

实验 3 因果图法与用例设计

1. 实验目标

- (1) 能够依据需求分析原因和结果。
- (2) 能够绘制因果图,标注相应的关系符号。
- (3) 能够将因果图转化成判定表。
- (4) 能够使用因果图法进行测试用例设计。

2. 背景知识

需求: 某旅馆住宿系统可办理房间选定、房款支付及房间管理相关业务。其需求描述如下: 游客的情况分为支付全部房款(即预期入住天数内所有房款)和支付部分房款(仅支付定金)。可选择“单人间”“双人间”或“豪华间”,若某类型房间有空房,则相应类型的房间被开启;若某类型房间无空房,则“房间已满”提示灯亮。无空房时,支付部分房款的游客选择该类型的房间,则该类型房间不被开启且提示办理退款。若此期间,该类型房间有客人退房,则“房间已满”提示灯灭,该类型房间的某间房被开启的同时提示房款不足。

界面原型: 旅馆住宿系统业务办理页面如图 3.1 所示。



图 3.1 旅馆住宿系统业务办理页面

首先,基于上述需求,思考采用等价类划分法如何进行测试用例设计。读者采用等价类划分法进行上述需求的测试用例设计时,不难发现设计出的测试用例存在如下特点。其一,数量甚少。其二,仅着重考虑了各项输入条件,并未考虑输入条件的各种组合情况。例如,选择不同的房款支付方式及房间类型,在“房间已满”和“房间空余”的不同前提下,产生的结果会有所差异,此情况在等价类划分法中并未涉及和考虑。其三,未考虑各输入情况之间的相互制约关系。例如,“支付房款”与“支付定金”不能同时选择,最多只能选择一个;“单人间”“双人间”和“豪华间”不能同时选择,最多仅能选择一个等,上述列举的两种情况在等价

类划分法中也并未涉及和考虑。

再如某保险公司的预约投保系统,界面原型如图 3.2 所示。读者针对此界面原型采用等价类划分法进行测试用例设计时,同样会忽略多种关系不能同时存在的情况。例如,“称谓”字段中“先生”与“女士”不能同时成立,最多仅能成立两者之一;“所在地”字段中,当选择某市时,该市所属省份必须同步出现,不能有选择某市后该市所属省份不出现的情况发生;“联系电话”字段中,“固定电话”“小灵通”及“手机号”至少填写一个即可。

预约投保

称谓: ☒ 先生 ☐ 女士

姓名:

所在地: 省 市 *

Email:

联系电话(请至少填写一项): *

固定电话:

小灵通:

手机号:

验证码: *

确定 重置

图 3.2 预约投保系统界面原型

读者充分理解了上述两实例后,则不难理解仅采用等价类划分法并不能很好地处理“当系统中输入项之间以及输入项与输出之间存在多种关系”时的测试用例设计问题,这正是引入因果图法的主要原因。

因果图法是从需求中找出因(输入条件)和果(输出或程序状态的改变),通过分析输入条件之间的关系(组合关系、约束关系等)以及输入和输出之间的关系绘制出因果图,再转化成判定表,从而设计出测试用例的方法,如图 3.3 所示。不难理解,该方法主要适用于各种输入条件之间存在某种相互制约关系或输出结果依赖于各种输入条件的组合时的情况。



