Path Slicing*

PLDI 2005

Ranjit Jhala

CS Department, UC San Diego

Rupak Majumdar

CS Department, UC Los Angeles

作者简介

Ranjit Jhala，美国加州大学圣地亚哥分校雅各布工程学院 ，计算机科学系教授。

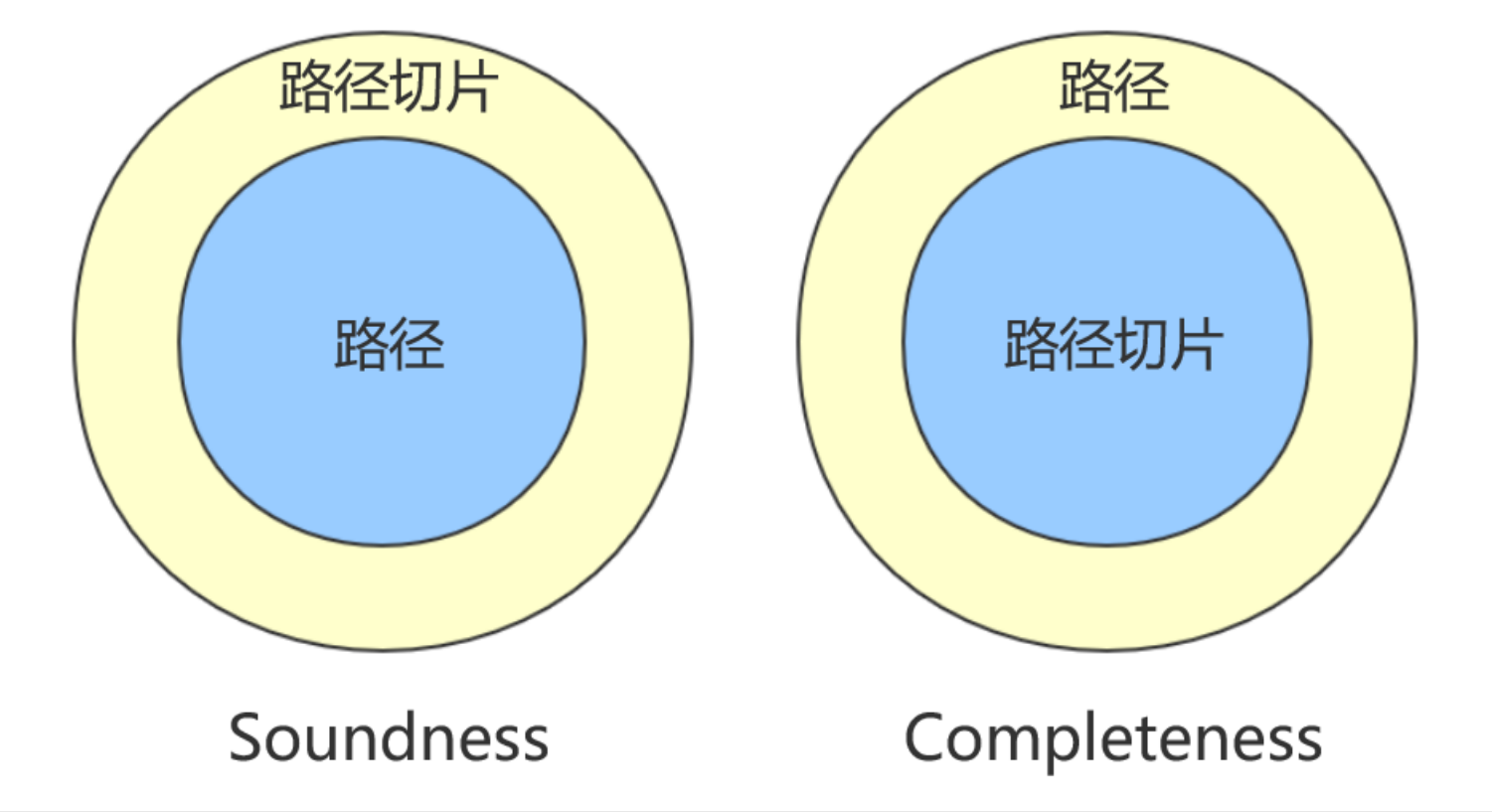
研究领域：类型系统，模型检查，程序分析和自动演绎

