Topics in Software Dynamic White-box Testing

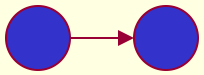
[书上3、4章]

路径覆盖

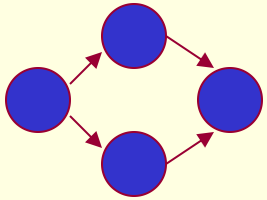
- 路径测试就是设计足够的测试用例，覆盖程序中所有可能的路径。这是最强的覆盖准则。但在路径数目很大时，真正做到完全覆盖是很困难的，必须把覆盖路径数目压缩到一定限度。

流图(flow graph)

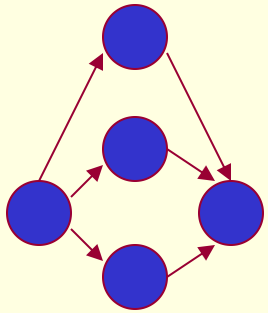
利用流图表示控制逻辑



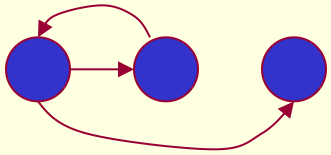
顺序结构



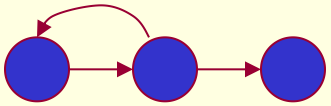
if 结构



Case 结构



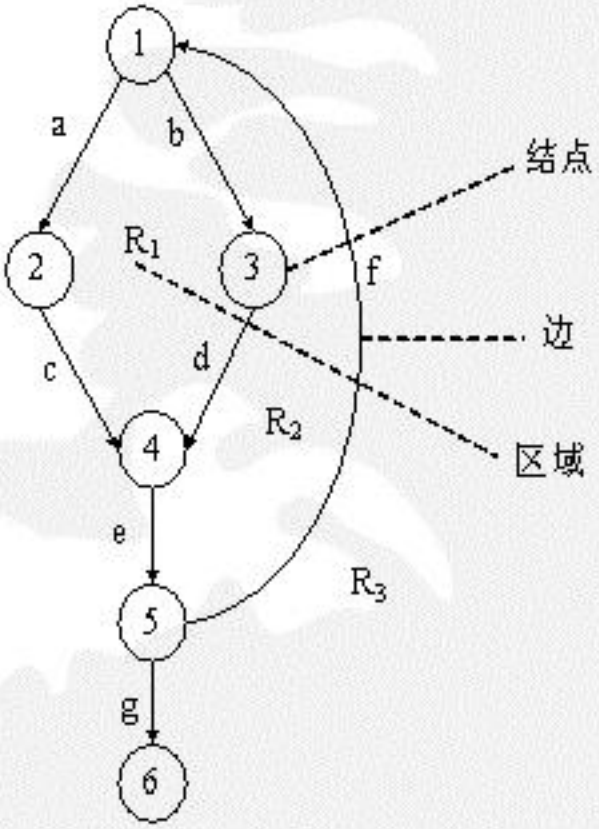
while 结构



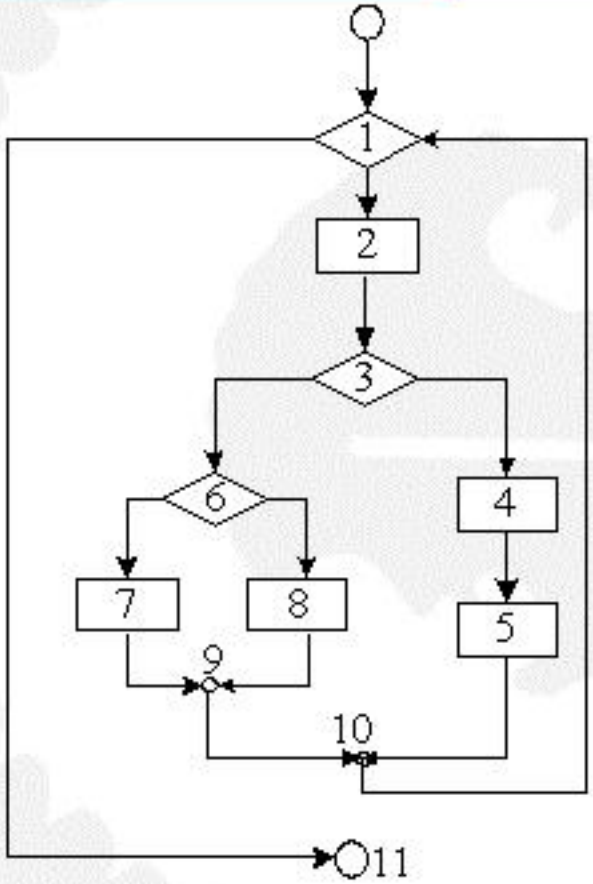
until 结构

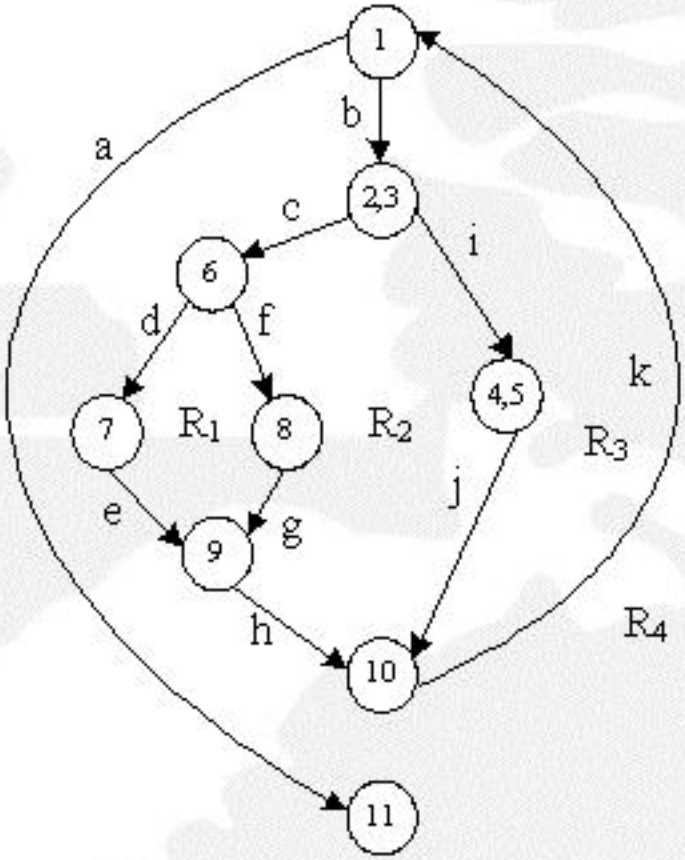
基本概念—控制流图

- 描述程序控制流的一种图示方式
- 结点：一个圆圈，表示单条或多条语句。
- 边：带箭头的线条，起始于一个结点，中止于一个结点。表示了控制流的方向。
- 区域：边和结点圈定的部分。



由框图导出控制流图





路径表达式概括表达路径

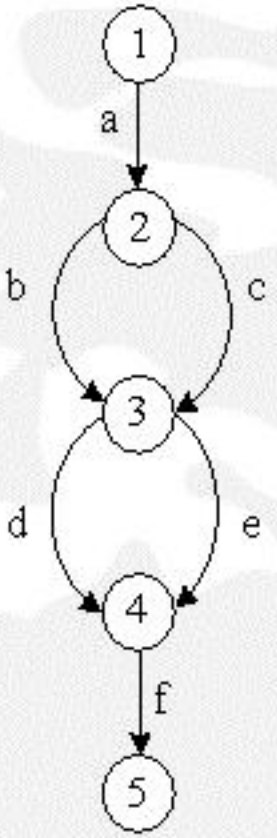
- 概括表示所有路径

  a(b+c)(d+e)f

- 拆分

  abdf+abef+acdf+acef

  所有的四条路径



路径覆盖

路径覆盖是相当强的逻辑覆盖，它保证程序中每条可能的路径都至少执行一次，因此更具代表性，暴露错误的能力也比较强。但为了做到路径覆盖，只需考虑每个判定式的取值组合，并没有检验表达式中条件的各种可能组合。如果将路径覆盖和条件组合覆盖结合起来，可以设计出检错能力更强的测试数据。

