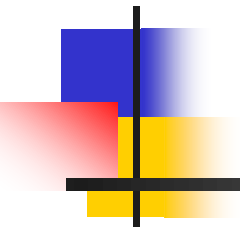


第3章 黑盒测试

张圣林
zhangsl@nankai.edu.cn
http://nkcs.iops.ai/courses/softwaretesting/

第3章 黑盒测试

- 3.1 黑盒测试的基本概念
- 常用黑盒测试方法
  - 3.2 等价类划分
  - 3.3 边界值分析法
  - 3.4 因果图法
  - 3.5 决策表法
  - 3.6 正交实验设计法
  - 3.7 状态图法
- 3.8 黑盒测试方法的比较与选择
- 3.9 黑盒测试工具

3.1 黑盒测试的基本概念

- 黑盒测试：不考虑内部实现，依据输入输出设计测试
- 白盒测试：依据内部的实现逻辑设计测试

例子：Compare(int a, int b)

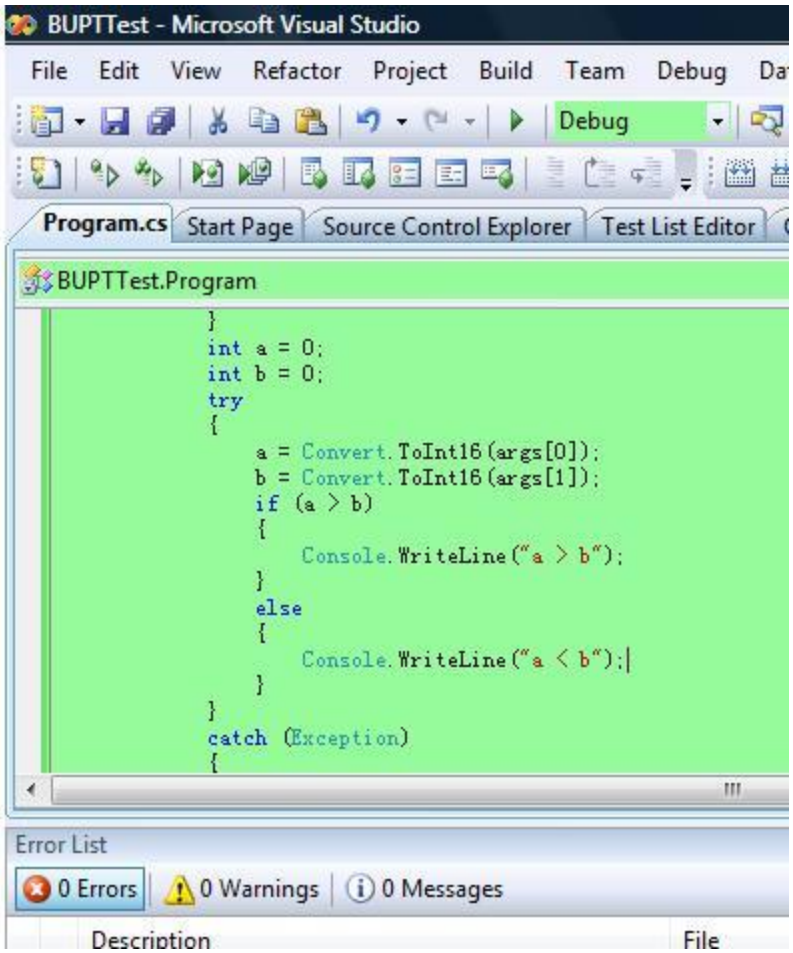
- 功能描述：
  - 输入：int a, int b
  - 输出：打印 “a<b”, “a=b”, or “a>b”

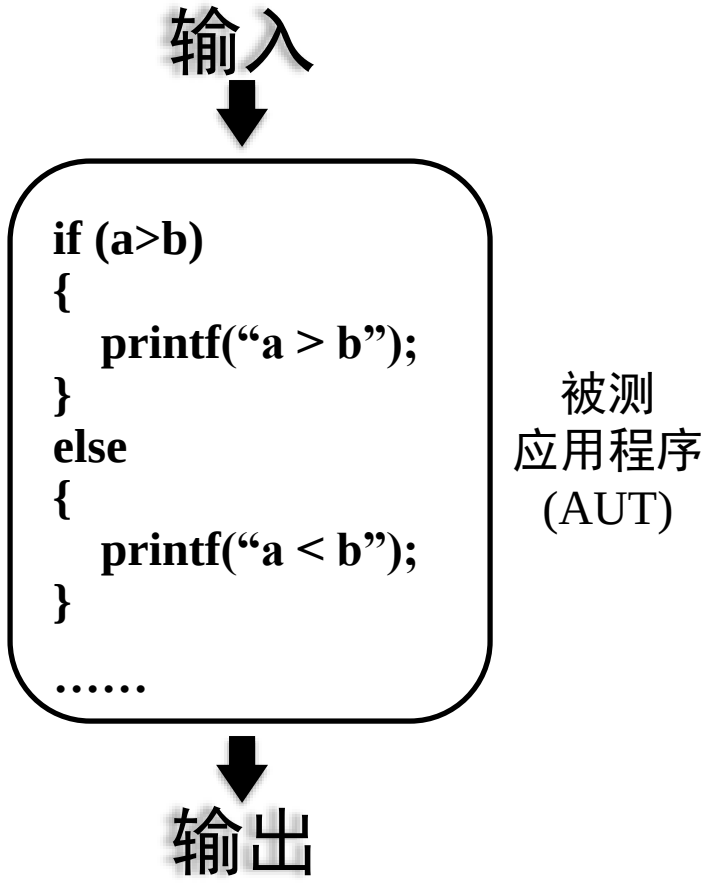
- 黑盒测试方法

输入 Input		期望输出 Expected Output	实际输出 Actual Output	测试通过？
a=0	b=1	a < b	a < b	**PASS**
a=0	b=0	a = b	**a < b**	**FAIL**
a=0	b=-1	a > b	a > b	**PASS**

例子：Compare(int a, int b)

- 白盒测试方法





灰盒测试

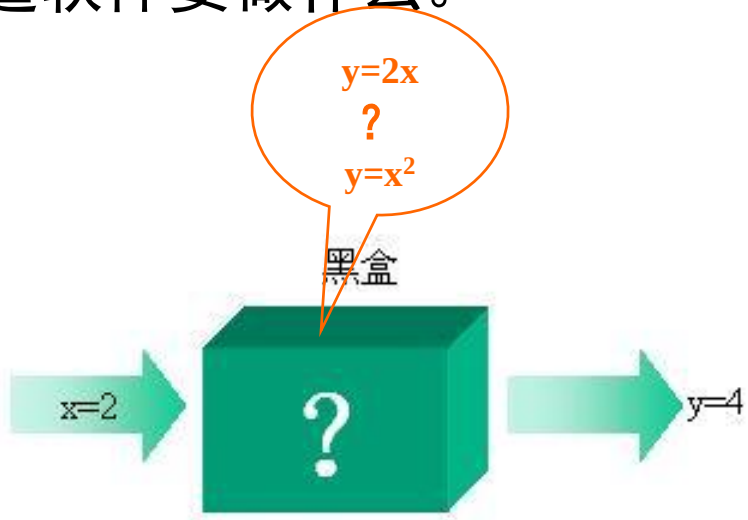
- 实际工作中，需要结合黑盒测试和白盒测试

测试用例

- 为实施测试而向被测试系统提供的输入数据、操作或各种环境设置以及期望结果的一个特定的集合。
- 解决要测什么、怎么测和如何衡量的问题
  - 测试数据、操作步骤、预期结果
- 测试用例的用途
  - 核实需求
  - 监督过程
  - 评估结果
  - 准确回归
  - 防止遗漏
  - 提高效率

3.1 黑盒测试的基本概念

黑盒测试是从一种从软件外部对软件实施的测试，也称功能测试或基于规格说明的测试。其基本观点是：任何程序都可以看作是从输入定义域到输出值域的映射，这种观点将被测程序看作一个打不开的黑盒，黑盒里面的内容(实现)是完全不知道的，只知道软件要做什么。



3.1 黑盒测试的基本概念

黑盒测试是从用户观点出发的测试，其目的是尽可能发现软件的外部行为错误。

- 检测软件功能能否按照需求规格说明书的规定正常工作，是否有功能遗漏；
- 检测是否有人机交互错误，是否有数据结构和外部数据库访问错误，是否能恰当地接收数据并保持外部信息（如数据库或文件）等的完整性；
- 检测行为、性能等特性是否满足要求等；
- 检测程序初始化和终止方面的错误等。

3.1 黑盒测试的基本概念

黑盒测试着眼于软件的外部特征，通过上述方面的检测，确定软件所实现的功能是否按照软件规格说明书的预期要求正常工作。

两个显著的优点

- 黑盒测试与软件具体实现无关，所以如果软件实现发生了变化，测试用例仍然可以使用；
- 设计黑盒测试用例可以和软件实现同时进行，因此可以压缩项目总的开发时间。

3.1 黑盒测试的基本概念

- 黑盒测试：计算器
  - 利用黑盒测试方法查出软件中所有的故障，只能采用穷举输入测试，即把所有可能的输入全部都用作测试输入。

3.1 黑盒测试的基本概念

- 黑盒测试：计算器测试用例
  - 1+1=2，1+2=3，……，1+4294967295
  - 2+1，……，4294967295+4294967295
  - 小数：1.0+0.1，1.0+1.1，……
  - 5<退格键>7+2
  - 非法输入：l+a.Z+I，1a1+2b2
  - 3个数相加、4个数相加
  - 减法、乘法、除法、求平方根、百分数和倒数
  - 无穷无尽……

3.1 黑盒测试的基本概念

穷举输入测试是不现实的。这就需要我们认真研究测试方法，以便能开发出尽可能少的测试用例，发现尽可能多的软件故障。

常用的黑盒测试方法有等价类划分、边界值分析、决策表测试、因果图、状态图、正交法、大纲法等，每种方法各有所长，我们应针对软件开发项目的具体特点，选择合适的测试方法，有效地解决软件开发中的测试问题。

3.2 等价类划分

- 等价类划分法是一种典型的黑盒测试方法，它完全不考虑程序的内部结构，只根据程序规格说明书对输入范围进行划分，把所有可能的输入数据，即程序输入域划分为若干个互不相交的子集，称为等价类，然后从每个等价类中选取少数具有代表性的数据构成测试用例，然后进行测试。