Rupak Majumdar，2003年博士毕业于美国加利福尼亚大学伯克利分校，现就职于德国凯撒斯劳滕Max Planck软件系统研究所

研究领域：计算机辅助验证和控制，软件验证和编程语言，逻辑和自动机理论

一、摘要

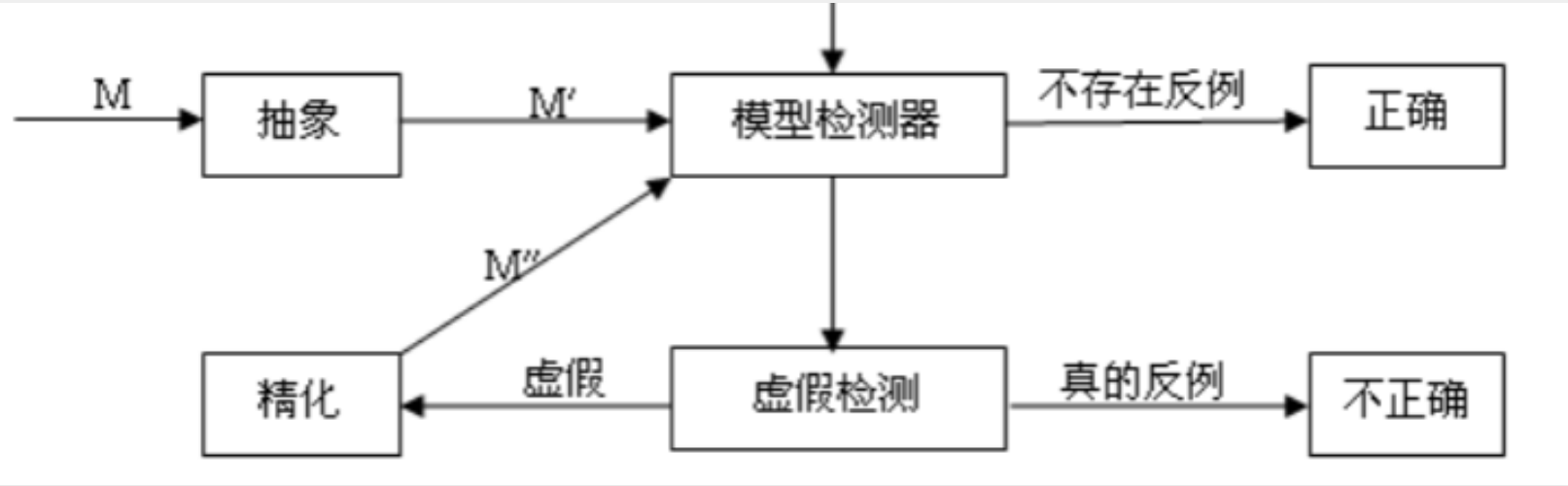
本文提出路径切片技术：输入一条到达目标位置程序路径，输出一条消除所有与目标位置可达性无关的操作的程序路径，这条输出的程序路径就叫做“路径切片”。



