软件测试方法与技术

软件测试方法：黑盒测试

内容大纲

- 边界值分析
- 等价类
- 决策表
- 因果图

1. 边界值分析

- 1.1 边界值分析概念
  - 长期的软件测试工作经验证明，大部分的缺陷是发生在输入或输出的边界条件上，而不是内部。边界值分析是一种最常用的黑盒测试方法之一。
  - 1）语言相关的边界
    - 当定义不同类型的数据时，其取值范围是不一样的。如果在定义变量时，没有充分考虑所使用数据的范围，那么就可能出现错误。
    - 表1给出以JAVA语言中的整型数据类型取值范围的例子，显然在进行代码编写时，必须考虑这个变量将来可能出现的边界是否和变量的类型相一致。

类型	长度(位)	范围
byte	8	$-2^7 \cdots 2^7-1$
short	16	$-2^{15} \cdots 2^{15}-1$
int	32	$-2^{31} \cdots 2^{31}-1$
long	64	$-2^{63} \cdots 2^{63}-1$

表1 Java语言整数的取值范围

1. 边界值分析

- Python是在首次赋值时决定该变量的类型。如果第1次赋值的变量恰好达到该变量所要求的长度，那么其能够获得正确的变量类型，否则也和JAVA语言一样可能会存在错误。
- Python的标准整数类型是最通用的数字类型。在大多数32位机器上，标准整数类型的取值范围是$-2^{31}\cdots 2^{31}-1$，也就是-2,147,483,648到2,147,483,647。如果在64位机器上使用64位编译器编译Python，那么整数的长度将是 64 位。

- 2）业务相关的边界
  - 业务相关的边界是由业务特性所决定的而产生的边界。在这些边界上的数值，往往具有特定的要求，如果没有合适处理，就会导致错误。

1. 边界值分析

- 分析边界值可以用图示分析法。图示分析法用图的形式表示出输入值的边界情况，包括三种类型的点：上点、离点和内点。上点是指边界上的点，离点是指距离上点最近的点，如果输入区间是开放的则离点在区间内部，如果输入区间是封闭的则离点区间外部。内点是在区间内部的任意点。用不同的图标标识上点、离点和内点，可以更加清晰地看到边界的周边取值，测试用例取值时可以参考这些点。
- 用黑色实点表示有效的上点，白色实点表示无效的上点，内部有斜线的点表示离点，内部有点的点表示内点，如图 1-1所示。

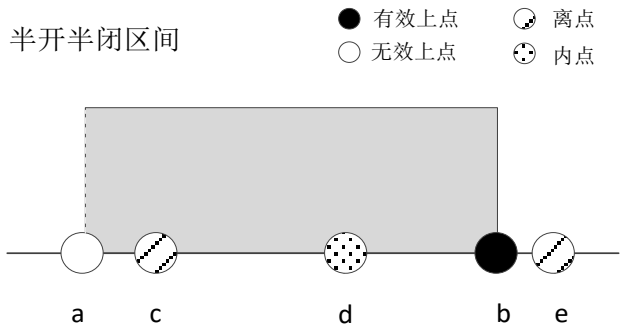


图 1 边界值的图示法

1. 边界值分析

- 1.2 边界值分析原则
  - 边界值分析设计设计测试用例时，进行边界值划分通常有以下几条规则：
    - （1）如果输入值有确定的范围并且是连续的，则测试数据可以取最小值、略大于最小值、正常值、略小于最大值以及最大值进行测试。例如，函数y=5x+2的定义域是[0,3]，则测试数据x可以取-1,0、0.1、2、2.9、3以及3.1。
    - （2）如果输入值有确定的范围并且是离散的，则测试数据可以取该离散范围内存在最小值、略大于最小值、正常值、略小于最大值以及最大值进行测试。
    - （3）如果输入数据有特殊的结构，则可以根据该结构的特性进行测试用例的设计，比较灵活。例如输入的数据是一份文件，则可以取文件开头和结尾的数据来进行测试。
    - （4）如果程序的规格说明给出的输入/输出域是有序集合（例如有序表、顺序文件等），则应选取集合中的第一个元素以及最后一个元素作为测试用例。
    - （5）分析规格说明，找出其他可能的边界条件。

1. 边界值分析

- 1.3 边界值确定和分析法
  - 1）一个输入参数
    - 若程序只包含一个输入参数，记为x。参数x的取值区域是一个一维的空间，x在这个空间占有一定的区间。在寻找输入参数x边界值时主要考虑三个因素：(1)区间是开区间还是闭区间，(2)区间是否存在界，(3)在内部是否都存在间断点。
    - 情况一：输入参数x的取值区域为闭区间[a,b]，如图2 所示。边界值是a和b，并且都是有效值。
      - 闭区间的取值如下：
      - （1）上点：a,b
      - （2）离点：d,e（分别无限接近点a和点b）
      - （3）内点：c
      - 测试用例的x取值应该为d、a、c、b和e这5个点的值，即包括上点、离点和内点。

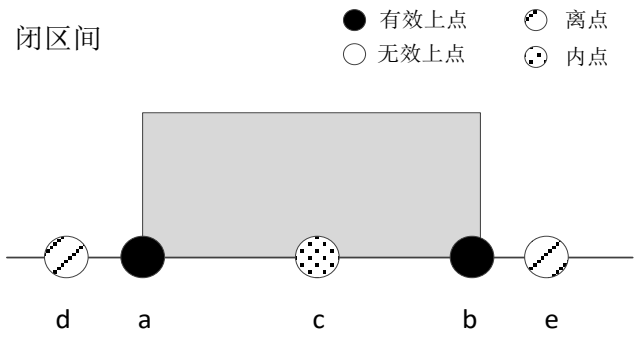


图2 闭区间取值

1. 边界值分析

- 情况二：输入参数x的取值区域为开区间(a,b)，如图3所示。边界值是a和b，并且都是无效值。
  - 开区间的取值如下：
  - （1）上点：a,b
  - （2）离点：c,d（分别无限接近点a和点b）
  - （3）内点：e
  - 测试用例的x取值应该为a、c、e、d和b这5个点的值，即包括上点、离点和内点。

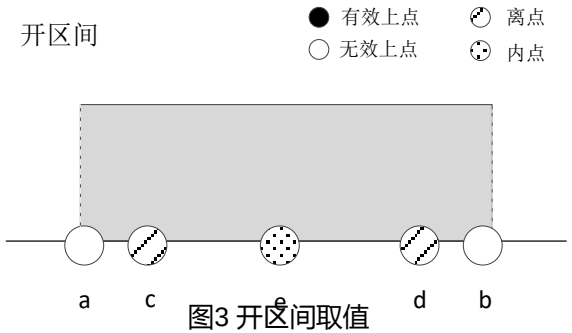


图3 开区间取值

- 情况三：输入参数x的取值区域为半开半闭区间(a,b]，如图 4所示。边界值是a和b，其中a是无效上点，b是有效上点。
  - 半开半闭区间的取值如下：
  - （1）上点：a,b
  - （2）离点：c,e（分别无限接近点a和点b）
  - （3）内点：d
  - 测试用例的x取值应该为a、c、d、b和e这5个点的值，即包括上点、离点和内点。

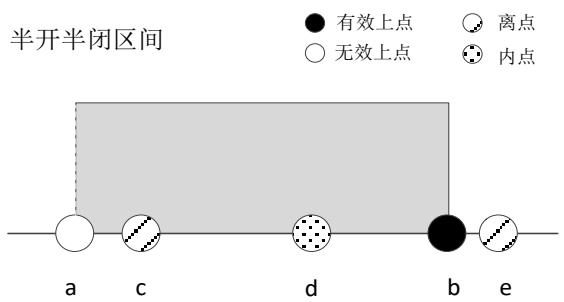


图4 半开半闭区间取值

1. 边界值分析

- 情况四：输入参数x的取值区域为无界区间[a,]，如图5所示。边界值是a，并且是有效的，另一个边界是正无穷大。实际测试的时候可以选择一个足够大的数来代替无穷大，图中用d表示。
  - 无界区间的取值如下：
  - （1）上点：a,d
  - （2）离点：b（无限接近点a）
  - （3）内点：c
  - 测试用例的x取值应该为b、a、c和d和这4个点的值，即包括上点、离点和内点。

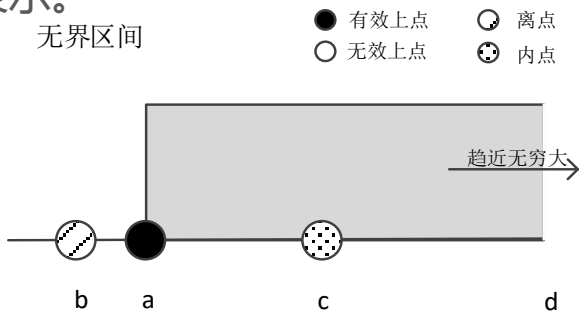


图5 无界区间是[a,]的情况

- 情况五：输入参数x的取值区域不是一个连续的区域，而是多个连续的区间。假设输入域是两个区间的并集[a,b]c,d]，如图 6所示。则边界值是a、b、c和d，并且都是有效的。
  - 多个连续区间取值如下：
  - （1）上点：a,b,c,d
  - （2）离点：e,g,h,j
  - （3）内点：f,i
  - 测试用例的x取值应该为e、a、f、b、g、h、c、i、d和j 这10个点，即包括上点、离点和内点。不能只考虑a和d这两个最小值和和最大值边界，而应该考虑所有连续区间的边界。

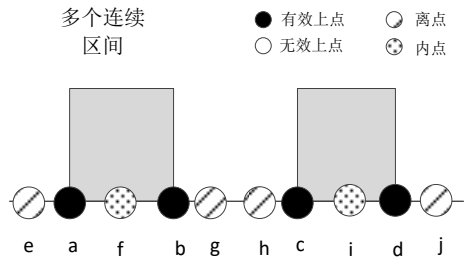


图6 多个连续区间取值

1. 边界值分析

- 情况六：输入参数x的取值区域为区间[a,b]，其中有一个点c是间断点，那么实际区间可表示为[a,c)(c,d]，如图7 所示。这里的c是一个间断点，可以认为c把一个连续的区域划分成了两个独立的区域，并且c是这两个区域的边界。这种情况下的边界值是a、b和c，其中a和b是有效的，c是无效的。
  - 有间断点区间取值如下：
  - （1）上点：a,b,c
  - （2）离点：d,e,f,g
  - （3）内点：h,i
  - 测试用例的x取值应该为d、a、h、f、c、g、i、b和e这五个点，即包括上点、离点和内点。不能只考虑a和b这两个边界。

