软件测试技术

User : testing_software@163.com
Password: testing

任课教师：侯鲲

bluebloodhk@163.com

内容提要

- 黑盒测试概述
- 黑盒测试方法

内容提要

- 黑盒测试概述
- 黑盒测试方法

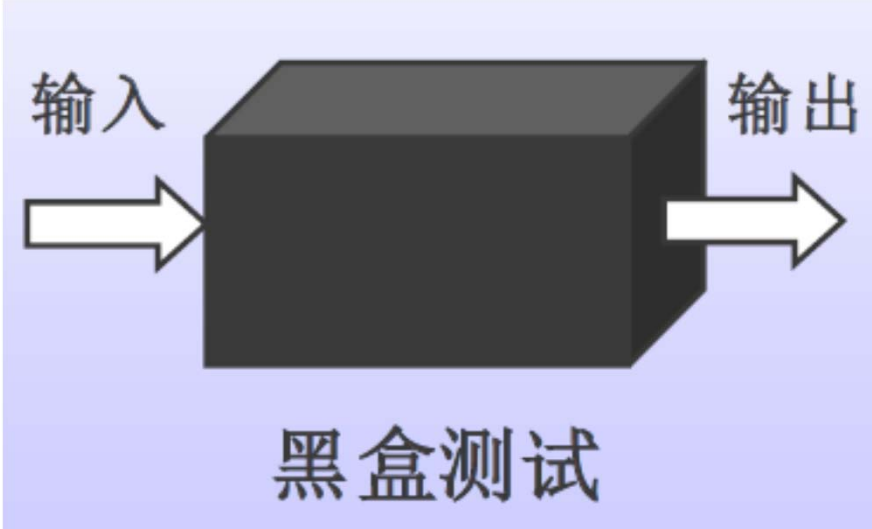
黑盒测试概述

- 黑盒测试概念
- 通过测试和失败测试
- 黑盒测试的应用范围
- 黑盒测试的过程
- 黑盒测试可能发现的错误
- 黑盒测试的优缺点
- 黑盒测试分类

黑盒测试概念

黑盒测试：

是基于需求说明书的软件测试，在这种测试下，不需要了解软件的内部结构，以及软件代码的具体实现。



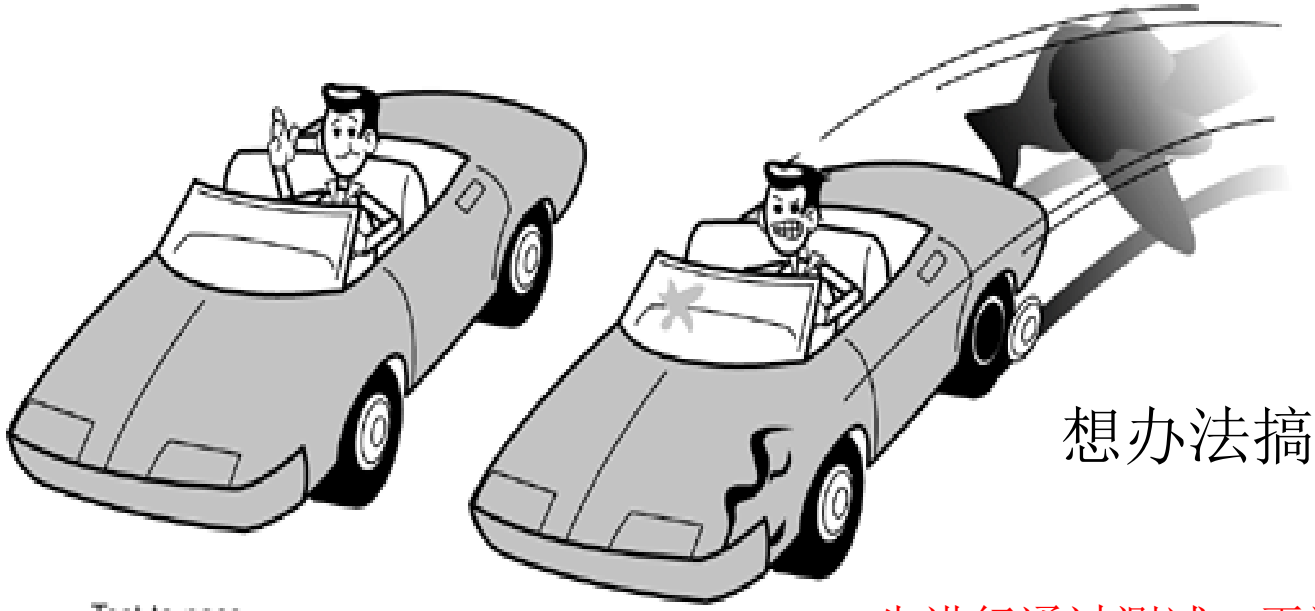
黑盒测试概念

- 在不了解软件结构和内部代码的情况下测试软件（软件处于运行状态）称为动态黑盒测试
- 动态黑盒测试也叫行为测试，因为测试的是软件在使用过程中的实际行为
- 有效的黑盒测试需要关于软件行为的定义，即软件需求规格说明书，这是动态黑盒测试的基础

通过性测试 vs 失效性测试

通过测试：对软件的正常功能进行测试，保证软件能够完成它声称的功能。一般选择有效测试案例进行测试

失败测试：又称为迫使出错测试，采取各种措施和手段使软件出错。目的是测试软件在各种极端情况下运行的状况。一般选择无效测试案例进行测试。

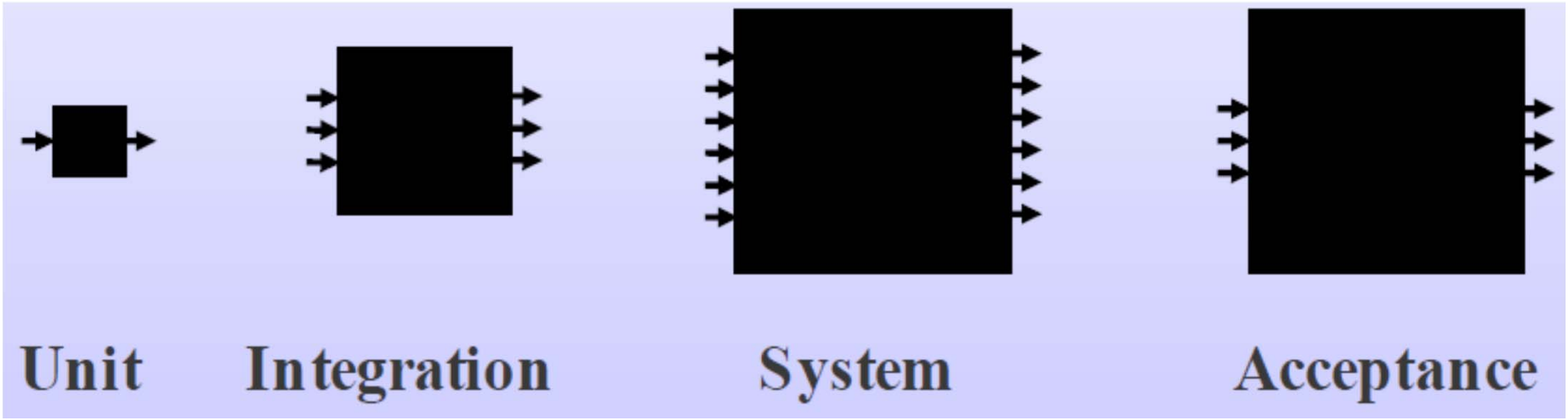


想办法搞垮软件

先进行通过测试，再进行失败测试

黑盒测试的应用范围

- 黑盒测试基于软件的基本需求，主要是针对软件功能的测试。
- 黑盒测试可以应用到系统开发的各个阶段，包括单元测试、集成测试、系统测试以及确认测试

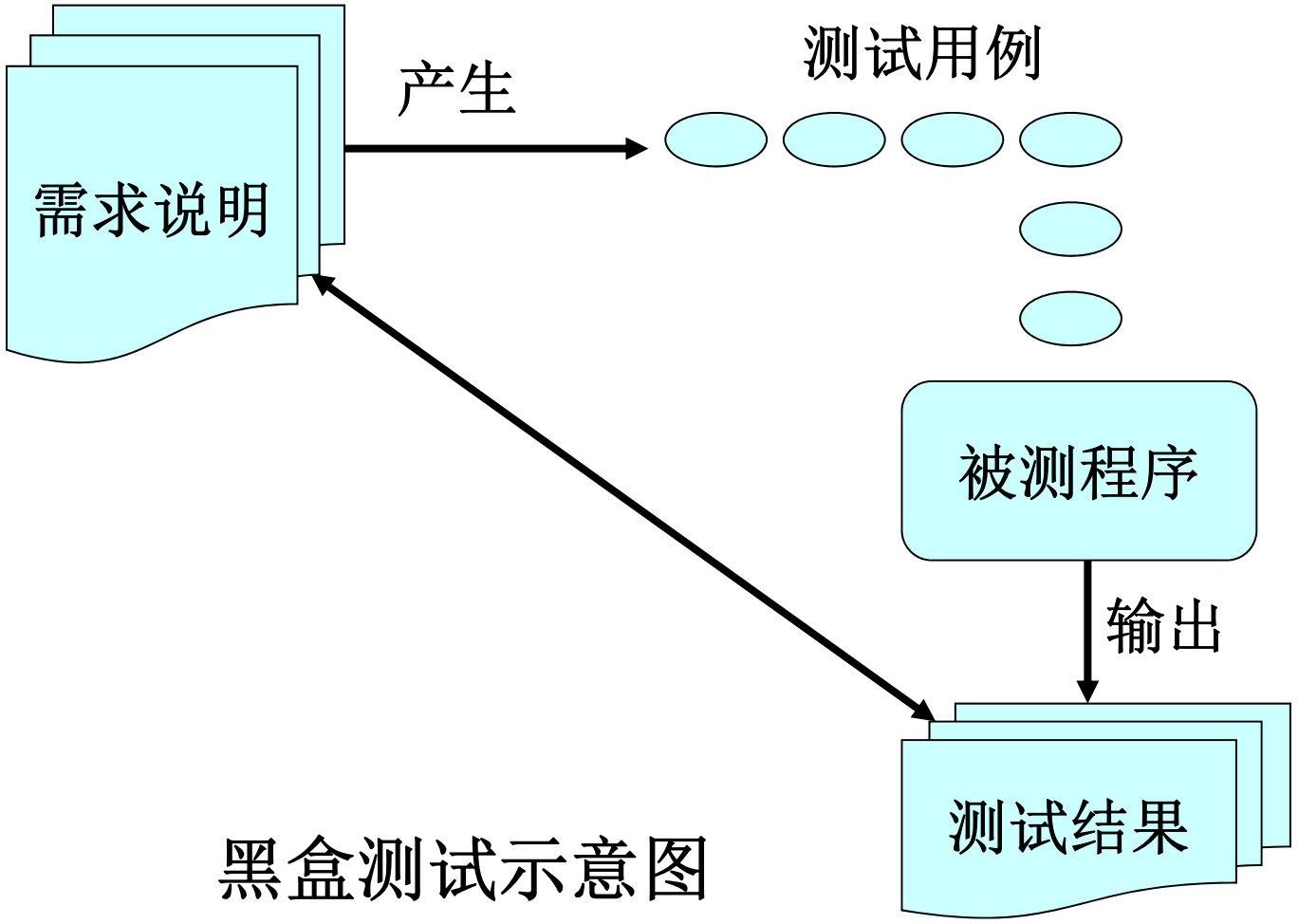


黑盒测试的基本过程

1. 分析需求规格说明书；
2. 基于需求规格说明书选择有效输入确认软件能够正确处理；选择无效输入验证软件是否能够处理它们；
3. 确定预期输出结果；
4. 用选择的输入构建测试用例；
5. 进行测试；
6. 比较实际输出结果和预期输出结果；
7. 确认被测试软件功能的正确性。

黑盒测试的基本过程

行为测试：　　输入 ——→ 输出



黑盒测试示意图

黑盒测试可能发现的错误

- 不正确的或遗失的功能
- 接口错误
- 数据结构或外部数据库访问错误
- 性能缺陷
- 初始化和终止错误

黑盒测试的特点

- 黑盒测试与软件的具体实现过程无关
- 黑盒测试用例的设计可以和软件实现同时进行
- 对于无法得到源代码的软件可以完成一定的测试
- 正规的黑盒测试可以指导测试人员选择高效和有效的，能发现软件缺陷的测试子集。