二、介绍

背景：

静态分析将控制流路径返回到特定位置，作为警告说明程序不安全。由于静态分析的保守性，此路径不一定真正错误。



二、介绍

用一个例子说明路径切片的必要性：

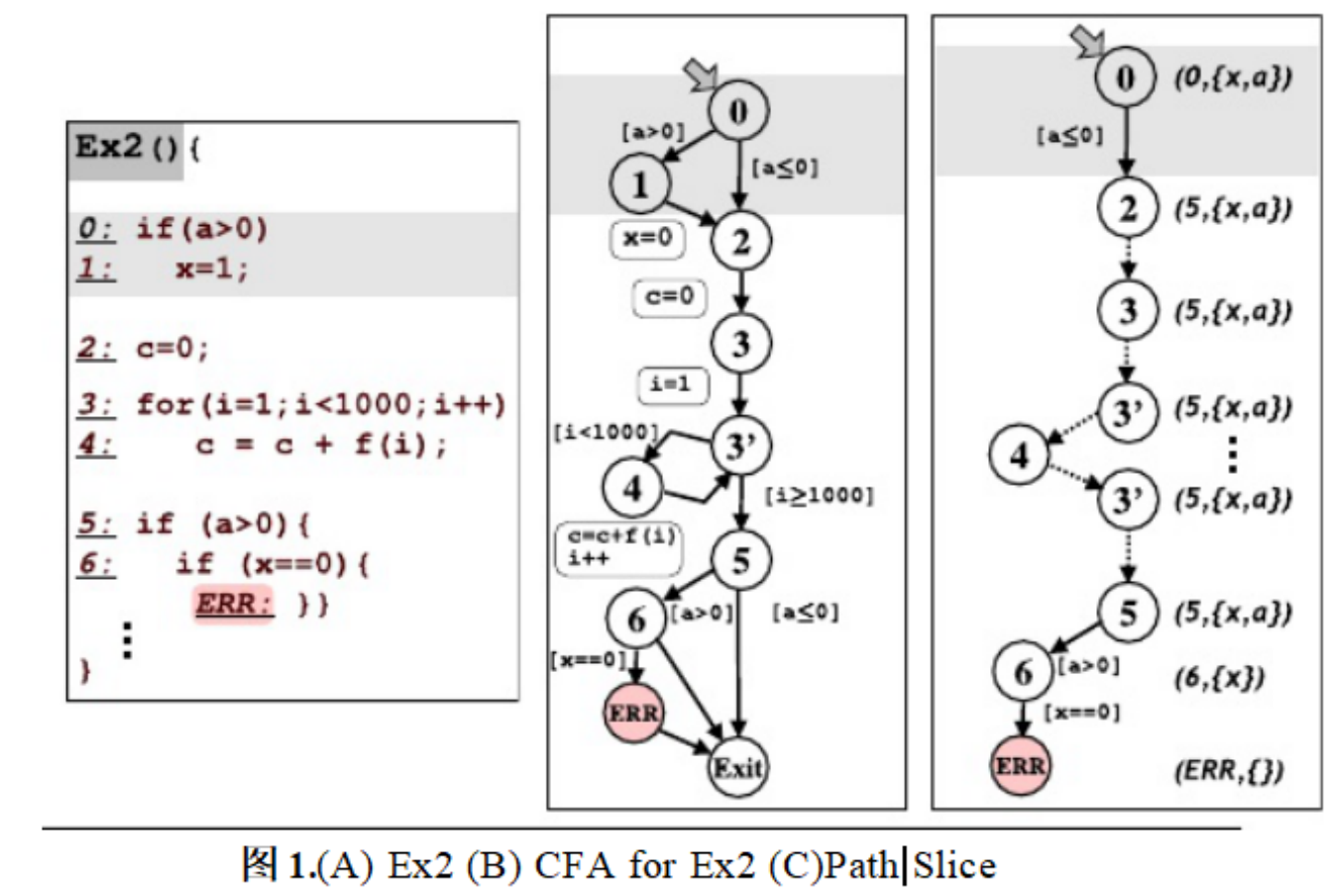


图 1.(A) Ex2 (B) CFA for Ex2 (C)Path|Slice

二、介绍

路径切片与**静态切片**和**动态切片**的区别：