3.2.1 等价类划分方法

所谓等价类是指输入域的某个互不相交的子集合，所有等价类的并便是整个输入域

1.划分等价类

- 有效等价类
  - 检验程序是否实现了规格说明预先规定的功能和性能。
- 无效等价类
  - 检查软件功能和性能的实现是否有不符合规格说明要求的地方。

3.2.1 等价类划分方法

2. 常用的等价类划分原则
   - 按区间划分
   - 按数值划分
   - 按数值集合划分
   - 按限制条件或规则划分
   - 细分等价类
- 同样，可按输出条件，将输出域划分为若干等价类

3.2.1 等价类划分方法

- 列出所有划分出的等价类表

输入条件	有效等价类	无效等价类

3.2.1 等价类划分方法

3. 等价类划分测试用例设计

在设计测试用例时应同时考虑有效等价类和无效等价类测试用例的设计。根据等价类表设计测试用例，具体步骤如下

（1）确认输入需求及输入项。

（2）为每个等价类规定一个唯一的编号。

（3）设计一个新的测试用例，尽可能多地覆盖尚未被覆盖的有效等价类，重复这一步，直到测试用例覆盖了所有的有效等价类（包括输出域）。

(4) 设计一个新的测试用例，使其覆盖并且只覆盖一个还没有被覆盖的无效等价类。重复这一步，直至测试用例覆盖了所有的无效等价类。

3.2.2 等价类划分法的测试运用

1.三角形问题的等价类测试

【例2.1】 输入三个整数a、b和c分别作为三角形的3条边，且三个整数介于1-100之间，通过程序判断由这3条边构成的三角形类型是：等边三角形、等腰三角形、一般三角形或非三角形（不能构成一个三角形）。

3.2.2 等价类划分法的测试运用

更详细地描述：

输入3个整数a、b和c分别作为三角形的三条边，要求a、b和c必须满足以下条件：

1、整数

2、3个数

3、边长大于等于1，小于等于100

4、任意两边之和大于第三边

输出为5种情况之一：

如果不满足条件1、2、3，则程序输出为“输入错误”。
如果不满足条件4，则程序输出为“非三角形”。
如果三条边相等，则程序输出为“等边三角形”。
如果恰好有两条边相等，则程序输出为“等腰三角形”。
如果三条边都不相等，则程序输出为“一般三角形”。

输入域等价类划分

有效等价类	编号	无效等价类	编号
a为整数	1	a非整数	13
b为整数	2	b非整数	14
c为整数	3	c非整数	15
三个数	4	大于3	16
		小于3	17
1<=a<=100	5	小于1	18
		大于100	19
1<=b<=100	6	小于1	20
		大于100	21
1<=c<=100	7	小于1	22
		大于100	23

输出域等价类划分

有效等价类	编号
非三角形	9
等边三角形	10
等腰三角形	11
一般三角形	12

覆盖有效等价类的测试用例

输入数据	输出结果	覆盖的等价类
(30,30,30)	等边三角形	1,2,3,4,5,6,7,10
(30,30,20)	等腰三角形	1,2,3,4,5,6,7,11
(30,40,50)	一般三角形	1,2,3,4,5,6,7,12
(30,40,90)	非三角形	1,2,3,4,5,6,7,9

覆盖无效等价类的测试用例

无效等价类	输入数据	输出结果	覆盖的等价类
a非整数	(11.1,10,10)	输入错误	13
b非整数	(10,4.4,10)	输入错误	14
c非整数	(9,10,2.3)	输入错误	15
大于3	(10,10,10,4)	输入错误	16
小于3	(10,10)	输入错误	17
a小于1	(0,10,10)	输入错误	18
a大于100	(101,60,60)	输入错误	19
b小于1	(10,-1,10)	输入错误	20
b大于100	(60,110,60)	输入错误	21
c小于1	(10,10,0)	输入错误	22
c大于100	(60,60,110)	输入错误	23

3.2.2 等价类划分法的测试运用

2.保险公司人寿保险保费计算程序的等价类测试

【例2.2】 某保险公司人寿保险的保费计算方式为：

保费＝投保额×保险费率

其中，保险费率根据年龄、性别、婚姻状况和抚养人数的不同而有所不同，体现在不同年龄、性别、婚姻状况和抚养人数，点数设定不同，10点及10点以上保险费率为0.6%，10点以下保险费率为0.1%；而点数又是由投保人的年龄、性别、婚姻状况和抚养人数来决定。

年龄			性别		婚姻状况		抚养人数
20~39	40~59	其它	M	F	已婚	未婚	1人扣0.5点，最多扣3点
6点	4点	2点	4点	3点	3点	5点	

3.2.2 等价类划分法的测试运用

分析程序规格说明中给出和隐含的对输入数据的要求，可以得出：

① 年龄：一位或两位非零整数，取值的有效范围为1~99。

② 性别：一位英文字符，只能取 ‘M’ 或 ‘F’ 值。

③ 婚姻：字符，只能取‘已婚’或‘未婚’。

④ 抚养人数：空白或字符‘无’或一位非零整数（1~9）

⑤ 点数 ：一位或两位非零整数，取值范围为8~19

通过对规格说明输入数据的取值分析，可以得出保险公司人寿保险保费计算程序的等价类。

等价类划分表

输入条件	有效等价类	编号	无效等价类	编号
年龄	20~39岁	1		
	40~59岁	2		
	1~19岁	3	小于1	12
	60~99岁		大于99	13
性别	‘M’	4	除‘M’和‘F’之外的其它字符	14
	‘F’	5		
婚姻	已婚	6	除‘已婚’和‘未婚’之外的其它字符	15
	未婚	7		
抚养人数	空白	8	除空白和‘无’和数字之外的其它字符	16
	无	9		
	1~6人	10	小于1	17
	6~9人	11	大于9	18

3.2.2 等价类划分法的测试运用

- 等价类测试存在两个问题
  - 规格说明往往没有定义无效测试用例的期望输出应该是什么样的。因此，测试人员需要花费大量时间来定义这些测试用例的期望输出。
  - 强类型语言没有必要考虑无效输入。传统等价类测试是诸如FORTRAN和COBOL这样的语言占统治地位年代的产物，那时这种无效输入的故障很常见。事实上，正是由于经常出现这种错误，才促使人们使用强类型语言。

3.3 边界值分析法

- 大量的软件测试实践表明，故障往往出现在定义域或值域的边界上，而不是在其内部。
- 为检测边界附近的处理专门设计测试用例，通常都会取得很好的测试效果。因此边界值分析法是一种很实用的黑盒测试用例方法，它具有很强的发现故障的能力。

3.3.1 边界值分析法的原理

- 1.边界条件
  - 边界是一些特殊情况。程序在处理大量中间数值时都是正确的，但是在边界处可能出现错误。边界条件就是软件计划的操作界限所在的边缘条件。
  - 一些可能与边界有关的数据类型有：数值，速度，字符，地址，位置，尺寸，数量等。同时，考虑这些数据类型的下述特征：
    - 第一个/最后一个，最小值 / 最大值，开始 / 完成，超过/在内，空 / 满，最短 / 最长，最慢/最快，最早/最迟，最高 / 最低，相邻 / 最远等。

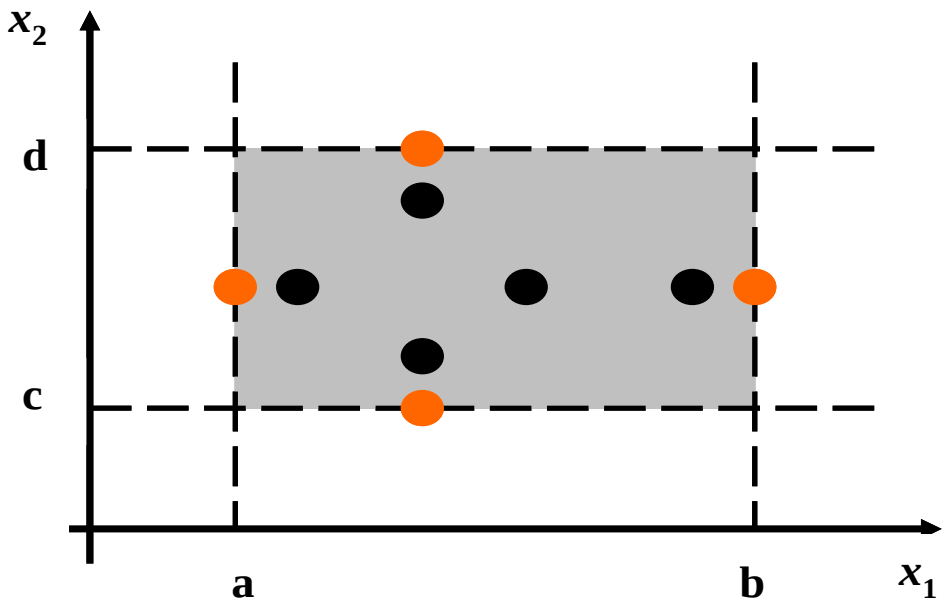
3.3.1 边界值分析法的原理

- 边界值和等价类密切相关，输入等价类和输出等价类的边界是要着重测试的边界情况。在等价类的划分过程中产生了许多等价类边界。边界是最容易出错的地方，所以，从等价类中选取测试数据时应该关注边界值。
- 在等价类划分基础上进行边界值分析测试的基本思想是，选取正好等于、刚刚大于或刚刚小于等价类边界的值作为测试数据，而不是选取等价类中的典型值或任意值做为测试数据。

3.3.1 边界值分析法的原理

- 2.边界值分析测试
  - 这里讨论一个有两个变量x1和x2的程序P。假设输入变量x1和x2在下列范围内取值:
    - $a \le x1 \le b$, $c \le x2 \le d$
  - 边界值分析利用输入变量的最小值（min），稍大于最小值(min+)，域内任意值（nom），稍小于最大值(max-)，最大值(max)来设计测试用例。即通过使所有变量取正常值，只使一个变量分别取最小值，略高于最小值、略低于最大值和最大值。