缺点

黑盒测试不能确认对软件测试的程度。无论多么聪明和勤奋的测试人员，都不可能通过黑盒测试测试程序的所有路径。

黑盒测试分类

按测试目的分：功能性测试及非功能性测试

- 功能性测试-验证被测试软件的功能正确性、一致性。
- 非功能性测试-验证软件的非功能属性，比如性能测试等。
- 由于非功能性测试的存在，黑盒测试较白盒测试有更广泛的应用。

内容提要

- 黑盒测试概述
- 黑盒测试方法

黑盒测试方法

- 黑盒测试方法
  - 等价类划分
  - 边界值分析
  - 因果图
  - 错误推测
  - 功能图法
  - …

等价类划分

- 把所有可能的输入数据，即程序的输入域划分成若干部分，然后从每部分中选取少数有代表性的数据做为测试用例。
- 传统的等价类划分测试的实现分为两步进行
  - 划分等价类
  - 确定测试用例

先从程序的功能说明书中找出各个输入条件，然后为每个输入条件划分两个或多个等价类，形成若干个互不相交的子集，称之为等价类。

精简海量测试用例集，同时保持有效性。

等价类

- 等价类中的各个数据对于揭露程序中的错误都是等效的。意味着：
  - 如果等价类中的一个测试案例发现缺陷，那么其它案例都将发现同样的软件缺陷
  - 如果等价类中的一个测试案例没有发现缺陷，那么其它案例也不会发现软件缺陷
- 等价类对于测试有两个重要的意义
  - 完备性——整个输入域提供完备性
  - 无冗余性——若互不相交则可保证无冗余性

等价类分类

划分等价类可分为两种情况：

有效等价类：完全满足程序输入的规格说明,有效、有意义的输入数据所构成的集合。

----检验程序是否满足规格说明所规定的功能和性能

无效等价类：　不满足程序输入要求，或者无效的输入数据所构成的集合。

----鉴别程序异常情况的处理，保障在输入错误或空时仍有异常保护，保证程序的可靠性

划分等价类的原则

- 到目前为止没有划分高质量等价类的标准方法，不同的功能说明可能使用不同的方法。
- 不同的等价类得到的测试用例质量不同。
- 在划分等价类时，可以参考下面的建议：

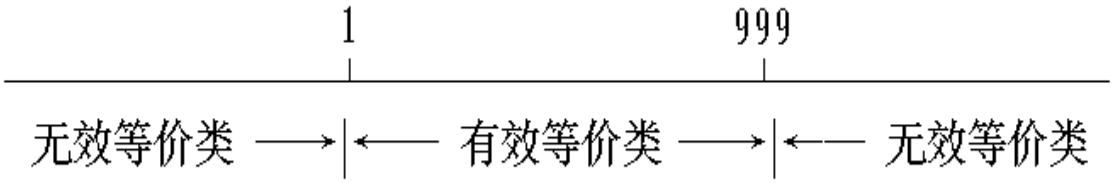
(1) 如果输入条件规定了取值范围，或值的个数，则可以确立一个有效等价类和两个无效等价类。

例如，在程序的规格说明中，对输入条件约束：

“…… 项数可以从1到999 ……”

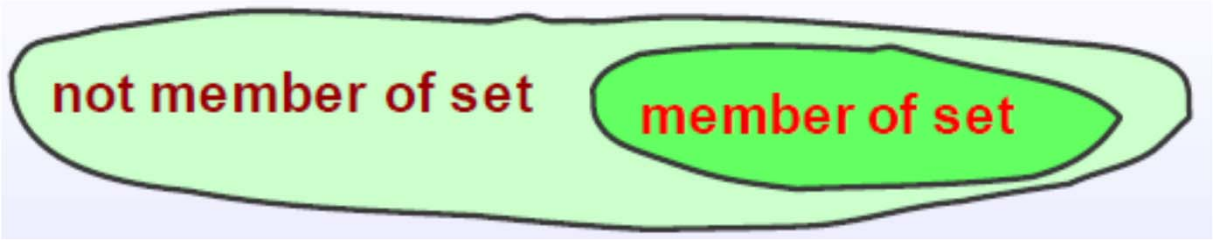
则有效等价类是“1≤项数≤999”

两个无效等价类是“项数＜1”或“项数＞999”。



划分等价类的原则

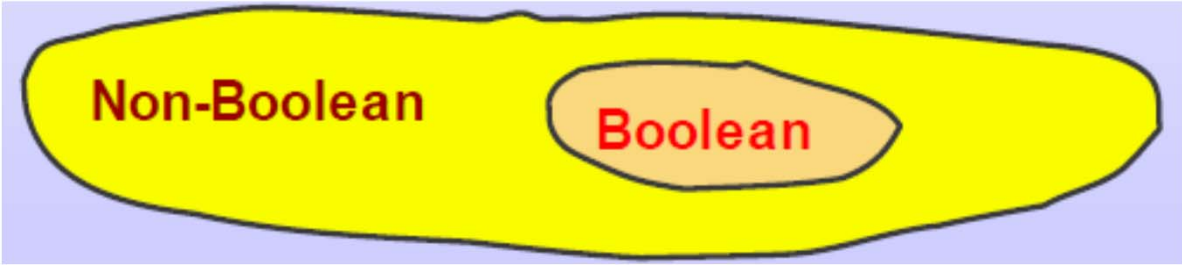
(2) 如果输入条件规定了输入值的集合，或者是规定了“必须如何”的条件，这时可确立一个有效等价类和一个无效等价类。



例如，在C语言中对变量标识符规定为“以字母或下划线打头的……串”。那么所有以字母或下划线打头的构成有效等价类，而不在此集合内（不以字母打头）的归于无效等价类。

划分等价类的原则

（3）如果输入条件是一个布尔量，则可以确定一个有效等价类和一个无效等价类。



划分等价类的原则

(4)如果规定了输入数据的一组值，而且程序要对每个输入值分别进行处理。　这时可为每一个输入值确立一个有效等价类，并针对这组值确立一个无效等价类，它是所有不允许的输入值的集合。

例如，在教师上岗方案中规定对教授、副教授、讲师和助教分别计算分数，做相应的处理。因此可以确定4个有效等价类为教授、副教授、讲师和助教，一个无效等价类，它是所有不符合以上身分的人员的输入值的集合。

划分等价类的原则

（5）如果规定了输入数据必须遵守的规则，则可以确立一个有效等价类（符合规则）和若干个无效等价类（从不同角度违反规则）。

例如，e-mail地址

（6）在确定已知的等价类中个元素在程序中的处理方式不同，则应将等价类进一步划分成更小的等价类。

确定测试用例

- 步骤
  1. 划分等价类
  2. 为每一个等价类规定一个唯一编号
  3. 设计一个测试用例，使其尽可能多地覆盖尚未覆盖的有效等价类， 重复这一步直到所有有效等价类均被测试用例所覆盖
  4. 设计一个测试用例，使其只覆盖一个无效等价类，重复这一步，直到所有无效等价类均被覆盖
- 表示法
  - 使用等价类表格有利于确定测试用例

条件	有效等价类	编号	无效等价类	编号

等价类测试用例的种类

- 针对是否对无效数据进行测试，可以将等价类测试分为
  - 一般等价类测试
    - 弱一般等价类测试用例
    - 强一般等价类测试用例
  - 健壮等价类测试
    - 弱健壮等价类测试用例
    - 强健壮等价类测试用例

等价类测试用例的种类

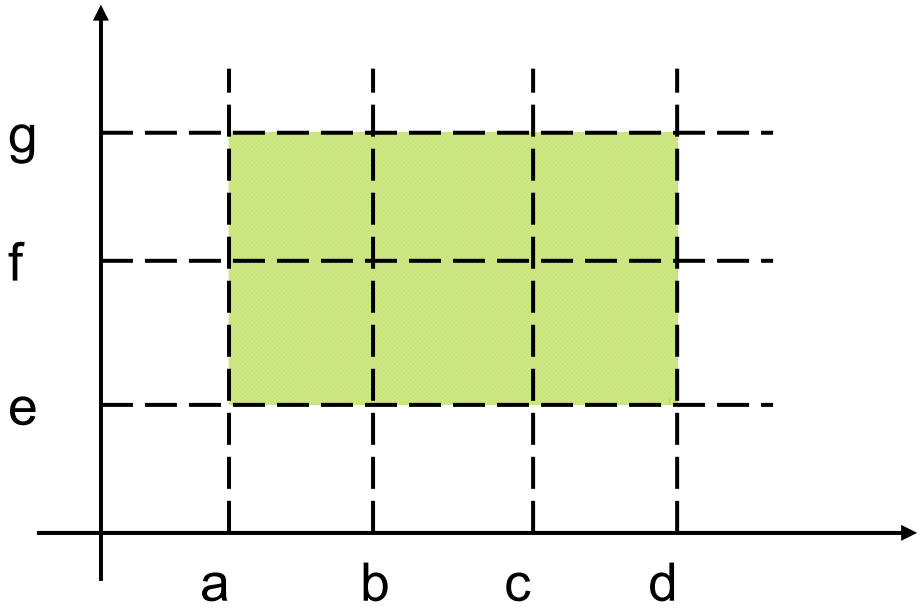
- 一般等价类测试：
  - 不考虑无效数据值
  - 测试用例使用每个等价类中的一个值
- 健壮等价类测试：
  - 主要的出发点是考虑了无效等价类
  - 对有效输入，测试用例从每个有效等价类中取一个值
  - 对无效输入，一个测试用例有一个无效值，其他值均取有效值

等价类测试用例的种类

有两个变量x_1和x_2的函数F。如果函数F实现为一个程序，则输入两个变量x_1和x_2会有一些(可能未规定)边界：

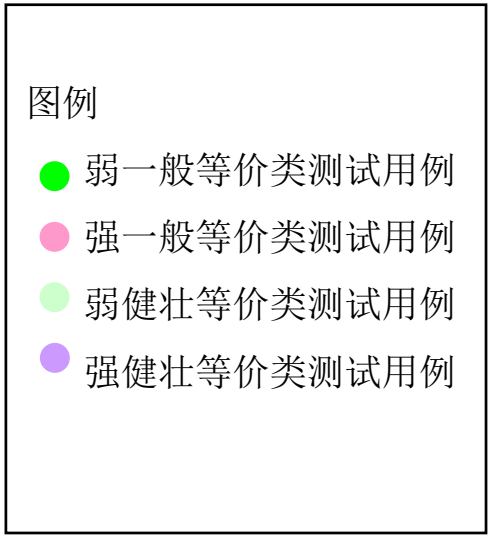
$a \leq x_1 \leq d$　区间为[a，b)，[b，c)，[c，d]

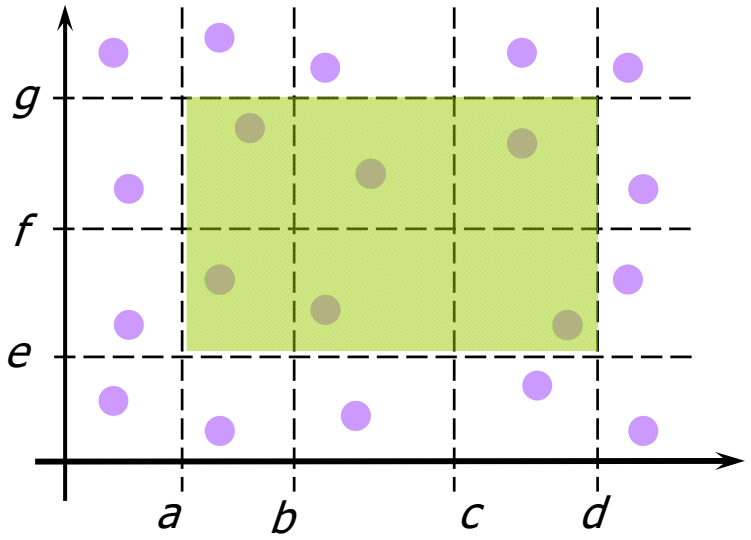
$e \leq x_2 \leq g$　区间为[e，f)，[f，g]



等价类测试用例的种类

- 函数$y = f(x_1, x_2)$ 输入变量的取值范围分别为:$x_1 \in [a,d]$, $x_2 \in [e,g]$, 根据规格说明划分得相应的等价类
  - X_1：有效等价类 [a, b) [b, c) [c, d]；无效等价类（-∞,a), (d, +∞)
  - X_2：有效等价类 [e,f) [f,g]；无效等价类（-∞,e), (g, +∞)