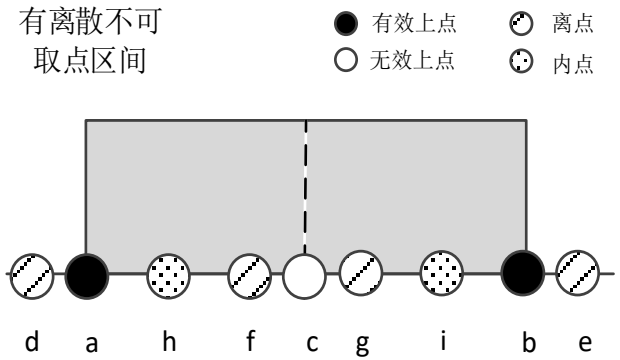


图7 有间断点区间取值

1. 边界值分析

- 2）两个输入参数
  - 两个输入参数，分别记为x和y，则输入域是由x和y共同确定的一个平面区域，围成该区域的直线或者曲线则是区域边界。在各个边界上的值可能是有效值，也可能是无效值。值得注意的是,只有一个输入参数时，只需要考虑一个参数的取值范围。当有两个输入参数，不仅仅要考虑单个输入参数的取值范围，还要考虑两个参数之间的共同约束。

1. 边界值分析

- 情况一：输入域是一些离散点，随机分布在二维空间中，如图8 所示。这种情况下可以分别单独确认输入参数x和y的边界，需要找到所有离散点中x的最小值和最大值以及y的最小值和最大值。如果要将所有情况都考虑进去，可以单独分析每个离散点。当离散点较少的时候测试比较方便，但当有许多离散点时会很费时间，而且效率很低。所以需要一种折中的方法，具体如下：将离散点最外面的点用直线相连，如图9 所示，连起来的线段表示的就是这个输入域的边界。
  - 离散点区域的取值如下：
  - （1）上点：线段ab、bd、cd和ac上的点（只有点a、b、c和d是有效值，其他都是无效值）。
  - （2）离点：线段ab、bd、cd和ac围成区域内的点（原本就存在的离散点是有效值，其他都是无效值）。
  - （3）内点：线段ab、bd、cd和ac围成区域外的点。
  - 这种方法也存在一定的缺陷，例如有一个点离所有的点很远，在连接最外面的点时会有很大的一部分内部区域是无效的，边界就变得没有意义，这时候需要将这个点单独取出。

1. 边界值分析

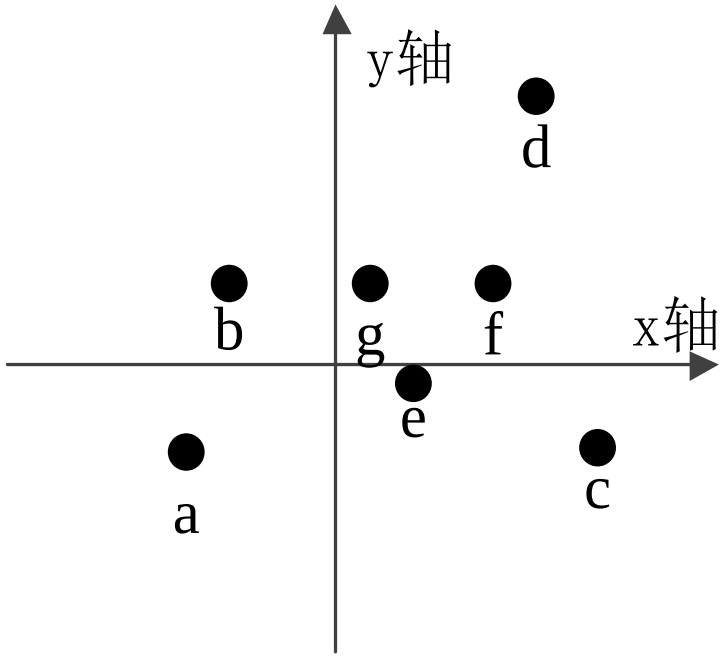


图8 情况一区域

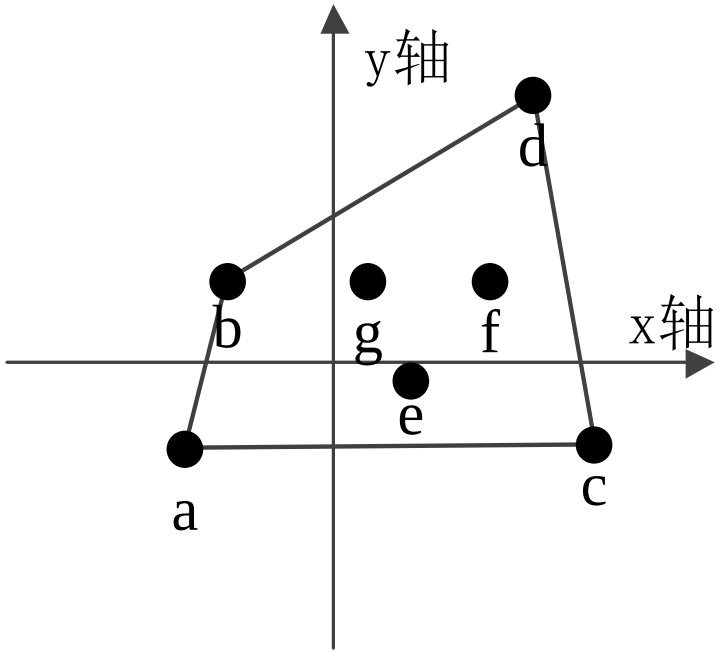


图9 情况一区域边界

1. 边界值分析

- 情况二：输入域是二维空间中的曲线或者直线，只有在这些线上的取值是有效的。图10中只有在圆周上的取值是有效值。首先单独分析输入参数x和y的边界，x的取值区间是[a,b]，y的取值区间是[c,d]。如果不深入考虑x和y之间的约束，就会得到四个边界x=a，x=b,y=c和y=d，这是一个矩形，如图11 所示。但是实际的边界是圆周，比上面四个边界确定的矩形边界要小很多。
  - 曲线或者直线区域的取值如下：
  - （1）上点：圆周上的点。
  - （2）离点：无限接近圆周的点（包括圆内和圆外的点）。
  - （3）内点：无（没有边界内的点）。

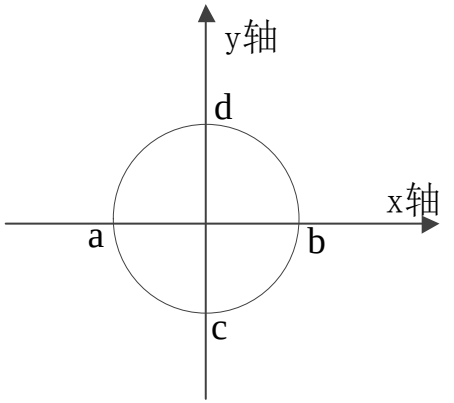


图10 曲线或者直线输入域

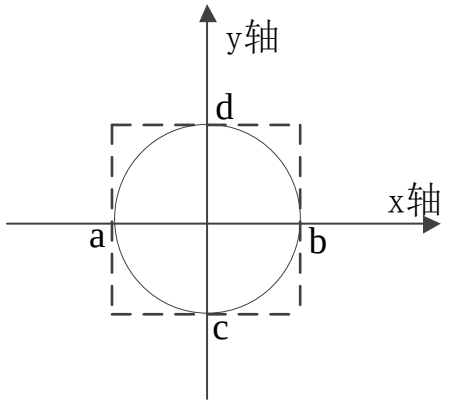


图11 不考虑约束的边界

1. 边界值分析

- 输入域是一条曲线或者直线，边界也就是这条线。在设计测试用例，要考虑边界上的边界点，而不是仅仅考虑x和y的各自边界的组合值。图 1-10 中的边界点是(a,0)、(b,0)、(0,c)和(0,d)，所以可以取这些点以及附近的点，具体的测试用例设计如表2 所示。

用例编号	x	y	预期输出
1	a−1	0	边界外
2	a	0	边界上
3	a+1	0	边界外
4	b−1	0	边界外
5	b	0	边界上
6	b+1	0	边界外
7	0	d−1	边界外
8	0	d	边界上
9	0	d+1	边界外
10	0	c−1	边界外
11	0	c	边界上
12	0	c+1	边界外

表2 情况二对应测试用例

1. 边界值分析

- 情况三：输入域是二维空间中的面，并且该面上的取值都是有效的，没有不可取的值。如图 12 所示。图中输入参数的取值区间是一个圆，而且圆内部的值都是有效的。这和情况二不同，情况二只有曲线或者直线上的值是有效的，情况三边界内的也是有效的。和情况二的分析过程类似，首先可以分别找出x和y的取值区间。x的取值区间[a,b]，y的取值区间[c,d]，同时区域边界是x和y的共同边界，需要满足一定的约束条件，图中显示边界是圆的圆周。
  - 取值如下：
  - （1）上点：圆周上的点。
  - （2）离点：无限接近圆周且在圆外的点。
  - （3）内点：圆内的点。

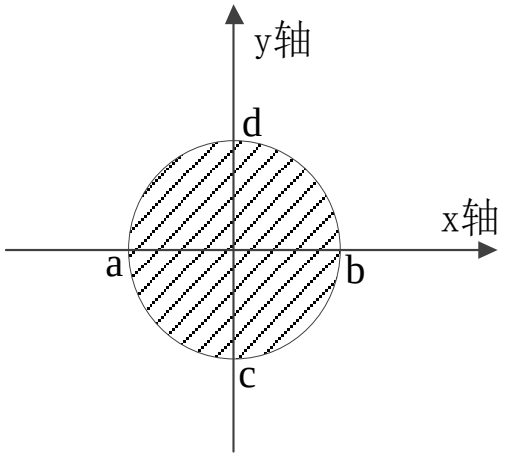


图12 情况三区域

1. 边界值分析

- 在设计测试用例时，输入域是一个面和输入域是一条线既有相同的地方又有不同的地方。相同的是输入域是一个面的边界还是一条曲线或者直线；不同的是输入域是一条面需要考虑边界内的正常值。针对边界，测试用例的设计和情况二相同。在这个基础上还要增加边界内的正常值，例如（0,0）。这和边界没有关系，但是也要进行测试。具体的测试用例设计如表 3 所示。

