软件分析

符号抽象

熊英飞
北京大学

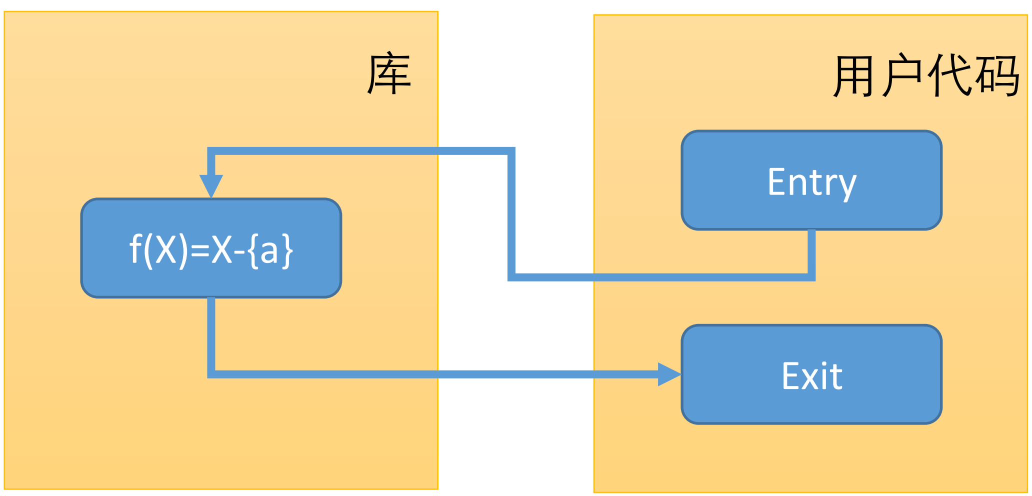
复习：函数抽象

- 给定伽罗瓦连接$(\boldsymbol{D}, \subseteq) \leftrightharpoons_{\alpha}^{\gamma}$ (虚, $\sqsubseteq$)
- 给定$\boldsymbol{D}$上的函数f和虚上的函数予

- 予是 f 的安全抽象,当且仅当
  - $(\alpha \circ f \circ \gamma)$(甲) $\sqsubseteq$ 予(甲)
  - 即$(f \circ \gamma)$(甲) $\sqsubseteq$ $(\gamma \circ$ 予$)$(甲)
- 予是 f 的最佳抽象,当且仅当
  - $\alpha \circ f \circ \gamma =$ 予
- 予是 f 的精确抽象,当且仅当
  - $f \circ \gamma = \gamma \circ$ 予
- 最佳抽象总是存在，但精确抽象不一定存在

复习：基于函数摘要的加速技术

- 将一个过程摘要成一个转换函数
- 如果节省下来的冗余计算大于摘要花费，则加速了程序分析
- 库函数可以提前做成摘要，在分析用户代码的时候直接使用摘要



能否保证函数摘要是最佳抽象?

- 之前的摘要沿着语法组合抽象语义
- 但抽象解释的组合会丢失精度
  - $\alpha(x - x) = \alpha(x) - \alpha(x)$
- 假设x为1，则
  - $\alpha(1) =$ 正
  - 正 − 正 =槑
- 但实际上执行x-x的结果恒为0
- 换句话说，最佳抽象在运算过程中丢失了。

将函数作为整体抽象

- 我们希望得到函数整体的最精确抽象
- $\alpha(x - x) =$ 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=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、抽象域和具体化函数γ,
  - 给定具体域集合S,
  - 甲为S的最小抽象当且仅当
    - $\mathrm{S} \subseteq \gamma(\text{甲})$且
    - 对任意乙,$\mathrm{S} \subseteq \gamma(\text{乙}) \Rightarrow \text{甲} \sqsubseteq \text{乙}$
- 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β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
  - 对任意抽象值甲满足S $\subseteq \gamma$(甲)的，我们有
    - $\forall x \in S. \{x\} \subseteq \gamma(\text{甲})$
  - 根据最小抽象的定义，我们有
    - $\forall x \in S. \beta(x) \sqsubseteq \text{甲}$
  - 即甲是$\{ \beta(x) \mid x \in S \}$的上界
  - 又因为$\sqcup_{x \in S} \beta(x)$是最小上界，所以$\sqcup_{x \in S} \beta(x) \sqsubseteq \text{甲}$