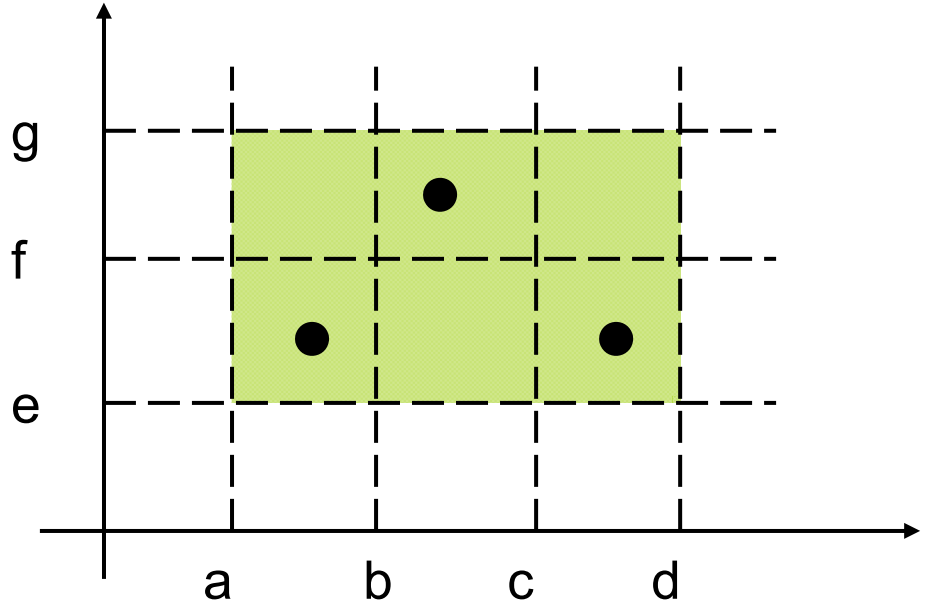




等价类测试用例的种类

弱一般等价类测试

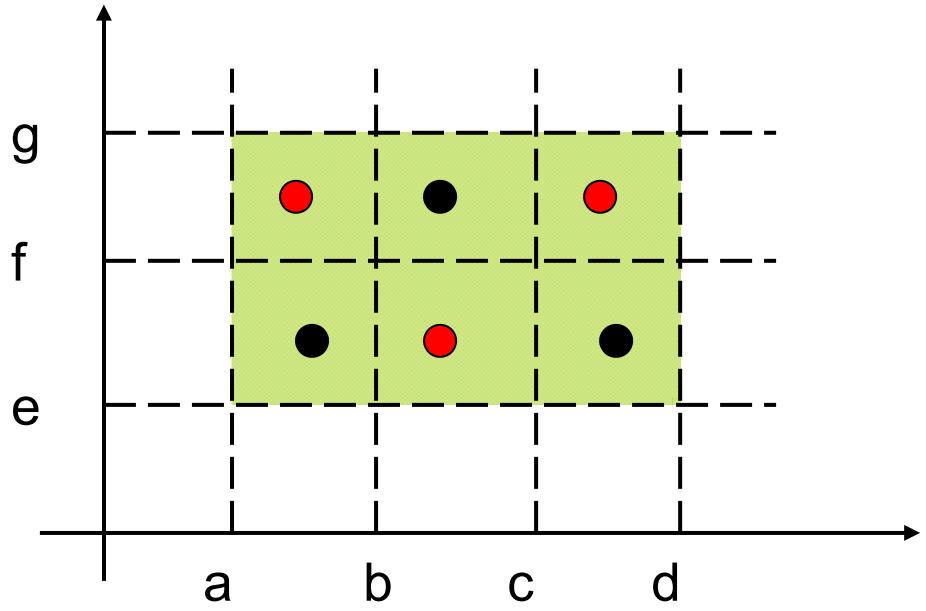
- 弱一般等价类测试是基于单缺陷假设的；
- 弱一般等价类测试通过使用一个测试用例中的每个等价类(区间)的一个变量实现。



等价类测试用例的种类

强一般等价类测试

- 强一般等价类测试是基于多缺陷假设的；
- 等价类笛卡儿积的每个元素对应的测试用例；

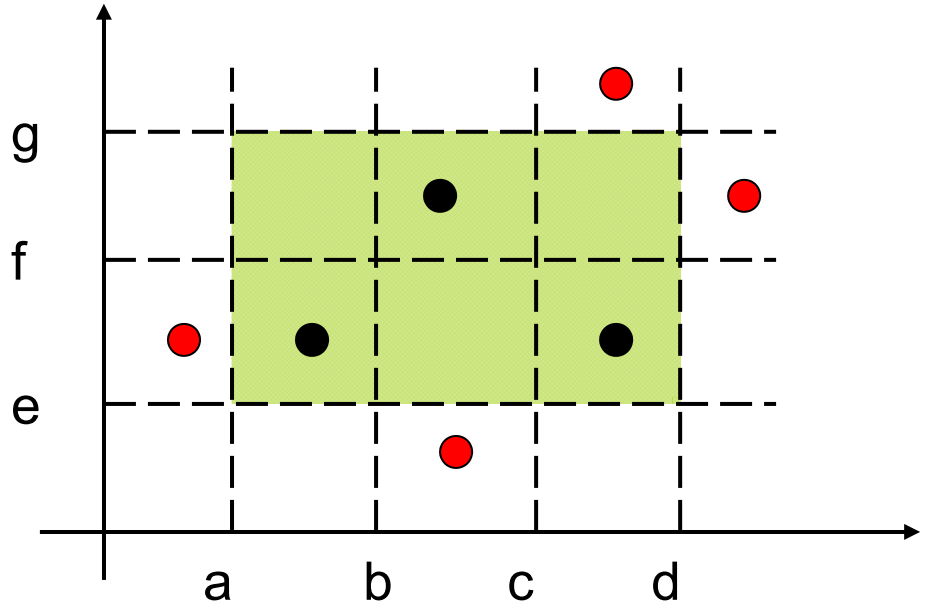


等价类测试用例的种类

弱健壮等价类测试

1．对于有效输入，使用每个有效类的一个值(就像我们在所谓弱一般等价类测试中所做的一样。)

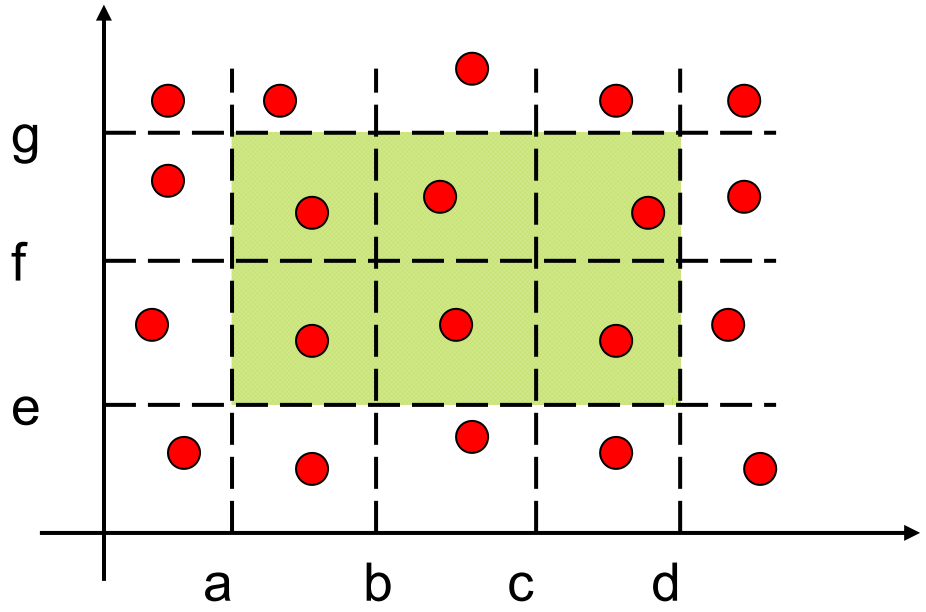
2．对于无效输入，测试用例将拥有一个无效值，并保持其余的值都是有效的。



等价类测试用例的种类

强健壮等价类测试

所有等价类笛卡儿积的每个元素中获得测试用例。



等价类测试用例的种类

- 健壮等价类测试存在两个问题：
  - 需要花费精力定义无效测试用例的期望输出
  - 对强类型的语言没有必要考虑无效的输入

举例

程序功能说明

- NextDate（年，月，日）是三个变量的函数。函数返回输入日期的下一个日期。变量年份，月份，日期都是整数值，且满足下面的条件： 1900≤年 ≤2060 ,1≤月 ≤12, 1≤日 ≤31.

例如：输入为1986年6月9日，则该函数的输出应为1986年6月10日。

举例

第一步：划分等价类

输入条件	*有效等价类*	*编号*	*无效等价类*	
年（Y）	1900≤年≤2060	Y1	年<1900	Y4
			年>2060	Y5
月（M）	1≤月≤12	M2	月<1	M6
			月>12	M7
日 （D）	1≤日≤31	D3	日<1	D8
			日>31	D9

举例

第二步：生成测试用例

输入条件	*有效等价类*	*编号*	*无效等价类*	
年（Y）	1900≤年≤2060	Y1	年<1900	Y4
			年>2060	Y5
月（M）	1≤月≤12	M2	月<1	M6
			月>12	M7
日 （D）	1≤日≤31	D3	日<1	D8
			日>31	D9

NextDate函数的弱/强一般等价类测试用例

用例ID	月份	日期	年	预期输出
WN1，SN1	6	15	1912	6/16/1912

举例

输入条件	有效等价类	编号	无效等价类	
年（Y）	1900≤年≤2060	Y1	年<1900	Y4
			年>2060	Y5
月（M）	1≤月≤12	M2	月<1	M6
			月>12	M7
日 （D）	1≤日≤31	D3	日<1	D8
			日>31	D9

NextDate函数的弱健壮等价类测试用例

用例ID	月份	日期	年	预期输出
WR1	6	15	1912	6/16/1912
WR2	-1	15	1912	无效输入
WR3	13	15	1912	无效输入
WR4	6	-1	1912	无效输入
WR5	6	32	1912	无效输入
WR6	6	15	1811	无效输入
WR7	6	15	2013	无效输入

举例

输入条件	有效等价类	编号	无效等价类	
年（Y）	1900≤年≤2060	Y1	年<1900	Y4
			年>2060	Y5
月（M）	1≤月≤12	M2	月<1	M6
			月>12	M7
日（D）	1≤日≤31	D3	日<1	D8
			日>31	D9

NextDate函数的强健壮等价类测试用例

- 用例个数：3*3*3 = 27
- 利用强健壮等价类获得的测试用例虽然覆盖了需测试的各个要点，但是测试用例个数却大幅度提高，而且包含了若干无意义的测试用例

部分用例

用例ID	月份	日期	年	预期输出
SR1	-1	15	1912	无效输入
SR2	6	-1	1912	无效输入
SR3	6	15	1811	无效输入
SR4	-1	-1	1912	无效输入
SR5	6	-1	1811	无效输入
SR6	-1	15	1811	无效输入
SR7	-1	-1	1811	无效输入

举例

NextDate函数的等价类另一种划分法

条件	有效等价类	编号	无效等价类	编号
年	闰年	1	年<1900	11
	平年	2	年>2060	12
月	1，3，5，7，8，10	3	月<1	13
	4，6，9，11	4	月>12	14
	2月	5		
	12月	6		
日	[1,28]	7	日<1	15
	29	8	日>31	16
	30	9		
	31	10		

举例

条件	有效等价类	编号	无效等价类	编号
年	闰年	1	年<1900	11
	平年	2	年>2060	12
月	1，3，5，7，8，10	3	月<1	13
	4，6，9，11	4	月>12	14
	2月	5		
	12月	6		
日	[1,28]	7	日<1	15
	29	8	日>31	16
	30	9		
	31	10		

NextDate函数的弱一般等价类测试用例

用例ID	月份	日期	年	预期输出	
WN1	7	31	2001	8/1/2000	等价类2,3,10
WN2	4	30	2006	5/1/2006	等价类2,4,9
WN3	2	29	2000	3/1/2000	等价类1,5,8
WN4	12	16	1901	12/17/1901	等价类2,6,7

举例

条件	有效等价类	编号	无效等价类	编号
年	闰年	1	年<1900	11
	平年	2	年>2060	12
月	1，3，5，7，8，10	3	月<1	13
	4，6，9，11	4	月>12	14
	2月	5		
	12月	6		
日	[1,28]	7	日<1	15
	29	8	日>31	16
	30	9		
	31	10		

NextDate函数的强一般等价类测试用例

测试用例的个数：M×D×Y = 2 * 4 * 4 = 32

举例

条件	有效等价类	编号	无效等价类	编号
年	闰年	1	年<1900	11
	平年	2	年>2060	12
月	1，3，5，7，8，10	3	月<1	13
	4，6，9，11	4	月>12	14
	2月	5		
	12月	6		
日	[1,28]	7	日<1	15
	29	8	日>31	16
	30	9		
	31	10		

