《软件分析与验证》

基于控制流自动机的程序验证
限界模型检验

贺飞

清华大学软件学院

2024 年 5 月 31 日

上节课内容：最强后置条件

- 最强后置条件定义
- 最强后置条件计算，量词消去
- 基于最强后置条件的程序验证

回顾：格局和可达图

设 Σ 为程序 P 中所有语句的集合，$G = (Loc, \Delta, l_{in}, l_{ex})$ 为程序 P 的控制流自动机。

定义

格局 (configuration) 是一个有序对 (l, s)，其中 $l \in Loc$ 是程序位置，$s \in State$ 是程序状态（即对程序所有变量的一组赋值）。

- 初始格局：位于程序初始位置且满足前置条件的格局
- 错误格局：位于程序退出位置且不满足后置条件的格局
- 可达格局：存在一条从初始格局到该格局的执行

定义

可达图（reachability graph）是一个有序对 (C_R, T)，其中 C_R 为可达格局集合，$T \subseteq C_R \times \Sigma \times C_R$ 为满足下面条件的变迁关系 (transition relation)：

$$\big((l, s), st, (l', s')\big) \in T \quad \Leftrightarrow \quad (l, st, l') \in \Delta \text{ 且 } (s, s') \in [\![st]\!]$$

目录

抽象格局与抽象可达图

定义

抽象格局 (abstract configuration) 是一个有序对 (l, φ)，其中 $l \in Loc$ 是程序位置，φ 是定义在程序变量上的逻辑公式。

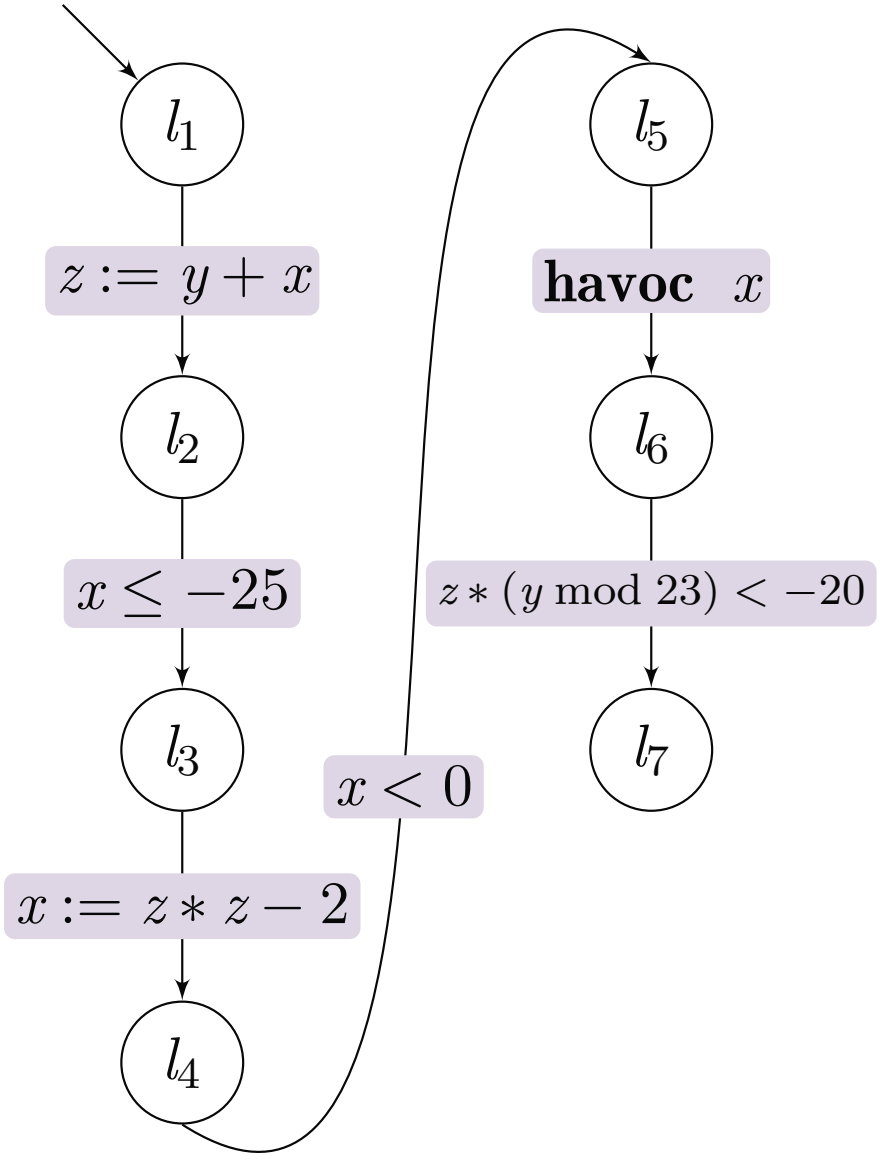
为了与抽象格局相区分，称前面定义的格局 (l, s) 为具体格局 (concrete configuration)。

