规格说明中向 4 种因果关系。



2） 因果图中使用了简单的逻辑符号，以直线联接左右结点。左结点表示输入状态（或称原因），右结点表示输出状态（或称结果）。

3） Ci 表示原因，通常置于图的左部；ei 表示结果，通常在图的右部。Ci 和 ei 均可取值 0 或 1，0 表示某状态不出现，1 表示某状态出现。

4. 因果图概念

1）关系

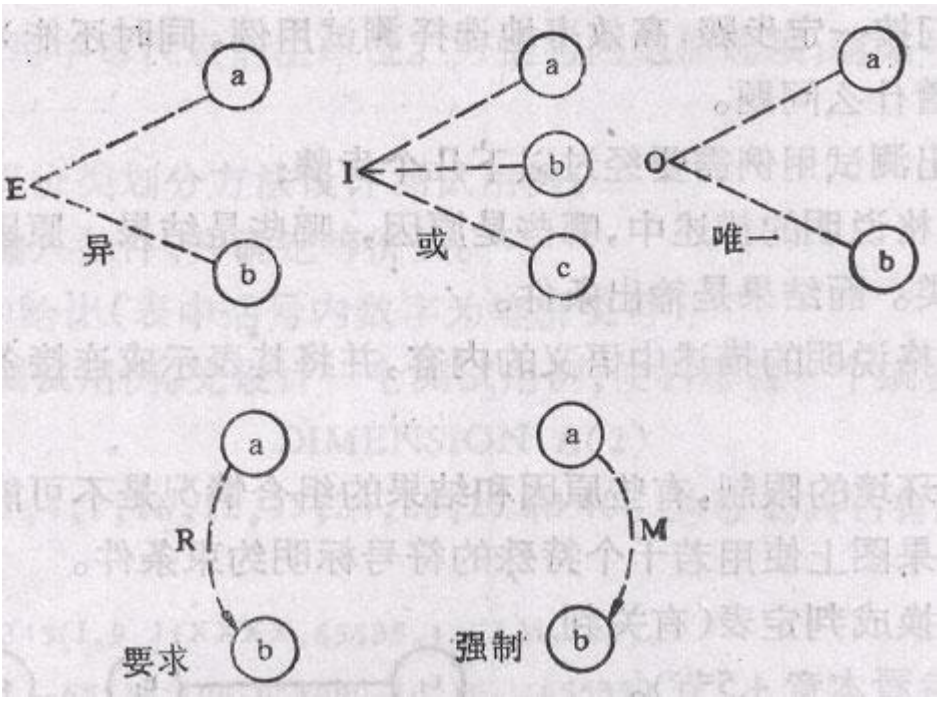
①恒等：若 ci 是 1，则 ei 也是 1；否则 ei 为 0。
②非：若 ci 是 1，则 ei 是 0；否则 ei 是 1。
③或：若 c1 或 c2 或 c3 是 1，则 ei 是 1；否则 ei 为 0。“或”可有任意个输入。
④与：若 c1 和 c2 都是 1，则 ei 为 1；否则 ei 为 0。“与”也可有任意个输入。

前面两者①，②，考虑的是单个输入（原因）和输出（结果）之间的关系：也就是输入为真(假)时，输出的值为真还是假的对应关系，很自然的，我们很容易想到有两种情况：一种和输入同真同假，一种和输入相反。

后面两者③，④，考虑的是多个输入之间的组合输入和输出之间的关系：我们也很容易想到：一种是组合关系为或(or)，一种组合关系为组合(and)，满足这种组合关系得出的输出才为真，否则为假

2) 约束

输入状态相互之间还可能存在某些依赖关系，称为约束。例如，某些输入条件本身不可能同时出现。输出状态之间也往往存在约束。在因果图中，用特定的符号标明这些约束。



A. 输入条件的约束有以下 4 类：

① E 约束（异）：a 和 b 中至多有一个可能为 1，即 a 和 b 不能同时为 1。
② I 约束（或）：a、b 和 c 中至少有一个必须是 1，即 a、b 和 c 不能同时为 0。
③ O 约束（唯一）：a 和 b 必须有一个，且仅有 1 个为 1。
④ R 约束（要求）：a 是 1 时，b 必须是 1，即不可能 a 是 1 时 b 是 0。

B. 输出条件约束类型

输出条件的约束只有 M 约束（强制）：若结果 a 是 1，则结果 b 强制为 0。

5. 采用因果图法设计测试用例的步骤

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)，并给每个原因和结果赋予一个标识符。

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)由于语法或环境限制，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，为表明这些特殊情况，在因果图上用一些记号表明约束或限制条件。