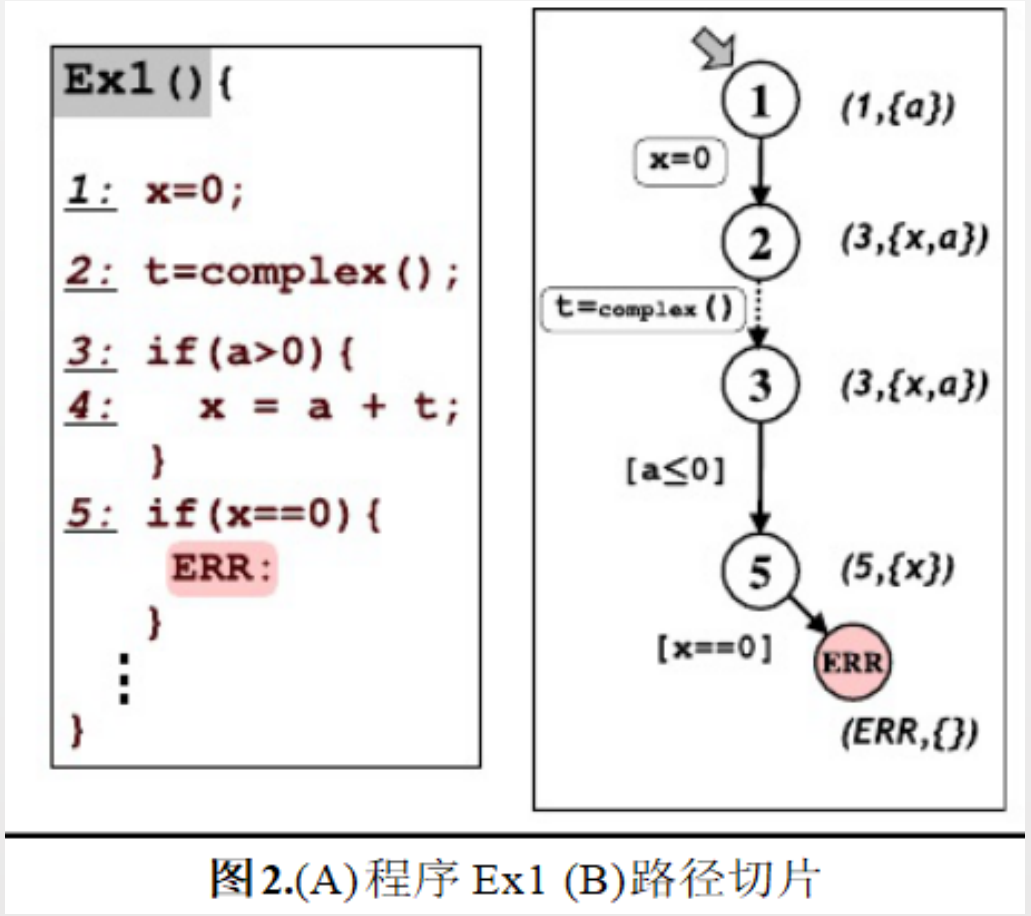


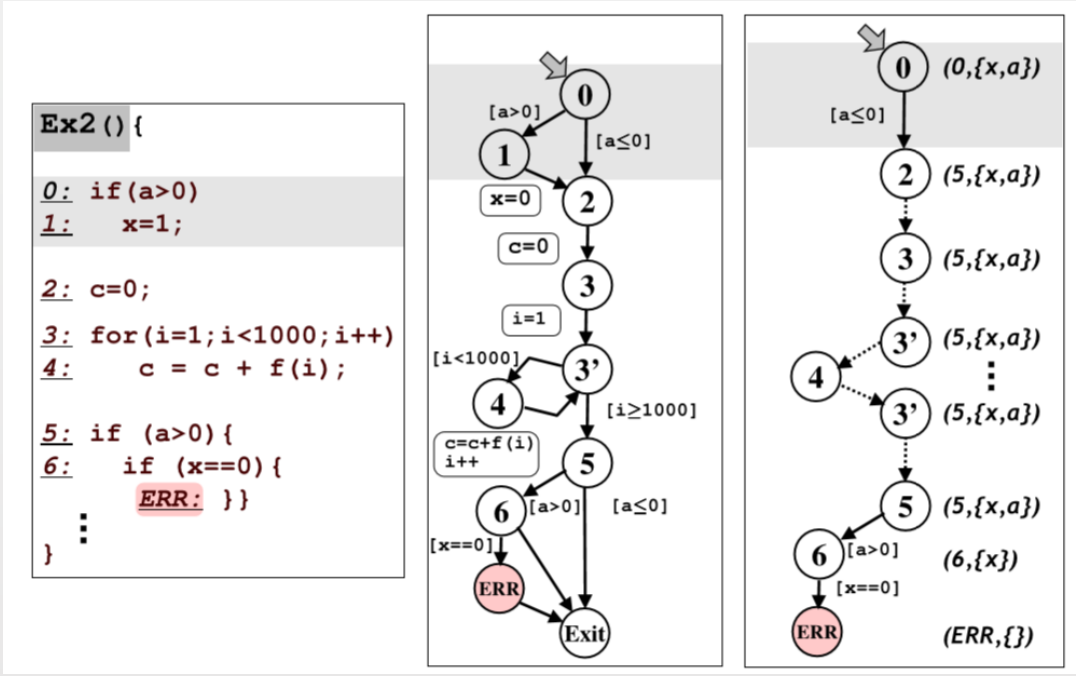
图2.(A)程序 Ex1 (B)路径切片

三、基本概念

控制流自动机（CFA）:本质上是每个函数的CFG（用操作标记边而不是用顶点标记边）

路径：路径是一个CFA边序列。一条路径对应于一系列标记边的操作。

路径切片：路径切片是通过删除路径上的一些边来获得的。