NextDate函数的弱健壮等价类测试用例

所有WN_i +

用例ID	月份	日期	年	预期输出
WR1	3	19	1830	无效输入
WR2	3	19	3000	无效输入
WR3	-2	10	2004	无效输入
WR4	15	10	2004	无效输入
WR5	8	-2	2004	无效输入
WR6	8	38	2004	无效输入

举例

条件	有效等价类	编号	无效等价类	编号
年	闰年	1	年<1900	11
	平年	2	年>2060	12
月	1，3，5，7，8，10	3	月<1	13
	4，6，9，11	4	月>12	14
	2月	5		
	12月	6		
日	[1,28]	7	日<1	15
	29	8	日>31	16
	30	9		
	31	10		

NextDate函数的强一般等价类测试用例

测试用例的个数：M×D×Y = 3 * 4 * 3 = 36

关于两种方案的讨论

每种结果都覆盖了一些应该测试的功能点

1. 第一种方案过多的关注于对无效等价类的测试，遗漏了很多NextDate问题需测试的关键点，比如平年闰年、二月、大小月、月底等等。
2. 第二种方案在第一种的结果的基础上，增加了对二月、年底等的测试；通过日常知识的运用，使得测试用例涵盖了大部分需测试的要点，但仍有些遗漏，例如年底、平年的二月等

指导方针和观察

- 等价类测试的弱形式(一般或健壮)不如对应的强形式的测试全面。
- 如果实现语言是强类型的(无效值会引起运行时错误)，则没有必要使用健壮形式的测试。
- 如果错误条件非常重要，则进行健壮形式的测试是合适的。
- 如果输入数据以离散值区间和集合定义，则等价类测试是合适的。当然也适用于如果变量值越界系统就会出现故障的系统。
- 通过结合边界值测试，等价类测试可得到加强。
- 如果程序函数很复杂，函数的复杂性可以帮助标识有用的等价类，就像**NextDate**函数一样。
- 强等价类测试假设变量是独立的，相应的测试用例相乘会引起冗余问题。如果存在依赖关系，则常常会生成“错误”测试用例，就像**NextDate**函数一样。
- 在发现“合适”的等价关系之前，可能需要进行多次尝试，就像**NextDate**函数例子一样。在其他情况下，存在“明显”或“自然”等价关系。如果不能肯定，最好对任何合理的实现进行再次预测。

等价类划分-练习

用等价类划分方法找出有效等价类和无效等价类，并作出测试用例。

练习**1**：

某城市电话号码由三部分组成，内容如下：第一部分是地区码，地区码可以是空白或三位数字；第二部分是前缀为非零和非一开头的三位数；第三部分是后缀为四位数。

练习1提示：

输入条件	*有效等价类*	*无效等价类*
地区码	空白（1） 三位数字（2）	一位数字（5） 二位数字（6） 多于三位数字(7) 非数字（8）
第二部分	非零开头的三位数和 非1开头的三位数(3)	以零开头的三位数(9) 以1开头的三位数(10) 非三位数（11） 非数字（12）
第三部分	后缀为四位数（4）	非四位数字（13） 非数字（14）

等价类划分-练习

练习**2**：报表日期

设某公司要打印**2001**～**2005**年的报表，报表日期为**6**位数字组成，其中，前**4**位为年份，后两位为月份。

等价类划分-练习

练习3：三角形问题

输入整数a、b、c，分别作为三角形的三条边。通过程序判断由三条边构成的三角形的类型为：等边三角形、等腰三角形、一般三角形，以及构不成三角形。

提示：

设3条边分别为A,B,C，设定一下三个输入条件：

- 如果要构成三角形的3条边，必须满足：
  A>0，B>0，C>0，且A+B>C，B+C>A，A+C>B；
- 如果是等腰的，则A=B，或B=C，或A=C；
- 如果是等边的，则A=B，且B=C，且A=C。

扩展：针对包含了直角三角形的扩展三角形问题来修改等价类集合

等价类划分-练习

输入域等价类划分（销售量）
输出域等价类划分（销售额）

练习4：佣金问题

前亚利桑那州境内的一位步枪销售商销售密苏里州制造商制造的步枪机lock、枪托stock和枪管barrel。枪机卖45美元，枪托卖30美元，枪管卖25美元。

销售商每月至少要售出一支完整的步枪，且生产限额是大多数销售商在一个月内可销售70个枪机、80个枪托和90个枪管。每访问一个镇子之后，销售商都给密苏里州步枪制造商发出电报，说明在该镇售出的枪机、枪托和枪管数量。到了月末，销售商要发出一封特殊的电报，通知-1个枪机被售出，这样制造商就知道当月的销售情况。

销售商的佣金如下计算：销售额不到（含）1000美元的部分为10%，1000（不含）到1800（含）美元的部分为15%，超过1800美元的部分为20%。佣金程序生成月份销售报告，汇总售出的枪机、枪托和枪管总数，销售商的总销售额以及佣金。

黑盒测试方法

- 黑盒测试方法
  - 等价类划分
  - 边界值分析
  - 因果图
  - 错误推测
  - 功能图法
  - …

边界值测试

NextDate例子中，等价类划分测试很难发现**X**月**31**日、**X**月**30**日和**12**月**31**日等情况下的软件失效。

人们从长期的测试工作经验得知，大量的错误是发生在输入或输出范围的边界上，而不是在输入范围的内部。因此针对各种边界情况设计测试用例，可以查出更多的错误。

边界值分析法 (BVA --Boundary Value Analysis)

边界值分析法

- “错误隐含在角落”（errors hide in the corner）
- 白盒测试中也应用到了边界值的测试思想，它不是黑盒测试的专利
- 通常边界值分析法是作为对等价类划分法的补充，这种情况下，其测试用例来自等价类的边界

等价类和边界值方法的区别：

（1）边界值不是从等价类中随便取一个数据作为代表，而是选一个或几个特定值，使这个等价类的每个边界都作为测试的目标。

（2）边界值分析不仅要考虑输入条件，而且要考虑输出条件（输出等价类）。

一般联合使用等价类划分和边界值分析两种方法。

边界值测试--边界条件

- 等价类区间之间的边缘即为边界，边界测试是指对等价类区间边界附近的数据进行测试
- 在实际的软件测试中，边界是指软件需求规格说明书中规定的操作界限的边缘条件。

边界值测试--边界条件

- 边界条件

  明确地定义在规格说明书中

- 次边界条件

  隐含在软件中，必须经过分析才能获得

- 默认、空白、空值、零值和无

- 非法、错误、不正确和垃圾数据

边界条件关键词

Numeric
Character
Position
Quantity

Speed
Location
Size

数字、字符、位置、质量、大小、速度、方位、尺寸、空间等
以上类型的边界值应该在：
最大/最小、首位/末位、上/下、最快/最慢、最高/最低、最短/最长、空/满等情况下。

关键词

First/Last
Start/Finish
Empty/Full
Slowest/Fastest
Largest/Smallest
Next-To/Farthest-From

Min/Max
Over/Under
Shortest/Longest
Soonest/Latest
Highest/Lowest

边界条件——次边界值条件

在多数情况下，边界值条件是基于应用程序的功能设计而需要考虑的因素，可以从软件的规格说明或常识中得到，也是最终用户可以很容易发现问题的。

然而，在测试用例设计过程中，某些边界值条件是不需要呈现给用户的，或者说用户是很难注意到的，但同时确实属于检验范畴内的边界条件，称为内部边界值条件或次边界值条件。

边界条件——次边界值条件

次边界值条件主要有下面几种：

- 数值的边界值：

  2的幂： 2（n） 和 2（n）-1

- 字符的边界值:

  ASCII表(UniCode)中不同类型的字符

- 其它边界值检验

字符	ASCII码值	字符	ASCII码值
空 (null)	0	A	65
空格 (space)	32	a	97
斜杠 (/)	47	Z	90
0	48	z	122
冒号 (:)	58	单引号 (‘)	96
@	64		

项	范围或值
位（bit）	0 或 1
字节（byte）	0 ~ 255
字（word）	0~65535（单字）或 0~4294967295（双字）
千（K）	1024
兆（M）	1048576
吉（G）	1073741824

边界值分析测试用例设计原则

1)、如果输入条件规定了取值范围，则应对该范围的边界内附近，恰好在边界和在边界外附近（无效等价类中）设计测试用例。

如：规定**1～5**千克邮件收费**2**元，应对**0.9**，**1**，**1.1**，**4.9**，**5**，**5.1**千克设计测试用例**(**健壮性**)**。

如果输入条件定义了数值区间(a,b)，且a,b是整数，除在a,b之间取正常点外，a,b,a-1,b-1,a+1,b+1都应被测试

边界值分析测试用例设计原则

2)、如果输入条件规定了数据的个数，则应对最少个数，最多个数，比最少个数少**1**，比最大个数多**1**等情况设计测试用例。

如：输入文件有**1～255**个记录，则应分别设计**0**，**1**，**255**，**256**个记录的输入文件的测试用例**(**健壮性**)**。

边界值分析测试用例设计原则

3)、如果程序规格说明中提到的输入或输出域是个有序的集合（如顺序文件，线性表等）。应选有序集的第一个和最后一个元素作为测试用例

4)、针对规格说明中的每个输出条件使用前面的原则。

如：计算折扣量，最低折扣为**0**元，最高为**1000**元，则要设计使它们恰好产生**0**元或**1000**元的结果。以及负值或稍大于**1000**元的结果（如果可能的话）。

边界值测试

从理论上来说，边界值可分为

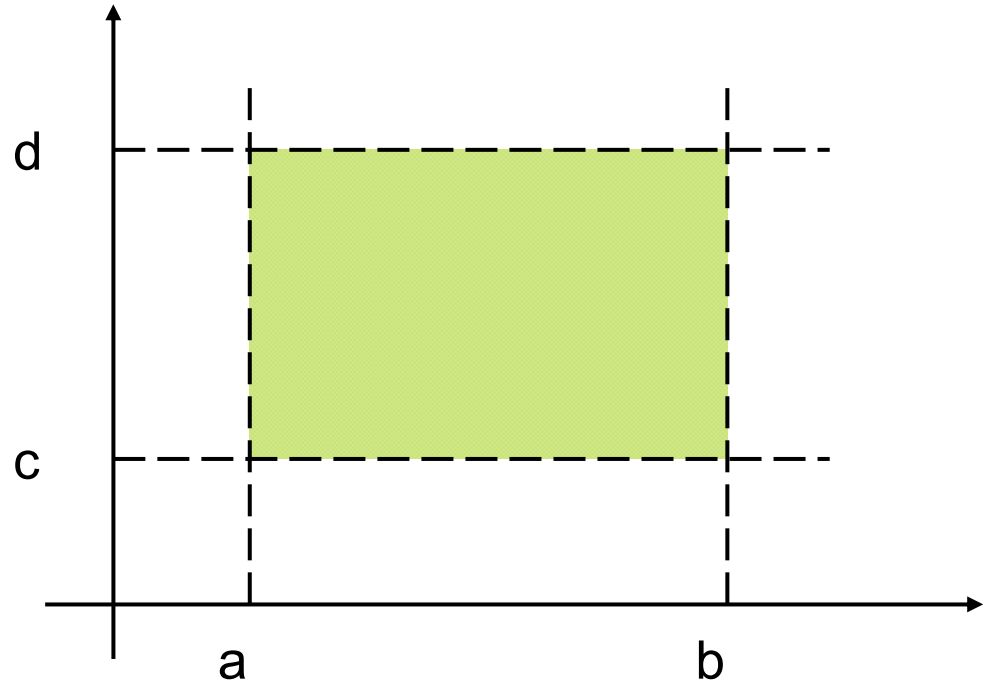
- 一般边界值：
- 一般最坏情况边界值：
- 健壮边界值：
- 健壮最坏情况边界值：