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
  - $[\![\varphi]\!]: \{ x \mid x > 0 \}$
- γ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
  - $[\![\varphi_1]\!] \subseteq [\![\varphi_2]\!] \Leftrightarrow \varphi_1 \rightarrow \varphi_2$

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
  - $\dot{\gamma}$从抽象值到具体域上SMT表达式的映射
  - $\dot{\mu}$从程序到具体域上SMT表达式的映射
  - β从具体值到最小抽象值的映射，即$\beta(x) = \hat{\alpha}(\{x\})$
  - 甲⊔乙：抽象值的并，返回甲乙的最小上界
  - 最小抽象值⊥ 使得$\gamma(\bot) = \emptyset$
- 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和抽象域上的输入甲，求抽象域上最精确的输出
  - 即：寻找在输入集合γ(甲)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 甲

result =⊥

While(sat($\dot{\mu}(\mathrm{p}) \wedge \dot{\gamma}(甲) \wedge \neg\dot{\gamma}(\mathrm{result})$))

    y=get-model()

    result=result $\sqcup$ $\beta(\mathrm{y})$

return result

示例

- 程序： x-x
- 输入： x=正

- 运行过程:
  - result =⊥， $r = x - x \wedge x > 0 \wedge \neg(\mathrm{false})$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 =⊥ , $r = x + y \land x > 0 \land y < 0 \land \neg(\text{false})$可满足， r=0
  - result=零， $r = x + y \land x > 0 \land y < 0 \land \neg(r = 0)$可满足， r=1
  - result=槑， $r = x + y \land x > 0 \land y < 0 \land \neg(true)$不可满足
  - 程序结束，返回槑

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$(\boldsymbol{D}, \subseteq) \underset{\hat{\alpha}}{\overset{\gamma}{\leftrightharpoons}} (\boldsymbol{\text{虚}}, \sqsubseteq)$ 上的最佳抽象，即
  - $\hat{\alpha} \circ f \circ \gamma$
- 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	槑
零	正
槑	槑

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

	正	负	零	槑	⊥
正	正				
负	负	正			
零	零	零	零		
槑	槑	槑	槑	槑	
⊥	⊥	⊥	⊥	⊥	⊥

方法2：直接计算函数

- 方法1要对每个抽象值分别计算，存在一定重复计算
- 解决方案：函数是输入输出对的集合，直接计算该集合的最小抽象

抽象域定义

- 抽象域：原抽象域上的函数（即抽象值对的集合）
- 函数的偏序关系：
  - $子_1 \sqsubseteq 子_2$当且仅当$\forall 甲. 子_1(甲) \sqsubseteq 子_2(甲)$
- $\boldsymbol{\gamma}(子)$定义为$\left\{ (a, b) \,\middle|\, b \in \bigcap_{a \in \gamma(甲)} \gamma \circ 子(甲) \right\}$
- 定理：$\boldsymbol{\hat{\alpha}}(f) = \hat{\alpha} \circ f \circ \gamma$
  - 证明：
    - 首先证明$f \subseteq \boldsymbol{\gamma}(\hat{\alpha} \circ f \circ \gamma)$
      - 对任意$(a, b) \in f$，我们有$a \in r \circ \beta(a)$，即$b \in f \circ r \circ \beta(a)$
      - 那么我们得到$b \in \gamma \circ \hat{\alpha} \circ f \circ r \circ \beta(a)$
    - 设存在子满足$f \subseteq \boldsymbol{\gamma}(子)$，证明$\hat{\alpha} \circ f \circ \gamma \sqsubseteq 子$
      - 即对任意甲，我们要证明$\hat{\alpha} \circ f \circ \gamma(甲) \sqsubseteq 子(甲)$
      - 因为$f \subseteq \boldsymbol{\gamma}(子)$，我们有$\forall a \in \gamma(甲), f(a) \in \gamma \circ 子(甲)$，即$f \circ \gamma(甲) \subseteq \gamma \circ 子(甲)$
      - 根据伽罗瓦连接的定义，得$\hat{\alpha} \circ f \circ \gamma \sqsubseteq 子$