图 3.3 因果图法介绍

下面以图 3.4 为例,对因果图进行初步介绍。

通过观察可以发现,图 3.4 中无法识别的符号较多,如 E、V 等。下面结合图 3.5,对因果图中的常用符号含义进行介绍。

因果图符号的种类繁多,常用符号介绍如下。

(1) CI: 原因。I 取“0”表示状态不出现,“1”表示状态出现,若有多状态,可用大于 1 的多个值表示。

(2) EI: 结果。I 取“0”表示状态不出现,“1”表示状态出现,若有多状态,可用大于 1 的多个值表示。

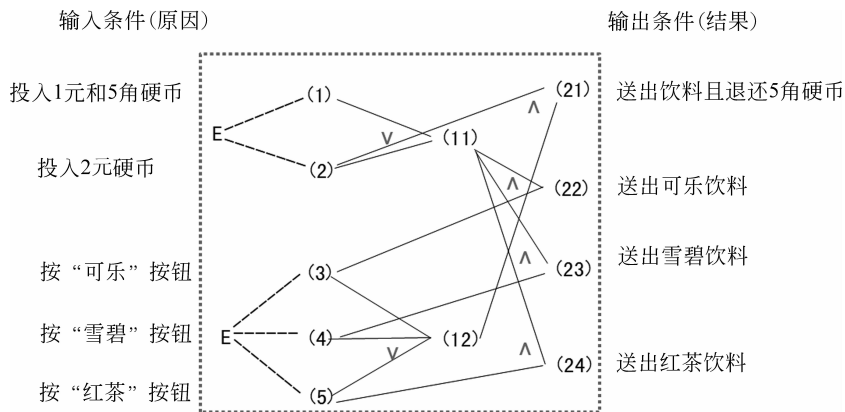


图 3.4 因果图初识

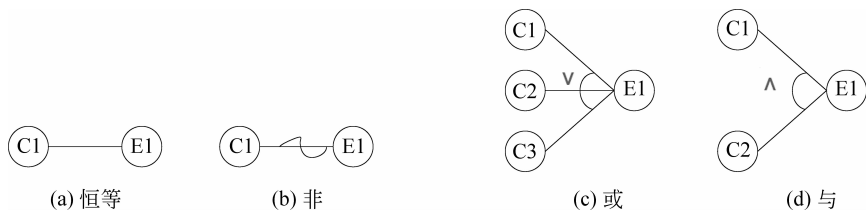


图 3.5 因果图符号

(3) 恒等：原因结果同时出现。

(4) 非：原因出现，结果不出现；原因不出现，结果出现。

(5) 或：只要有一个原因出现，结果就出现；原因都不出现，结果就不出现。

(6) 且：原因都出现，结果才出现。

注意：图 3.5 所示的每个结点表示一个状态。

为了表示原因与原因之间、结果与结果之间可能存在的约束条件，因果图中通常还附加一些表示约束条件的符号，如图 3.6 所示。

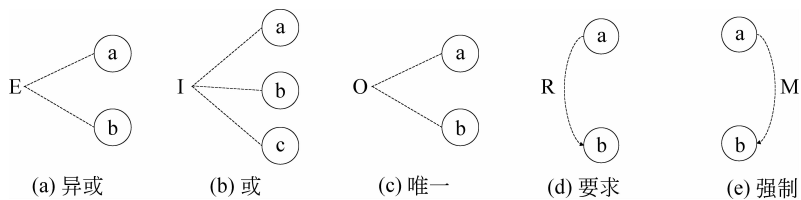


图 3.6 带约束条件的因果图符号

约束符号也包含多种类型，分别从输入考虑和从输出考虑两方面进行归类如下。

(1) 从输入考虑。

① E(互斥/异或)：表示 a、b 两个原因不会同时成立，最多只有一个成立。

② I(包含)：a、b、c 三个原因中至少有一个必须成立。

③ O(唯一)：a、b 两个原因中必须有一个，且仅有一个成立。

④ R(要求)：当 a 出现时，b 必须也出现，不可能出现 a 而不出现 b。

(2) 从输出考虑。

M(强制或屏蔽): a 是 1 时, b 必须是 0; a 是 0 时, b 的值不确定。

下面再次对某保险公司的预约投保系统的“预约投保”页面进行分析, 结果如图 3.7 所示。

预约投保

称谓: ☒ 先生 ☐ 女士

姓名: *

所在地: 宁夏 省 银川 市 *

Email: *

联系电话(请至少填写一项): *

固定电话:

小灵通:

手机号:

验证码: 2372 *

确定 重置

E (互斥/异或)
若必填: 0 (唯一)

R (要求)

I (包含)

图 3.7 预约投保页面各字段关系分析

到目前为止, 已经强调了因果图中各符号的含义, 究竟如何使用因果图法进行测试用例设计呢? 可参照如下步骤。

- (1) 分析需求, 提取原因和结果, 并赋予标识符;
- (2) 分析需求, 提取因果关系, 并表示成因果图;
- (3) 标明因果图中的约束条件;
- (4) 将因果图转换成判定表;
- (5) 为判定表中每一列表示的情况设计测试用例。