抽象格局 (l, φ) 对应一个满足 φ 的具体格局的集合，即

$$\{(l, s) \mid s \models \varphi\}$$

分别记 C, C^α, C_R 和 C_R^α 为程序中具体格局、抽象格局、可达具体格局和可达抽象格局的集合。

示例



以左边的控制流图为例：

- $(l_1, \{x \mapsto -25, y \mapsto 2, z \mapsto 1\})$ 是一个具体格局
- $(l_1, y \geq 1)$ 是一个抽象格局，代表在 l_1 位置上、状态满足 $y \geq 1$ 的所有具体格局
- **问**：上面的具体格局被上面的抽象格局所包含吗？

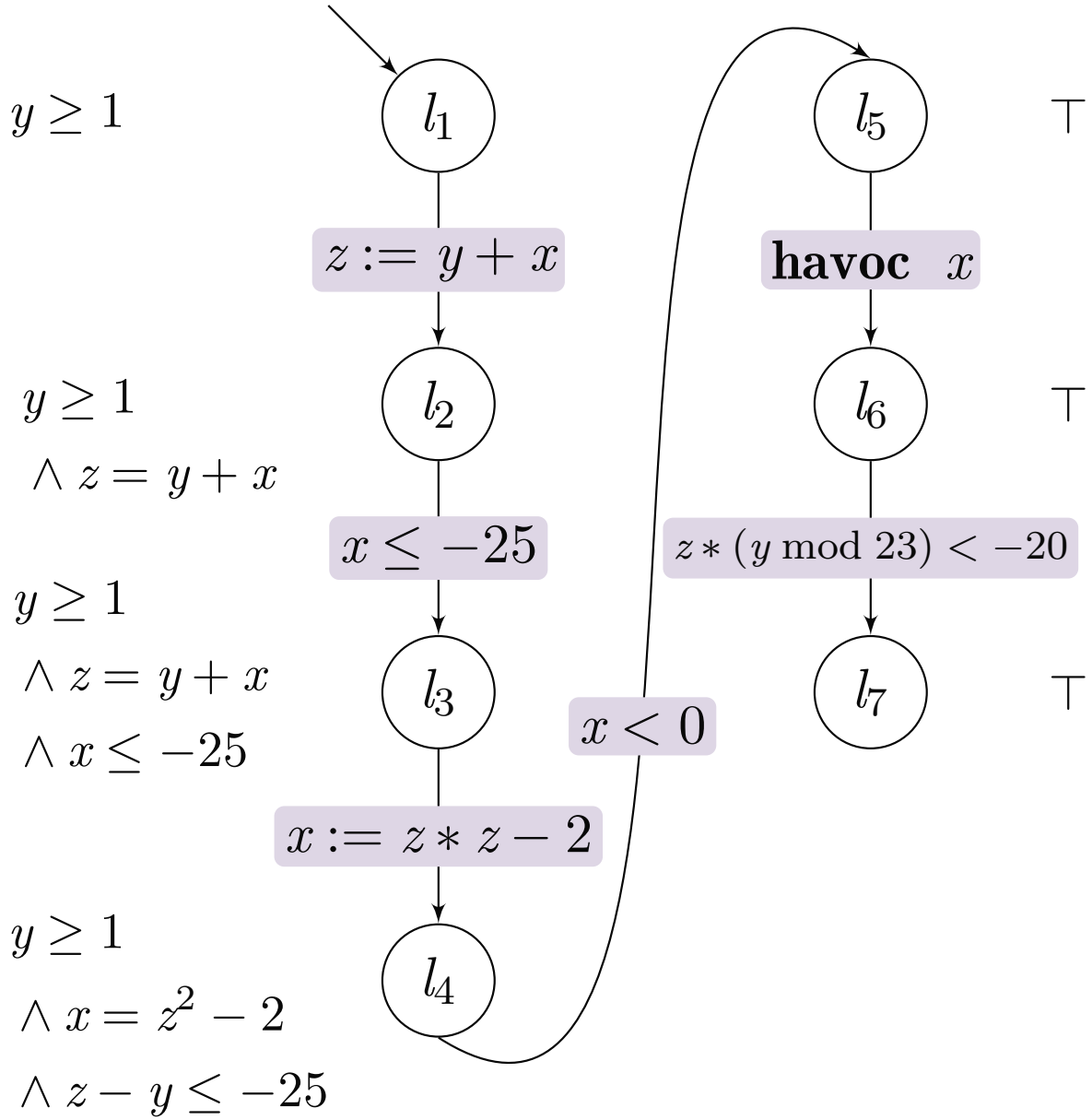
设 Σ 为程序 P 中所有语句的集合，$G = (Loc, \Delta, l_{in}, l_{ex})$ 为程序 P 的控制流自动机。