定义RSY算法需要的操作

- 函数抽象合并
  - $(\text{子}_1 \sqcup \text{子}_2)(\text{甲}) = \text{子}_1(\text{甲}) \sqcup \text{子}_2(\text{甲})$
  - 即合并对应输入上的输出
- 最小函数抽象
  - $\text{子}_\bot(\_) = \bot$
- β在输入输出对上的扩展$\boldsymbol{\beta}\big((x, y)\big) = \hat{\alpha} \circ (x, y) \circ \gamma$，即
  - $\boldsymbol{\beta}\big((x, y)\big)(\text{甲}) = \begin{cases} \bot, & \neg(x \in \gamma(\text{甲})) \\ \beta(y), & x \in \gamma(\text{甲}) \end{cases}$
- $\dot{\gamma}$在函数上的扩展：
  - 依次翻译输入输出对
  - [正, 负], … 翻译为
  - $x > 0 \to r < 0 \land \cdots$

用RSY算法抽象程序

- 输入：程序p

result =[illegible]$_{\perp}$

While(sat($\dot{\mu}(\mathrm{p}) \land \neg\dot{\gamma}(\mathrm{result})$))

    y=get-model()

    result=result $\sqcup$ $\beta(\mathrm{y})$

return result

示例

- 程序： x-x

- 运行过程:
  - result =子$_{\perp}$,
    $r = x - x \wedge \neg(x > 0 \rightarrow false \wedge \cdots)$可满足， [x,r]=[1, 0]
  - result ={[正, 零], [负, ⊥], [零, ⊥], [槑, 零]},
    $r = x - x \wedge \neg(x > 0 \rightarrow r = 0 \wedge \cdots \wedge (true \rightarrow r = 0))$
    可满足， [x,r]=[-1, 0]
  - result ={[正, 零], [负, 零], [零, ⊥], [槑, 零]},
    $r = x - x \wedge \neg(\ldots)$可满足， [x,r]=[0, 0]
  - result ={[正, 零], [负, 零], [零, 零], [槑, 零]},
    $r = x - x \wedge \neg(\ldots)$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φ，抽象域虚，寻找抽象域中关于公式φ的最小抽象甲，即满足
  - $[\![\varphi]\!] \subseteq \gamma(\text{甲}) \wedge$
  - $\forall \text{乙}$：$[\![\varphi]\!] \subseteq \gamma(\text{乙}) \rightarrow \gamma(\text{甲}) \subseteq \gamma(\text{乙})$

局限性

- 符号抽象需要不断计算最小抽象并合并
- 在较大或者无限的抽象域（比如区间）上表现不好
  - 无法按输入输出对记录
  - 直接在函数定义中记录⊔可能导致很大的函数
- 最新工作
  - 将最佳摘要合成问题看做程序合成问题
  - 程序合成只能保证合成出正确的抽象，不能保证最佳
  - 修改CEGIS为两轮
    - 首先检查合成的程序是否正确，否则产生新的要被满足的样例
    - 然后检查合成的程序是否精确，否则产生新的要不被满足的样例
      - 看做新的程序合成问题，同时合成更精确的程序和该程序不满足的样例
  - 论文
    - Kanghee Park, Loris D'Antoni, Thomas W. Reps: Synthesizing Specifications. Proc. ACM Program. Lang. 7(OOPSLA2): 1787-1816 (2023)
    - Pankaj Kumar Kalita, Sujit Kumar Muduli, Loris D'Antoni, Thomas W. Reps, Subhajit Roy: Synthesizing abstract transformers. Proc. ACM Program. Lang. 6(OOPSLA2): 1291-1319 (2022)

作业

- 考虑下面函数
    - int abs(int x, int y) {
    -   if (x > y) return x – y;
    -   else return y – x;
    - }
- 假设抽象域是{⊥,正零, 负,槑}
- 请给出一个可能的符号抽象执行过程，给出result如何一步步从[illegible]$_{\perp}$演变为最小抽象，并给出每一步的反例

Result	反例
[illegible]$_{\perp}$	…
…	…

参考资料

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