边界值测试

函数$y = f(x_1,x_2)$ 输入变量的取值范围分别为:$x_1 \in [a,b]$,

$x_2 \in [c,d]$,

如果函数F实现为一个程序，则输入两个变量x_1和x_2会有一些(可能未规定)边界：



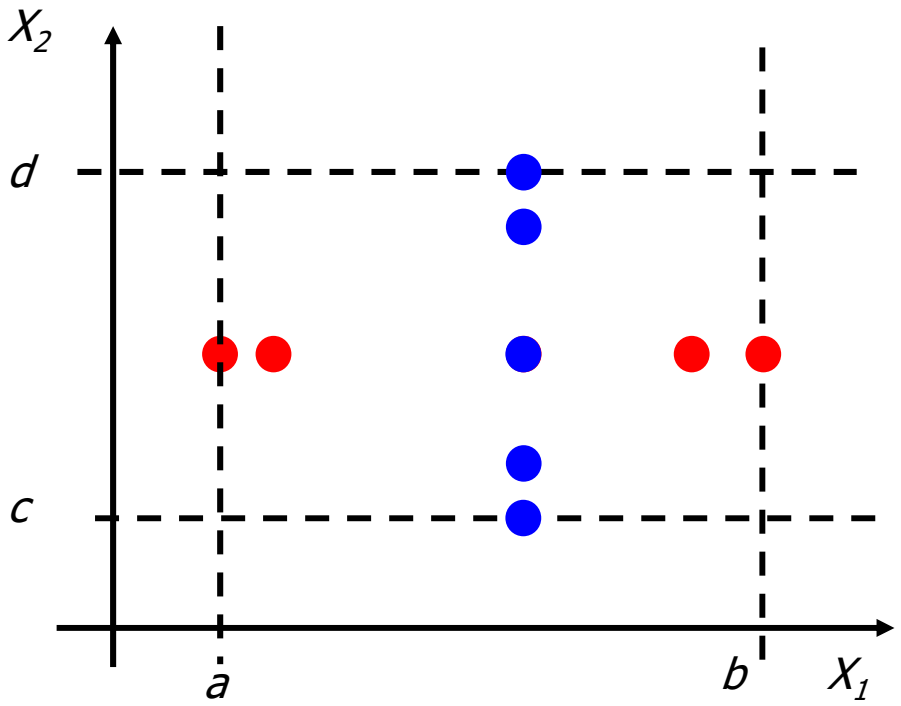
边界值分析基于一种关键假设，在可靠性理论叫做“单缺陷”假设，关注的是输入空间的边界。

边界值测试

函数$y = f(x_1,x_2)$ 输入变量的取值范围分别为:$x_1 \in [a,b]$, $x_2 \in [c,d]$,

一般边界值：

仅考虑有效区间单个变量边界值；用在最小值，略高于最小值，正常值，略低于最大值和最大值处取变量的值。如果被测变量个数为n，则测试用例个数为4n+1



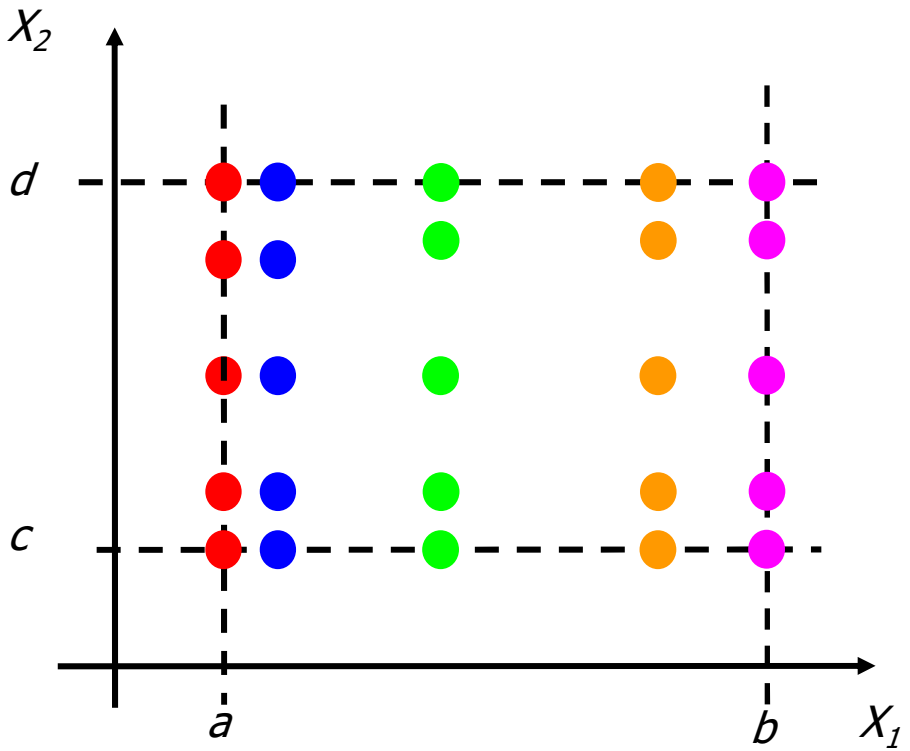
一般边界值

边界值测试

函数$y = f(x_1,x_2)$输入变量的取值范围分别为:$x_1 \in [a,b]$, $x_2 \in [c,d]$,

一般最坏情况边界值:

仅考虑有效区间多个变量边界值同时作用；用各个变量的最小值，略高于最小值，正常值，略低于最大值和最大值的笛卡尔积集。如果被测变量个数为n，则测试用例个数为5^n



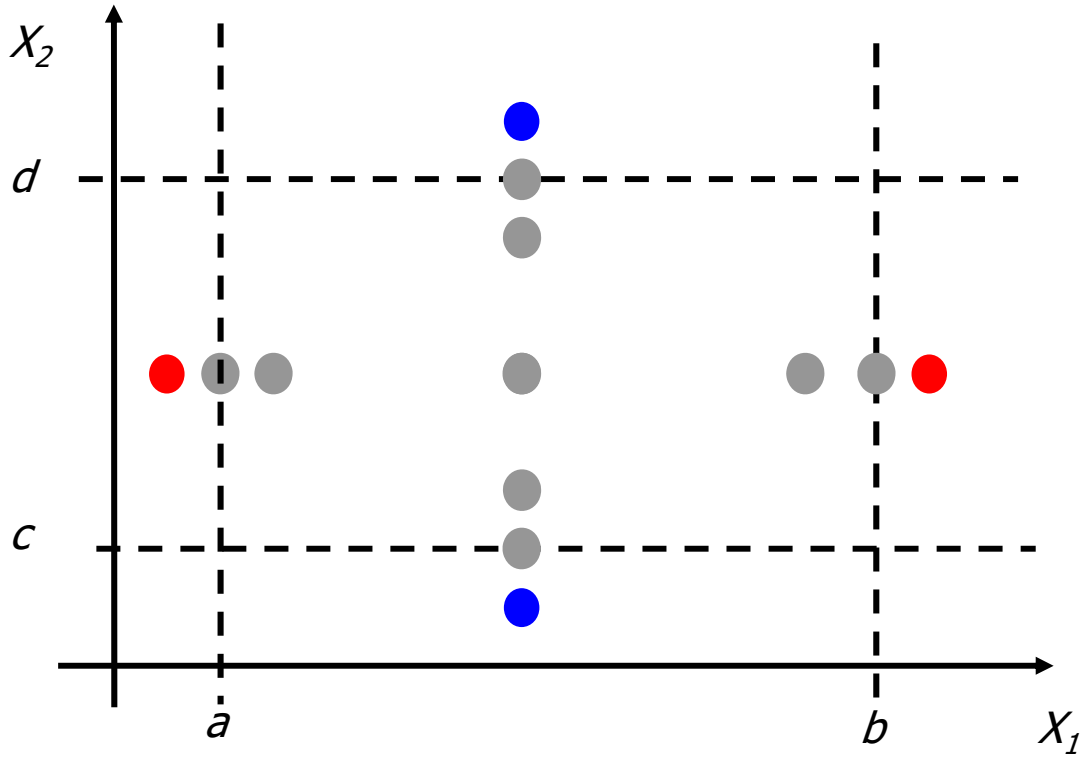
一般最坏情况边界值

边界值测试

函数$y = f(x_1,x_2)$输入变量的取值范围分别为:$x_1 \in [a,b]$, $x_2 \in [c,d]$,

健壮边界值:

同时考虑有效区间和无效区间单个变量边界值；除了在最小值，略高于最小值，正常值，略低于最大值和最大值处取变量的值，还要在略超过最大值以及略小于最小值之处值。如果被测变量个数为n，则测试用例个数为6n+1



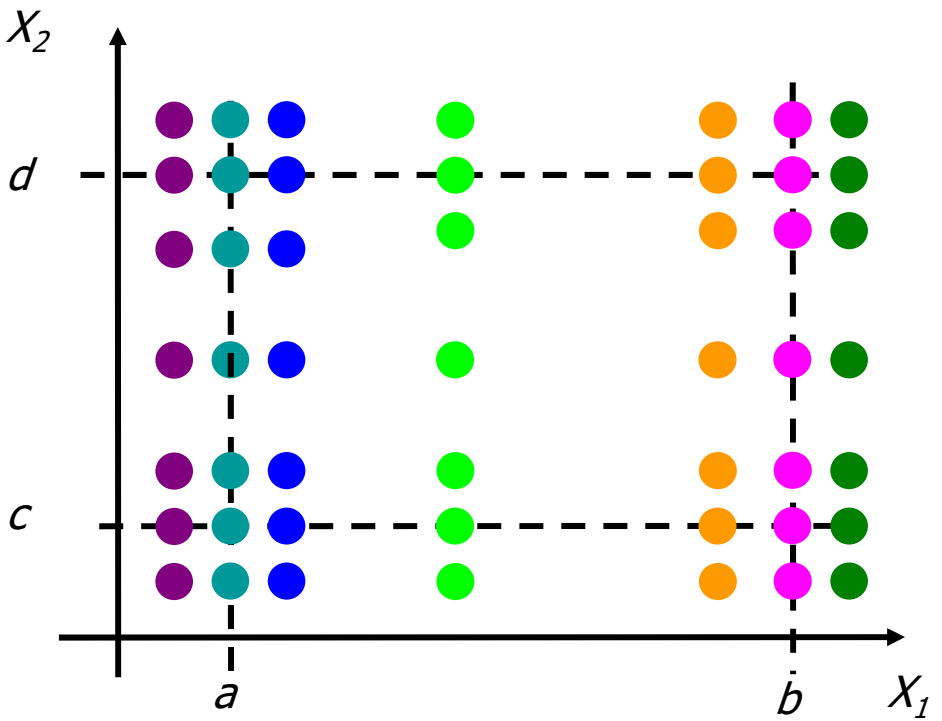
健壮边界值

边界值测试

函数$y = f(x_1, x_2)$ 输入变量的取值范围分别为:$x_1 \in [a,d]$, $x_2 \in [e,g]$,

健壮最坏情况边界值:

同时考虑有效区间和无效区间多个变量边界值同时作用；用各个变量的略小于最小值，最小值，略高于最小值，正常值，略低于最大值，最大值和略超过大值的完全组合。如果被测变量个数为n，则测试用例个数为7^n



健壮最坏边界值

举例

三角形问题有三个输入（整数），即三条边a、b、c，其取值范围为：

$1 \leqslant a \leqslant 200$

$1 \leqslant b \leqslant 200$

$1 \leqslant c \leqslant 200$

(1) 三角形问题的一般边界值测试用例

a = {1，2，100，199，200}
b = {1，2，100，199，200}
c = {1，2，100，199，200}

用例	A	B	C	预期输出
1	100	100	1	等腰三角形
2	100	100	2	等腰三角形
3	100	100	100	等边三角形
4	100	100	199	等腰三角形
5	100	100	200	非三角形
6	100	1	100	等腰三角形
7	100	2	100	等腰三角形
8	100	199	100	等腰三角形
9	100	200	100	非三角形
0	1	100	100	等腰三角形
11	2	100	100	等腰三角形
12	199	100	100	等腰三角形
13	200	100	100	非三角形

（2）三角形问题的一般最坏情况边界值测试用例

a = {0，1，2，100，199，200，201}
b = {0，1，2，100，199，200，201}
c = {0，1，2，100，199，200，201}

最坏情况测试用例集合 = a × b × c =
{1，2，100，199，200} ×
{1，2，100，199，200} ×
{1，2，100，199，200}

（3）三角形问题的健壮边界值测试用例

a = {1，2，100，199，200}
b = {1，2，100，199，200}
c = {1，2，100，199，200}

（3）三角形问题的健壮边界值测试用例

用例	A	B	C	预期输出
1	100	100	0	非法输入
2	100	100	1	等腰三角形
3	100	100	2	等腰三角形
4	100	100	100	等边三角形
5	100	100	199	等腰三角形
6	100	100	200	非三角形
7	100	100	201	非法输入
8	100	0	100	非法输入
9	100	1	100	等腰三角形
10	100	2	100	等腰三角形
11	100	199	100	等腰三角形
12	100	200	100	非三角形
13	100	201	100	非法输入
14	0	100	100	非法输入
15	1	100	100	等腰三角形
16	2	100	100	等腰三角形
17	199	100	100	等腰三角形
18	200	100	100	非三角形
19	201	100	100	非法输入