用例编号	x	y	预期输出
1	a-1	0	边界外
2	a	0	边界上
3	a+1	0	边界内
4	b-1	0	边界内
5	b	0	边界上
6	b+1	0	边界外
7	0	d-1	边界内
8	0	d	边界上
9	0	d+1	边界外
10	0	c-1	边界外
11	0	c	边界上
12	0	c+1	边界内
13	0	0	边界内

表3 情况三对应的测试用例

1. 边界值分析

- 情况四：输入域是二维空间中的面，其中这个面的内部有无效值，如图 13 所示。图中显示的是一个环，这时候就不仅仅存在外边界，同时还存在内边界。外边界x的取值范围为[a,b]，y的取值范围为[e,f]；内边界y的取值范围为[e,f]，y的取值范围为[h,g]。可以发现，在边界上，一个x值可能对应一个y值、两个y值和4个y值，例如x取0时，边界上y的取值有4个，分别是d、g、h和c。
  - 取值如下：
  - （1）上点：内圆周和外圆周上的点。
  - （2）离点：内圆周以内和外圆周以外的点。
  - （3）内点：圆环上的点。

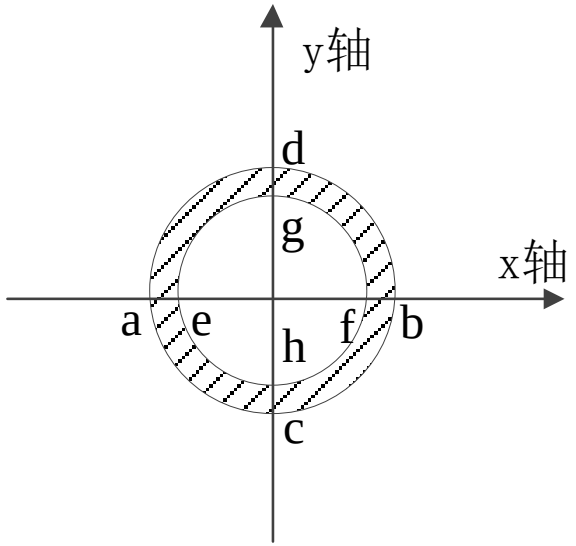


图13 情况四区域

1. 边界值分析

- 这种情况和情况3的测试用例设计方法基本一样，在考虑的时候要把外边界和内边界都考虑进去，同时还需考虑两个边界之间的点。

表4 情况四对应的测试用例

用例编号	x	y	预期输出
1	a-1	0	边界外
2	a	0	边界上
3	(a+e)/2	0	边界内
4	e	0	边界上
5	e+1	0	边界外
6	f-1	0	边界外
7	f	0	边界上
8	(f+b)/2	0	边界内
9	b	0	边界上
10	b+1	0	边界外
11	0	g-1	边界外
12	0	g	边界上
13	0	(g+d)/2	边界内
14	0	d	边界上
15	0	d+1	边界外
16	0	c-1	边界外
17	0	c	边界上
18	0	(c+h)/2	边界内
19	0	h	边界上
20	0	h+1	边界外

1. 边界值分析

- 3）三个输入参数
  - 三个输入参数的情况相对于前面分析的两类情况更加复杂，它的输入域是在一个三维的空间中，可能是点、线、平面，也可能是一个三维的立体空间。如果用 X_1, X_2, X_3 分别表示三个输入参数，则输入域可表示为 $D=\{(X_1, X_2, X_3) \mid P(X_1, X_2, X_3)\}$ 。当输入域是一个三维的立体空间该如何确定边界。首先，根据输入域的定义分析得到三个输入参数的边界。其次，找到各个参数之间的联系或者约束。三维的立体空间通常是由面围成的，这些面就是输入参数的边界。例如输入域是一个立体的三角形，则边界就是围绕成这个三角形的三个三角平面。在测试时，每个边界（面）都要考虑。

1. 边界值分析

- 4）间接推断隐含边界
  - 实际测试中，测试人员可能不能直接从软件规格说明书中得出直接约束条件，这就需要找到隐含的关系才能推断分析得到输入域的边界。下面将从四种不同的情况分析如何间接推断边界。
  - 情况一：输入参数和输出之间的逻辑关系未知，但有一个参数与输入输出都有逻辑关系，称其为中间参数。如果已知中间参数的约束条件，可根据中间参数和输入参数之间的逻辑关系求得输入参数的约束条件，再分析边界情况。
  - 例如有一个程序检查水桶质量，需计算圆柱形水桶的体积。已知水桶高度为h，程序的输入参数是水桶底部半径r，同时要求水桶底面积S不得大于m平方分米，则如何分析这个程序的输入边界？（公式用V=S×h)

1. 边界值分析

- 输入参数是半径r，中间参数是底面积S，输出是体积V。水桶是圆柱体，则水桶底部是一个圆，底面积 $S = \pi r^2$ 。隐藏的约束条件是水桶底部半径r要大于0，所以输入参数r的一个边界是0。另外中间参数，则可推断得到输入参数半径。可以看出r的另一个边界是，并且是可取的，这是由中间参数S以及它们之间的逻辑推断得到的，如下表5所示。

表5 包含中间参约束的示例

中间参数（底面积S）	输入参数（底面半径r）	
S ≤ m	r ≤ $\sqrt{(m/\pi)}$	0 < r ≤ $\sqrt{(m/\pi)}$
S ＞ 0（隐含条件）	r > 0	

1. 边界值分析

- 情况二：已知输入参数和输出之间的逻辑关系，并且有另一个参数与输入参数有一定的逻辑关系，称其为输入参数的相关参数。根据相关参数和输入参数的逻辑关系，可以推断出输入参数的约束条件，再进一步得到边界。
- 例如，同样以检查水桶质量程序为例，已知水桶高度h，程序的输入参数改为底面积S，并且要求半径r≤m，则如何分析这个程序的输入边界？（公式用V=S×h）
- 输入参数是底面积S，相关参数是半径r，输出是体积V。已知半径r≤m，计算$S=\pi r^2 \le \pi m^2$，可得，输入参数S的约束条件是小于等于4π平方分米。则输入参数S的一个边界是 4π，并且是可取的。这个推断过程用到了S的相关参数r的约束条件以及它们之间的逻辑关系，如表6所示。

表6 包含相关参数约束的示例

相关参数（底面半径r）	输入参数（底面积S）	
r≤m	$S \le \pi m^2$	$0 < S \le \pi m^2$
r>0（隐含条件）	S>0	

1. 边界值分析

- 情况三：输入参数和输出之间的逻辑关系未知，已知中间参数和输出之间的逻辑关系以及输入参数和中间参数定的逻辑关系。不同于第一种情况，中间参数的约束条件未知，但是已知中间参数的相关参数约束条件。既然相关参数的约束条件已知，可以根据与中间参数的逻辑关系，得到中间参数的约束条件。然后再根据中间参数和输入参数的逻辑关系，得到输入参数的约束条件，最后得到输入边界。
- 同样以检查水桶质量程序为例，已知水桶高度h，程序的输入参数为水桶底面积S，并且要求水桶底部周长C≤m。则如何分析这个程序的输入边界？（公式用$V = \pi r^2 \times h$）
- 输入参数是底面积S，中间参数是半径r，相关参数是周长C，输出是体积V。由约束条件周长C≤m分米，以及公式$C = 2\pi r$ 计算得到半径$r \le m/2\pi$。又根据公式$S = \pi r^2$ 得到底面积$S \le m\hat{}2/(4 \cdot \pi)$。则输入参数S的一个边界是$m\hat{}2/4\pi$，且是可取的。C的一个隐藏约束条件C>0，可以推断出。这个推断过程用到周长的约束条件、周长与半径的逻辑关系和半径与面积的逻辑关系，如表7所示。

1. 边界值分析

表7 中间参数和输入输出均有关系

中间参数的相关参数（周长）	中间参数（底面半径r）	输入参数（底面积S）	
C≤m	$r \le \frac{m}{2\pi}$	$S \le \frac{m^2}{4\pi}$	$0 < S \le \frac{m^2}{4\pi}$
C>0（隐含条件）	r>0	S>0	

- 情况四：输入参数和输出之间的逻辑关系已知，同时有两个输入参数的相关参数，记为相关参数1和相关参数2。已知输入参数和两个相关参数之间的逻辑关系，两个相关参数的约束条件也都明确。既然相关参数的约束条件已知，可以根据与输入参数的逻辑关系，得到输入参数的约束条件。值得注意的是，相关参数收到两个相关参数的二重约束。
- 同样以检查水桶质量程序为例，已知水桶高度h，程序的输入参数为水桶底部半径r，并且要求水桶底部周长C≤ m_1，水桶底面积 $S \ge m_2$ 。则如何分析这个程序的输入边界？（公式用 $V = \pi r^2 \times h$ ）

1. 边界值分析

- 输入参数是半径r，相关参数1是底面积S，相关参数2是周长c，输出是体积V。由约束条件周长 $C \le m_1$ 分米、底部面积 $S \ge m_2$ 平方分米、公式 $C = 2\pi r$ 和 $S = \pi r^2$ 计算得到半径 $1 \le r \le 2$。则输入参数r边界为1和2，且都是可取的。这个推断过程用到周长1和底部面积S的约束条件以及它们与半径r的逻辑关系。

表8 输入和输出关系推导约束关系

相关参数1（周长）	相关参数1（底面积S）	输入参数底面半径r）	
$C \le m_1$	—	$r \le \frac{m_1}{2\pi}$	$\sqrt{\frac{m_2}{\pi}} \le r \le \frac{m_1}{2\pi}$
—	$S \ge m_2$	$r \ge \sqrt{\frac{m_2}{\pi}}$	
C>0（隐含条件）	—	r>0	
—	S>0（隐含条件）	r>0	

1. 边界值分析

- 5）根据输出条件分析边界值
  - 软件测试中经常会遇到这样的情况：规格说明书中对输入域没有进行过多的说明，但是对输出域有一定的说明，或者对输出结果有一定的预测。这种情况下可以根据输出条件来确定输入域。根据不同输出对应着不同的输入，如果已知输出的边界也就可以倒推出输入的边界值。例如，某银行的ATM自动取款机的余额为30000人民币，即输出的上限是30000，则用户取款的输入金额的上边界也是30000。

