《软件分析与验证》

推导循环不变式

贺飞
清华大学软件学院

2024 年 5 月 17 日

上节课：终止性

- 终止性证明的基础——良基关系
- 终止性证明的依据——秩函数
- 终止性证明的示例

本节课的内容

演绎程序验证:

- 部分正确性验证依赖合适的循环不变式
- 完全正确性验证依赖合适的秩函数

寻找合适的循环不变式和秩函数:

- 都是不可判定问题
- 演绎程序验证领域最富挑战性的问题

下面讨论手工推导循环不变式的一些策略

基本事实

基本事实

为推导合适的循环不变式，首先从循环中提取一些明确的基本事实，包括：

- 相关变量进入循环前的初始值
- 循环索引的边界值
- 循环索引是如何更新的
- 循环的进入条件
- 相关数组的边界范围

此外，还需要特别关注程序中与规约相关的一些事实

```
int i = l;
while (i <= u)
{
    if (a[i] == e) return 1;
    i = i + 1;
}
```

基本事实：

- 初始值：$i := l$
- 循环条件：$i \leq u$
- 循环索引更新：$i := i + 1$

基于上述观察，容易得到下面的推测：

$$l \leq i \leq u + 1$$

思考： 这里为什么是 $u + 1$ 而不是 u?

示例：嵌套循环

```
i = |a| - 1;
while (i > 0){
  j = 0;
  while (j < i){
    if (a[j] > a[j+1]){
      t = a[j];
      a[j] = a[j+1];
      j = j + 1;
      a[j] = t;
    }
  }
  i = i - 1;
}
```

外循环：

- 初始值： $i := |a| - 1$
- 循环条件： $i > 0$
- 循环索引更新： $i := i - 1$

于是，可推测：

$$-1 \leq i < |a|$$

思考：为什么 $-1 \leq i$ 而不是 $0 \leq i$？

示例：嵌套循环

```
i = |a| - 1;
while (i > 0){
  j = 0;
  while (j < i){
    if (a[j] > a[j+1]){
      t = a[j];
      a[j] = a[j+1];
      j = j + 1;
      a[j] = t;
    }
  }
  i = i - 1;
}
```

内循环：

- 初始值：$j := 0$
- 循环条件：$j < i$
- 循环索引更新：$j := j + 1$

于是，可推测：

$$0 \leq j \leq i$$

注意 i 的值来自外层循环，有：

- $-1 \leq i < |a|$（外层不变式）
- $i > 0$（外层循环进入条件）

所以

$$0 \leq j \leq i \ \wedge \ 0 < i < |a|$$

不变式增强

不变式增强

一般而言，由基本事实推测得到的不变式还比较弱，很难直接用来证明程序的正确性（使得规约的后置条件成立）

我们常常还需要从后置条件出发，通过计算最弱前置条件来对不变式进行增强

1. 假设通过计算确定 γ 需要在程序某位置成立，但却不被当前循环不变式所支持
2. 从该位置出发，至循环头或过程入口，计算 γ 的最弱前置条件
3. 尽可能泛化计算得到的结果
4. 重复第 2、3 步，直至得到一个满足要求的循环不变式

示例：线性查找


```
int i = l;
while (i <= u)
{
    if (a[i] == e) return 1;
    i = i + 1;
}
return 0;
```

基于基本事实得到的循环不变式：

$$l \leq i \leq u + 1$$

该算法的后置条件：

$$rv = 1 \leftrightarrow \exists i.\ l \leq i \leq u \wedge a[i] = e$$

考虑下面的基本路径：

$$
\begin{aligned}
&\{l \leq i \leq u+1\} \\
&\ \textbf{assume}\ i > u; \\
&rv := 0 \\
&\{\psi :\ \ rv = 1 \leftrightarrow \exists j.\ l \leq j \leq u \wedge a[j] = e\}
\end{aligned}
$$

对应的验证条件为：

$$l \leq i \leq u+1 \wedge i > u \rightarrow \neg(\exists j.\ l \leq j \leq u \wedge a[j] = e)$$

- 该式的前提与数组 a 无关，显然无法推导出结论
- 换言之，当前不变式不足以证明程序的后置条件

示例：线性查找

考虑下面的基本路径：

$$\{l \leq i \leq u+1\}$$
$$\textbf{assume } i > u;$$
$$rv := 0$$
$$\{\psi : \ rv = 1 \leftrightarrow \exists j.\ l \leq j \leq u \wedge a[j] = e\}$$

从后置条件出发，计算最弱前置条件，得到：

$$wp(\textbf{assume } i > u; rv := 0, \psi)$$
$$\Leftrightarrow i > u \rightarrow (\forall j.\ l \leq j \leq u \rightarrow a[j] \neq e)$$

- 该式可看作为在循环终止时，为了让后置条件成立而对循环不变式的最低要求

还需考虑下面的基本路径：

$$
\begin{aligned}
&\{l \leq i \leq u+1\} \\
&c_0: \quad \textbf{assume } i \leq u; \\
&c_1: \quad \textbf{assume } a[i] \neq e; \\
&c_2: \quad i := i+1; \\
&\{I: \ i > u \rightarrow (\forall j.\ l \leq j \leq u \rightarrow a[j] \neq e)\}
\end{aligned}
$$

从上一步计算得到的公式出发，沿着这条基本路径继续向前计算最弱前置条件：

$$
\begin{aligned}
& wp(c_0; c_1; c_2, I) \\
\Leftrightarrow\ & i \leq u \rightarrow (a[i] \neq e \rightarrow I[i \mapsto i+1]) \\
\Leftrightarrow\ & i \leq u \wedge a[i] \neq e \wedge i+1 > u \rightarrow (\forall j.\ l \leq j \leq u \rightarrow a[j] \neq e) \\
\Leftrightarrow\ & i = u \wedge a[i] \neq e \rightarrow (\forall j.\ l \leq j \leq u \rightarrow a[j] \neq e) \\
\Leftrightarrow\ & i = u \wedge a[i] \neq e \rightarrow (\forall j.\ l \leq j < u \rightarrow a[j] \neq e)
\end{aligned}
$$

至此，为保证后置条件成立，我们得到了两个在循环头必须成立的公式：

$$i > u \to (\forall j.\ l \leq j \leq u \to a[j] \neq e)$$
$$i = u \land a[i] \neq e \to (\forall j.\ l \leq j < u \to a[j] \neq e)$$

将上述两式泛化为：

$$\forall j.\ l \leq j < i \to a[j] \neq e$$

于是，可以将循环不变式增强为：

$$l \leq i \leq u + 1 \land (\forall j.\ l \leq j < i \to a[j] \neq e)$$

利用 SMT 求解器确认循环不变式

下节课内容

- 程序的自动机表示

谢谢!