（4）三角形问题的一般最坏情况边界值测试用例

a = {0，1，2，100，199，200，201}
b = {0，1，2，100，199，200，201}
c = {0，1，2，100，199，200，201}

最坏情况测试用例集合 = a × b × c =
{0，1，2，100，199，200，201} ×
{0，1，2，100，199，200，201} ×
{0，1，2，100，199，200，201}

关于使用边界值测试

1. 适合使用边界值分析的情况：

如果被测程序是多个独立变量的函数，如果变量受物理量的限制，则很适合。

- 举一个例子，菲尼克斯的航空港国际机场1992年6月26日被迫关闭，因为空气温度达到122° F。飞行员在起飞之前不能设置特定设备：该设备能够接受的最大空气温度是120° F 。

2. 对于输出变量的边界上同样可以考虑使用边界值测试

3. 虽然边界值测试是以黑盒测试方法的面貌出现，但是其思想不仅可以用于测试功能，同样可以将边界值测试的思想用于测试代码

4. 不适合使用： 边界值分析对布尔变量没有什么意义

边界值测试-练习

练习1：有函数**f(x,y,z)**，其中**x∈[1900,2100]**，**y∈[1,12]**，**z∈[1,31]**的。

请写出该函数采用边界值分析法设计的测试用例。

练习2：**NextDate**是一个有三个变量（月份、日期和年）的函数，函数返回输入日期后面的那个日期。

变量月、日和年都具是整数，且满足以下条件：

1 ≤ 月份 **≤ 12**

1 ≤ 日期 **≤ 31**

1812 ≤ 年 **≤ 2012**

黑盒测试方法

- 黑盒测试方法
  - 等价类划分
  - 边界值分析
  - 因果图 和 决策表）
  - 错误推测
  - 功能图法
  - …

因果图

产生背景

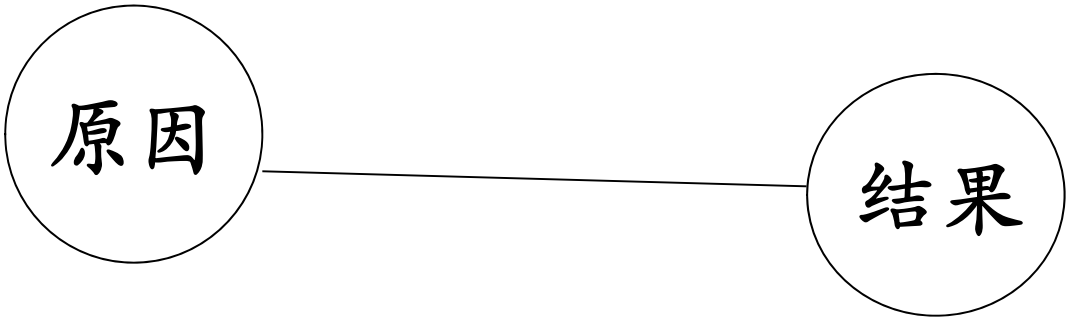
等价类划分和边界值分析法都是着重考虑输入条件，没有考虑输入之间的组合、制约关系。这样虽然各种输入条件可能出错的情况已经测试到了，但多个输入条件组合起来可能出错的情况却被忽视了。

因果图

定义

考虑输入条件之间的联系、各种组合，相应产生多个动作来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。

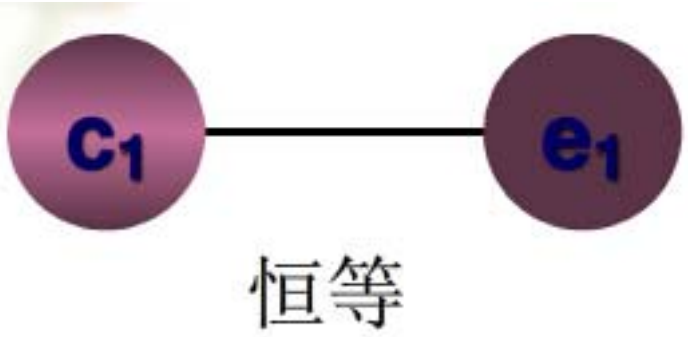
因果图



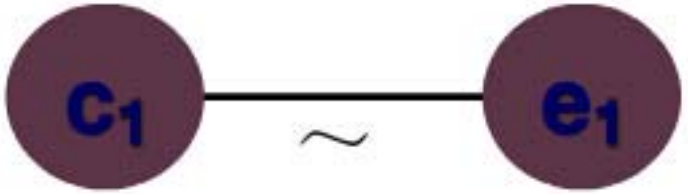
- 通常在因果图中用Ci表示原因，用Ei表示结果，各结点表示状态，可取值“0”或“1”。“0”表示某状态不出现，“1”表示某状态出现。

因果图

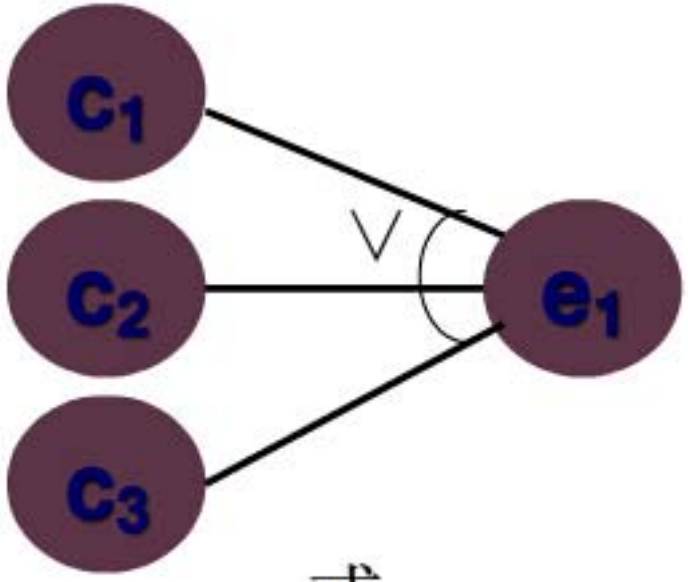
原因与结果之间的关系：



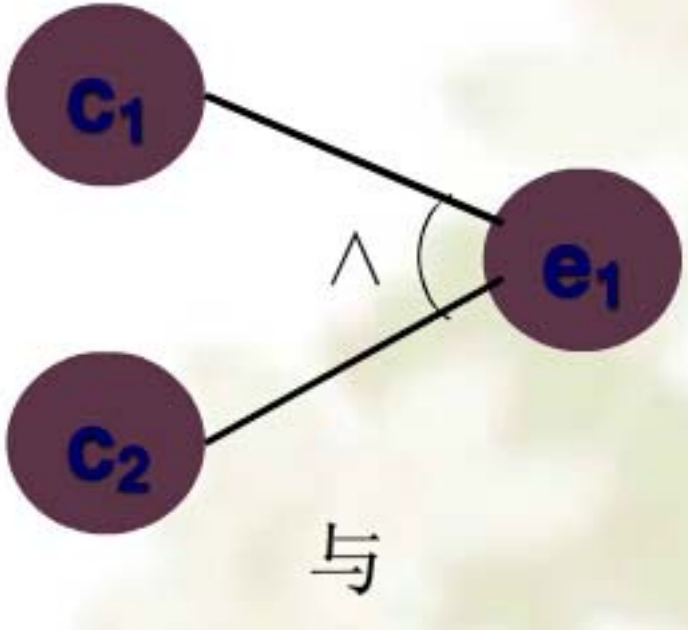
恒等



非



或



与

因果图中的基本符号

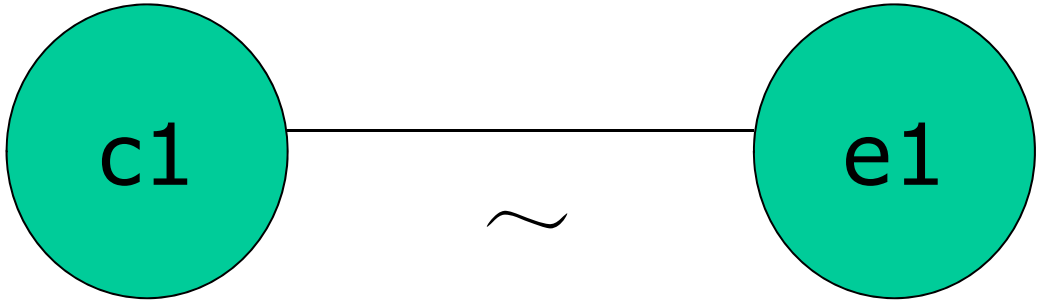
原因与结果之间的关系：

- 恒等：若c1是1，则e1也为1，否则e1为0；

因果图中的基本符号

原因与结果之间的关系：

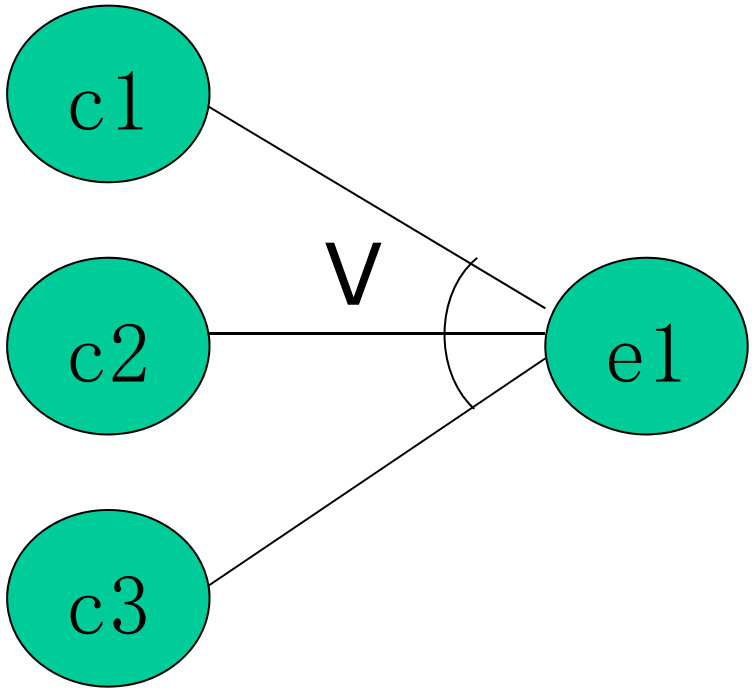
- 非：若c1是1，则e1为0，否则e1为1；用符号“~”表示。



因果图中的基本符号

原因与结果之间的关系：

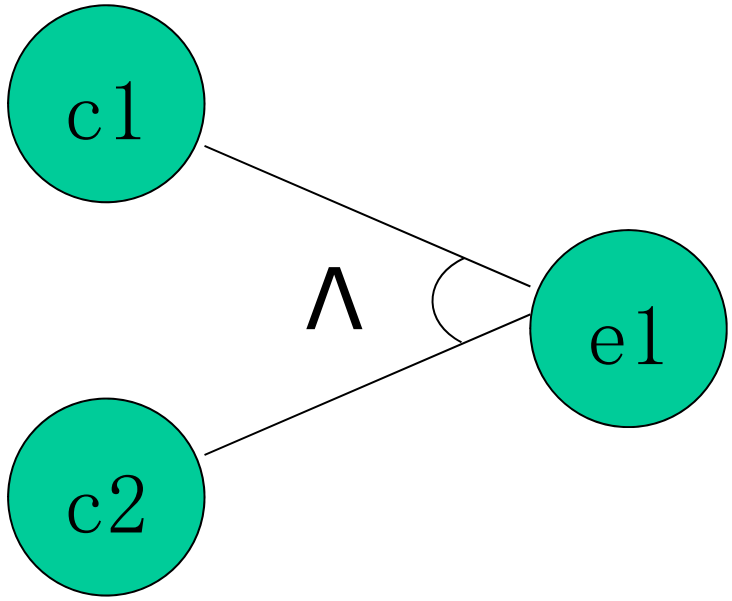
- 或：若c1或c2或c3是1，则e1是1，否则e1为0，“或”可有任意个输入；用符号“V”表示。