1. 边界值分析

- 1.4 边界值测试举例
  - 有一个三角形ABC，现在根据角的大小来判断三角形的形状。在能组成三角形的前提下，三个角形是锐角、直角、还是钝角。输入参数是 $\angle A$ 、$\angle B$ 和 $\angle C$ 的角度。其中 $\angle A$ 、$\angle B$ 和 $\angle C$ 的取值范围都是 $(0^{\circ},180^{\circ})$ 0° 和 180° 都不可取。
  - 由问题描述可知，$\angle A$ 、$\angle B$ 和 $\angle C$ 的和必须是 180° 才能构成三角形。现将输入参数简单记为变量A、B和C，输入域是三维空间中的一个超平面，这个三角形面的方程表达式是：A＋B＋C=180，其中1<A<180，0<B<180，0<C<180，如图14所示。超平面外面的边界是不可取，是因为、和不能取。

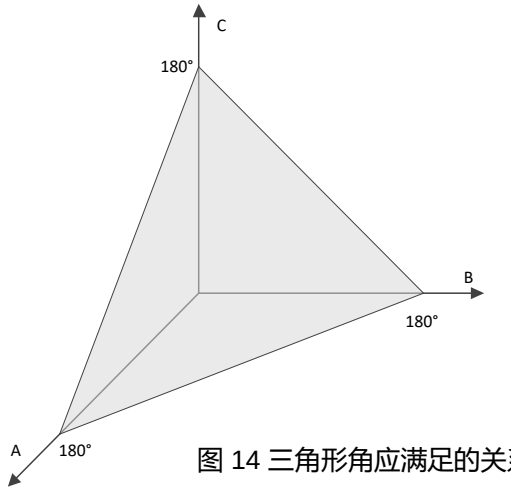


图 14 三角形角应满足的关系

1. 边界值分析

- 根据输出（三角形的形状），可以分为四个等价类，分别是输出结果是钝角三角形、直角三角形、锐角三角形和非三角形。
  - 钝角三角形对应的输入条件是$\angle A$ 、$\angle B$和 $\angle C$中间有一个角是钝角，即大于 90°；
  - 直角三角形对应的输入条件是$\angle A$ 、$\angle B$和 $\angle C$ 中间有一个角是直角，即等于90°；
  - 锐角三角形对应的输入条件是$\angle A$ 、$\angle B$和 $\angle C$ 的角都锐角，即都小于 90°
  - 非三角形对应的输入条件是A+B+C≠180。
- 根据等价类的边界可以将整个输入域进行重新划分，如图15所示。整个输入域被分为了四个小的三角形，其中最中间的小三角形表示锐角三角形，外面三个小的三角形表示钝角三角形，中间小三角形的边界表示直角三角形。

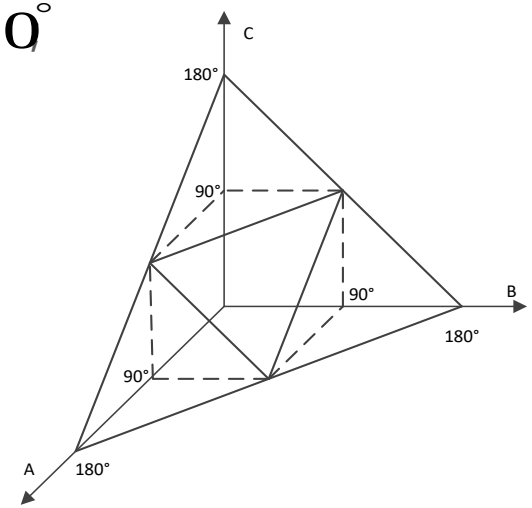


图15 不同三角形所处的区域

1. 边界值分析

- 设计测试用例，首先得知A的取值区间为，测试用例的A可取取0、45、90、135和180。如图16所示，表示A取不同的值对应的输入域上的直线。当A=180时，是边界上的一个点，只需分析边界点周围的情况即可。当A=135时，可以从图中看到，虚线与2个边界相交，所以设计测试用例时要考虑边界外的点、边界上的点以及边界内的点，总共5个点。当A=45时，虚线与4个边界相交，设计测试用例时要考虑边界外的点、边界上的点以及边界内的点，总共9个点。当A=90和A=0时，正好是边界，则需要分析边界上特殊的点，一般是指边界相交的点。除此之外，这些点的附近也要分析。

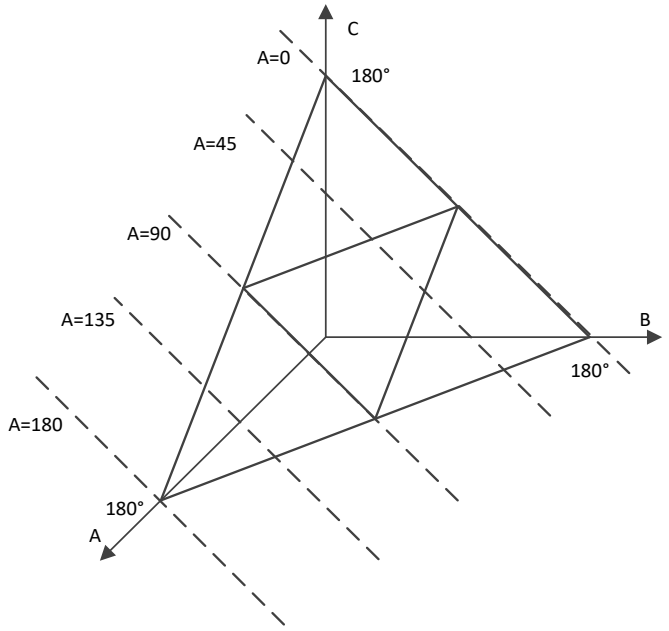


图16 不同三角型边界取值分析

1. 边界值分析

- 具体的用例设计如表9所示。

表9 三角形问题的测试用例

用例编号	A	B	C	预期输出
1	0	0	179	不是三角形
2	0	0	180	不是三角形
3	0	1	180	不是三角形
4	0	90	89	不是三角形
5	0	90	90	不是三角形
6	0	91	90	不是三角形
7	0	180	-1	不是三角形
8	0	180	0	不是三角形
9	0	181	0	不是三角形
10	45	0	136	不是三角形
11	45	0	135	不是三角形
12	45	5	130	钝角三角形
13	45	45	90	直角三角形
14	45	60	75	锐角三角形
15	45	90	45	直角三角形
16	45	130	5	钝角三角形
17	45	135	0	不是三角形
18	45	136	0	不是三角形
19	90	-1	90	不是三角形
20	90	0	90	不是三角形
21	90	0	91	不是三角形
22	90	45	45	直角三角形
23	90	90	-1	不是三角形
24	90	90	0	不是三角形
25	90	91	0	不是三角形
26	135	-1	45	不是三角形
27	135	0	45	不是三角形
28	135	5	35	钝角三角形
29	135	45	0	不是三角形
30	135	46	0	不是三角形
31	180	-1	0	不是三角形
32	180	0	0	不是三角形
33	180	0	1	不是三角形

2. 等价类

- 2.1 等价类概念
  - 将大量的输入数据（有效的和无效的）划分为若干等价类时，等价类覆盖的范围通常是指输入域的子集合，在该集合中的所有数据对于发现的错误的作用是等效的。
  - 等价类测试通常基于这样的假定：等价类中的某一代表值等价于这个等价类中所有其他值的测试，等价的含义是发现缺陷的能力是一样的。
  - 当利用某个等价类中的输入值进行测试时发现了错误，那么使用该等价类中所有输入值进行测试也会发现同样的错误；反之，如果使用等价类中值无法发现错误，那么该等价类中的其他数据也无法发现错误。

2. 等价类

- 例如，对学生的考试成绩进行评级，其中评级的标准如表 10 所示。

表10 学生成绩评级参考

成绩范围	级别
90～100	优
80～89	良
70～79	中
60～69	差
<60	不及格

- 当某个学生输入成绩时，学生系统能够自动判断出该生成绩的级别。当输入91、92、95时，判定为优；当输入81、83时，判定为良；当输入72、78时，判定为中；当输入66、68时，判定为差；当输入10、59时，判定为不及格。因此，(91,92,95)、(81,83)、(72,78)、(66,68)、(10,59)均可以被看做等价类。

2. 等价类

- 等价类包括：有效等价类和无效等价类。
  - 有效等价类：对于程序的规格说明而言，有效等价类是合理的、有意义的数据所构成的集合。利用有效等价类可以检验程序是否能够实现规格说明中规定的功能和性能。
  - 无效等价类：对于程序的规格说明而言，无效等价类是不合理的、没有意义的数据构成的集合。测试人员主要利用无效等价类检验程序是否正确处理不符合规格说明要求的输入。
- 等价类划分是将程序的输入域划分为若干部分，然后从每个部分中选取少数代表性数据作为测试用例。每类的代表性数据在测试中的作用等价于这一类中的其他值。使用等价类划分法设计测试用例通常按照以下三步：
  - 第一步：划分等价类；并对每一个等价类规定一个唯一的编号；
  - 第二步：设计测试用例，使其尽可能多地覆盖尚未覆盖的有效等价类；重复该步骤直至所有的有效等价类均被覆盖；
  - 第三步：设计新的测试用例，直至所有的无效等价类均被覆盖。

2. 等价类

- 边界值分析法与等价类划分法都是黑盒测试中常用的方法，而边界值分析法是在等价类划分法的基础进行的测试。这两种方法有两个不同之处：
  - （1）等价类划分法是在已分好的等价类中随机选取输入数据作为该等价类的代表，而边界值分析法是在已分好的等价类中选取边界值进行测试；
  - （2）等价类划分法仅仅考虑的是输入数据的划分，而边界值分析法还需要考虑输出结果的划分。

2. 等价类

- 2.2 等价类的划分及依据
  - 原则1：如果输入条件规定了取值范围、取值个数，则可以确立一个或者多个有效等价类或者无效等价类。
    - 例如，在程序规格说明中给出的输入条件为：“输入值为0到100之间的整数”，则有效等价类为“0输入值100”，无效等价类为“输入值0”或者“输入值100”。
  - 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/一定”的条件，可以确立一个有效等价类和一个无效等价类。
    - 例如，在设置密码时规定密码必须不能为纯数字，则所有由数字构成的密码组成无效等价类，其他的就可以组成有效等价类。