3.3.1 边界值分析法的原理

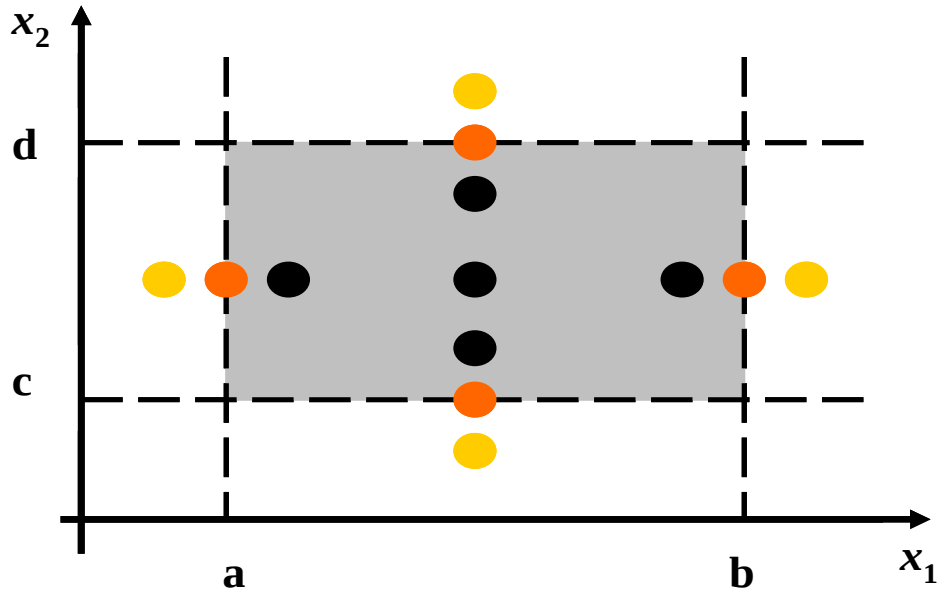


对于一个n变量的程序，边界值分析测试会产生4n+1个测试用例。

3.3.1 边界值分析法的原理

- 3. 健壮性边界值测试
  - 健壮性测试是边界值分析的一种扩展。
  - 变量除了取min，min＋，nom，max－，max五个边界值外，还要考虑采用一个略超过最大值（max+）以及一个略小于最小值(min-)的取值，看看超过极限值时系统会出现什么情况。

3.3.1 边界值分析法的原理



健壮性边界值测试将产生6n+1个测试用例。

健壮性测试最有意义的部分不是输入，而是预期的输出，观察例外情况如何处理。

3.3.2 边界值分析法的测试运用

- 三角形问题的边界值分析测试用例设计
  - 设输入在1~100之间取值

测试用例	a	b	c
Test1	60	60	1
Test2	60	60	2
Test3	60	60	60
Test4	50	50	99
Test5	50	50	100
Test6	60	1	60
Test7	60	2	60
Test8	50	99	50
Test9	50	100	50
Test10	1	60	60
Test11	2	60	60
Test12	99	50	50
Test13	100	50	50

3.3.2 边界值分析法的测试运用

- 2.加法器边界值测试用例设计
  - 【例2.4】 加法器程序计算两个1～100之间整数的和。
    - 对于加法器程序，根据输入要求可将输入空间划分为三个等价类，即1个有效等价类（1～100之间），两个无效等价类（小于1，大于100）。

3.3.2 边界值分析法的测试运用

- 为此，可综合考虑输入数据的取值范围和类型划分等价类，结果如下

编号	输入条件	所属类别	编号	输入条件	所属类别
1	1～100之间整数	有效等价类	5	非数值（字母）	无效等价类
2	<1整数	无效等价类	6	非数值（特殊字符）	无效等价类
3	>100整数	无效等价类	7	非数值（空格）	无效等价类
4	小数	无效等价类	8	非数值（空白）	无效等价类

3.3.2 边界值分析法的测试运用

- 加法器边界值测试用例

测试用例	输入数据		预期输出
	加数1	加数2	和
Test1	1	50	51
Test 2	2	50	52
Test 3	99	50	149
Test 4	100	50	150
Test 5	50	1	51
Test 6	50	2	52
Test 7	50	99	149
Test 8	50	100	150
Test 9	0	50	提示“请输入1～100间的整数”
Test 10	50	0	提示“请输入1～100间的整数”
Test11	101	50	提示“请输入1～100间的整数”
Test12	50	101	提示“请输入1～100间的整数”
Test13	0.2	50	提示“请输入1～100间的整数”

3.3.2 边界值分析法的测试运用

- 应用边界值分析法进行测试用例设计时，应遵循以下一些原则
  - 如果输入条件对取值范围进行了限定，则应以边界内部以及刚超出范围边界外的值作为测试数据。
  - 如果对取值的个数进行了界定，则应分别以最大、稍小于最大、稍大于最大、最小、稍小于最小、稍大于最小个数作为测试用例。
  - 对于输出条件，同样可以应用上面提到的两条原则来进行测试用例设计。
  - 如果程序规格说明书中指明输入或者输出域是一个有序的集合，如顺序文件、表格等，则应注意选取有序集合中的第一个和最后一个元素作为测试数据。

3.3.3 边界值分析法的局限性

- 基于函数(程序)输入定义域的测试方法，是所有测试方法中最基本的。
- 这类测试方法都有一种假设，即输入变量是真正独立的。
- 如果不能保证这种假设，则这类方法不能产生令人满意的测试用例。

3.4 因果图法

- 等价类划分法和边界值分析方法都是着重考虑输入条件，如果程序输入之间没有什么联系，采用等价类划分和边界值分析是一种比较有效的方法。
- 如果输入之间有关系，例如，约束关系、组合关系，这种关系用等价类划分和边界值分析是很难描述的，测试效果难以保障，因此必须考虑使用一种适合于描述对于多种条件的组合，产生多个相应动作的测试方法，因果图正是在此背景下提出的。

3.4.1 因果图法的原理

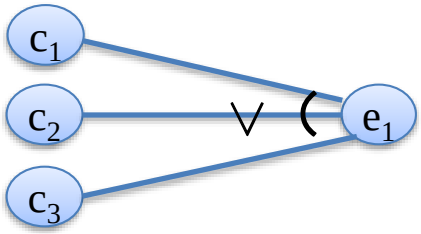
- 因果图

（a）恒等

如果c_1是1，则e_1也是1；否则e_1是0

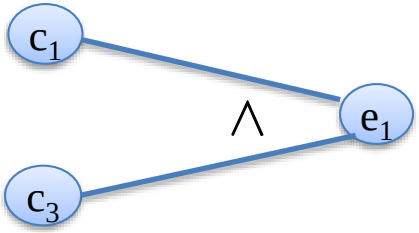
（b）非

如果c_1是1，则e_1是0；否则e_1是1



（c）或

如果c_1或c_2或c_3是1，则e_1也是1；否则e_1是0



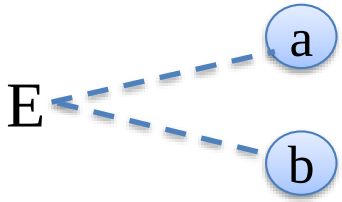
（d）与

如果c_1和c_2是1，则e_1也是1；否则e_1是0

3.4.1 因果图法的原理

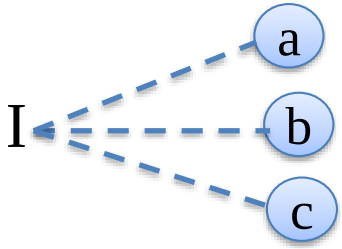
- 约束：在实际问题中，输入状态之间还可能存在某些依赖关系，称之为约束。例如，某些输入条件不可能同时出现。
  - 输入条件的约束
    - E（Exclusive，异或）：表示至多1个为1；
    - I（Inclusive，或）：表示至少1个为1；
    - O（One and Only，唯一）：只有一个为1；
    - R（Require，需要）：表示a是1，则b必须是1；
  - 输出条件的约束
    - M（Mask，强制）：表示a是1，则b必须是0。

3.4.1 因果图法的原理



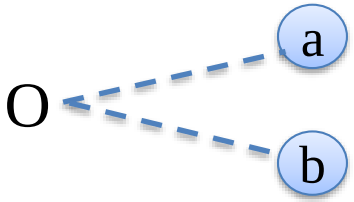
（a）异或

a和b至多一个可能为1



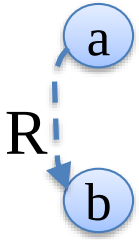
（b）或

a、b和c至少一个为1



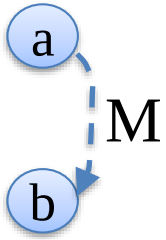
（c）唯一

a和b必须有且仅有一个为1



（d）要求

表示：a是1，b必是1



（e）强制

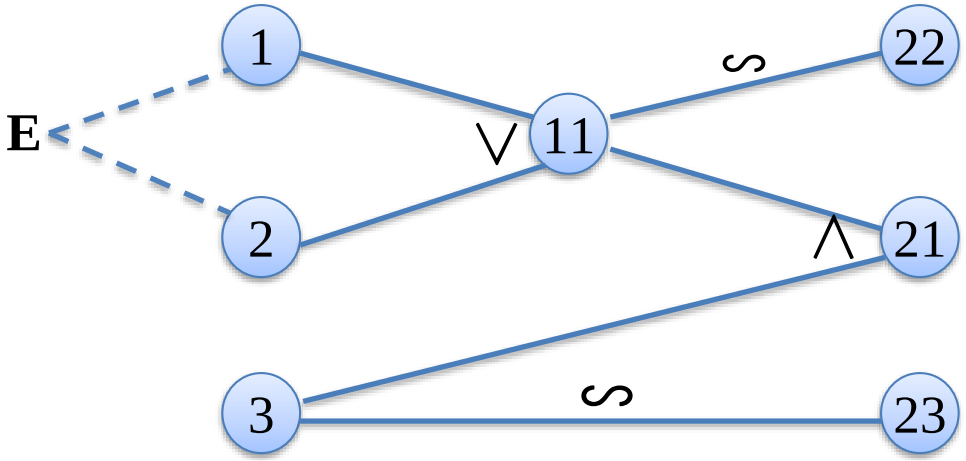
结果a是1，则结果b必是0

3.4.1 因果图法的原理

- 因果图法测试用例的设计步骤
  - 确定软件规格中的原因和结果。分析规格说明中哪些是原因（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。
  - 确定原因和结果之间的逻辑关系。分析软件规格说明中的语义，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。
  - 确定因果图中的各个约束。由于语法或环境的限制，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。
  - 把因果图转换为决策表。
  - 根据决策表设计测试用例。