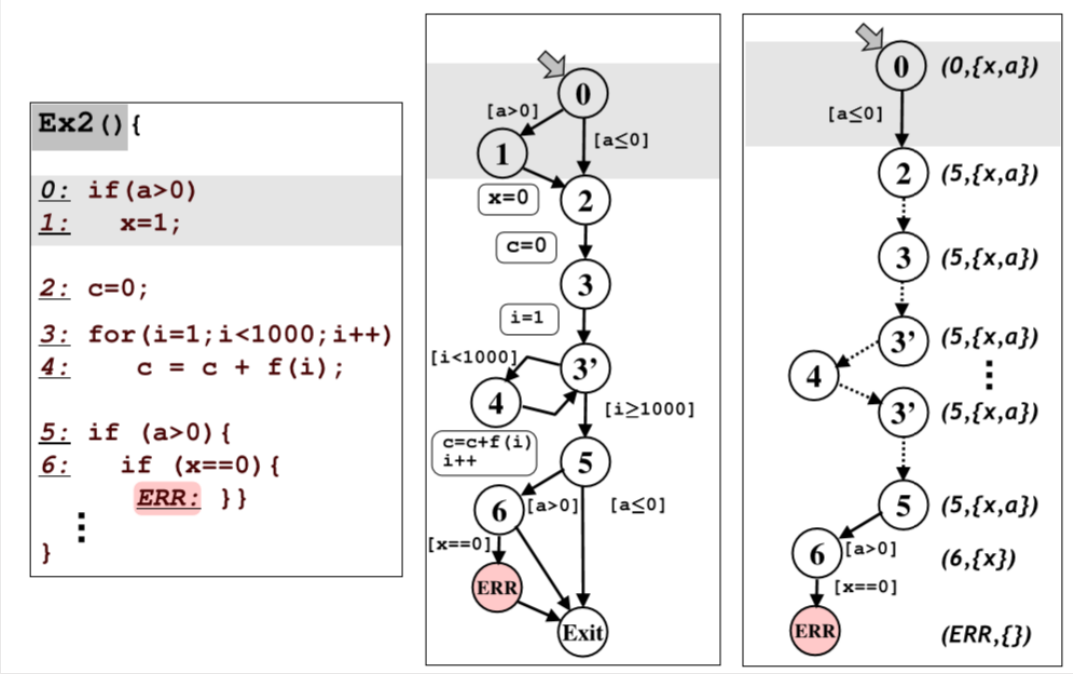
四、计算路径切片

算法中几个参数的定义：

1.活动集合（live set）：一个路径相关活动值的集合，活动值通过踪迹后缀决定错误位置是否可达。

2.步骤位置（step location）：最后加入路径切片那条边的起始位置。

3.切片后缀：已经加入路径后缀的边集



四、计算路径切片

程序Take：

格式：Take.(L,PCs).(PC,OP,PC')