2. 等价类

- 原则3：如果输入数据为一组限定值，同时需要对输入的值进行处理，此时可以为所有输入值确定一个有效等价类。此外，针对该值可以确立一个无效等价类，即所有不允许的输入值集合。
  - 例如，为计算出差补贴，出差的住宿、交通工具以及餐费都给予报销。住宿宾馆按照3星级标准，交通工具限制为汽车、火车、飞机和轮船，餐费的报销也按照相应的标准。因此可以确定的有效等价类包括3星及3星以下的宾馆，汽车、火车、飞机和轮船，相应的伙食补助标准。除此以外，不符以上条件的均可以被划分为无效等价类，如4星级宾馆，三轮车等的输入值集合。
- 原则4：如果规定了输入数据必须遵守的规则，那么可以建立符合规则的有效等价类和其他若干无效等价类。
  - 例如，Java中的变量命名必须以$、字母、下划线开头。这时的有效等价类可以是以$、字母、下划线开头的名称，若干个无效等价类则是不是以$、字母、下划线开头的名称。

2. 等价类

- 原则5：如果明确知道已划分的等价类中不同元素在程序中的处理方式不一样，那么将该等价类进一步划分为更小的等价类。

- 除了上述原则外，等价类划分还需要依据输入数据的类型、输入数据的层次关系、输入数据的维度和输出条件等信息做进一步考虑。
  - 依据一：根据输入数据的类型来划分等价类，主要分为两种情况：连续型和离散型。
    - 1）连续型
      - 连续型是指输入数据属于连续型数据。输入数据有一个连续的取值区间，在这个区间上可以取测试数据。
      - 对于这种情况，首先要分析连续性数据取值区间上不同取值所代表的意义。根据不同的取值意义，找到相区别的边界值，再进行等价类划分。

2. 等价类

- 例如，在一个处理水状态的程序中，在不同的温度下具有不同的属性，程序需要根据不同的状态执行不同的操作。0摄氏度以下的水呈固态，0摄氏度～100摄氏度之间水呈液态，100摄氏度以上的水呈气态。对于输入的水温而言，温度值是连续型变量，其中0摄氏度和100摄氏度是等价类的边界值，如图 17 所示。其中，(-6.5,-3)、(13,46)、(110,118)是等价类固态、液态和气态的数据。

图17 不同温度下水的状态

- 2）离散型
  - 离散型是指输入数据属于离散型数据。输入数据的范围并不是一个连续的区间，而是离散地分布在区域中。
  - 离散的输入数据之间也是有关联的，一般方法是分析输入数据的属性，然后根据输入数据的一个或者多个属性特点对离散的数据进行分类，可能会得到不同的等价类。

2. 等价类

- 例如，在一个某高校的教学管理系统中，若需要依据老师的职称以及所属的学院信息做不同处理，那么职称和所属院系可以作为等价类划分的依据。依据职称可将老师分为教授、副教授、讲师等。其中（张三，李四，王五）、（赵一，钱二，孙三）是离散等价类数据，如图 18 左边的图所示。如果依据的所属院系来分，可将老师可分为信息学院、化工学院、商学院等。在职称属性中张三、李四和王五是属于一个等价类，而在赵一、李四和孙三是属于一个等价类，如图 18 右边的图所示。离散的输入数据根据不同属性有不同的等价类划分。如果进行属性叠加分类，则等价类分得更细。

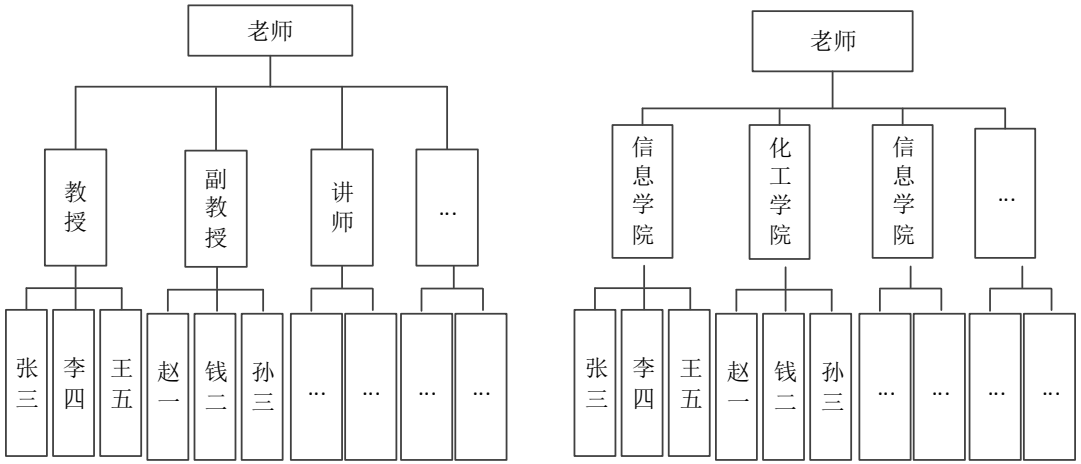


图18 不同的数据属性可以划分不同的等价类

2. 等价类

- 依据二：根据输入数据的层次关系来划分等价类，可以得到多层子等价类。
  - 实际测试的时候，先从大范围来划分等价类，可以快速得到大的等价类。如果没有达到测试的粒度要求，需要对已有的等价类进一步划分，得到子等价类。子等价类的划分也是有必要的，有些情况下的等价类没有反映出其内部的差别，导致测试不全面。
  - 例如，在教学管理系统，教授根据起级别可以做进一步的划分，例如二级教授、三级教授、四级教授等。划分的粒度如何，依据业务处理是否区分这些信息。
- 依据三：根据输入数据的维度来划分等价类，主要分为两种情况：单维和多维。
  - 单维，当输入数据只有一个，只需要分析这一个输入数据的特性，并以此分析输入数据的连续性，然后划分等价类。
  - 多维，不同于单维的输入数据，多维涉及到多个数据变量，就会存在多种组合。其中一种情况是各维度之间没有必然的关系，则应先对每一个输入变量各自划分等价类，然后将所有的维度相互组合形成新的多维等价类。另一种情况是各维度之间是有一定的关联的，各个变量的组合是表示一定的意义的（有些组合可能没有意义），则根据组合的意义来划分等价类，而不是单纯的相互组合。

2. 等价类

- 例如，用户注册信息中需要填写用户名和密码。对用户名的要求是未注册过的，则可以将用户名分为已注册的和未注册的。对密码的长度要求是6位以上20位以下，则可以将密码分为长度在6位以下、长度在6位以上20位以下，以及长度20位以上的等价类。如果同时考虑用户名和密码的关系，那么可以构成6个测试用例。
- 例如，用户登录信息中需要填写用户名和密码。登录成功的要求是用户名存在且对应的密码输入正确。可以看到这种情况下的用户名和密码是关联的。所以测试等价类可以首先分为两大类：登陆成功和登录失败。其中登录失败中又可以分为用户名不存在和用户名存在但是密码错误两个类。这种情况的等价类划分考虑的是输入变量间的组合关系，而不是简单的相互组合。例如用户名不存在和密码正确这种组合是没有意义的，不可能存在。

- 依据四：根据输出条件来划分等价类。
  - 规格说明书中并不一定对输入条件进行约束说明，很多情况下是对输出条件进行说明。不同的输出对应不同的等价类，所以输出对输入也是有一定的约束。

2. 等价类

- 2.3 等价类测试举例
  - 以NOIP1182食物链问题为例，来说明等价类测试的方法。具体的问题描述如下：
    - 动物王国中有三类动物A,B,C，这三类动物的食物链构成了有趣的环形。A吃B， B吃C，C吃A。 现有N个动物，以1 - N编号。每个动物都是A,B,C中的一种，但是并不知道它到底是哪一种。 用两种说法对这N个动物所构成的食物链关系进行描述：
      - 第一种说法是"1 X Y"，表示X和Y是同类。
      - 第二种说法是"2 X Y"，表示X吃Y。
    - 此人对N个动物，用上述两种说法，一句接一句地说出K句话，这K句话有的是真的，有的是假的。当一句话满足下列三条之一时，这句话就是假话，否则就是真话。
      - （1）当前的话与前面的真话冲突，就是假话。
      - （2）当前的话中X或Y比N大，就是假话。
      - （3）当前的话表示X吃X，就是假话。
    - 根据给定的N（1 <= N <= 50,000）和K句话（0 <= K <= 100,000），输出假话的总数。

2. 等价类

- 由上面的描述可知，程序的输入是动物的数量N和K句话，由于每一句话包含了两个动物，分别用X和Y标示，那么实际参数是N、X和Y。实际测试时，输入的一句话可以进行等价类划分，首先可以分为真话和假话两大类，不存在又不是真话又不是假话的情况。其中假话可以继续划分成3个子等价类，分别是：
  - （1）当前的话与前面的真话冲突；
  - （2）当前的话中X或Y比N大；
  - （3）当前的话表示X吃X。
- 真话的要求其实就是无法满足假话的任一条件，即必须同时满足当前的话与前面的某些真的话不冲突、当前的话中X和Y都比N小以及当前的话表示的不是X吃X。真话也可以划分为两类，分别是用第一种说法和用第二种说法。具体的等价类划分如图 19 所示，大的等价类是真话和假话，真话又根据说的话类型分为两个子等价类，同样假话根据假话的条件分为了三个子等价类。

2. 等价类

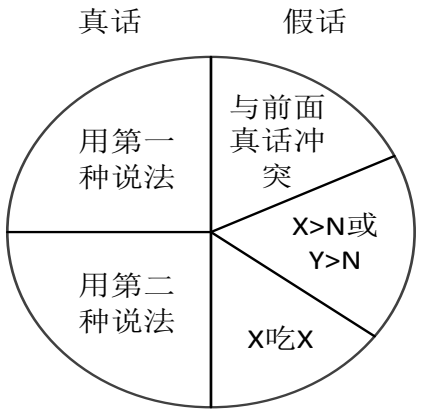


图19 第1个维度的等价类划分

表 11 第一种划分方式的测试用例设计

用例编号	测试用例	预期输出（1表示假话）
1	1 1 4（1和4是同类）	0
2	2 5 1（5吃1）	0
3	2 3 1（3吃1）	1
4	1 6 1（6和1是同类）	1
5	2 3 3（3吃3）	1