注意: 原因常常是输入条件或输入条件的等价类; 结果常常是输出条件。

下面以旅馆住宿系统为例, 针对忽略房间状态和考虑房间状态两种不同的需求情况, 采用因果图法进行测试用例设计。

3. 实验任务

【任务 3.1】 旅馆住宿系统测试用例设计(忽略房间状态)。

需求: 某旅馆住宿系统可办理房间选定、房款支付及房间管理相关业务, 系统默认房间资源始终保持充足的状态。其需求描述如下: 游客的情况分为支付全部房款(即预期入住天数内所有房款)和支付部分房款(仅支付定金)。选择“单人间”“双人间”或“豪华间”, 则相应类型的房间被开启。若游客支付的房款不足, 则在开启房间的同时系统提示房款不足。

界面原型: 旅馆住宿系统业务办理页面(忽略房间状态)如图 3.8 所示。

问题: 采用因果图法进行测试用例设计。



图 3.8 旅馆住宿系统业务办理页面(忽略房间状态)

前提条件：对需求进行分析后可发现，该需求的输入项与输入项之间以及输入项与结果之间存在多种关系，此时采用因果图法进行测试用例设计更为合适。

第 1 步，分析需求，找出原因和结果，即输入和输出。

原因：

- (1) 游客支付全部房款。
- (2) 游客支付部分房款。
- (3) 游客选择“单人间”。
- (4) 游客选择“双人间”。
- (5) 游客选择“豪华间”。

结果：

- (21) 该类型房间被打开且提示房款支付不足。
- (22) 某单人间被打开。
- (23) 某双人间被打开。
- (24) 某豪华间被打开。

第 2 步，绘制因果图，并标注相应的关系符号，如图 3.9 所示。图 3.9 中，所有原因结点显示于左侧，所有结果结点显示于右侧，中间结点表示处理的中间状态。

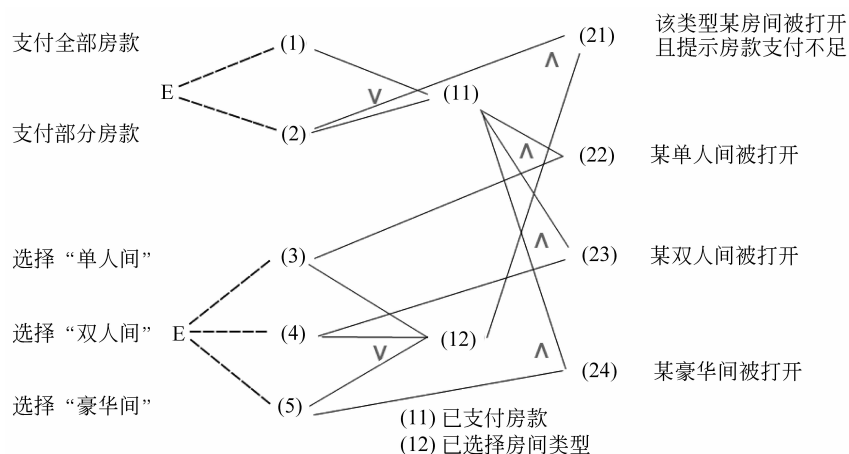


图 3.9 业务办理_因果图(忽略房间状态)

中间结点：

(11) 已支付房款。

(12) 已选择房间类型。

注意：中间结点的设立并非必须要完成的工作，但是它的设立可使绘制出的因果图更简单和美观，阅读起来也较为方便。

第 3 步，转换成判定表，如表 3.1 所示。

表 3.1 业务办理_判定表(忽略房间状态)

输入条件	游客支付全部房款	(1)	1	1	1	1	0	0	0	0	0	0	0
	游客支付部分房款	(2)	0	0	0	0	1	1	1	1	0	0	0
	游客选择“单人间”	(3)	1	0	0	0	1	0	0	0	1	0	0
	游客选择“双人间”	(4)	0	1	0	0	0	1	0	0	0	1	0
	游客选择“豪华间”	(5)	0	0	1	0	0	0	1	0	0	0	1
中间结果	已支付房款	(11)	1	1	1	1	1	1	1	1	0	0	0
	已选择房间类型	(12)	1	1	1	0	1	1	1	0	1	1	1
输出结果	该类型房间被打开且提示房款支付不足	(21)	0	0	0	0	1	1	1	0	0	0	0
	某单人间被打开	(22)	1	0	0	0	1	0	0	0	0	0	0
	某双人间被打开	(23)	0	1	0	0	0	1	0	0	0	0	0
	某豪华间被打开	(24)	0	0	1	0	0	0	1	0	0	0	0
测试用例			Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