定义 (抽象执行)

抽象执行（abstract execution）是满足下列条件的抽象格局序列 $(l_0, \varphi_0), \ldots, (l_n, \varphi_n)$：存在一个语句序列 $st_1, \ldots, st_n$，使得对任意 $i \in \{0, \ldots, n-1\}$ 下列条件都成立：

- $(l_i, st_i, l_{i+1}) \in \Delta$
- $sp(\varphi_i, st) \Rightarrow \varphi_{i+1}$

一条抽象执行的示例


$y \geq 1$
l_1
$z := y + x$
$y \geq 1$
$\wedge\, z = y + x$
l_2
$x \leq -25$
$y \geq 1$
$\wedge\, z = y + x$
$\wedge\, x \leq -25$
l_3
$x := z * z - 2$
$y \geq 1$
$\wedge\, x = z^2 - 2$
$\wedge\, z - y \leq -25$
l_4
$x < 0$
l_5
$\top$
havoc x
l_6
$\top$
$z * (y \bmod 23) < -20$
l_7
$\top$

定义 (抽象可达图)

抽象可达图 (abstract reachability graph) 是一个有序对 (C_R^α, T^α)，其中 C_R^α 为抽象格局的集合，$T^\alpha \subseteq C_R^\alpha \times \Sigma \times C_R^\alpha$ 为满足下面条件的抽象变迁关系 (abstract transition relation)：

- $(l_{in}, \varphi_{pre}) \in C_R^\alpha$，
- $\forall (l, \varphi) \in C_R^\alpha$，若 $\varphi \neq \bot$ 且存在 $(l, st, l') \in \Delta$，则一定存在另一个抽象格局 (l', φ')，使得 $sp(\varphi, st) \Rightarrow \varphi'$，并且 $((l, \varphi), st, (l', \varphi')) \in T^\alpha$。

抽象可达图中的任意抽象格局 $(l, \varphi) \in C_R^\alpha$ 都是可达的，即存在一条从初始抽象格局 (l_{in}, φ_{pre}) 到该格局的抽象执行。

精确抽象可达图

定义

精确抽象可达图（precise abstract reachability graph）是满足下面条件的抽象可达图：

$$\forall \left((l,\varphi), st, (l',\varphi')\right) \in T^{\alpha},\ sp(\varphi, st) \Leftrightarrow \varphi'$$