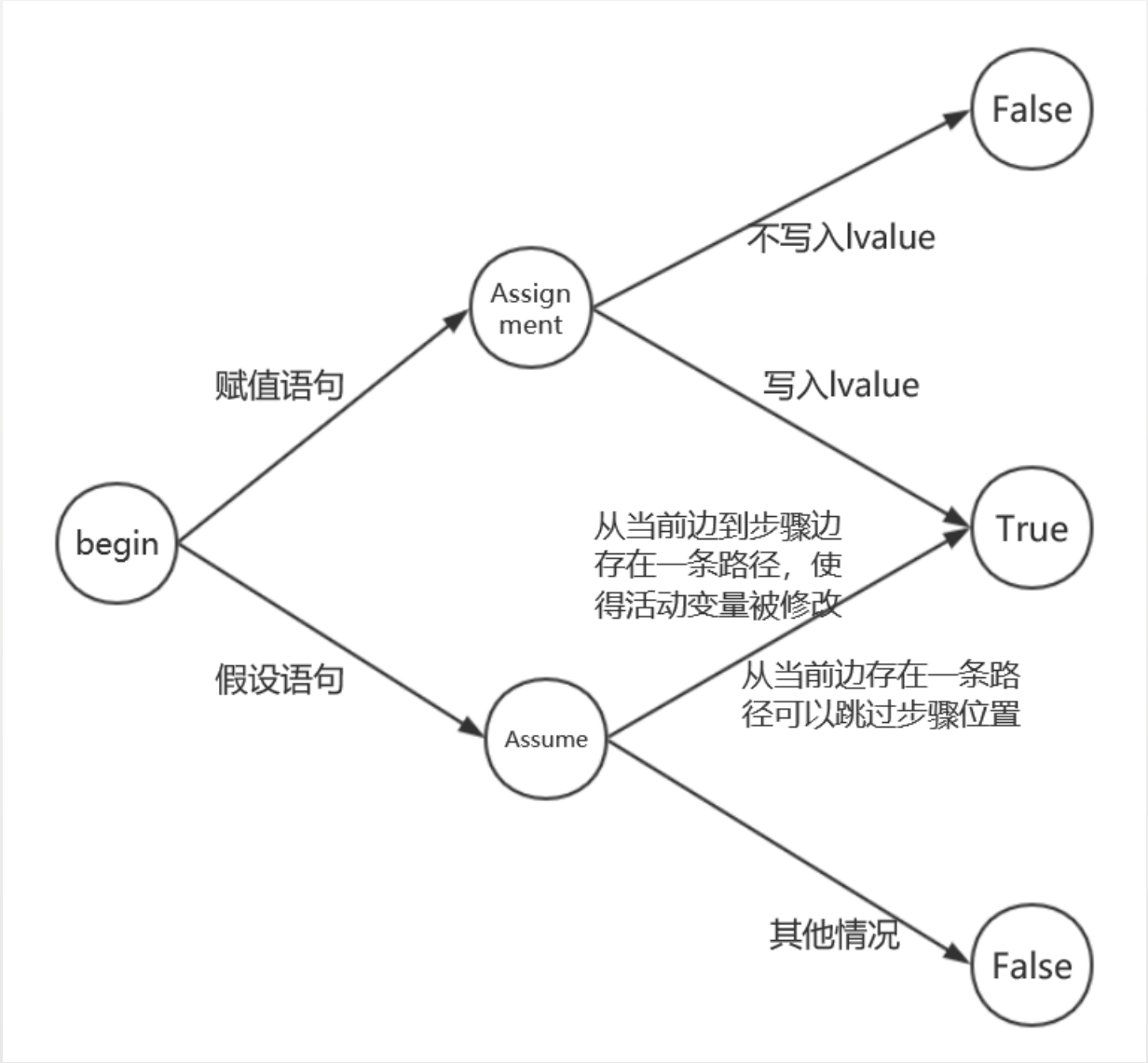
输入：

1.活动值(live lvalues)集合L；

2.当前步骤位置PCs；

3.边（PC,OP,PC'）

输出：布尔值，表示这条边是否应该加入切片



四、计算路径切片

Algorithm 1 PathSlice

Input: Program Path π.
Output: Path Slice π'.

1: $\pi' := [\cdot]$
2: $i := |\pi|$
3: $Live := \emptyset$; $(\cdot, \cdot, pc_{step}) := \pi.i$
4: **while** $i \geq 1$ **do**
5: $\quad e := \pi.i$
6: $\quad tk := \mathsf{Take}.(Live, pc_{step}).e$
7: $\quad$ **if** tk **then**
8: $\quad\quad \pi' := e :: \pi'$
9: $\quad\quad (pc, \mathsf{op}, \cdot) := e$
10: $\quad\quad Live := (Live \setminus \mathsf{Wt.op}) \cup \mathsf{Rd.op}$
11: $\quad\quad pc_{step} := pc$
12: $\quad i := i - 1$
13: **return** π'

```
A = 0;
B = 1;
C = A + B;
if (C > 0):
  ERR;
```