- 如果现在N=5，K=4，并且已经输入了三句话，分别是"2 1 3"（1吃3）、"2 3 2"（3吃2）以及"2 2 1"（2吃1）。最后的输出结果是假话的个数，因为前面三句都是真话，所以假话的个数是1就代表最后一句话是假，否则为真。根据上述两种等价类划分方式可以设计不同的测试用例来测试这个算法，第一种划分方式的测试用例设计如表 11 所示。

2. 等价类

- 除了上面的分类，还可以有第二种分类方式。首先把话分为用第一种说法和用第二种说法两大类。每一类又可以分为真话和假话两类。假话可以再进一步划分，但是不像上一种划分方式，都根据假话的三个条件分为三类。用不同种说法说的假话划分子类情况不同：用第二种说法说的假话仍然可根据假话的三个条件分为三类；用第一种说法说的假话可分为两类，分别满足前面两个条件，第三个条件不存在，因为第一种说法X和Y是同类，不存在X吃X的情况，则不需要再划分这个子类。具体的等价类划分如图 20 所示
- 根据第二种划分等价类的方法，可以设计相关的测试用例，如表 12 所示。

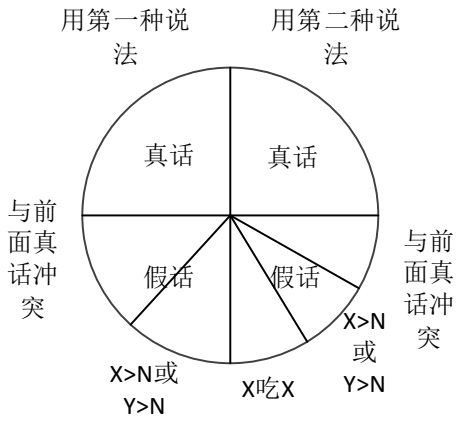


图 20 第2个维度的等价类划分

表 12 第二种划分方式的测试用例设计

用例编号	测试用例	预期输出（1表示假话）
1	1 1 4（1和4是同类）	0
2	1 2 3（2和3是同类）	1
3	1 6 1（6和1是同类	1
4	2 5 1（5吃1）	0
5	2 3 1（3吃1）	1
6	2 6 1（6吃1）	1
7	2 3 3（3吃3）	1

3. 决策表

- 3.1 决策表概念
  - 决策表也被称为判定表(Decision Table)，适合描述在不同逻辑条件取值组合的情况下需要执行的动作。决策表通常由4个部分组成，如图 21 所示。
    - 条件桩(Condition Stub):列出问题中可能出现的条件，一般情况下条件的次序是无关紧要的。
    - 动作桩(Action Stub):列出解决问题可能采取的操作，一般情况下操作的排列顺序没有约束。
    - 条件项(Condition Entry):针对所有条件的取值列出不同条件取值的组合。
    - 动作项(Action Entry):在条件项各种取值的情况下应该采取的动作。

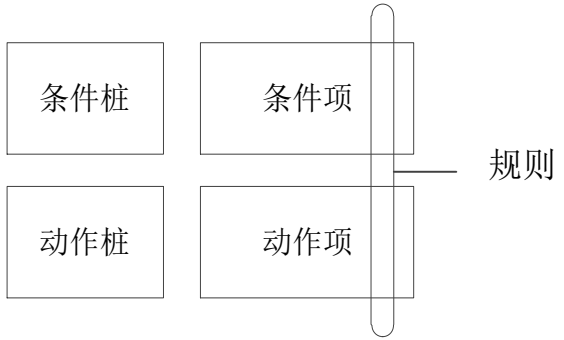


图 21 决策表结构

  - 为便于阅读可以人为指定所有动作的排列顺序，一般采用相应的“规则”进行操作。所谓的“规则”就是任何一个条件组合以及相关的执行操作，在决策表中对应为纵向贯穿条件项和动作项的一列。因此，决策表中所列出的条件取值组合就表示有多少规则。

3. 决策表

- 3.2 决策表的建立
  - 通常根据软件规格说明，按照以下步骤建立决策表。
    - （1）确定规则数目。如果有n个条件，每个条件的取值为(0,1)两种，便有 2^n 种规则。
    - （2）列出所有的条件桩和动作桩。
    - （3）输入条件项。
    - （4）输入动作项，制定初始决策表。
    - （5）简化，合并相似或者相同的动作。

3. 决策表

- Beizer指出了适合使用决策表设计测试用例的条件：
  - （1）规格说明以决策表的形式给出或者和容易转换成决策表。
  - （2）输入条件的排列顺序不影响操作执行。
  - （3）规则的排列顺序不影响操作执行。
  - （4）当某一规则的条件已满足，同时确定需要执行的操作时，无需检验其他规则。
  - （5）如果某一规则需要执行多个操作，这些操作的顺序不会造成影响。
- 决策表结构如表 13所示。

规则 / 选项		规则1	规则2	规则3、4	规则5、6	规则7	规则8
条件	c1	T	T	T	F	F	F
	c2	T	T	F	—	T	F
	c3	T	F	—	T	F	F
动作	a1	Y					
	a2		Y	Y			
	a3				Y		Y
	a4					Y	

表13 决策表的结构

3. 决策表

- 表中的c1、c2和c3表示条件桩，a1、a2、a3和a4表示动作桩，决策表的每一列对应一条规则，每条规则包括一个条件组合（条件项）以及相关的执行操作（动作项）。每个条件的取值为(0,1)两种，用T和F表示。如果有n个条件，便有 2^n 条规则。
- 决策表适用于以下的情况：
  - （1）输入变量和输出变量之间有因果关系。
  - （2）输入变量之间存在着一定的逻辑关系。
  - （3）输入域可以进行分解，且对应不同的输出。
- 用决策表可以设计测试用例，但是不是所有的情况都适合，下面举出几点适用的条件：
  - （1）通过规格说明书可以比较容易地转换成测试表
  - （2）条件的排列顺序不影响执行的动作
  - （3）规则的排列顺序不影响执行的动作
  - （4）动作的执行没有排列顺序
- 决策表不适合有执行顺序的动作或者重复的动作，在使用决策表时要考虑是否适用。

3. 决策表

- 3.3 决策表的简化
  - 建立决策表后，部分决策表可以进行简化，决策表的简化包括3个方面：
    - 1）合并相似项
      - 当两个条件项对应相同的动作，且只有一个条件的值是不同的，则可以将这两个条件项进行合并，合并后的条件项需要将不同值的条件设为不关心，用“—”表示。如图 22所示。不同值的条件对结果不影响，被称为无关项或者不关心条目，用“—”或“N/A”表示。

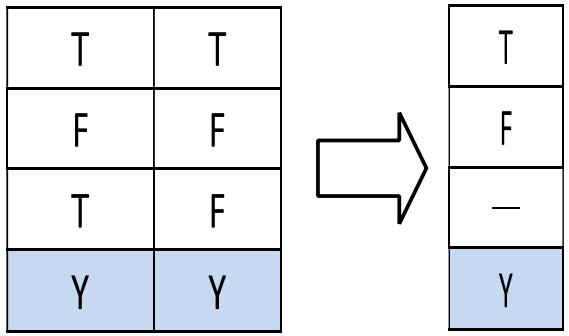


图22 合并相似项

3. 决策表

- 2）合并包含项
  - 如果一个条件项已经包括了另一个的条件项，执行的动作相同则也可以进行合并，如图 23 所示。一般情况下，当有无关项出现，则需要注意是否有其他的条件项已经被其包含，如果包含，则保留无关项，删除包含项。

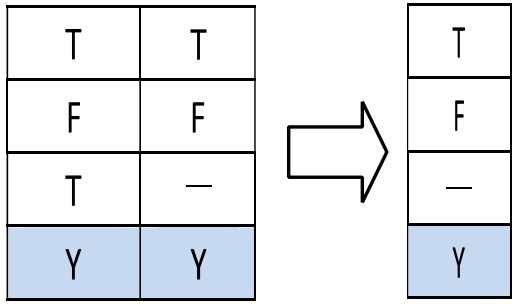


图23 合并包含项

表 14 合并不可能项

选项 \ 规则		规则1	规则2	规则3
条件	c1	T	—	—
	c2	—	T	—
	c3	—	—	T
动作	a1			
	a2			
	a3			

- 3）合并不可能项
  - 如果决策表中有相互排斥条件，则当一个选项为真时，其他几个条件就一定是假的。简化时将其他条件用“—”表示。这里的不关心条目表示“必须失败”，而不是随意取值。例如，有三个条件c1、c2和c3是3个互斥的条件，则决策表如表 14所示。规则1的含义是当c1为真时，c2和c3一定是假的。在设计测试用例时要注意“—”的具体含义。

3. 决策表

- 合并条件项的过程其实就是对决策表进行简化，方便测试用例的设计。下面举一个例子说明规则的合并过程。
- 现有一台ATM自动取款机，插入银行卡后必须输入正确的密码才能取款。如果账户内的余额不足会提示卡内余额不足，如果ATM内现金不足会提示ATM内现金不足。具体的决策表如表15所示。

表15 ATM决策表

选项 \ 规则		1	2	3	4	5	6	7	8
条件	c1：密码是否正确	T	T	T	T	F	F	F	F
	c2：账户内余额是都足够	T	T	F	F	T	T	F	F
	c3：ATM内现金是否足够	T	F	—	F	T	F	T	F
动作	a1：取款成功	Y							
	a2：提示密码错误					Y	Y	Y	Y
	a3：提示卡内余额不足			Y	Y				
	a4：提示ATM内现金不足		Y						

- 分析上面的表，可以发现条件项3包含了条件项4，则可以直接删除规则4。条件项5和6是相似项，对应相同的动作，只有c3不同，则可以将这两个条件项合并。同理条件项6和7是相似项，也可以进行合并。这里还发现一个有趣的现象，条件项5和条件项7也是相似项，条件项6和条件项8也是相似项，将这四个条件项合并，得到最终的决策表如表16 所示。

表16 化简后的ATM决策表

选项 \ 规则		1	2	3，4	5-8
条件	c1：密码是否正确	T	T	T	F
	c2：账户内余额是都足够	T	T	F	—
	c3：ATM内现金是否足够	T	F	—	—
动作	a1：取款成功	Y			
	a2：提示密码错误				Y
	a3：提示卡内余额不足			Y	
	a4：提示ATM内现金不足		Y		