3.4.2 因果图法的测试运用

- 某个软件规格说明书中规定：第一列字符必须是*或#，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M；如果第二列字符不正确，则给出信息N。
- 原因
  - 1——第一列字符是*；
  - 2——第一列字符是#；
  - 3——第二列字符是一数字。
- 结果
  - 21——修改文件；
  - 22——给出信息M；
  - 23——给出信息N。



3.4.2 因果图法的测试运用

		1	2	3	4	5	6
原因	1	1	1	0	0	0	0
	2	0	0	1	1	0	0
	3	1	0	1	0	1	0
结果	21	1	0	1	0	0	0
	22	0	0	0	0	1	1
	23	0	1	0	1	0	1
测试用例		*3	*M	#5	#N	C2	DY
		ok	N	ok	N	M	M,N

3.5 决策表法

1.决策表

决策表是把作为条件的所有输入的各种组合值以及对应输出值都罗列出来而形成的表格。

它能够将复杂的问题按照各种可能的情况全部列举出来，简明并避免遗漏。因此，利用决策表能够设计出完整的测试用例集合。

3.5.1 决策表法

规则 / 选项		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
问题	能编写程序？	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y
	熟悉软件工程？	N	N	N	N	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y
	对书中内容感兴趣？	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y
	理解书中内容？	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y
建议	学习C/C++语言		√	√	√		√	√	√								
	学习软件工程		√	√	√					√	√	√	√				
	继续阅读		√		√		√	√	√		√		√			√	√
	放弃学习	√				√								√	√		

3.5.1 决策表法

决策表通常由条件桩、条件项、动作桩和动作项4部分组成。

	条件桩	规则 1	规则 2	规则 3	规则 4	规则 5	规则 6	
条件桩	C1	T	T	T	F	F	F	条件项
	C2	T	T	F	T	T	F	
	C3	T	F		T	F		
	动作桩							
动作桩	A1	X	X		X		X	动作项
	A2	X			X	X		
	A3		X	X				

决策表的组成

动作项和条件项紧密相关，指出在条件项的各组取值情况下应采取的动作。

规则：任何一个条件组合的特定取值及其相应要执行的操作。

3.5.1 决策表法

2.决策表的构造及化简

构造决策表可采用以下5个步骤：

（1）列出所有的条件桩和动作桩。

（2）确定规则的个数。

（3）填入条件项。

（4）填入动作项，得到初始决策表。

（5）简化决策表，合并相似规则。

3.5.1 决策表法

- 决策表的化简

对于n个条件的决策表，相应有2^n个规则（每个条件分别取真、假值），当n较大时，决策表很繁琐。实际使用决策表时，常常先将它简化。决策表的简化是以合并相似规则为目标。即若表中有两条以上规则具有相同的动作，并且在条件项之间存在极为相似的关系，便可以合并。

3.5.1 决策表法

选项 \ 规则		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
问题	能编写程序？	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y
	熟悉软件工程？	N	N	N	N	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y
	对书中内容感兴趣？	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y
	理解书中内容？	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y
建议	学习C/C++语言		√	√	√		√	√	√								
	学习软件工程		√	√	√					√	√	√	√				
	继续阅读		√		√		√	√	√		√		√			√	√
	放弃学习	√				√								√	√		

3.5.1 决策表法

规则 / 选项		1，5	2，4	3	6，7，8	9，11	10，12	13，14	15，16
问题	能编写程序？	N	N	N	N	Y	Y	Y	Y
	熟悉软件工程？	—	N	N	Y	N	N	Y	Y
	对书中内容感兴趣？	N	—	Y	—	—	—	N	Y
	理解书中内容？	N	Y	N	—	N	Y	—	—
建议	学习C/C++语言		√	√	√				
	学习软件工程		√	√		√	√		
	继续阅读		√		√		√		√
	放弃学习	√						√	

3.5.2 决策表法的测试运用

下面以NextDate函数为例，讨论决策表测试用例的设计。

【例2.5】 NextDate函数输入为month(月份)、day(日期)和year(年)，输出为输入后一天的日期。例如，如果输入为：1964年8月16日，则输出为1964年8月17日。要求输入变量month、day和year都是整数值，并且满足以下条件：

Con1. 1≤month≤12

Con2. 1≤day≤31

Con3. 1900≤year≤2050

3.5.2 决策表法的测试运用

采用决策表法则可以通过使用“不可能动作”的概念表示条件的不可能组合，来强调这种依赖关系。

为了获得下一个日期，NextDate函数需要执行的操作只有如下5种：

① day变量值加1；

② day变量值复位为1；

③ month变量值加1；

④ month变量值复位为1；

⑤ year变量值加1。

3.5.2 决策表法的测试运用

如果将注意力集中到NextDate函数的日和月问题上，并仔细研究动作桩。可以在以下的等价类集合上建立决策表。

M_1: {month: month有30天}；

M_2: {month: month有31天，12月除外}；

M_3: {month: month是12月}；

M_4: {month: month是2月}；

D_1:{day:1≤day≤27}；

D_2: {day:day=28}；

D_3: {day:day=29}；

D_4: {day:day=30}；

D_5: {day:day=31}；

Y_1:{year:year是闰年}；

Y_2: {year:year不是闰年}

3.5.2 决策表法的测试运用

NextDate函数的决策表

选项 \ 规则		1	2	3	4	5	6	7	8	9	10	11
条件	C1:month在	M_1	M_1	M_1	M_1	M_1	M_2	M_2	M_2	M_2	M_2	M_3
	C2:day在	D_1	D_2	D_3	D_4	D_5	D_1	D_2	D_3	D_4	D_5	D_1
	C3:year在	—	—	—	—	—	—	—	—	—	—	—
动作	A1:不可能					√						
	A2:day加1	√	√	√			√	√	√	√		√
	A3:day复位				√						√	
	A4:month加1				√						√	
	A5:month复位											
	A6:year加1											

3.5.2 决策表法的测试运用

NextDate函数的决策表

选项 \ 规则		12	13	14	15	16	17	18	19	20	21	22
条件	C_1:month在	M_3	M_3	M_3	M_3	M_4	M_4	M_4	M_4	M_4	M_4	M_4
	C_2:day在	D_2	D_3	D_4	D_5	D_1	D_2	D_2	D_3	D_3	D_4	D_5
	C_3:year在	—	—	—	—	—	Y_1	Y_2	Y_1	Y_2	—	—
动作	A_1:不可能									√	√	√
	A_2:day加1	√	√	√		√	√					
	A_3:day复位				√			√	√			
	A4:month加1							√	√			
	A5:month复位				√							
	A6:year加1				√							

3.5.2 决策表法的测试运用

NextDate函数的决策表

选项 \ 规则		1	2	3	4	5	6	7	8	9	10	11
条件	C_1:month在	M_1	M_1	M_1	M_1	M_1	M_2	M_2	M_2	M_2	M_2	M_3
	C_2:day在	D_1	D_2	D_3	D_4	D_5	D_1	D_2	D_3	D_4	D_5	D_1
	C_3:year在	—	—	—	—	—	—	—	—	—	—	—
动作	A_1:不可能					√						
	A_2:day加1	√	√	√			√	√	√	√		√
	A_3:day复位				√						√	
	A_4:month加1				√						√	
	A_5:month复位											
	A_6:year加1											

3.5.2 决策表法的测试运用

NextDate函数的决策表

选项	规则	12	13	14	15	16	17	18	19	20	21	22
条件	C_1:month在	M_3	M_3	M_3	M_3	M_4	M_4	M_4	M_4	M_4	M_4	M_4
	C_2:day在	D_2	D_3	D_4	D_5	D_1	D_2	D_2	D_3	D_3	D_4	D_5
	C_3:year在	—	—	—	—	—	Y_1	Y_2	Y_1	Y_2	—	—
动作	A_1:不可能									√	√	√
	A_2:day加1	√	√	√		√	√					
	A_3:day复位				√			√	√			
	A4:month加1							√	√			
	A5:month复位				√							
	A6:year加1				√							

3.5.2 决策表法的测试运用

可进一步简化这22个测试用例。简化后的决策表如下所示。

		1~3	4	5	6~9	10	11~14	15	16	17	18	19	20	21~22
条件	C_1:month在	M_1	M_1	M_1	M_2	M_2	M_3	M_3	M_4	M_4	M_4	M_4	M_4	M_4
	C_2:day在	D_1~D_3	D_4	D_5	D_1~D_4	D_5	D_1~D_4	D_5	D_1	D_2	D_2	D_3	D_3	D_4,D_5
	C_3:year在	-	-	-	-	-	-	-	-	Y_1	Y_2	Y_1	Y_2	-
动作	A_1:不可能			√									√	√
	A_2:day加1	√			√		√		√	√				
	A_3:day复位		√			√		√			√	√		
	month加1		√			√					√	√		
	month复位							√						
	A_6:year加1							√						

3.5.2 决策表法的测试运用

根据简化后的决策表，可设计测试用例如下表所示。

测试用例	month	day	year	预期输出
Test1-3	6	16	2001	
Test4	6	30	2001	
Test5	6	31	2001	不可能
Test6-9	1	16	2001	
Test10	1	31	2001	
Test11-14	12	16	2001	
Test15	12	31	2001	
Test16	2	16	2001	
Test17	2	28	2004	
Test18	2	28	2001	
Test19	2	29	2004	
Test20	2	29	2001	不可能
Test21-22	2	30	2001	不可能