第 4 步，可将表 3.1 作为确定测试用例的依据。设计测试用例如表 3.2 所示。

表 3.2 业务办理_测试用例设计(忽略房间状态)

编号	输 入	预 期 结 果
1	游客支付全部房款，选择“单人间”	某单人间被打开
2	游客支付全部房款，选择“双人间”	某双人间被打开
3	游客支付全部房款，选择“豪华间”	某豪华间被打开
4	游客支付全部房款，未选择任何类型的房间	所有房间均不被打开
5	游客支付部分房款，选择“单人间”	某单人间被打开且系统提示房款支付不足
6	游客支付部分房款，选择“双人间”	某双人间被打开且系统提示房款支付不足
7	游客支付部分房款，选择“豪华间”	某豪华间被打开且系统提示房款支付不足
8	游客支付部分房款，未选择任何类型的房间	所有房间均不被打开
9	游客不进行支付，选择“单人间”	所有房间均不被打开
10	游客不进行支付，选择“双人间”	所有房间均不被打开
11	游客不进行支付，选择“豪华间”	所有房间均不被打开

【任务 3.2】 旅馆住宿系统测试用例设计(考虑房间状态)。

需求：某旅馆住宿系统可办理房间选定、房款支付及房间管理相关业务。其需求描述如下：游客的情况分为支付全部房款(即预期入住天数内所有房款)和支付部分房款不足(仅支付定金)。可选择“单人间”“双人间”或“豪华间”，若某类型房间有空房，则相应类型的房间被开启；若某类型房间无空房，则“房间已满”提示灯亮。无空房时，支付部分房款的游客选择该类型的房间，则该类型房间不被开启且提示办理退款。若此期间，该类型房间有客人退房，则“房间已满”提示灯灭，该类型房间的某间房被开启的同时提示房款不足。

界面原型：旅馆住宿系统业务办理页面(考虑房间状态)如图 3.10 所示。



图 3.10 旅馆住宿系统业务办理页面(考虑房间状态)

问题：采用因果图法进行测试用例设计。

前提条件：对需求进行分析后可发现，该需求的输入项与输入项之间以及输入项与结果之间存在多种关系，此时采用因果图法进行测试用例设计更为合适。

第 1 步，分析需求说明，找出原因和结果，即输入和输出。

原因：

- (1) 该类型房间有空房。
- (2) 游客支付部分房款。
- (3) 游客支付全部房款。
- (4) 游客选择“单人间”。
- (5) 游客选择“双人间”。
- (6) 游客选择“豪华间”。

结果：

- (21) 该类型房间“房间已满”灯亮。
- (22) 提示办理退款。
- (23) 提示房款不足。
- (24) 某单人间被打开。
- (25) 某双人间被打开。
- (26) 某豪华间被打开。

第 2 步，绘制因果图，并标注相应关系符号，如图 3.11 所示。图 3.11 中，所有原因结点显示于左侧，所有结果结点显示于右侧，中间结点表示处理的中间状态。

(14) 房款已支付。

该类型房间有空房

支付部分房款

支付全部房款

选择“单人间”

选择“双人间”

选择“豪华间”

(1)

(2)

(3)

(11)

(12)

(13)

(14)

(21)

(22)

(23)

(24)

(25)

(26)