四、计算路径切片

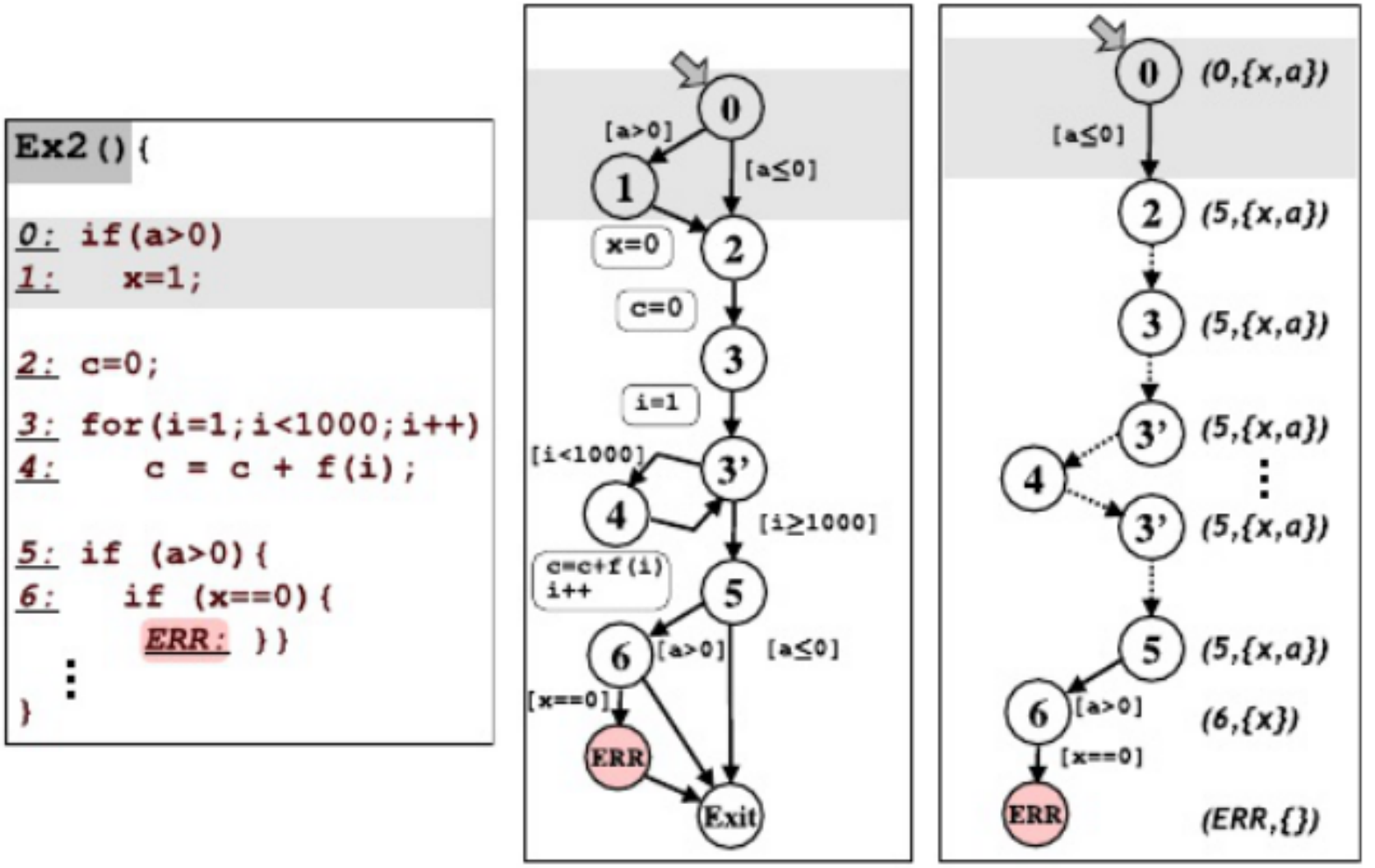


图 1.(A) Ex2 (B) CFA for Ex2 (C)Path|Slice

五、实验分析

实现了在BLAST软件模型检查器中生成路径切片的算法

Program	Description	LOC	Procedures	Number of checks	Results	Total time	Max time	Number of refinements
fcron 2.9.5	cron daemon	12K/14K	121	10/25	10/0/0	22.95	9.56	15
wuftpd 2.6.2	ftp server	24K/35K	205	33/59	30/3/0	2417.41	412.08	74
make 3.80	make	30K/39K	296	19/44	18/1/0	89.8	32.7	35
privoxy 3.03	web proxy	38K/51K	291	15/54	13/2/0	107.5	69.1s	13
ijpeg	jpeg compression	31K/37K	403	21/43	21/0/0	128.0s	121.6	23
openssh 3.5.1	ssh server	50K/114K	745	24/84	23/0/1	2211.5	554.1	135

Table 1. Benchmarks and analysis times.