3. 决策表

- 3.4 决策表规则数统计
  - 完备决策表（所有规则都列出来）的规则条目就是决策表中列的数目，即一列表示一条规则。但实际分析中，经常会将决策表进行简化，就会有无关项，从而影响规则条数统计。下面将进行具体分析。
    - 原则一：对于没有互相排斥条件的决策表，如果有n个条件，则产生条规则。如果决策表中没有无关项，每一列对应一个规则；如果决策表中有无关项，则每出现一个无关项，该列的规则数目乘以2。
    - 原则二：对于有互相排斥条件的决策表，如果决策表中没有无关项，则和原则一一样，每一列对应一个规则；如果决策表中有无关项，则需要分析无关项的含义。如果无关项的含义是条件无关，则仍然是每出现一个无关项，该列的规则数目乘以2。如果无关项的含义是“必须失败”，需要扩展决策表使决策表完备，然后再计算规则数，不能直接计算。

3. 决策表

- 例如，表17显示的是含有互相排斥条件的决策表，如果直接按照原则一，决策表中增加了规则条数统计这一行，最后得到的规则条数是4+4+4=12。
- 这12个规则中是有重复的规则，同时又没有覆盖所有的规则，如表18所示。

表17 含有互斥条件的决策表

选项 \ 规则		规则1	规则2	规则3
条件	c1	T	—	—
	c2	—	T	—
	c3	—	—	T
规则条数统计		4	4	4
动作	a1			
	a2			
	a3			

表18 将表17展开以后的决策表

选项 \ 规则		1	2	3	4	5	6	7	8	9	10	11	12
条件	c1	T	T	T	T	T	T	F	F	T	T	F	F
	c2	T	T	F	F	T	T	T	T	T	F	T	F
	c3	T	F	T	F	T	F	T	F	T	T	T	T
规则条数统计		1	1	1	1	1	1	1	1	1	1	1	1
动作	a1												
	a2												
	a3												

3. 决策表

- 从上表中可以看到，规则1、规则5和规则9是重复的，规则2和规则6是重复的，规则3和规则10是重复的，规则7和规则11 是重复的。去掉重复的规则，最后得到12-5=7条规则。实际上表中还缺少了所有条件都为假的规则，所以实际上的规则数是8条。所以，必须按照原则二来统计规则条数。需要扩展决策表到原始的完备表，删除重复的规则和增加缺少的规则，如表19所示。

表19 实际的决策表

选项 \ 规则		1	2	3	4	5	6	7	8
条件	c1	T	T	T	T	F	F	F	F
	c2	T	T	F	F	T	T	F	F
	c3	T	F	T	F	T	F	T	F
规则条数统计		1	1	1	1	1	1	1	1
动作	a1								
	a2								
	a3								

3. 决策表

- 3.5 规则标特性
  - 决策表直接会影响到测试用例的设计，在建立决策表阶段需要关注决策表是否满足3个特性：完备性、无冗余性和一致性。
  - 首先，完备性是指决策表中需要包括所有的规则。例如，表20表示的决策表没有满足完备性，缺少了全部为假的条件项，则需要增加这一条件项。

表20 不完备的决策表

选项 \ 规则		1	2	3	4	5
条件	c1	T	T	T	F	F
	c2	T	F	F	—	T
	c3	—	T	F	T	F
规则条数统计		1	1	1	1	1
动作	a1	Y		Y		
	a2		Y			Y
	a3				Y	

3. 决策表

- 其次，无冗余性是指决策表中没有重复的规则。例如，表21中的条件项1包含了条件项2，而且它们对应的动作是一致的，决策表存在冗余。需要修改决策表，删除重复的条件项。直接冗余发生的比较少但也有可能发生，例如决策表中有完全相同的两列。发生冗余通常和无关项有关，无关项包含很多种情况，如果把包含的情况也列出来就会出现冗余。关注无关项的冗余是检查决策表是否冗余很重要的的一点。

表21 冗余决策表

选项 \ 规则		1	2	3	4	5
条件	c1	T	T	T	T	F
	c2	T	F	T	F	—
	c3	—	T	T	F	—
规则条数统计		1	1	1	1	1
动作	a1	Y	Y			
	a2			Y		Y
	a3				Y	

3. 决策表

- 最后，一致性是指决策表中没有相互冲突的规则，表示两个规则的条件项相同或者一个条件项包含另一个条件项，但是执行的动作不同。表22中条件项1包含了条件项2，但是它们执行的动作不同，说明决策表不一致，需要分析实际情况再重新设计规则。

表22 不一致的决策表

选项 \ 规则		1	2	3	4	5
条件	c1	T	T	T	T	F
	c2	T	F	T	F	—
	c3	—	T	T	F	—
规则条数统计		1	1	1	1	1
动作	a1	Y				
	a2		Y	Y		Y
	a3				Y	

3. 决策表

- 3.6 决策表测试用例设计
  - 本节以一个货运计费系统为例来说明决策表测试用例的设计。
  - 货运收费标准如下：如果收货地点在本省以内，重量小于15公斤，快件10元/公斤，慢件5元/公斤；重量大于15公斤，超出部分每公斤收费增加3元/公斤。如果收货地点在本省以外，重量小于15公斤，快件15元/公斤，慢件10元/公斤；重量大于15公斤，超出部分每公斤收费增加3元/公斤（重量用字母W表示）。
  - 根据以上描述分析得到，收费的条件有3个：
    - c1：是否在本省以内
    - c2：是否快件
    - c3：重量是否超出15公斤

3. 决策表

- 将上述条件两两组合可以得到8种情况，但因为省内快件和省外慢件收费方式相同，实际得到的收费方式有6种：
  - a1：5 × W
  - a2：10 ×W
  - a3：5 × W ＋ 3 × (W －15)
  - a4：10 × W ＋ 3 × (W －15)
  - a5：15 × W
  - a6：15 × W ＋ 3 × (W －15)
- 根据条件和动作（收费）之间的关系可以得到如表23所示的决策表。

3. 决策表

表23 货运计费问题的决策表

	条件	1	2	3	4	5	6	7	8
条件	c1：是否在本省以内?	T	T	T	T	F	F	F	F
	c2：是否快件?	F	F	T	T	F	F	T	T
	c2：重量是否超出15公斤?	F	T	F	T	F	T	F	T
收费	a1：	Y							
	a2：			Y		Y			
	a3：		Y						
	a4：				Y		Y		
	a5：							Y	
	a6：								Y

- 根据上述决策表可以设计一组测试用例，如表24所示。

表24 货运问题的测试用例

用例编号	测试用例	预期输出（收费）
1	省内慢件，10公斤	
2	省内慢件，20公斤	
3	省内快件，10公斤	
4	省内快件，20公斤	
5	省外慢件，10公斤	
6	省外慢件，20公斤	
7	省外快件，10公斤	
	省外快件，20公斤	

4. 因果图

- 4.1 因果图概念
  - 因果图，顾名思义，就是包含了原因、结果以及它们之间关系的一种图。在因果图中，原因和原因之间、原因和结果之间都存在一定的关联。原因是指输入条件或者输入条件的等价类，结果是指输出条件。
  - 在因果图设计测试用例的过程中，需要用一定的图形符号来表示一些约束和限制。表示原因和结果之间关系有四中，分别是恒等、非、或和与，具体如图24所示。左侧的节点表示原因（输入条件）的状态，用C1表示，右侧的节点表示结果（输出条件）的状态，用E1表示。用一条直线连接两个节点，并用相对应的逻辑符号来表示原因和结果之间的逻辑关系。节点状态可取“0”或者“1”,其中“0”表示“不存在”状态，“1”表示“存在”状态。

4. 因果图

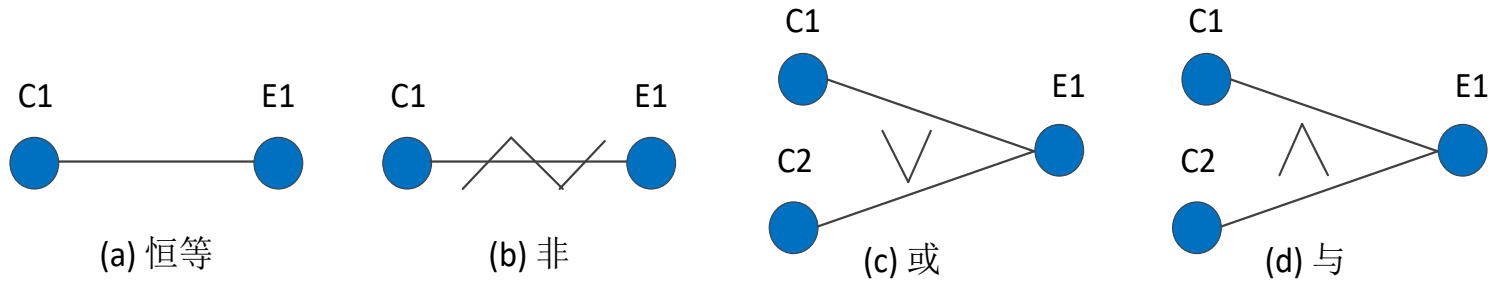
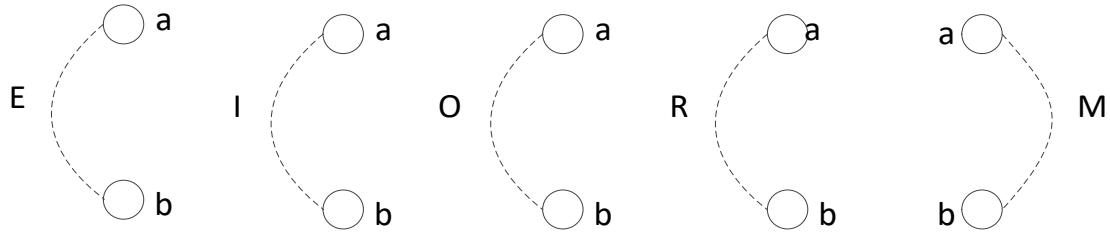