在精确抽象可达图中，抽象格局 (l,φ) 关于语句 st 的后继抽象格局唯一。

一个程序的抽象可达图可以有很多个，但其精确抽象可达图有且仅有一个（不考虑格局中逻辑公式的不同等价形式）。

示例

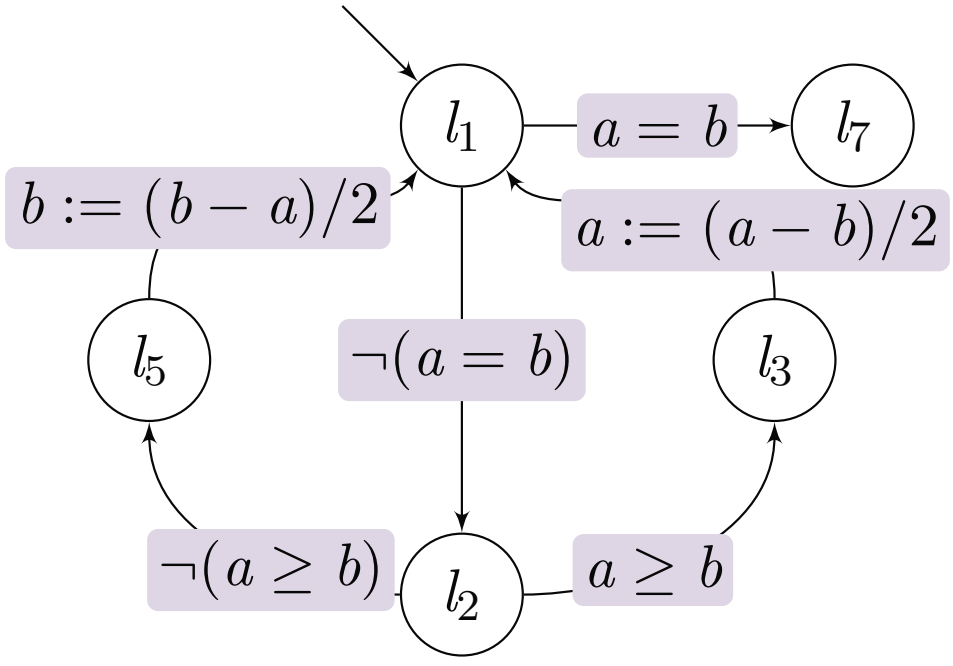
验证下面求最大公约数的算法实现是否正确。注意：形式化地表达最大公约数是困难的，因此我们在后置条件中只刻画了最大公约数的一个性质。

```
1  while (!(a == b)) {
2    if (a >= b) {
3      a = (a - b) / 2;
4    } else {
5      b = (b - a) / 2;
6    }
7  }
```

$$\varphi_{pre}: \ a_{in} = a \land b_{in} = b \land a_{in} > 0 \land b_{in} > 0$$