4)把因果图转换为判定表。

5)把判定表的每一列拿出来作为依据,设计测试用例。

二.实战演习

1.某软件规格说明书包含这样的要求：第一列字符必须是 A 或 B，第二列字符必须是一个数字，在此情况下进行文件的修改，但如果第一列字符不正确，则给出信息 L；如果第二列字符不是数字，则给出信息 M。

解答：

1) 找出原因(输入)和结果(输出)：

原因：

C1.第一列字符为 A？；

C2.第一列字符为 B？；

C3.第二列字符为数字？

技巧：如上，查找原因时将输入有效等价类和输入无效等价类看作是同一个原因的正反面取值，即查找有效等价类，然后加上问号

结果：

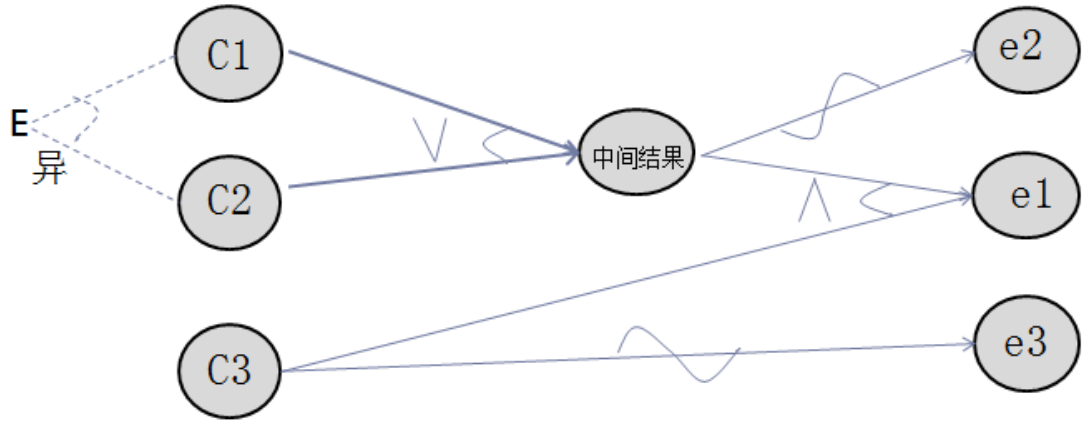
e1.修改文件

e2.给出信息 L

e3.给出信息 M

2) 其对应的因果图如下：

考虑到原因 1 和原因 2 不可能同时为 1，因此在因果图上施加 E 约束。



注意，学习添加中间结果节点

3) 根据因果图建立判定表。

A	B	C	D	E	F	G	H	I	J
条件(原因)	C1	1	1	1	1	0	0	0	0
	C2	1	1	0	0	1	0	1	0
	C3	1	0	1	0	1	1	0	0
	中间结果			1	1	1	0	1	0
动作(结果)	e1			1	0	1	0	0	0
	e2			0	0	0	1	0	1
	e3			0	1	0	0	1	1

表中，C1 和 C2 是不可能同时为 1 的，即不可能同时出现，所以应排除这两种情况。

4）用例设计

针对每一条规则(C,D 列除外)设计一条用例

2.有一个处理单价为 5 角钱的饮料的自动售货机软件测试用例的设计。其规格说明如下：若投入 5 角钱或 1 元钱的硬币，押下〖橙汁〗或〖啤酒〗的按钮，则相应的饮料就送出来。若售货机没有零钱找，则一个显示〖零钱找完〗的红灯亮，这时在投入 1 元硬币并押下按钮后，饮料不送出来而且 1 元硬币也退出来；若有零钱找，则显示〖零钱找完〗的红灯灭，在送出饮料的同时退还 5 角硬币。

