



软件分析

# 符号抽象

熊英飞

北京大学



# 抽象解释是非组合式的

- 程序设计语言的语义通常用组合的方式定义
  - $\mu(x * y + y) = \mu(x * y) + \mu(y) = \mu(x) * \mu(y) + \mu(y)$
- 抽象解释的组合会丢失精度
  - $\sigma(x - x) = \sigma(x) - \sigma(x)$
- 假设x为1，则
  - $\sigma(1) = \text{正}$
  - $\text{正} - \text{正} = \text{靛}$
- 但实际上执行x-x的结果恒为0



# 将表达式作为整体抽象

- 将表达式看做函数，我们希望得到表达式的最精确抽象
- $\sigma(x - x) = \text{零}$
- 如何得到这样的最精确抽象？
  - 可能的表达式种类无限，无法一一定义

# 符号抽象Symbolic Abstraction



- 2004年由Tom Reps等人提出
- 利用SMT Solver的求解能力，自动找到函数的最精确抽象



# 抽象域计算问题

- 给定程序和抽象域上的输入，求抽象域上最精确的输出
- 如，给定
  - $x = \text{负}$
  - 求  $x - x$  在抽象域上的计算结果
- 答案：零



# 最小抽象

- 如何定义最精确的抽象?
- 最小抽象:
  - 给定具体域集合 $S$ 和具体化函数 $\gamma$ ,
  - 甲为 $S$ 的最小抽象当且仅当
    - $S \subseteq \gamma(\text{甲})$ 且
    - 对任意乙,  $S \subseteq \gamma(\text{乙}) \Rightarrow \text{甲} \sqsubseteq \text{乙}$
- $S$ 的最小抽象记为 $\hat{\alpha}(S)$



# 最小抽象的存在性

- 最小抽象不一定存在
  - $\gamma(\text{甲}) = \{1, 2\}$
  - $\gamma(\text{乙}) = \{2, 3\}$
  - $\{2\}$ 没有最小抽象



# 最小抽象存在性

- 定义 $\beta$ 为从具体值到最小抽象值的映射，即 $\beta(x) = \hat{\alpha}(\{x\})$
- 定理：给定具体值的集合 $S$ ，如果对任意 $x \in S$ ， $\beta(x)$ 都有定义，则该集合的最小抽象 $\hat{\alpha}(S)$ 满足
  - $\hat{\alpha}(S) = \sqcup_{x \in S} \beta(x)$
- 证明：
  - 容易证明 $S \subseteq \gamma(\sqcup \{ \beta(x) \mid x \in S \})$ ，接下来证明这个抽象最小
  - 对任意抽象值 $\alpha$ 满足 $S \subseteq \gamma(\alpha)$ 的，我们有
    - $\forall x \in S. \{x\} \subseteq \gamma(\alpha)$
  - 根据最小抽象的定义，我们有
    - $\forall x \in S. \beta(x) \sqsubseteq \alpha$
  - 即 $\alpha$ 是 $\{ \beta(x) \mid x \in S \}$ 的上界
  - 又因为 $\sqcup_{x \in S} \beta(x)$ 是最小上界，所以 $\sqcup_{x \in S} \beta(x) \sqsubseteq \alpha$





# 逻辑与集合

- 明确逻辑和集合的关系对后续理解有帮助
- 任何逻辑表达式定义了一个集合：满足该表达式的值的集合
  - $\varphi: x > 0$
  - 定义了
  - $\llbracket \varphi \rrbracket: \{x \mid x > 0\}$
- $\gamma$ 可以写成从抽象值到逻辑表达式的映射
- 子集关系也就对应了逻辑蕴含关系
  - $\llbracket \varphi_1 \rrbracket \subseteq \llbracket \varphi_2 \rrbracket \Leftrightarrow \varphi_1 \rightarrow \varphi_2$



# RSY算法

- Tom Reps等人2004年的论文提出RSY算法
- 假设抽象域和具体域上定义了如下操作和特殊值
  - $\gamma$ 从抽象值到SMT表达式的映射
  - $\mu$ 从程序到SMT表达式的映射
  - $\beta$ 从具体值到最小抽象值的映射，即 $\beta(x) = \hat{\alpha}(\{x\})$
  - 甲 $\sqcup$ 乙：抽象值的并，返回甲乙的最小上界
  - 最小抽象值 $\perp$ 使得 $\gamma(\perp) = \emptyset$
- 注意以上操作都是计算机可表示的



# RSY算法

- 抽象域计算问题：
  - 给定程序 $p$ 和抽象域上的输入 $\alpha$ ，求抽象域上最精确的输出
  - 即：寻找在输入集合 $\gamma(\alpha)$ 下， $p$ 的输出集合的最小抽象
- $\hat{\alpha}(S) = \sqcup_{x \in S} \beta(x)$
- 基本原理：不断调用SMT Solver寻找在 $S$ 中但不在当前结果中的元素 $x$ ，然后将 $\beta(x)$ 并入当前结果



# RSY算法

- 输入：程序  $p$
- 输入：  $p$  的抽象输入  $\alpha$

result =  $\perp$

While(sat( $\mu(p) \wedge \gamma(\alpha) \wedge \neg \gamma(\text{result})$ ))

$y = \text{get-model}()$

    result = result  $\sqcup \beta(y)$

return result



# 示例

- 程序：  $x-x$
- 输入：  $x=正$
- 运行过程：
  - $result = \perp$ ,  $r = x - x \wedge x > 0 \wedge \neg(false)$ 可满足,  $r=0$
  - $result=零$ ,  $r = x - x \wedge x > 0 \wedge \neg(r = 0)$ 不可满足
  - 程序结束, 返回零