$$\varphi_{post}: \ a_{in} \bmod a = 0 \land b_{in} \bmod b = 0$$

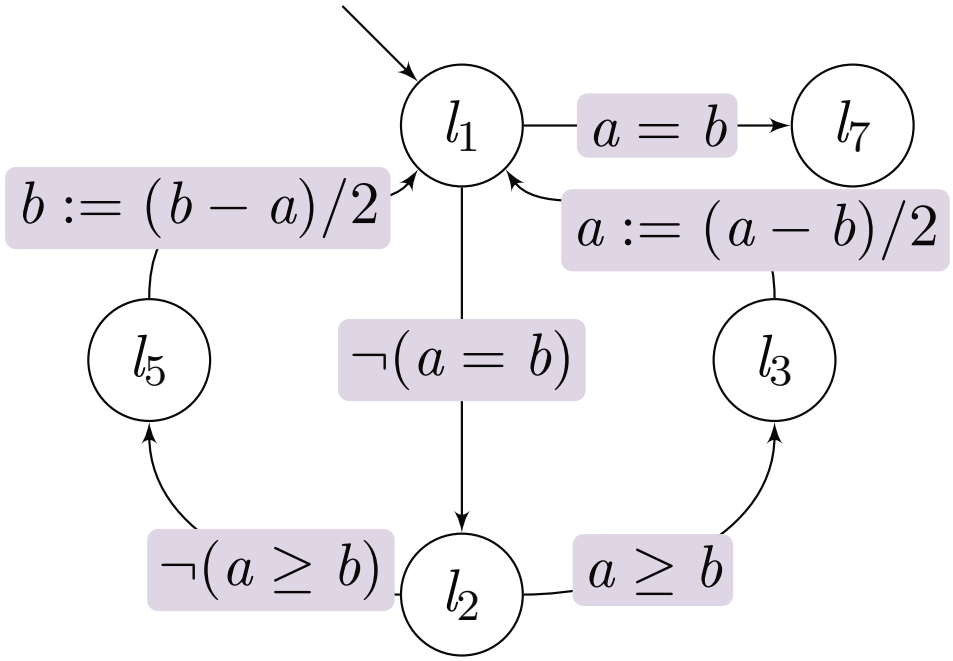
例：最大公约数算法实现



	a	b
l_1	9	15
l_2	9	15
l_5	9	15
l_1	9	3
l_2	9	3
l_3	9	3
l_1	3	3
l_7	3	3

通过

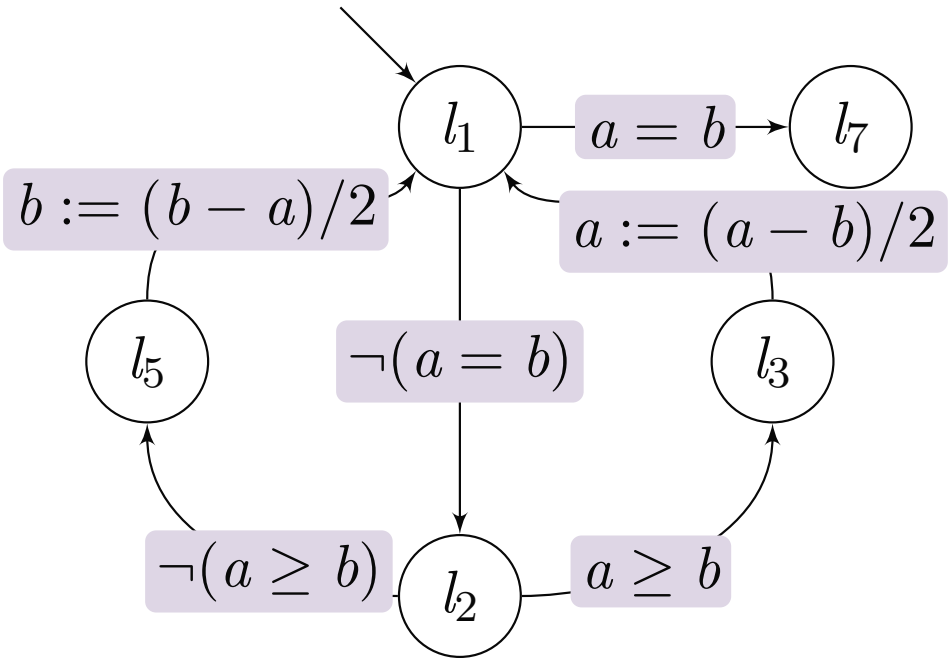
例：最大公约数算法实现



	a	b
l_1	40	24
l_2	40	24
l_3	40	24
l_1	8	24
l_2	8	24
l_5	8	24
l_1	8	8
l_7	8	8

通过

例：最大公约数算法实现



	a	b
l_1	11	5
l_2	11	5
l_3	11	5
l_1	3	5
l_2	3	5
l_5	3	5
l_1	3	1
l_2	3	1
l_3	3	1
l_1	1	1
l_7	1	1