3.7 状态图法

- 案例分析

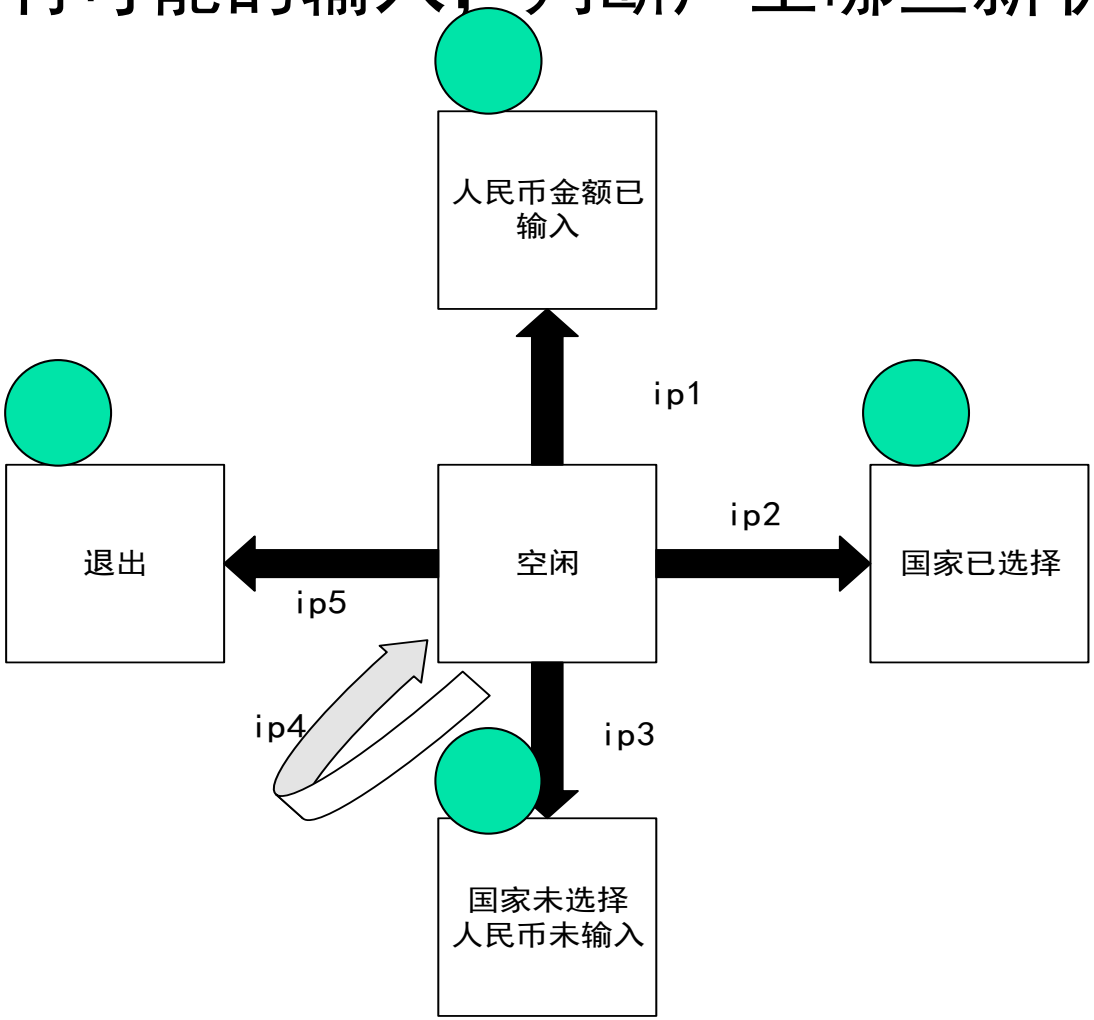
3.7.1 如何画出状态图

- 第一步：列出被测系统的输入事件

输入代号	输入事件
ip1	输入人民币金额
ip2	按下国家按钮
ip2.1	按下美国
ip2.2	按下加拿大
ip2.3	按下欧共体
ip2.4	按下日本
ip3	按下“计算”按钮
ip4	按下“清除”按钮
ip5	按下“退出”按钮
ip6	在错误消息中按下“确定”按钮

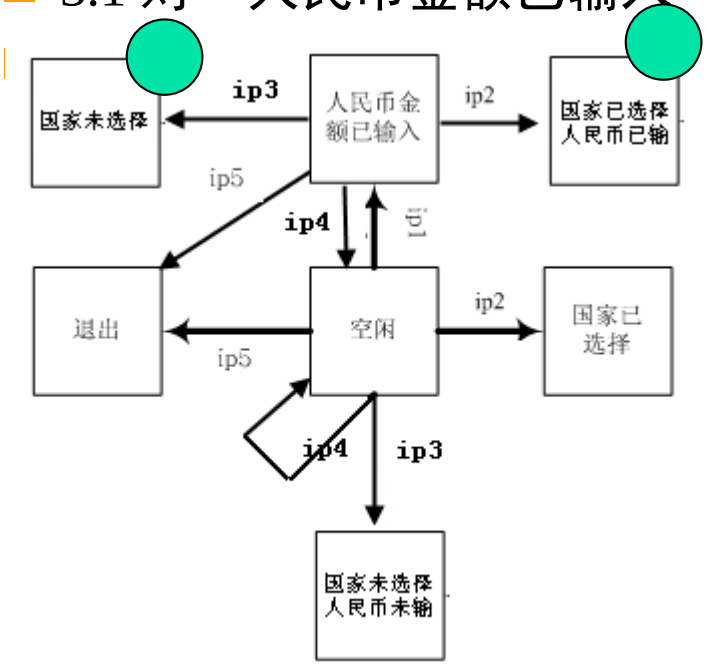
3.7.1 如何画出状态图

- 第二步：对空闲状态（程序刚启动时的状态）加所有可能的输入，判断产生哪些新状态。



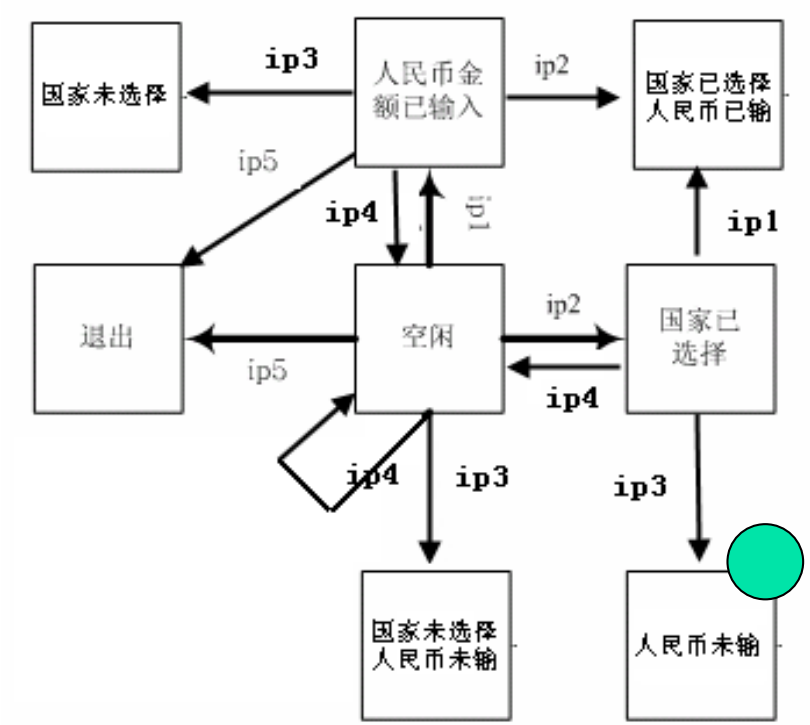
3.7.1 如何画出状态图

- 第三步：对第二步产生的每个新状态分别加所有可能的输入。
  - 3.1 对“人民币金额已输入”加所有可能的输入。



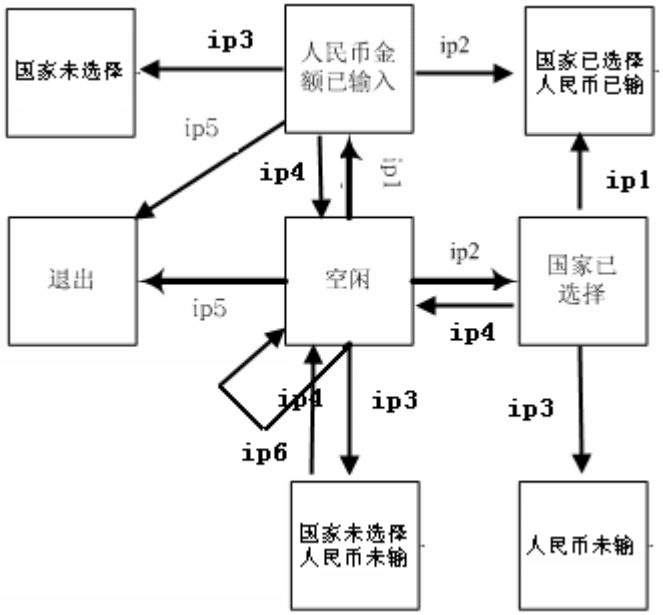
3.7.1 如何画出状态图

- 第三步：对第二步产生的每个新状态分别加所有可能的输入。
  - 3.1 对“人民币金额已输入”加所有可能的输入。
  - 3.2 对“国家已选择”再加所有可能的输入（图中加ip5输入的线省略了，因为其指向退出状态，不产生新状态）。



3.7.1 如何画出状态图

- 第三步：对第二步产生的每个新状态分别加所有可能的输入。
  - 3.1 对“人民币金额已输入”加所有可能的输入。
  - 3.2 对“国家已选择”再加所有可能的输入（图中加ip5输入的线省略了，因为其指向退出状态，不产生新状态）。
  - 3.3对“国家未选择、人民币未输”加所有可能的输入（ip6）