1) 分析这一段说明，列出原因和结果

原因：

C1：投入 1 元钱？

C2：投入 5 角钱？

C3：押下橙汁？

C4：押下啤酒？

C5：售货机有零钱找？

C6：中间结果，按下按钮？

C7：中间结果，找钱成功？

结果：

e1：送出橙汁

e2：送出啤酒

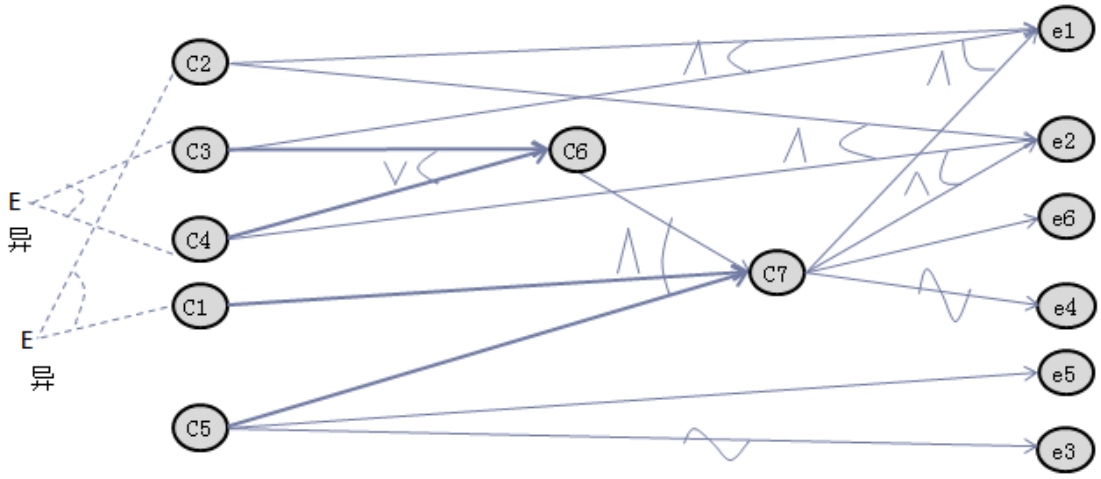
e3：高亮【零钱找完】的红灯

e4：退出 1 元硬币

e5：熄灭【零钱找完】的红灯

e6：退出 5 角硬币

2）画出因果图



说明：因果图需要对需求和逻辑理解很透彻，不同的理解画出的因果图不同，自然设计难易程度也就不一样，个人建议少用因果图，多用场景法，因为相比之下，场景法设计用例实施起来会比较容易

3）转换为判定表

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
条件(原因)	C1：投入1元钱？	1	-	1	1	1	1	0	0	0	0	0	0	0	0
	C2：投入5角钱？	1	-	0	0	0	0	1	1	1	1	0	0	0	0
	C3：押下橙汁？	-	1	1	1	0	0	1	1	0	0	1	1	0	0
	C4：押下啤酒？	-	1	0	0	1	1	0	0	1	1	0	0	1	1
	C5：售货机有零钱找？	-	-	1	0	1	0	1	0	1	0	1	0	1	0
	C6：中间结果，按下按钮？	-	-	1	1	1	1	1	1	1	1	1	0	0	1
	C7：中间结果，找钱成功？	-	-	1	0	1	0	-	-	-	-	-	-	-	-
动作(结果)	e1：送出橙汁			1	0	0	0	1	1	0	0	0	0	0	0
	e2：送出啤酒			0	0	1	0	0	0	1	1	0	0	0	0
	e3：高亮【零钱找完】的红灯			0	1	0	1	0	1	0	1	0	1	0	1
	e4：退出1元硬币			0	1	0	1	0	0	0	0	0	0	0	0
	e5：熄灭【零钱找完】的红灯			1	0	1	0	1	0	1	0	1	0	1	0
	e6：退出5角硬币			1	0	1	0	0	0	0	0	0	0	0	0

4）简化判定表

A	B	C	D	E	F	G	H	I	J	K
条件(原因)	C1：投入1元钱？	1	1	1	0	0	0	0	0	0
	C2：投入5角钱？	0	0	0	1	1	1	1	0	0
	C3：押下橙汁？	1	–	0	1	1	0	0	–	–
	C4：押下啤酒？	0	–	1	0	0	1	1	–	–
	C5：售货机有零钱找？	1	0	1	1	0	1	0	1	0
	C6：中间结果，按下按钮？	1	1	1	1	1	1	1	1	1
	C7：中间结果，找钱成功？	1	0	1	–	–	–	–	–	–
动作(结果)	e1：送出橙汁	1	0	0	1	1	0	0	0	0
	e2：送出啤酒	0	0	1	0	0	1	1	0	0
	e3：高亮【零钱找完】的红灯	0	1	0	0	1	0	1	0	1
	e4：退出1元硬币	0	1	0	0	0	0	0	0	0
	e5：熄灭【零钱找完】的红灯	1	0	1	1	0	1	0	1	0
	e6：退出5角硬币	1	0	1	0	0	0	0	0	0

5）用例设计

参考文章：测试用例设计白皮书_张元礼