图24 因果图的图形符号

- 1）恒等（identity）：表示原因和结果一一对应关系。如果C1=0，则E1=0；如果C1=1，则 E1=1。
- 2）非（not）:表示原因和结果的相反关系。如果C1=0，则E1=1；如果C1=1，则E1=0。
- 3）或（or）：表示多个原因中只要有一个是存在状态，则结果就存在。如果C1或C2是1，则E1是1，否则E1是0。
- 4）与（and）：表示多个原因都是存在状态，结果才存在。如果C1和C2都是1，则E1是1，否则E1是0。

4. 因果图

- 除了原因和结果之间存在联系，各原因之间以及各结果之间也会存在一些联系，这些联系称为“约束”。在因果图中，有相应的约束符号表示各种约束，例如互斥、包含、唯一、要求和屏蔽。具体的约束符号表示如图25所示。



(a)E（互斥） (b)I（包含） (c)O（唯一） (d)R（要求） (e)M（屏蔽）

图25 因果图的约束符号

- （1）异（E约束）：表示多个原因中最多有一个是成立的。即a和b中最多有一个是1，可能都是0，但不可能都是1。
- （2）或（I约束）：表示多个原因中至少有一个是成立的。即a和b中至少有一个是1，可能都是1，但不可能都是0,。
- （3）唯一（O约束）：表示多个原因中有且只有一个是成立的。即a=1，b=0或者a=0，b=1，a和b中有且只有一个为1。
- （4）要求（R约束）：表示多个原因同时成立或者同时不成立。即a和b同时为0或者1。
- （5）强制（M约束）：表示a=1时，b强制为0。其他情况下没有约束。

4. 因果图

- 4.2 因果图设计
  - 利用因果图设计测试用例的具体步骤如下：
    - 1）阅读软件规格说明书并进行分析，找到原因和结果，即输入条件和输出条件，用统一的标识符对原因和和结果进行标识。
    - 2）仔细分析软件规格说明书中的语义，确定各原因之间、各结果之间以及原因和结果之间的关系，并将这些关系转化成因果图。
    - 3）因为原因和原因、结果和结果以及原因和结果之间有关联，需要在因果图中用一些图形符号来表示这些约束和限制。
    - 4）分析因果图中的各种条件关系，将其转化为判定表。
    - 5）根据判定表中的每一列数据设计一个测试用例。

4. 因果图

- 4.3 利用因果图设计测试用例
  - 以升降横移类立体车库为例，如图26所示。它是采用载车板升降或横移来存取车辆，每个立体车库有一个位置是没有载车板（空位），是为了给载车板的移动提供移动空间。现在具体分析两层的升降横移类立体车库，如图所示。由图可知，两层的车位中至少一个是空位，用于其他载车板的移动。第一层停放的车辆可以直接取车，第二层的车辆需要下降到第一层再取车。注意，在下降载车板时要确保下面是空位。如果不是，则需要移动其它的载车板来制造这种情况。停入车库的每辆车都有相应的编号，当需要取车时，会自动根据编号来获取车的位置和空位的位置并通过移动载车板将车从车库取出。

4. 因果图

- 在不同位置上的车子移动的方向，如图27所示。

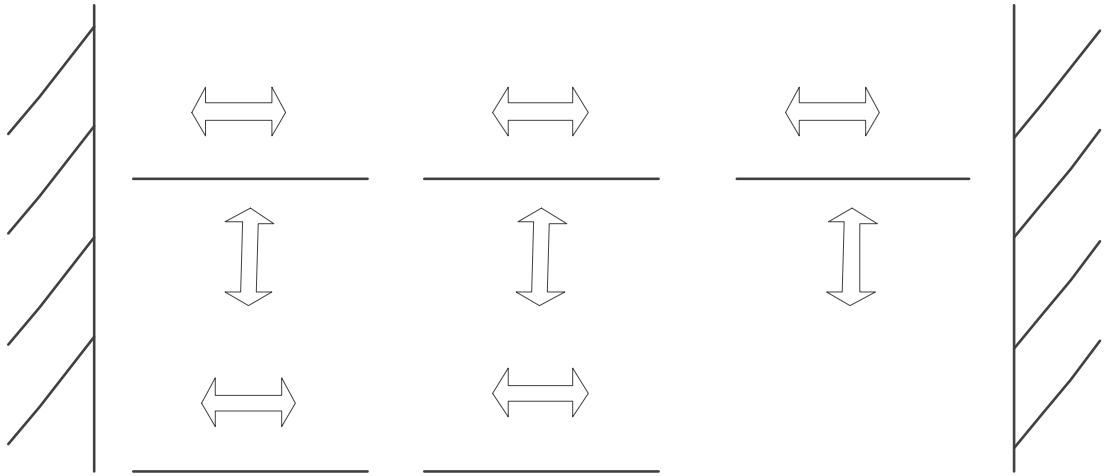


图27 车子移动方向示意图

4. 因果图

- 下面对取车的过程分析原因和结果，记录产生的中间状态，如表25所示。

表25 取车过程的原因和结果

原因	C1：取车位在第一层 C2：取车位在第二层 C3：空位在第一层 C4：空位在第二层 C5：空位在取车位正下方 C6：空位不在取车位正下方
中间状态	L1：开始时取车位在第一层且空位在第一层 L2：开始时取车位在第一层且空位在第二层 L3：开始时取车位在第二层且空位在取车位正下方 L4：开始时取车位在第二层且空位在第一层但不在取车位下方 L5：开始时取车位在第二层且空位在第二层 L6：取车位在第一层且空位在第二层 L7：取车位在第二层且空位在取车位正下方 L8：取车位在第二层且空位在第一层但不在取车位下方
结果	E1：直接取车

4. 因果图

- 根据原因和结果的分析，可以得到因果图，如图28所示。

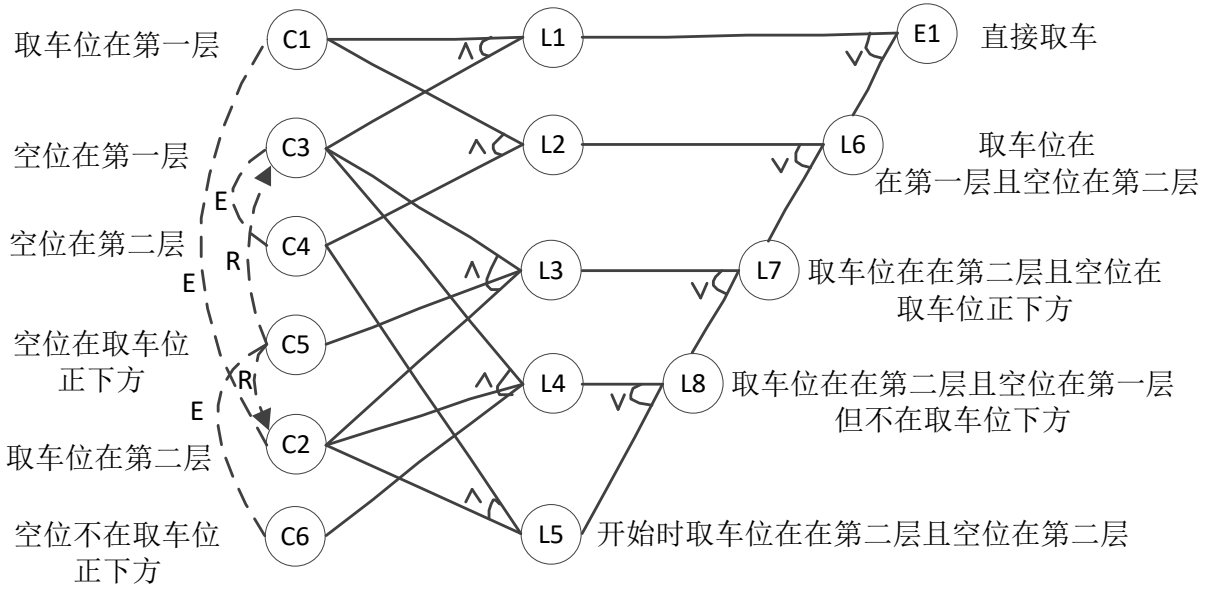


图28 取车过程的因果图

- 注意：从中间节点L5到L8，需要上升空位下方载车板让出下层空位；从中间节点L8到L7，需要横移其他载车板让出取车位下方空位；从中间节点L7到L6，需要下降取车位到第一层。

4. 因果图

- 根据因果图可以得到相应的决策表，如表26取车问题的决策表所示。

表26 取车问题的决策表

序号			1	2	3	4	5	6	7	8	9	10
输 入		C1：取车位在第一层	1	1	1	0	0	0	0	0	0	0
		C2：取车位在第二层	0	0	0	1	1	1	1	0	0	0
		C3：空位在第一层	1	0	0	1	1	0	0	1	0	0
		C4：空位在第二层	0	1	0	0	0	1	0	0	1	0
		C5：空位在取车位正下方	—	—	—	1	0	—	—	—	—	—
		C6：空位不在取车位正下方	—	—	—	0	1	—	—	—	—	—
中间节点	开始状态	L1：开始时取车位在第一层且空位在第一层	1	0	0	0	0	0	0	0	0	0
		L2：开始时取车位在第一层且空位在第二层	0	1	0	0	0	0	0	0	0	0
		L3：开始时取车位在第二层且空位在取车位正下方	0	0	0	1	0	0	0	0	0	0
		L4：开始时取车位在第二层且空位在第一层但不在取车位下方	0	0	0	0	1	0	0	0	0	0
		L5：开始时取车位在第二层且空位在第二层	0	0	0	0	0	1	0	0	0	0
	中间状态	L6：取车位在第一层且空位在第二层	0	1	0	1	1	1	0	0	0	0
		L7：取车位在第二层且空位在取车位正下方	0	0	0	1	1	1	0	0	0	0
		L8：取车位在第二层且空位在第一层但不在取车位下方	0	0	0	0	1	1	0	0	0	0
输出		E1：直接取车	1	1	0	1	1	1	0	0	0	0

4. 因果图

- 通过上面的决策表可以设计相关的测试用例，如表27所示。

表27 取车问题的测试用例

用例编号	测试用例	预期输出
1	开始时取车位在第一层且空位在第一层	直接取车
2	开始时取车位在第一层且空位在第二层	直接取车
3	开始时取车位在第二层且空位在取车位正下方	直接取车
4	开始时取车位在第二层且空位在第一层但不在取车位下方	直接取车
5	开始时取车位在第二层且空位在第二层	直接取车