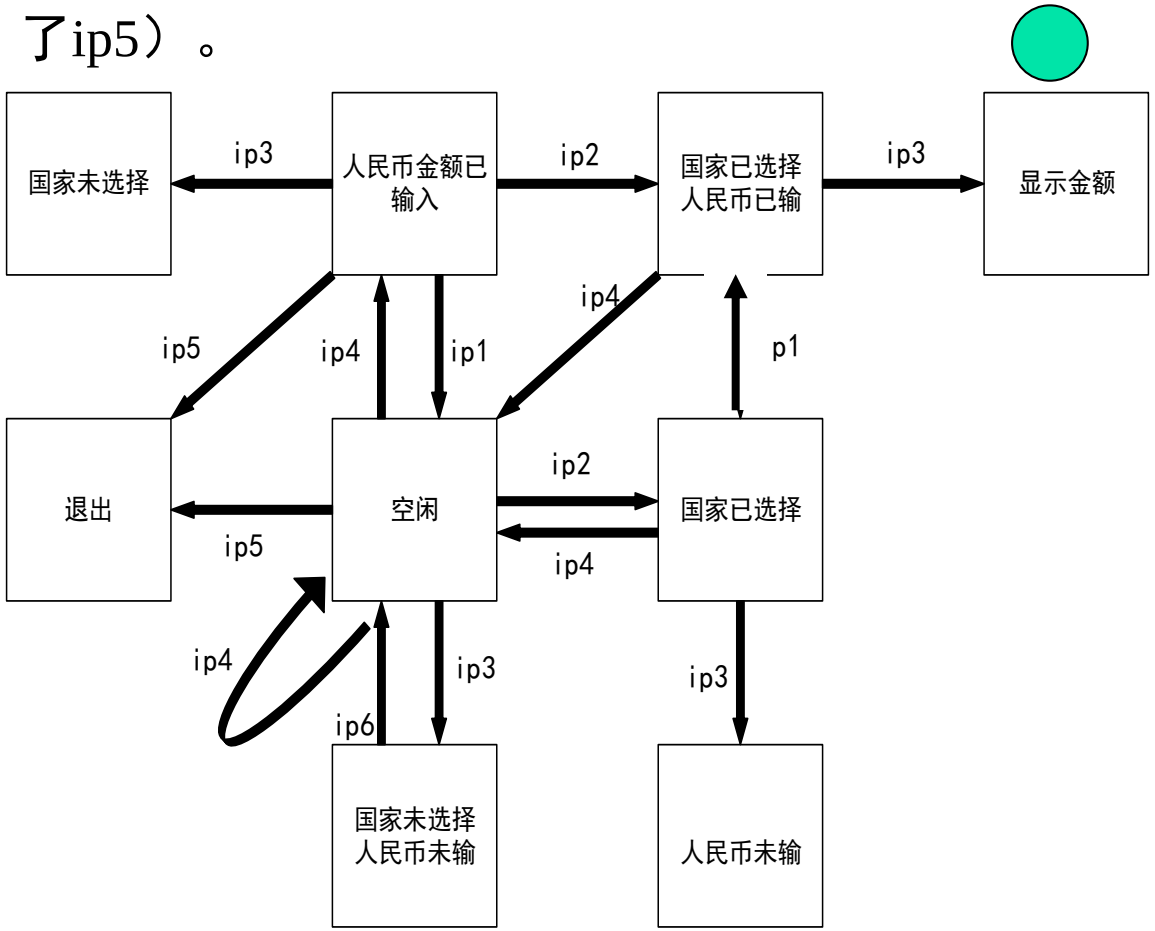


3.7.1 如何画出状态图

- 第三步：对第二步产生的每个新状态分别加所有可能的输入。
  - 3.1 对“人民币金额已输入”加所有可能的输入。
  - 3.2 对“国家已选择”再加所有可能的输入（图中加ip5输入的线省略了，因为其指向退出状态，不产生新状态）。
  - 3.3对“国家未选择、人民币未输”加所有可能的输入（ip6）
  - 3. 4对“退出”加所有可能的输入（没有）

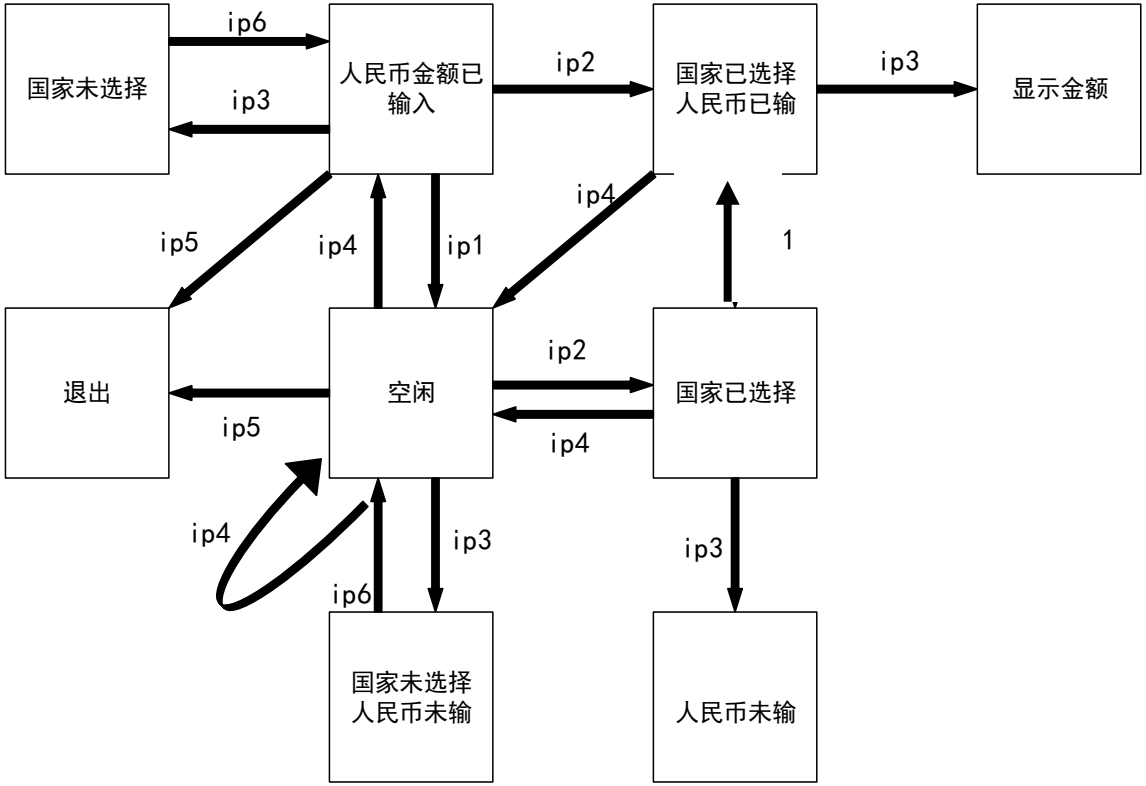
3.7.1 如何画出状态图

- 第四步：对第三步产生的每个新状态分别加所有可能的输入。
  - 4.1 对“国家已选择、人民币已输”加所有可能的输入（省略了ip5）。



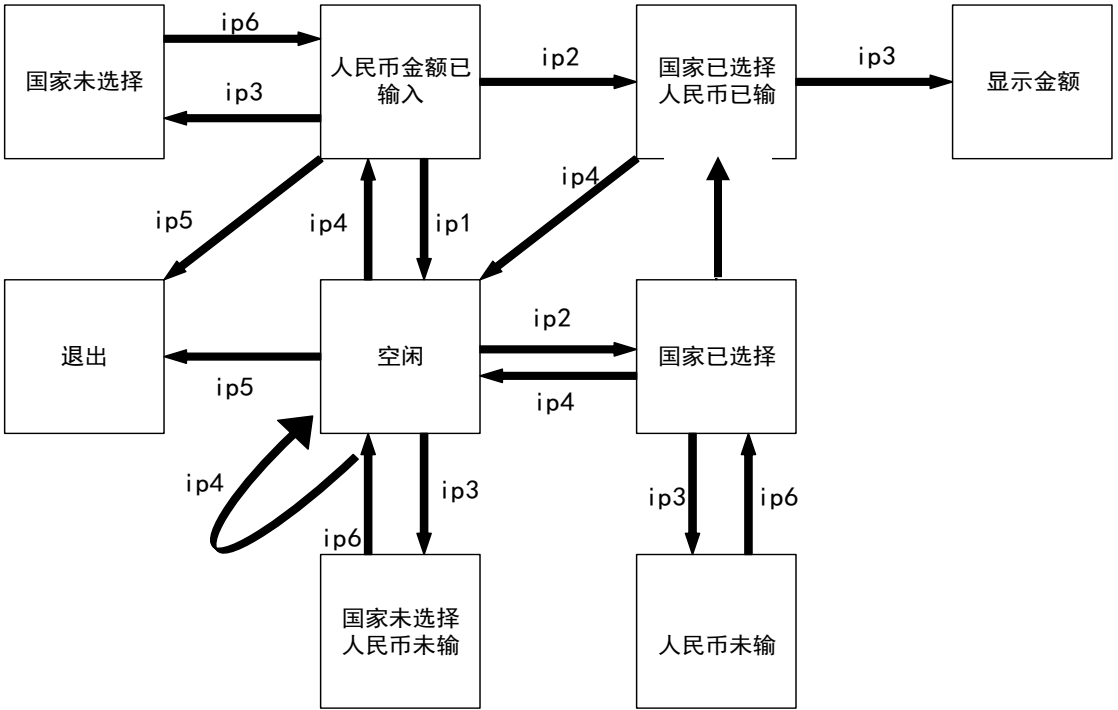
3.7.1 如何画出状态图

- 第四步：对第三步产生的每个新状态分别加所有可能的输入。
  - 4.1 对“国家已选择、人民币已输”加所有可能的输入（省略了ip5）。
  - 4. 2对“国家未选择”加所有可能的输入（只有ip6）



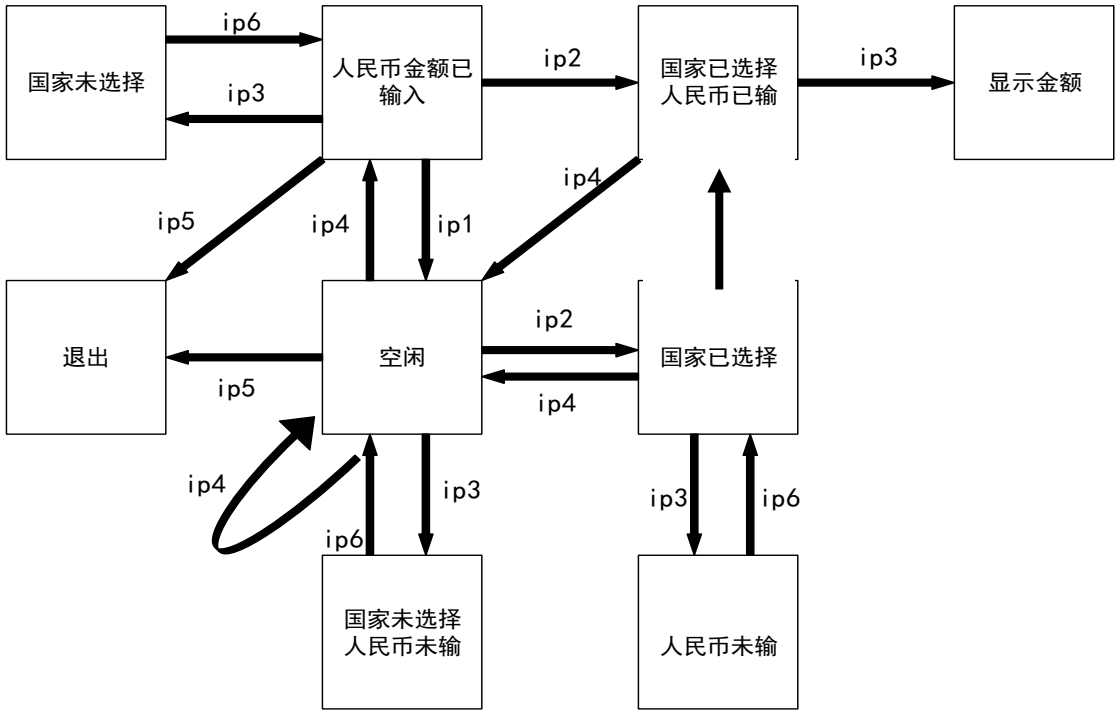
3.7.1 如何画出状态图

- 第四步：对第三步产生的每个新状态分别加所有可能的输入。
  - 4.1 对“国家已选择、人民币已输”加所有可能的输入（省略了ip5）。
  - 4. 2对“国家未选择”加所有可能的输入（只有ip6）
  - 4. 3对“人民币未输”加所有可能的输入（只有ip6）