该类型房间“房间已满”灯亮

提示办理退款

提示房款不足

某单人间被打开

某双人间被打开

某豪华间被打开

(11) 支付房款不足且已选择房间类型

(12) 已选择房间类型

(13) 该类型房间有空房并且提示房款支付不足

(14) 房款已支付

续表

编号	输 入	预 期 结 果
10	游客不进行支付,选择“双人间”且有空房	所有房间均不被打开且“房间已满”灯为灭的状态
11	游客不进行支付,未选择任何类型的房间且有空房	所有房间均不被打开且“房间已满”灯为灭的状态
12	游客不进行支付,选择“豪华间”且有空房	所有房间均不被打开且“房间已满”灯为灭的状态
13	游客支付部分房款,选择“单人间”且没有空房	“房间已满”灯为亮的状态且系统提示办理退款
14	游客支付部分房款,选择“双人间”且没有空房	“房间已满”灯为亮的状态且系统提示办理退款
15	游客支付部分房款,未选择任何类型的房间	“房间已满”灯为亮的状态
16	游客支付部分房款,选择“豪华间”且没有空房	“房间已满”灯为亮的状态且系统提示办理退款
17	游客支付全部房款,选择“单人间”且没有空房	“房间已满”灯为亮的状态且系统提示办理退款
18	游客支付全部房款,选择“双人间”且没有空房	“房间已满”灯为亮的状态且系统提示办理退款
19	游客支付全部房款,未选择任何类型的房间	“房间已满”灯为亮的状态
20	游客支付全部房款,选择“豪华间”且没有空房	“房间已满”灯为亮的状态且系统提示办理退款
21	游客不进行支付,选择“单人间”且没有空房	所有房间均不被打开且“房间已满”灯为亮的状态
22	游客不进行支付,选择“双人间”且没有空房	所有房间均不被打开且“房间已满”灯为亮的状态
23	游客不进行支付,未选择任何类型的房间且没有空房	所有房间均不被打开且“房间已满”灯为亮的状态
24	游客不进行支付,选择“豪华间”且没有空房	所有房间均不被打开且“房间已满”灯为亮的状态

注意:需求中描述当房间没有空余时,“房间已满”灯亮。但是,读者会发现界面原型中并未显示“灯”。在此,值得一提的是,实际项目中界面原型可能会采用其他方式来实现需求说明中的要求,所采取的表现方式或许更加易于理解和使用。建议开发人员确定界面原型后,及时与客户进行沟通并确认,便于后续工作在此基础上顺利开展。

思考:若此处采用等价类划分法设计测试用例,又该如何考虑呢?

4. 拓展训练

【练习 3.1】 采用因果图法针对以下需求进行测试用例设计。

需求: 输入的第一个字符必须是#或*,第二个字符必须是一个数字,此情况下进行文件的修改。若第一个字符不是#或*,则给出信息 N;若第二个字符不是数字,则给出信息 M。

【练习 3.2】 采用因果图法针对以下需求进行测试用例设计。

需求: 有一个自动售货机,若投入 5 角或 1 元的硬币,按下“橙汁”或“啤酒”按钮,则相应的饮料就送出来。若售货机没有零钱,则显示“零钱找完”的红灯亮,这时再投入 1 元硬币并按下按钮后,饮料不送出来而且 1 元硬币也退出来;若售货机有零钱,则显示“零钱找完”的红灯灭,投入 1 元硬币并按下按钮后,在送出饮料的同时退还 5 角硬币。

实验 4 决策表法与用例设计

1. 实验目标

- (1) 理解决策表法的原理。
- (2) 能够使用决策表法进行测试用例设计。
- (3) 能够在真实项目中灵活运用决策表法。

2. 背景知识

在使用因果图法进行测试用例设计的过程中用到了判定表。判定表又称决策表,是使用决策表法进行测试用例设计的核心,是分析和表达多逻辑条件下执行不同操作的情况的有效工具。因此,决策表法是一种能够将复杂逻辑关系和多条件组合情况表达得较为明确的方法,适用于程序中输入、输出较多或输入与输出之间相互制约条件较多的情况。在所有黑盒测试方法中,基于决策表法的测试是最严格、最具有逻辑性的。

下面通过表 4.1 所示实例加以说明。

表 4.1 决策表实例

问题与建议		1	2	3	4	5	6	7	8
问题	是否劳累	Y	Y	Y	Y	N	N	N	N
	是否喜欢	Y	Y	N	N	Y	Y	N	N
	是否难理解	Y	N	Y	N	Y	N	Y	N
建议	重听一遍					√			
	继续进行						√		
	进行下一题							√	√
	休息	√	√	√	√				