通过

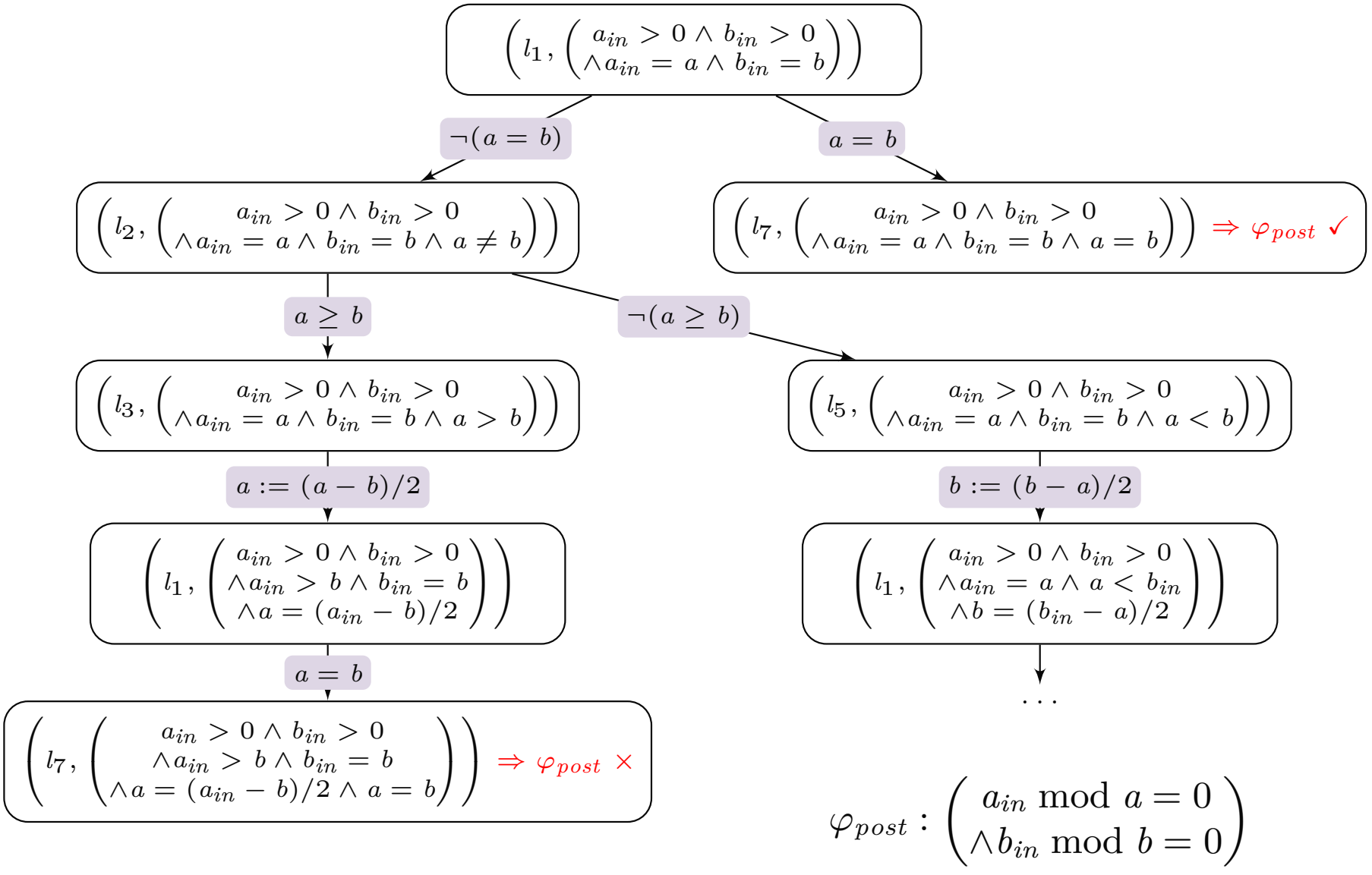
例：最大公约数算法实现

基于测试的方法：

- 没有找到违反程序规约的测例
- 这些测例具有很好的覆盖率
  - 100% 语句覆盖
  - 100% 分支覆盖
  - 输入为质数的边界情况已覆盖
- 测试工程师很有可能会认为该程序是正确的

真的如此吗？
让我们尝试建立此程序的精确抽象可达图。

例：最大公约数算法实现



$$\varphi_{post} : \begin{pmatrix} a_{in} \bmod a = 0 \\ \wedge b_{in} \bmod b = 0 \end{pmatrix}$$

定义

一个抽象格局 (l, φ) 称为

- 初始抽象格局，如果 $l = l_{in}$ 且 $\varphi \Leftrightarrow \varphi_{pre}$
- 错误抽象格局，如果 $l = l_{ex}$ 且 $\varphi \not\Rightarrow \varphi_{post}$