独立路径

- 定义
  - 从入口到出口的路径，至少经历一个从未走过的边。这样形成的路径叫独立路径。
- 优点
  - 减少路径数量
  - 包含所有的边和结点
- 缺点
  - 简化循环结构

基本路径测试步骤

① 根据程序的逻辑结构画出程序框图
② 根据程序框图导出流图
③ 计算流图G的环路复杂度V(G)
④ 确定只包含独立路径的基本路径集
⑤ 设计测试用例

程序框图 ==> 流图 ==> 基本路径 ==> 测试用例

白盒路径测试技术

环复杂度计算法：

三种方法之一：

1.流图的区域数量应该对应于环复杂度

2.给定流图G的环复杂度V(G)定义为：V(G)=E-N+2

其中：E为流图中的边数量，N为流图中的节点数量

3.给定流图G的环复杂度V(G)也可以定义为：V(G)=P+1

其中：P为流图中的判断节点数量

Step3 确定基本路径的集合

流图的环形复杂度正好＝基本路径的数目

计算V(G)的不同方法

(1) V(G) =区域个数 = 4

(2) V(G) =边的条数-节点个数+2 = 11-9+2=4

(3) V(G) =判定节点个数+1 = 3+1=4

Step3 确定基本路径的集合

流图的环形复杂度正好是基本路径的数目

确定只包含独立路径的基本路径集

Path1：1-11
Path2：1-2-3-4-5-10-1-11
Path3：1-2-3-6-8-9-10-1-11
Path4：1-2-3-6-7-9-10-1-11

一条新路径必须包含一条新边

测试用例

输入		通过路径	输出	
nPosX	nPosY		nPosX	nPosY
-1	1	1 – 11	-1	1
1	1	1 – 2 – 3 – 4 – 5 – 10 – 1 – 11	0	0
1	-3	1 – 2 – 3 – 6 – 8 – 9 – 10 – 1 – 11	-1	-3
1	-1	1 – 2 – 3 – 6 – 7 – 9 – 10 – 1 – 11	-3	-1