因果图中的基本符号

原因与结果之间的关系：

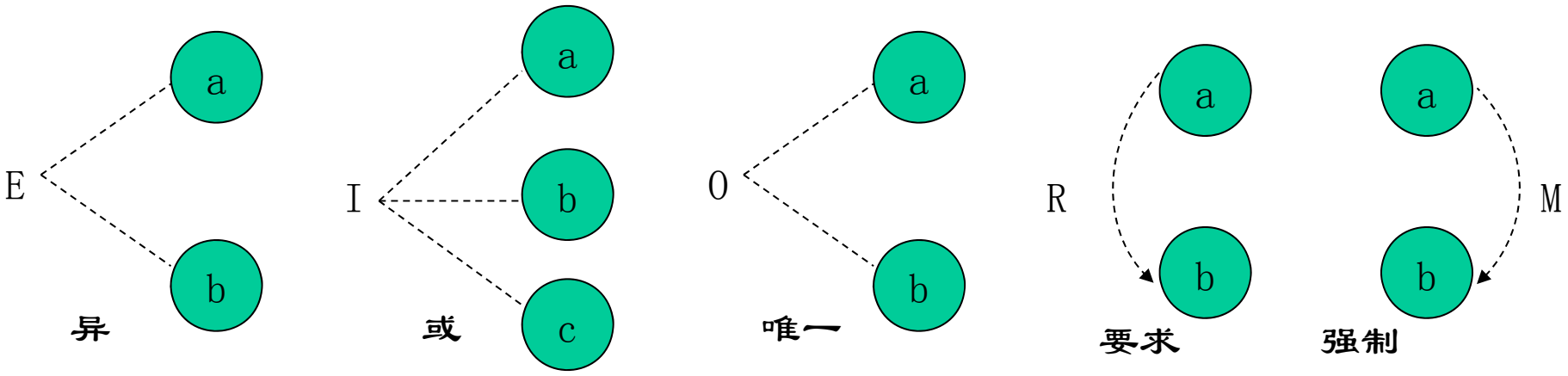
与：若c1和c2都是1，则e1为1，否则e1为0，“与”也可有任意个输入。用符号“∧”表示。



因果图中的基本符号

输入状态之间的关系：

在实际问题当中输入状态相互之间还可能存在某些依赖关系，称为“约束”

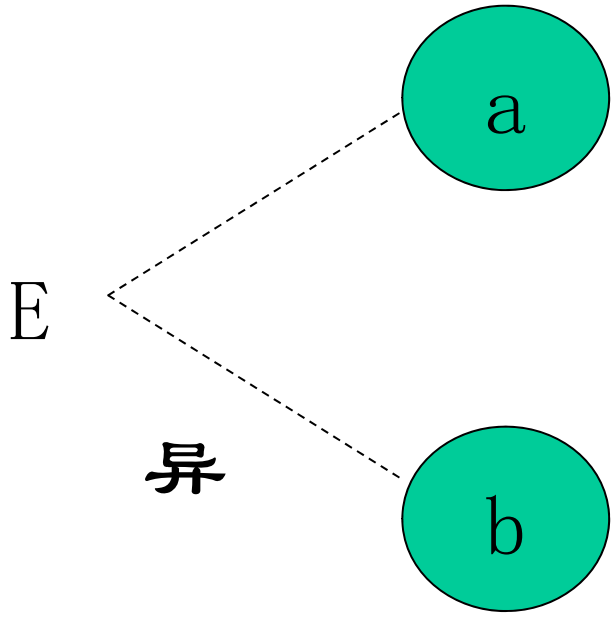


(a) (b) (c) (d) (e)

因果图中的基本符号

输入状态之间的关系：

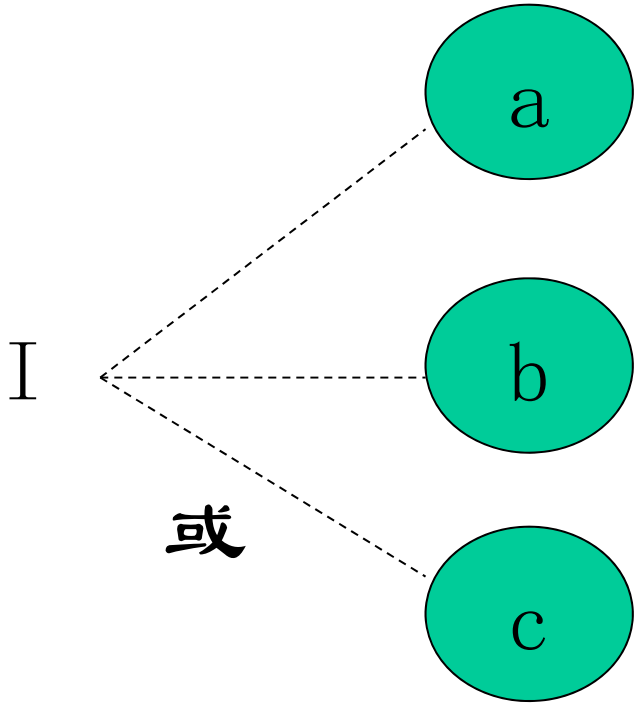
E约束（异，排他）：a和b中最多有一个可能为1，即a和b不能同时为1；



因果图中的基本符号

输入状态之间的关系：

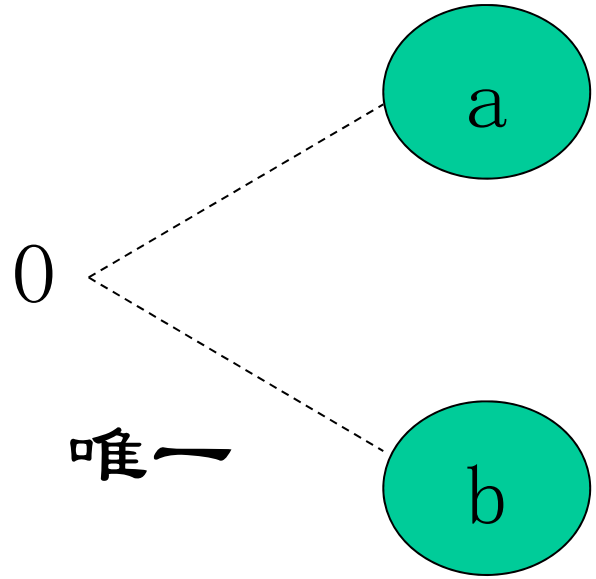
I约束（或，包含）：a、b、c中至少有一个必须是1，即a、b、c不能同时为0；



因果图中的基本符号

输入状态之间的关系：

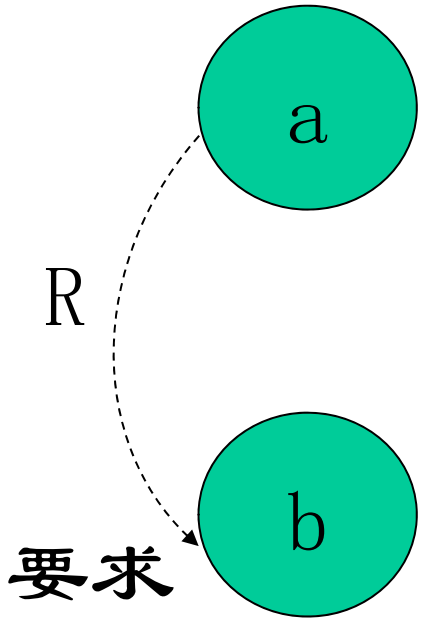
O约束（唯一）：**a**和**b**必须有一个且仅有一个为**1**；



因果图中的基本符号

输入状态之间的关系：

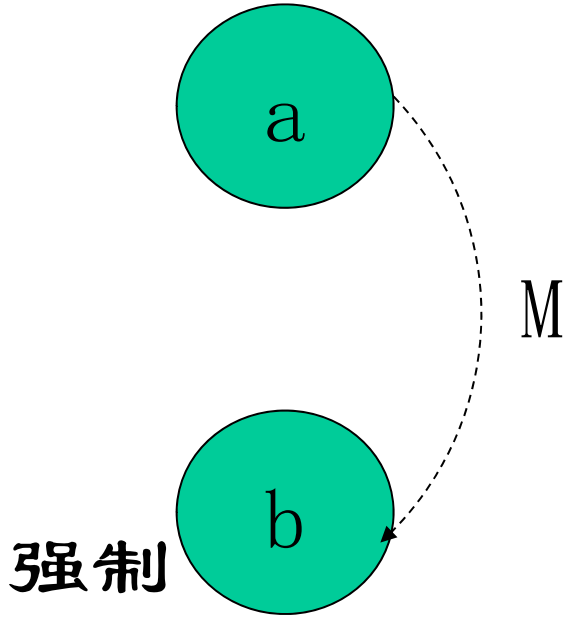
R约束（要求）：a是1时，b必须是1；



因果图中的基本符号

输出状态之间的关系：

M约束（强制）：若结果a是1，则结果b强制为0。



因果图

用例设计步骤

①分析并列出需求或规格说明书中原因和结果。

②找出关系，将原因和结果连接成因果图。

③把因果图转换成判定表。

④把判定表中每一列表示的情况写成测试用例。

- 因：输入条件
- 果：输出结果或者程序状态的改变
- 关系：原因与结果的关系，原因与原因之间的关系

因果图

实例1

- 某个软件的规格说明书规定：第一个字符必须是#或*，第二个字符必须是一个数字字符，在此情况下进行约定的文件修改操作。
- 如果第一个字符不是#或*，则给出信息N；
- 如果第二个字符不是数字，则给出信息M。