例

例如，在上一页示例中：

$$\big(l_1, \big(a_{in} > 0 \wedge b_{in} > 0 \wedge a_{in} = a \wedge b_{in} = b\big)\big)$$

是一个初始抽象格局，而

$$\big(l_7, \big(a_{in} > 0 \wedge b_{in} > 0 \wedge a_{in} > b \wedge b_{in} = b \wedge a = (a_{in} - b)/2 \wedge a = b\big)\big)$$

是一个错误抽象格局。

引理

程序可达格局集合 C_R 含错误格局当且仅当程序的精确抽象可达图中含错误抽象格局。

定理

给定程序 P 和规约 $(\varphi_{pre}, \varphi_{post})$，程序满足规约的充要条件是程序的精确抽象可达图中不含错误抽象格局。

证明.

程序满足规约 $\Leftrightarrow$ C_R 中不含错误格局 $\Leftrightarrow$ 程序精确抽象可达图中不含错误抽象格局。 □

验证算法

可达图构造算法

给定控制流自动机 $G = (Loc, \Delta, l_{in}, l_{ex})$ 和前置条件 φ_{pre}

算法 $ConstructRG(G, \varphi_{pre})$

$C_R \leftarrow \emptyset, T \leftarrow \emptyset, wl \leftarrow \emptyset$
for all $s \in \{\varphi_{pre}\}$ **do**
  $C_R \leftarrow C_R \cup \{(l_{in}, s)\}, wl \leftarrow wl \cup \{(l_{in}, s)\}$
while $wl \neq \emptyset$ **do**
  $(l, s) \leftarrow \text{RemoveFirst}(wl)$
  for all $(l, st, l') \in \Delta$ **do**
    for all s' with $(s, s') \in [\![st]\!]$ **do**
      $T \leftarrow T \cup \{((l, s), st, (l', s'))\}$
      if $(l', s') \notin C_R$ **then**
        $C_R \leftarrow C_R \cup (l', s'), wl \leftarrow wl \cup \{(l', s')\}$
return (C_R, T)

本质上是一个宽度优先搜索算法，可能不终止！

给定控制流自动机 $G = (Loc, \Delta, l_{in}, l_{ex})$ 和前置条件 φ_{pre}

算法 $ConstructARG(G, \varphi_{pre})$

$C_R^\alpha \leftarrow \{(l_{in}, \varphi_{pre})\},\ T^\alpha \leftarrow \emptyset,\ wl \leftarrow \{(l_{in}, \varphi_{pre})\}$
while $wl \neq \emptyset$ **do**
  $(l, \varphi) \leftarrow \textsc{RemoveFirst}(wl)$
  for all $(l, st, l') \in \Delta$ **do**
    $\varphi' \leftarrow sp(\varphi, st)$
    $T^\alpha \leftarrow T^\alpha \cup \{((l, \varphi), st, (l', \varphi'))\}$
    if $(l', \varphi') \notin C_R^\alpha$ **then**
      $C_R^\alpha \leftarrow C_R^\alpha \cup (l', \varphi')$
      if $\varphi' \neq \bot$ **then**
        $wl \leftarrow wl \cup \{(l', \varphi')\}$
return (C_R^α, T^α)

也称符号执行（symbolic execution）算法，仍可能不终止

应用 $ConstructARG$ 算法时，做以下修改：

- 添加 (l', φ') 到图中时，检查其是否错误抽象格局
- 如果 (l', φ') 是错误抽象格局
  - 令 $(l_{in}, \varphi_{pre}), \ldots, (l', \varphi')$ 为从初始抽象格局到 (l', φ') 的执行，而 $st, \ldots, st'$ 为这条执行经过的语句序列
  - 终止算法，报告程序不满足规约，并返回上述语句序列

有界正确性（确保算法终止）：

- 从初始抽象格局出发，只遍历 k 步以内可达的抽象格局；
- 如果没有找到错误抽象格局，终止算法并断言程序在执行不超过 k 条语句时是正确的。

总结：基于精确抽象可达图的程序验证算法

- 抽象格局、抽象执行、抽象可达图
- 精确抽象可达图
- 可达图和精确抽象可达图的构造
- 限界模型检验算法

下节课内容

- 基于谓词抽象的程序验证

谢谢!