不难理解,决策表能够依据各种可能的情况将看似复杂的问题全部罗列出来,简明且无遗漏。同理可得,在软件测试中,利用决策表法也能够设计出完整的测试用例集合。

图 4.1 所示为决策表模型图。

决策表模型图中包含条件桩、条件项、动作桩和动作项四项元素,简要解释如下。

(1) 条件桩:问题的所有条件的集合,包含各种条件,其中各条件次序无严格限制。

(2) 条件项:问题的所有条件的各种取值的集

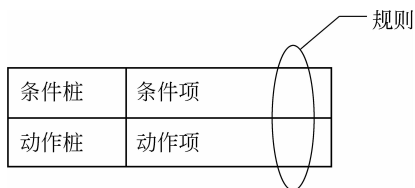


图 4.1 决策表模型图

合,包含条件桩中各种条件的各种取值的组合,其中各条件次序无严格限制。

(3) 动作桩: 问题的所有可采取操作的集合,包含各种可采取的操作,其中各操作次序无严格限制。

(4) 动作项: 针对条件项的各种组合的取值情况下,应该采取的对应操作。

需要提醒的是,决策表中任何一个条件组合的特定取值及其相应要执行的操作称为规则。

不难理解,决策表法实质为直接把测试输入中所有可能的情况进行组合,并汇总所对应的操作结果,这也是决策表法的优势所在。显然,利用决策表法能够设计出各种组合类型的完整测试用例集合。但不得不承认,决策表法并非十全十美,它不能表达重复执行的动作,如循环结构等。因此,读者应辩证地看待该方法。

至此,读者已对决策表法有了相关见解,如何使用决策表法成为下一步要研究的重点。读者可参照以下步骤进行测试用例设计。

- (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。
- (5) 简化决策表,合并类似规则或相同动作。

注意:

① 针对“确定规则的个数”,需要提醒的是,若某决策表中有 n 个条件,且每个条件可取真、假两值,则共有 2^n 条规则;若某决策表中有 n 个条件,且每个条件可取 $1、2、3、\dots、m$,则共有 m^n 条规则。

② 针对决策表的简化过程,提醒以下两点: 其一,若表中有两条或两条以上的规则具有相同的操作,且在条件项之间存在较为类似的关系,则可进行规则合并;其二,规则合并后得到的条件项用符合“—”表示,代表执行的动作与该条件的取值无关,即成为无关条件。

读者已从理论层面上认识了决策表法,下面将以旅馆住宿系统及经典的 NextDate 函数为例,从实践角度进一步介绍决策表法的应用。

3. 实验任务

【任务 4.1】 旅馆住宿系统测试用例设计。

需求: 为了进一步扩大业务和提升营业额,旅馆住宿系统支持房间预订、定金支付及会员卡办理功能,且规定在旅游旺季客房紧张的情况下,优先为进行了房间预订且已支付定金或持有会员卡的游客办理房间入住。

问题: 采用决策表法进行测试用例设计。

前提条件: 需求中输入与输出之间相互制约的条件较多,故适合采用决策表法设计测试用例。

第 1 步,分析需求,列出所有的条件桩和动作桩。

条件桩:

- (1) 是否进行房间预订。
- (2) 是否已支付定金。

(3) 是否为旅馆会员。

动作桩:

(1) 优先办理房间入住。

(2) 做其他处理。

第 2 步,确定规则的个数。在此有 3 个条件,且每个条件有两种取值(是或否),故应有 $2 \times 2 \times 2 = 8$ 种规则。此步骤得出表 4.2 所示的决策表。

表 4.2 旅馆住宿系统_决策表_确定规则个数

条件和动作		规 则							
		1	2	3	4	5	6	7	8
条件	是否进行房间预订								
	是否已支付定金								
	是否为旅馆会员								
动作	优先办理房间入住								
	做其他处理								

第 3 步,填入条件项。条件项即条件桩中各种条件的各种取值的组合,其中各条件次序无严格限制。此步骤得出表 4.3 所示的决策表。

表 4.3 旅馆住宿系统_决策表_填入条件项

条件和动作		规 则							
		1	2	3	4	5	6	7	8
条件	是否进行房间预订	Y	Y	Y	Y	N	N	N	N
	是否已支付定金	Y	Y	N	N	Y	Y	N	N
	是否为旅馆会员	Y	N	Y	N	Y	N	Y	N
动作	优先办理房间入住								
	做其他处理								

第 4 步,填入动作项。此步骤得出表 4.4 所示的决策表。

表 4.4 旅馆住宿系统_决策表_填入动作项

条件和动作		规 则							
		1	2	3	4	5	6	7	8
条件	是否进行房间预订	Y	Y	Y	Y	N	N	N	N
	是否已支付定金	Y	Y	N	N	Y	Y	N	N
	是否为旅馆会员	Y	N	Y	N	Y	N	Y	N
动作	优先办理房间入住	X	X	X		X		X	
	做其他处理				X		X		X