# 示例

- 程序：  $x+y$
- 输入：  $x=正$ ，  $y=负$
- 运行过程：
  - $result = \perp$ ，  
 $r = x + y \wedge x > 0 \wedge y < 0 \wedge \neg(false)$ 可满足，  $r=0$
  - $result=零$ ，  $r = x + y \wedge x > 0 \wedge y < 0 \wedge \neg(r = 0)$ 可满足，  $r=1$
  - $result=躲$ ，  $r = x + y \wedge x > 0 \wedge y < 0 \wedge \neg(true)$ 不可满足
  - 程序结束，返回躲



# 从值的抽象到程序的抽象

- RSY算法可以计算具体值的最小抽象
- 定义函数 $f$ 的最小抽象为 $\hat{\alpha} \circ f \circ \gamma$
- 如何得到函数 (=程序) 的最小抽象?



# 方法1： 计算每个输出

- 因为抽象域往往大小有限，抽象域上的函数可以直接用输入输出对来记录
- 针对每个输入通过RSY算法计算输出即可

$$f(x)=x+5$$

☞

输入	输出
⊥	⊥
正	正
负	✖
零	正
✖	✖

$$f(x,y)=x*y$$

☞

	正	负	零	✖	⊥
正	正				
负	负	正			
零	零	零	零		
✖	✖	✖	✖	✖	
⊥	⊥	⊥	⊥	⊥	⊥





# 方法2：直接计算函数

- 定义函数的抽象域，直接用RSY算法计算出函数的最小抽象
  - 一次求解的结果可用于更新多个输入的输出，提高效率
- 函数的抽象域：包括值抽象域上的所有函数
- 函数的偏序关系：
  - $\mathcal{F}_1 \sqsubseteq \mathcal{F}_2$  当且仅当  $\forall \theta. \mathcal{F}_1(\theta) \sqsubseteq \mathcal{F}_2(\theta)$
- $\gamma(\mathcal{F})$  定义：
  - $f \in \gamma(\mathcal{F})$  当且仅当  $\forall \theta. f \circ \gamma(\theta) \sqsubseteq \mathcal{F}(\gamma(\theta))$
- 定理：
  - $\hat{\alpha}(f) = \hat{\alpha} \circ f \circ \gamma$
  - 证明：易证  $f \in \gamma(\hat{\alpha} \circ f \circ \gamma)$ 。设存在  $\mathcal{F}$  满足  $f \in \gamma(\mathcal{F})$ ，则  $\forall \theta. f \circ \gamma(\theta) \sqsubseteq \mathcal{F}(\gamma(\theta))$ ，即  $\forall \theta. \hat{\alpha} \circ f \circ \gamma(\theta) \sqsubseteq \mathcal{F}(\theta)$



# 定义RSY算法需要的操作

- 函数抽象合并
  - $(\gamma_1 \sqcup \gamma_2)(\alpha) = \gamma_1(\alpha) \sqcup \gamma_2(\alpha)$
  - 即合并对应输入上的输出
- 最小函数抽象
  - $\gamma_{\perp}(\_) = \perp$
- $\beta$ 在函数上的扩展
  - $\beta([x, y])(\alpha) = \begin{cases} \perp, & \neg(x \in \gamma(\alpha)) \\ \beta(y), & x \in \gamma(\alpha) \end{cases}$
- $\gamma$ 在函数上的扩展：
  - 依次翻译输入输出对
  - [正, 负], ... 翻译为
  - $x > 0 \rightarrow r < 0 \wedge \dots$



# 用RSY算法抽象程序

- 输入：程序p

result =  $\perp$

While(sat( $\mu(p) \wedge \neg \gamma(\text{result})$ ))

    y=get-model()

    result=result  $\sqcup \beta(y)$

return result



# 示例

- 程序：  $x-x$
- 运行过程：
  - $result = \exists \perp$ ,  
 $r = x - x \wedge \neg(x > 0 \rightarrow false \wedge \dots)$  可满足,  $[x,r]=[1, 0]$
  - $result = \{[正, 零], [负, \perp], [零, \perp], [躲, 零]\}$ ,  
 $r = x - x \wedge \neg(x > 0 \rightarrow r = 0 \wedge (true \rightarrow r = 0) \dots)$   
可满足,  $[x,r]=[-1, 0]$
  - $result = \{[正, 零], [负, 零], [零, \perp], [躲, 零]\}$ ,  
 $r = x - x \wedge \neg(\dots)$  可满足,  $[x,r]=[0, 0]$
  - $result = \{[正, 零], [负, 零], [零, 零], [躲, 零]\}$ ,  
 $r = x - x \wedge \neg(\dots)$  不可满足



# 符号抽象问题

- 抽象域计算问题和程序抽象问题可以统一成如下符号抽象问题
- 给定逻辑公式 $\varphi$ ，抽象域**虚**，寻找抽象域中关于公式 $\varphi$ 的最小抽象甲，即满足
  - $\llbracket \varphi \rrbracket \subseteq \gamma(\text{甲}) \wedge$
  - $\forall \text{乙}: \llbracket \varphi \rrbracket \subseteq \gamma(\text{乙}) \rightarrow \gamma(\text{甲}) \subseteq \gamma(\text{乙})$



# 参考资料

- Thomas W. Reps, Aditya V. Thakur: Automating Abstract Interpretation. VMCAI 2016. 3-40