3.7.1 如何画出状态图

- 第五步：对第四步产生的每个新状态分别加所有可能的输入。
  - 5.1 对“显示金额”加所有可能的输入，经分析，不再有新的状态产生，即此程序有如下9个状态：
    - 空闲
    - 遗漏国家和人民币
    - 国家已选择
    - 人民币已输入
    - 遗漏人民币信息
    - 遗漏国家信息
    - 完成两种输入
    - 显示等价金额
    - 退出



3.7.2 编写测试用例

- 测试用例流程表

状态	用例1	用例2	用例3	用例4	用例5
空闲	1	1	1	1	1
遗漏国家和人民币消息	2				
国家已选择		2	2，4		
人民币已输入				2	2, 4
遗漏人民币消息			3		
遗漏国家消息					3
完成两种输入		3	5	3	5
显示等价金额		4	6	4	6
退出	3		7		7

3.7.2 编写测试用例

- 减少测试用例的方法
  - 每种状态至少访问一次。无论用什么方法，每一种状态都必须测试。
  - 测试看起来最常见最普遍的状态转换。我们可以根据审查产品说明书时分析收集到的信息确定某些用户情况可能比其他更常见。
  - 测试状态之间最不常用的分支。这些分支是最容易被产品设计者和程序员忽视的。
  - 测试所有错误状态及其返回值。错误没有得到正确处理、错误提示信息不正确、修复错误时未正确恢复软件等情况是常有的。
  - 利用工具自动执行状态转换测试。

其他黑盒测试方法

- 特殊值测试
  - 测试人员使用其领域知识、使用类似程序的测试经验等信息开发测试用例时，常常使用特殊值测试。这种方法不使用测试策略，只根据“最佳工程判断”来设计测试用例。
  - 特殊值测试特别依赖测试人员的能力。
- 故障猜测法
  - 人们靠经验和直觉猜测程序中可能存在的各种软件故障，从而有针对性地编写检查这些故障的测试用例。

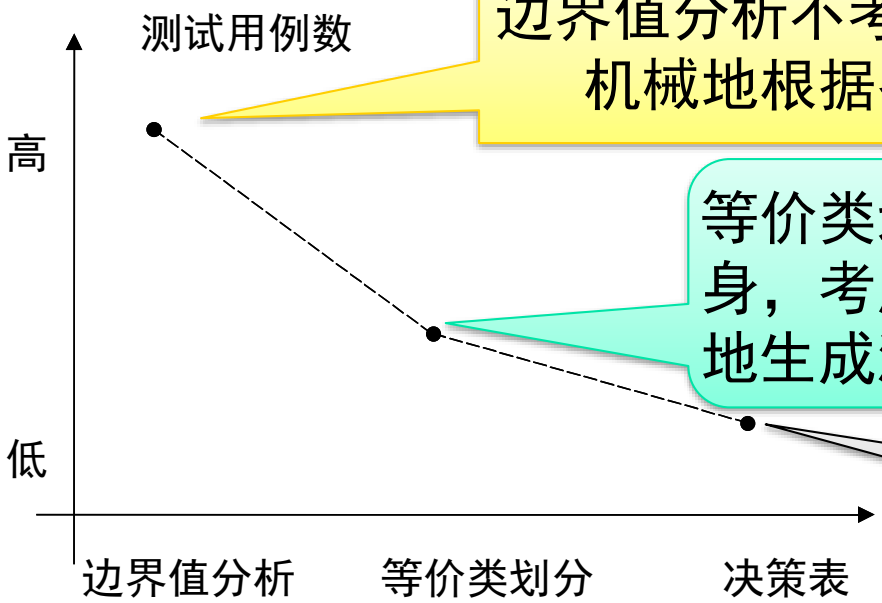
3.8 黑盒测试方法的比较与选择

- 上面研究了几种典型的黑盒测试方法，这些测试方法的共同特点是它们都把程序看作是一个打不开的黑盒，只知道输入到输出的映射关系，根据规范说明设计测试用例。
- 等价类分析测试中，通过等价类划分来减少测试用例的绝对数量。
- 边界值分析方法则通过分析输入变量的边界值域设计测试用例。
- 因果图测试方法考虑到输入条件和输出结果间的依赖关系和制约关系。
- 决策表方法全面的列出可能输入组合，并通过制约关系和合并的方法来减少测试用例。
- 状态图法侧重于过程间的转换动作及转换状态。

3.8 黑盒测试方法的比较与选择

1. 测试工作量

以边界值分析、等价类划分和决策表测试方法来讨论它们的测试工作量，即生成测试用例的数量与开发这些测试用例所需的工作量。



每种测试方法的测试用例数量

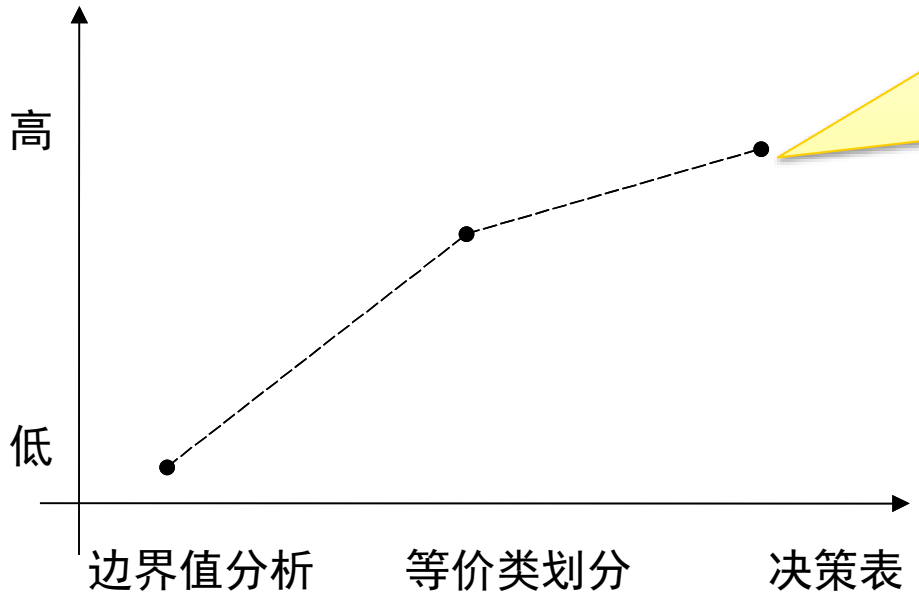
边界值分析不考虑数据或逻辑依赖关系，机械地根据各边界生成测试用例。

等价类划分则关注数据依赖关系和函数本身，考虑如何划分等价类，随后也是机械地生成测试用例。

决策表技术最精细，既要考虑数据，又要考虑逻辑依赖关系。

3.8 黑盒测试方法的比较与选择

每种方法开发测试用例所需的工作量趋势。



开发决策表测试用例生成所需的工作量最大。
但生成的测试用例少，机器执行时间短。

边界值分析测试方法使用简单，但会生成大量测试用例，机器执行时间长。

每种方法设计测试用例的工作量趋势

测试方法研究的目的就是在开发测试用例工作量和测试执行工作量之间做一个令人满意的折中。

3.8 黑盒测试方法的比较与选择

2.测试有效性

解释测试有效性是很困难的。因为我们不知道程序中的所有故障，因此我们也不可能知道给定方法所产生的测试用例是否能够发现这些。

所能够做的，只是根据不同类型的故障，选择最有可能发现这种缺陷的测试方法(包括白盒测试)。根据最可能出现的故障种类，分析得到可提高测试有效性的实用方法。通过跟踪所开发软件中的故障的种类和密度，也可以改进这种方法。

3.8 黑盒测试方法的比较与选择

2.测试有效性

利用程序的已知属性，选择处理这种属性的方法。

在选择黑盒测试方法时一些很有用的属性有：

- 变量是否表示物理量或逻辑量?
- 在变量之间是否存在依赖关系?
- 是假设单缺陷，还是假设多缺陷?
- 是否有大量例外处理?