第 5 步,简化决策表,合并类似规则或相同动作。经分析可知,规则 1 与规则 2、规则 5 与规则 7、规则 6 与规则 8 可进行合并,此步骤得出表 4.5 所示的决策表。

表 4.5 旅馆住宿系统_简化后的决策表

条件和动作		规 则				
		1	2	3	4	5
条件	是否进行房间预订	Y	Y	Y	N	N
	是否已支付定金	Y	N	N	—	—
	是否为旅馆会员	—	Y	N	Y	N
动作	优先办理房间入住	X	X		X	
	做其他处理			X		X

至此,依据表 4.5 所示的所有规则可得出最终测试用例,如表 4.6 所示。

表 4.6 旅馆住宿系统_最终测试用例

编号	输 入 条 件	输 入 数 据	预 期 结 果
1	已进行房间预订且已支付定金	房间编号、类型、定金金额	优先办理入住
2	已进行房间预订、未支付定金,是旅馆会员	房间编号、类型、会员卡卡号	优先办理入住
3	已进行房间预订、未支付定金,不是旅馆会员	房间编号、类型	做其他处理
4	未进行房间预订,是旅馆会员	会员卡卡号	优先办理入住
5	未进行房间预订,不是旅馆会员		做其他处理

注意:

- ① 实际使用决策表时,常常优先进行化简。
- ② 体会因果图法与决策表法的不同。

【任务 4.2】 NextDate 函数测试用例设计。

需求: NextDate 函数包含了三个输入变量,分别为 Month(月份)、Day(日期)和 Year(年);函数输出为输入日期的后一天的日期。例如输入为 2010 年 10 月 10 日,则输出为 2010 年 10 月 11 日。其中,输入变量 Month、Day 和 Year 都为整数,且取值范围满足 $1 \leq \text{Month} \leq 12, 1 \leq \text{Day} \leq 31, 1980 \leq \text{Year} \leq 2020$ 。

问题: 采用决策表法进行测试用例设计。

前提条件: 需求中存在输入、输出较多或输入与输出之间相互制约条件较多的情况,故适合采用决策表法进行测试用例设计。

第 1 步,分析需求,列出所有的条件桩和动作桩。

条件桩:

- (1) Month。
- (2) Day。
- (3) Year。

动作桩:

- (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。

注意：为获得输入日期的后一天的日期，NextDate 函数仅须执行上述 5 种类型的操作。

第 2 步，确定规则的个数。依据“若某决策表中有 n 个条件，且每个条件可取真、假两种值，则共有 2^n 条规则；若某决策表中有 n 个条件，且每个条件可取 $1、2、3、\cdots、m$ ，则共有 m^n 条规则。”可知，在此有 3 个条件，且每个条件的具体取值如下。

(1) Month 可有以下 4 种取值：

- ① M1={Month 有 30 天}。
- ② M2={Month 有 31 天,12 月除外}。
- ③ M3={Month 为 12 月}。
- ④ M4={Month 为 2 月}。

(2) Day 可有以下 5 种取值：

- ① D1={ $1\leqslant\text{Day}\leqslant 27$ }。
- ② D2={Day=28}。
- ③ D3={Day=29}。
- ④ D4={Day=30}。
- ⑤ D5={Day=31}。

(3) Year 可有以下 2 种取值：

- ① Y1={Year 为是闰年}。
- ② Y2={Year 为非闰年}。

综上所述，应有 $4\times 5\times 2=40$ 种规则。

第 3 步，填入条件项。条件项即条件桩中各种条件的各种取值的组合，其中各条件次序无严格限制。实际使用决策表时，常常优先进行化简，在此，填入条件项时即可结合实际情况进行适当化简。

填入条件项的过程中，以 Day 为填写基准（以条件中取值情况最多的条件项为基准），进行四组数据的填入（因为 Month 有 4 种取值），而 Year 仅对“Month=M4,且 Day=D2 或 D3”时的情况有影响。此步骤得出表 4.7 所示的决策表。

表 4.7 NextDate 函数_决策表_填入条件项

条件和动作		规 则																					
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
条件	Month	M1	M1	M1	M1	M1	M2	M2	M2	M2	M2	M3	M3	M3	M3	M3	M4	M4	M4	M4	M4	M4	M4
	Day	D1	D2	D3	D4	D5	D1	D2	D3	D4	D5	D1	D2	D3	D4	D5	D1	D2	D2	D3	D3	D4	D5
	Year	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	Y1	Y2	Y1	Y2	—	—