五、实验分析

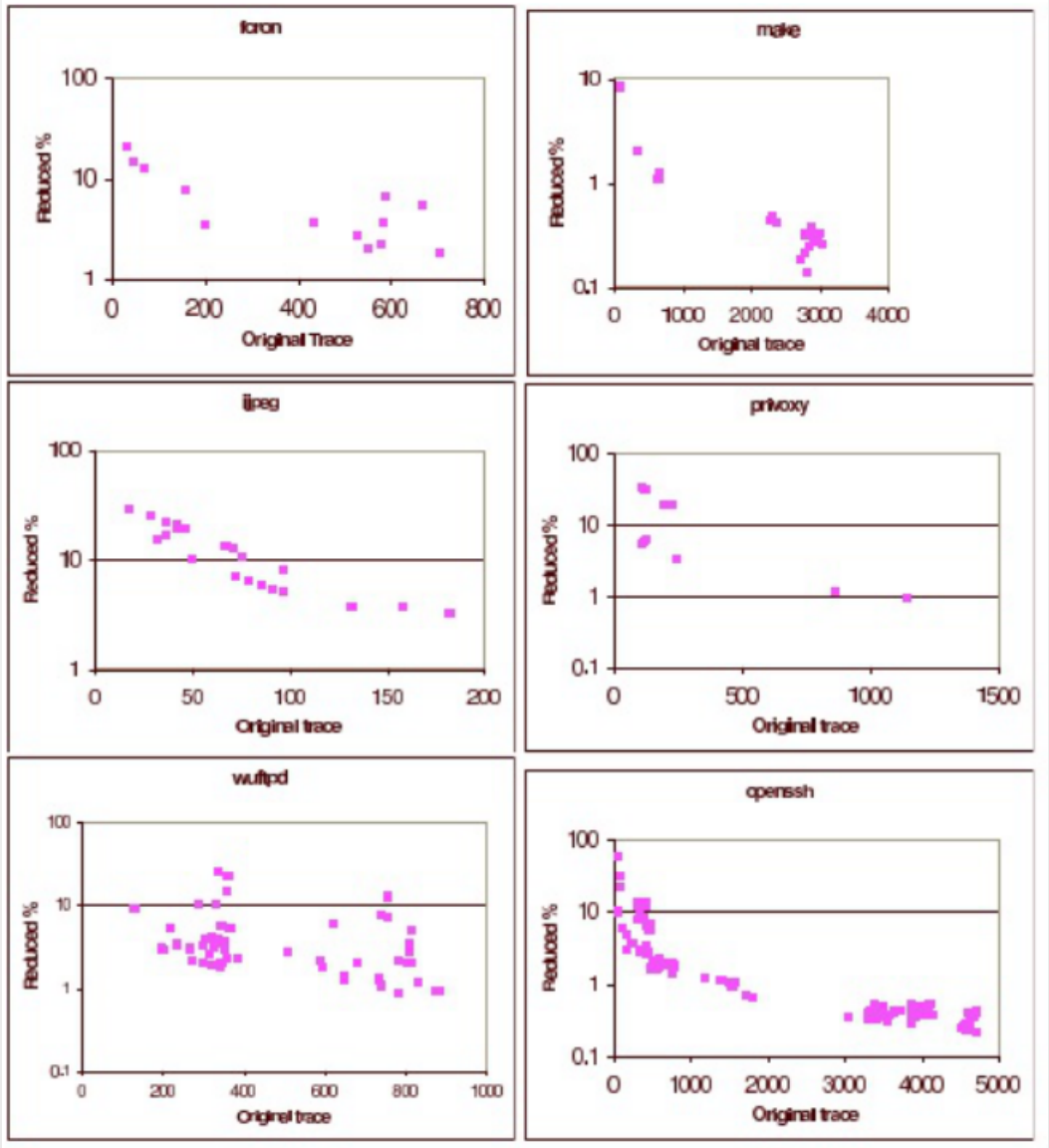


图 5 PathSlice 效果

六、工作局限性

1.在上下文无关的可达性算法中使用深度优先搜索，这会导致很长的反例。解决方案：通过研究广度优先搜索算法，它能找到最短的反例。

2.BLAST对堆的建模不精确。如果有一个散列表(以链表数组的形式构建)，用于保存从通道名(字符串)到文件指针的映射，那么BLAST无法推断文件指针是否被放在这些链表中。

3.当前路径切片代码实现的效率低下。