因果图

实例1

步骤一：分析并列出需求或规格说明书中原因和结果。

原因：c1－第一个字符是“#”
c2－第一个字符是“*”
c3－第二个字符是一个数字

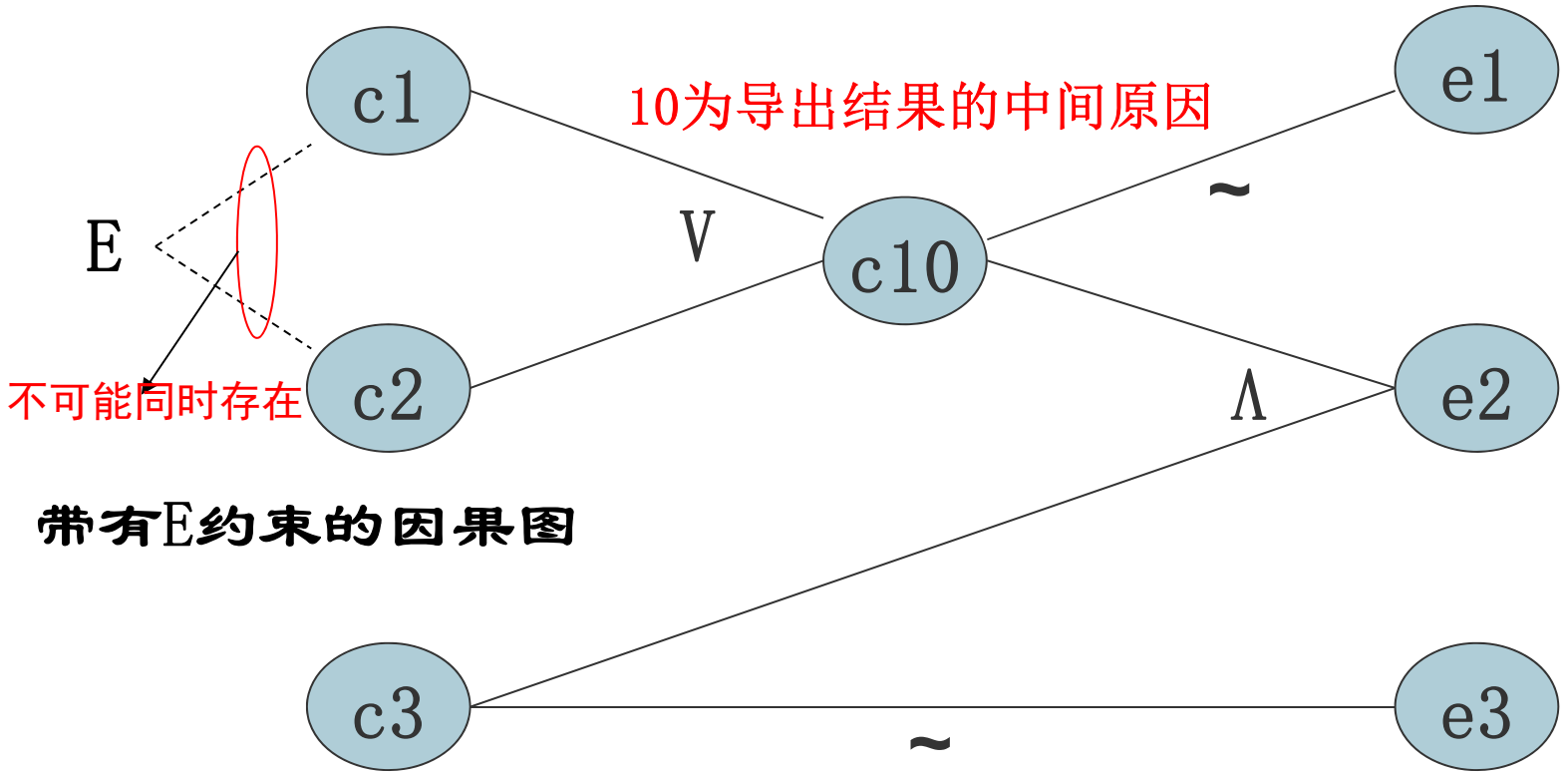
结果：e1－给出信息N
e2－修改文件
e3－给出信息M

因果图

原因：c1－第一个字符是“#”
c2－第一个字符是“*”
c3－第二个字符是一个数字
结果：e1－给出信息N
e2－修改文件
e3－给出信息M

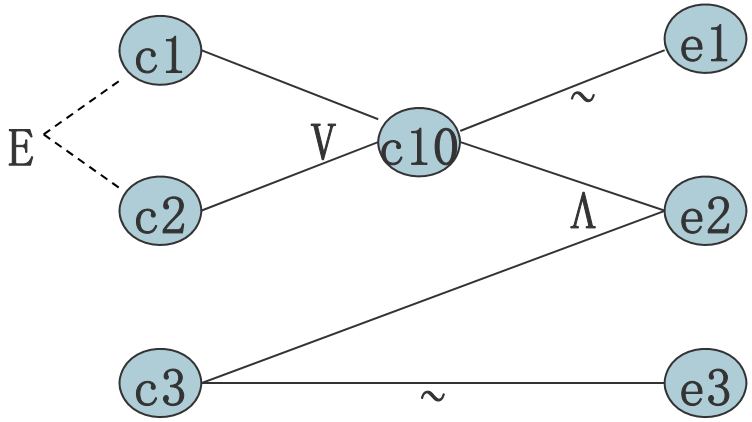
实例1

步骤二：找出关系，将原因和结果连接成因果图。



带有E约束的因果图

因果图



实例1

步骤三：把因果图转换成判定表。

	1	2	3	4	5	6	7	8
C1 C2 C3 C	1 1 1	1 1 0	1 0 1 1	0 1 1 1	1 0 0 1	0 1 0 1	0 0 1 0	0 0 0 0
E1 E2 E3 不可能	 √	 √	 √ 	 √ 	 √ 	 √ 	√ 	√ √
测试用例			#3	*4	#B	*M	C2	CM

因果图

实例1

步骤四：把判定表中每一列写成测试用例。

- 最左边两列，原因**c1**和**c2**同时为**1**不可能，排除掉，根据表可设计出**6**个测试用例。
  - **Test1**：输入数据－**#3**　预期输出－－修改文件
  - **Test2**：输入数据－***4**　预期输出－－修改文件
  - **Test3**：输入数据－**#B**　预期输出－－给出信息**N**
  - **Test4**：输入数据－***M**　预期输出－－给出信息**N**
  - **Test5**：输入数据－**C2**　预期输出－－给出信息**M**
  - **Test6**：输入数据－**CM**　预期输出－－给出信息**M**和**N**

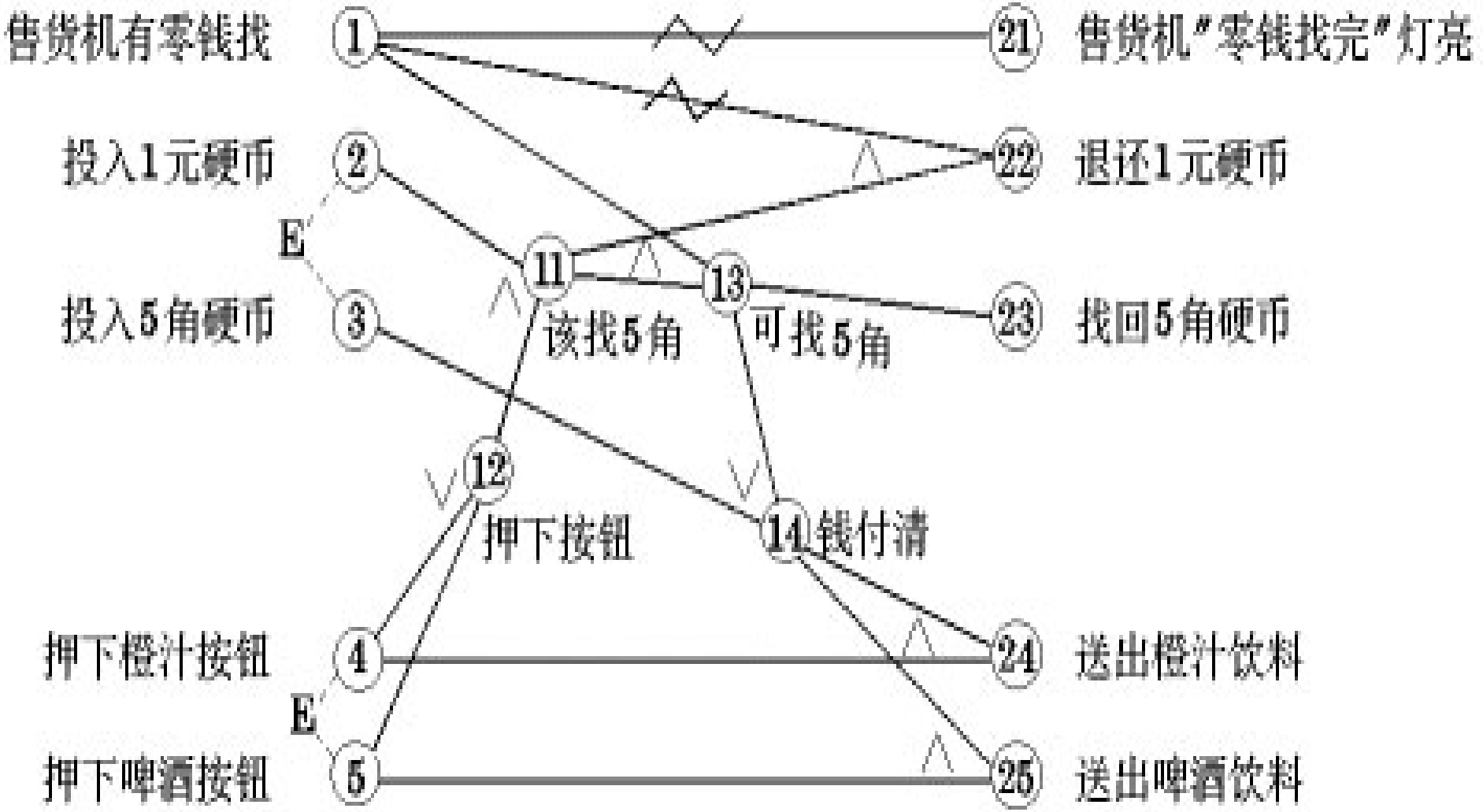
因果图

实例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：
1. 售货机有零钱找
2. 投入1元硬币
3. 投入5角硬币
4. 押下橙汁按钮
5. 押下啤酒按钮
结果：
21. 售货机〖零钱找完〗灯亮
22. 退还1元硬币
23. 退还5角硬币
24. 送出橙汁饮料
25. 送出啤酒饮料

2）画出因果图，如图所示。



中间结点：

11. 投入**1**元硬币且押下饮料按钮

12. 押下〖橙汁〗或〖啤酒〗的按钮

13. 应当找**5**角零钱并且售货机有零钱找

14. 钱已付清

根据因果图建立的决策表

序号		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2
条件	①	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	②	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
	③	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
	④	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
	⑤	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
中间结果	⑪						1	1	0		0	0	0		0	0	0						1	1	0		0	0	0		0	0	0
	⑫						1	1	0		1	1	0		1	1	0						1	1	0		1	1	0		1	1	0
	⑬						1	1	0		0	0	0		0	0	0						0	0	0		0	0	0		0	0	0
	⑭						1	1	0		1	1	1		0	0	0						0	0	0		1	1	1		0	0	0
结果	㉑						0	0	0		0	0	0		0	0	0						1	1	1		1	1	1		1	1	1
	㉒						0	0	0		0	0	0		0	0	0						1	1	0		0	0	0		0	0	0
	㉓						1	1	0		0	0	0		0	0	0						0	0	0		0	0	0		0	0	0
	㉔						1	0	0		1	0	0		0	0	0						0	0	0		1	0	0		0	0	0
	㉕						0	1	0		0	1	0		0	0	0						0	0	0		0	1	0		0	0	0
测试用例							Y	Y	Y		Y	Y	Y		Y	Y							Y	Y	Y		Y	Y	Y		Y	Y	

使用因果图法的优点

（ 1）考虑到了输入情况的各种组合以及各个输入情况之间的相互制约关系。

（ 2）能够帮助测试人员按照一定的步骤，高效率的开发测试用例。

（3）因果图法是将自然语言规格说明转化成形式语言规格说明的一种严格的方法，可以指出规格说明存在的不完整性和二义性。

不同测试方法的工作量比较

工作量主要体现在设计测试和实现测试用例两方面。
设计测试的工作量：
边界值分析 < 等价类分析 < 因果图/决策表

实现测试用例的工作量（也是测试用例的数量）：
边界值分析 > 等价类分析 >因果图/决策表

在实践中，需要对于实际情况进行平衡，达到两个方面的平衡。

因果图

练习1

- 一图书销售系统，其中一加工为“优惠处理”，条件是：顾客的营业额必须大于1000元。同时信誉好，或者虽然信誉不好，但是20年以上的老主顾。

- 要求使用因果图法设计测试用例

因果图

练习2

- 在地图查找项中，输入完全地址和模糊地址能查找出地址；输入错误或不输入地址则提示错误信息或不显示。

- 要求画出因果图和判定表

因果图

练习3

- 有一个处理单价为1元5角的盒装饮料的自动售货机软件。若投入1元5角硬币，按下“可乐”，“雪碧”或“红茶”按钮，相应的饮料就送出来。若投入的是两元硬币，在送出饮料的同时退还5角硬币。

练习4

某规格说明规定：输入的第一列字符必须是A或B，第二列字符必须是一个数字。第一、二列都满足条件时执行操作H；如果第一列字符不正确，则给出信息L；如果第二列字符不正确，则给出信息R。

根据上述要求画出因果图，并设计测试用例。

分析:
原因:
1—第一列字符是A
2—第一列字符是B
3—第二列字符是一数字
结果:
21—执行操作H
22—给出信息L
23—给出信息R

练习5

- 奖金计算软件
  - 该软件可以计算某公司的年终奖，该公司员工分为普通员工和管理人员；
  - 员工表现分为普通、优秀和特殊贡献（普通、优秀员工都可以有特殊贡献，不同组合所得工资是不同的）
  - 根据员工的分类和表现，将奖金分为1类奖金，2类奖金，……

- 使用该软件时，输入员工的种类和表现，就会输出相应的奖金类别。
- 请为该软件设计测试用例
- 提示：
  - 这道题概括比较笼统，不涉及具体细节，只是一个总体的框架。

- 解题思路：

  (1)找到所有的输入条件（原因）和输出条件（结果），并为其编号

  – 输入条件
    - 员工类别：普通员工A1、管理人员A2
    - 员工表现：普通B1、优秀B2、特殊贡献B3
  – 输出条件
    - 奖金类别：1类奖金C1、2类奖金C2.....

- （2）分析输入条件之间的关系，根据需求我们知道，A1和A2是互斥的（一个人不可能既是普通员工又是管理人员），B1和B2是互斥的（一个人不可能既表现普通又表现优秀），B1和B3，B2和B3可以同时满足。由此分析，我们可以列出所有的输入条件排列组合：

- 普通员工
  - A1+B1→C1
  - A1+B2 →C2
  - A1+B1+B3 →C3
  - A1+B2+B3 →C4
- 管理人员
  - A2+B1→C5
  - A2+B2 →C6
  - A2+B1+B3 →C7
  - A2+B2+B3 →C8

- （3）根据对输入输出条件的分析，画出因果图

- （4）根据对输入输出条件的分析，编写测试用例

用例编号	输　入	输 出
1	普通员工，表现普通	1类奖金
2	普通员工，表现优秀	2类奖金
3	普通员工，表现普通，且有特殊贡献	3类奖金
4	普通员工，表现优秀，且有特殊贡献	4类奖金
5	管理人员，表现普通	5类奖金
6	管理人员，表现优秀	6类奖金
7	管理人员，表现普通，且有特殊贡献	7类奖金
8	管理人员，表现优秀，且有特殊贡献	8类奖金