续表

条件和动作		规 则																					
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
动作	无效																						
	Day 加 1																						
	Day 复位																						
	Month 加 1																						
	Month 复位																						
	Year 加 1																						

第 4 步,填入动作项。此步骤得出表 4.8 所示的决策表。

表 4.8 NextDate 函数_决策表_填入动作项

条件和动作		规 则																					
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
条件	Month	M1	M1	M1	M1	M1	M2	M2	M2	M2	M2	M3	M3	M3	M3	M3	M4	M4	M4	M4	M4	M4	M4
	Day	D1	D2	D3	D4	D5	D1	D2	D3	D4	D5	D1	D2	D3	D4	D5	D1	D2	D2	D3	D3	D4	D5
	Year	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	Y1	Y2	Y1	Y2	—	—
动作	无效					✓														✓	✓	✓	
	Day 加 1	✓	✓	✓			✓	✓	✓	✓		✓	✓	✓	✓		✓	✓					
	Day 复位				✓						✓					✓			✓	✓			
	Month 加 1				✓						✓								✓	✓			
	Month 复位															✓							
	Year 加 1															✓							

第 5 步,简化决策表,合并类似规则或相同动作。经分析可知,规则 1、规则 2 与规则 3,规则 6、规则 7、规则 8 与规则 9,规则 11、规则 12、规则 13 与规则 14,以及规则 21 与规则 22 可进行合并,此步骤得出表 4.9 所示的决策表。

表 4.9 NextDate 函数_简化后决策表

条件和动作		规 则													
		1~3	4	5	6~9	10	11~14	15	16	17	18	19	20	21	22
条件	Month	M1	M1	M1	M2	M2	M3	M3	M4	M4	M4	M4	M4	M4	
	Day	D1~D3	D4	D5	D1~D4	D5	D1~D4	D5	D1	D2	D2	D3	D3	D4	D5
	Year	—	—	—	—	—	—	—	—	Y1	Y2	Y1	Y2	—	—
动作	无效			✓									✓	✓	✓
	Day 加 1	✓			✓		✓		✓	✓					
	Day 复位		✓			✓		✓			✓	✓			

续表

条件和动作		规 则													
		1~3	4	5	6~9	10	11~14	15	16	17	18	19	20	21	22
动作	Month 加 1		✓			✓					✓	✓			
	Month 复位							✓							
	Year 加 1							✓							

至此,依据表 4.9 所示的所有规则可得出最终测试用例,如表 4.10 所示。

表 4.10 NextDate 函数_最终测试用例

编号	输 入 条 件	输 入 数 据	预 期 结 果
1	输入 3 个变量值	Year=2010、Month=9、Day=10	2010-9-11
2	输入 3 个变量值	Year=2010、Month=9、Day=30	2010-10-1
3	输入 3 个变量值	Year=2010、Month=9、Day=31	无效
4	输入 3 个变量值	Year=2010、Month=10、Day=10	2010-10-11
5	输入 3 个变量值	Year=2010、Month=10、Day=31	2010-11-1
6	输入 3 个变量值	Year=2010、Month=12、Day=10	2010-12-11
7	输入 3 个变量值	Year=2010、Month=12、Day=31	2011-1-1
8	输入 3 个变量值	Year=2010、Month=2、Day=10	2010-2-11
9	输入 3 个变量值	Year=2012、Month=2、Day=28	2012-2-29
10	输入 3 个变量值	Year=2010、Month=2、Day=28	2010-3-1
11	输入 3 个变量值	Year=2012、Month=2、Day=29	2012-3-1
12	输入 3 个变量值	Year=2010、Month=2、Day=29	无效
13	输入 3 个变量值	Year=2010、Month=2、Day=30	无效

注意: 请读者尝试采用等价类划分法、边界值分析法和决策表法进行测试用例设计,并体会等价类划分法、边界值分析法与决策表法的不同。

4. 拓展练习

【练习】 采用决策表法针对以下需求进行测试用例设计。

需求: 订购单的检查规则为,若订购单金额超过 600 元且未过期,则发出批准单和提货单;若订购单金额超过 600 元,但过期了,则不发批准单;如果订购单金额低于 600 元,则不论是否过期都发出批准单和提货单,在过期的情况下还需发出通知单。