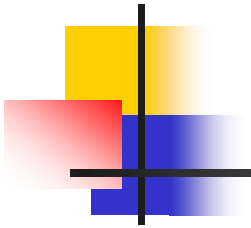
3.8 黑盒测试方法的比较与选择

下面列出一些黑盒测试方法选择的初步的“专家系统”

- 如果变量引用的是物理量，可采用边界值分析测试和等价类测试
- 如果变量是独立的，可采用边界值分析测试和等价类测试
- 如果变量不是独立的，可采用决策表测试
- 如果可保证是单缺陷假设，可采用边界值分析和健壮性测试
- 如果程序包含大量例外处理，可采用健壮性测试和决策表测试
- 如果变量引用的是逻辑量，可采用等价类测试用例和决策表测试

黑盒测试方法小结

- 常用的黑盒测试方法有等价类划分、边界值分析、决策表法等。
- 每种黑盒测试方法各有所长，应针对软件开发项目的具体特点，选择适当的测试方法，设计高效的测试用例，有效地将软件中隐藏的故障揭露出来。



3.9 黑盒测试工具介绍

- 黑盒测试工具是指测试软件功能和性能的工具，主要用于集成测试、系统测试和验收测试。
- 本节主要介绍几款常用的功能测试工具，性能测试工具则在第6章中介绍。

3.9 黑盒测试工具介绍

- 3.9.1 黑盒测试工具概要

黑盒测试是在已知软件产品应具有的功能的条件下，在完全不考虑被测程序内部结构和内部特性的情况下，通过测试来检测每个功能是否都按照需求规格说明的规定正常使用。

黑盒测试工具又分为：功能测试工具和性能测试工具。

① 功能测试工具:功能测试工具主要用于检测被测程序能否达到预期的功能要求并能正常运行。

② 性能测试工具:性能测试工具主要用于确定软件和系统性能。

3.9.1 黑盒测试工具概要

功能测试工具一般采用脚本录制(Record)/回放(Playback)原理，模拟用户的操作，然后将被测系统的输出记录下来，并同预先给定的标准结果进行比较。在回归测试中使用功能测试工具，可以大大减轻测试人员的工作量，提高测试效果。

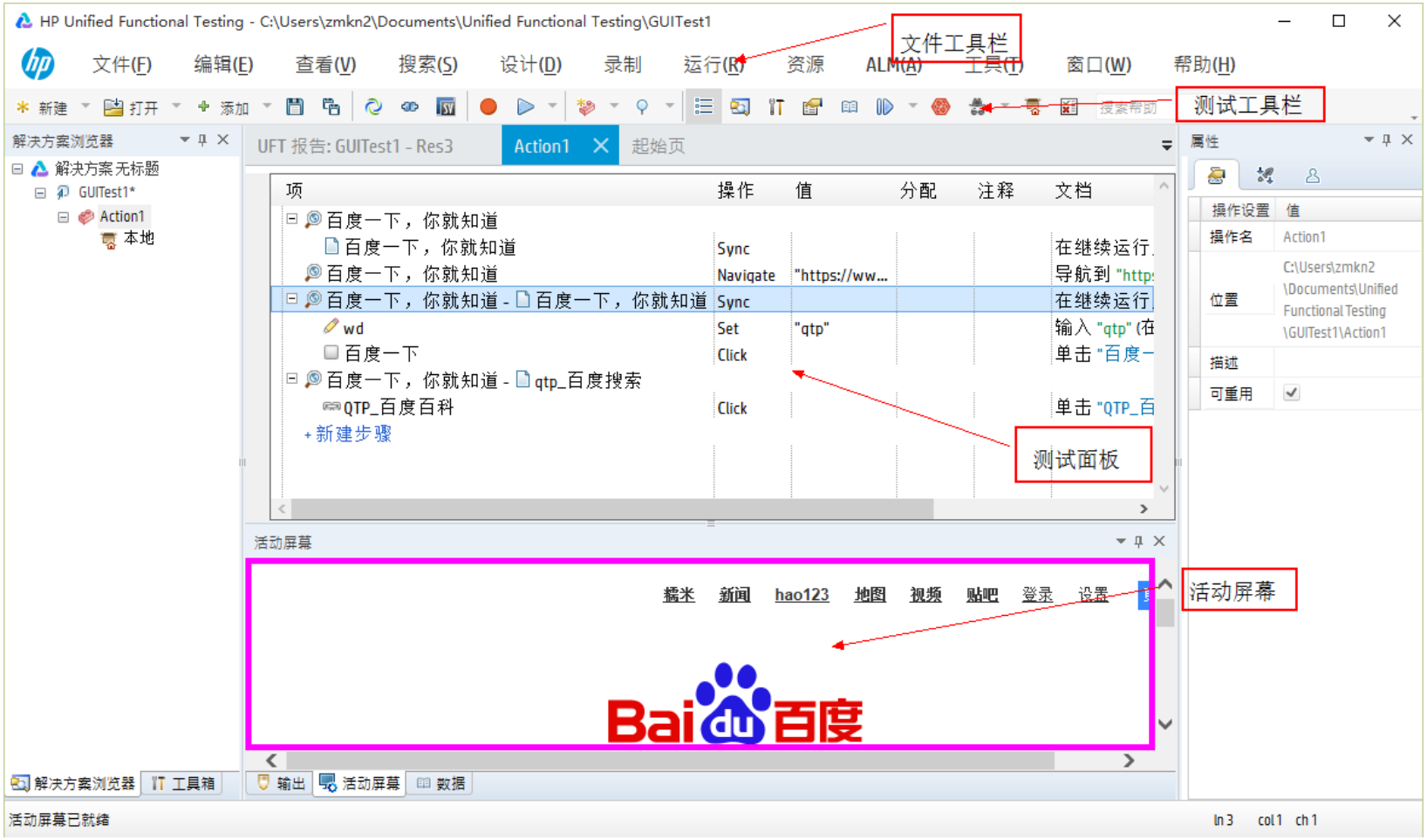
功能测试工具不太适合于版本变动较大的软件。

主流的黑盒功能测试工具包括Mercury Interactive公司的QTP，IBM Rational公司的TeamTest和Robot，Compuware公司的QACenter，ThoughtWorks公司开发的Selenium等。

3.9.2 黑盒功能测试工具—QTP

- 是Mercury Interactive公司的关键字驱动的功能测试工具，QTP属于新一代自动化测试解决方案，能够支持所有常用环境的功能测试
- 支持的脚本语言：VBScript
- 脚本调试工具：Microsoft Script Debugger
- 有超过WinRunner的趋势

3.9.2 黑盒功能测试工具—QTP



QTP的主界面

3.9.2 黑盒功能测试工具—QTP

- QTP的测试流程
  - 设计测试用例
    - 测试方法
    - 测试用例：操作步骤、测试数据、期望结果
  - 录制脚本
    - 录制正常的业务流程
  - 保存脚本
  - 增强脚本：参数化、添加检查点checkpoint
  - 执行测试
  - 分析结果

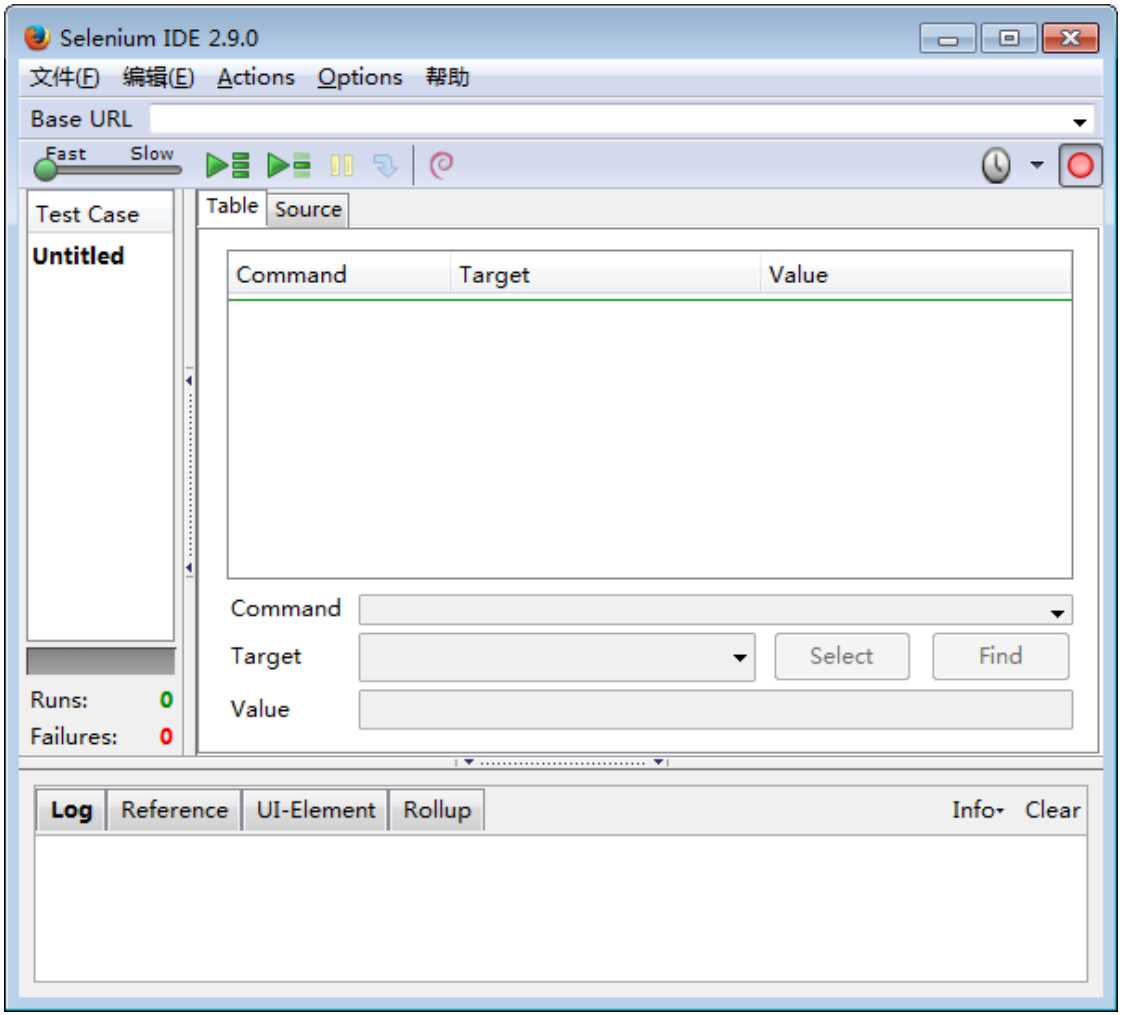
3.9.2 黑盒功能测试工具—QTP

QTP适合测试版本比较稳定的软件产品，在一些界面变化不大的回归测试中非常有效，但对于界面变化频率较大的软件，则体现不出QTP的优势。

3.9.3 黑盒功能测试工具—Selenium

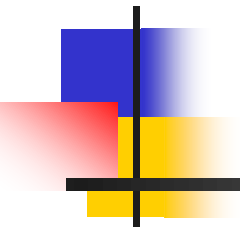
- Selenium是ThoughtWorks©编写的用于Web应用程序进行功能测试的工具
- Selenium测试直接运行在浏览器中，能够模拟真实的用户的操作
- 测试你的应用程序看是否能够很好得工作在不同浏览器和操作系统之上
- 创建回归测试检验软件功能和用户需求
- http://www.seleniumhq.org/

3.9.3 黑盒功能测试工具—Selenium



3.9.3 黑盒功能测试工具—Selenium

- Selenium的特点
  - 测试兼容性
  - 移动领域的自动化支持
  - 便于集成在工程项目中
  - 自动化截图



谢谢！