



软件科学基础

Separation Logic & Incorrectness Logic a Quick Look

熊英飞
北京大学



扩展霍尔逻辑验证指针

```
void swap(int *x, int* y) {  
    *x = (*x)^(*y);  
    *y = (*x)^(*y);  
    *x = (*x)^(*y);  
}
```

下面的霍尔三元组成立吗？

$$\{ *x = a \wedge *y = b \} \text{swap}(x, y) \{ *x = b \wedge *y = a \}$$



扩展霍尔逻辑验证指针

- 不成立，因为x和y可能是别名（指向同一块地址）
- 一个成立的霍尔三元组如下所示

$$\{x \neq y \rightarrow *x = a \wedge *y = b\} \text{swap}(x, y) \{x \neq y \rightarrow *x = b \wedge *y = a\}$$

- 对于指针程序的验证需要反复论证别名
 - 如链表需要无环，两个链表连接不能连接自己的一部分
- 能否把别名的论证变成逻辑的一部分？



分离逻辑Separation Logic

- 扩展了霍尔逻辑，将“无别名”作为逻辑符号的一部分
- 源于John C. Reynolds, Peter O'Hearn, Samin Ishtiaq and Hongseok Yang等人在99-01年期间发表的四篇论文
- 是Facebook的程序分析工具Infer的理论基础（理论上）
- 本世纪程序验证上最重要的工作之一
 - 获得2016年的CAV Award和哥德尔奖



分离逻辑概念

- Store: 从变量名到变量值的映射
 - 表示栈
- Heap: 从内存地址到内存值的部分映射
 - 表示堆
 - $h_1 \perp h_2$ 表示两个堆不交，即定义域交集为空
 - $h_1 \cup h_2$ 表示合并两个不交的堆



分离逻辑断言

- 语法

$$\begin{aligned} \text{Ass} &:= && \mathbf{emp} \\ &| && \text{Exp}_1 \mapsto \text{Exp}_2 \\ &| && \text{Ass}_1 * \text{Ass}_2 \\ &| && \text{Ass}_1 \text{ -* } \text{Ass}_2 \\ &| && \text{其他一阶逻辑式子} \\ \text{Exp} &:= && \text{程序表达式} \end{aligned}$$

- 语义

- 三元关系 $s, h \models P$ 表示谓词 P 在 Store s 和 Heap h 上成立



emp

- $s, h \models \mathbf{emp}$
 - 表示h为空，对所有内存地址都未定义



$Exp_1 \mapsto Exp_2$

- $s, h \models e \mapsto e'$
 - 表示地址 e 保存了值 e' ，即 $h([e]_s) = [e']_s$ 且 h 对 $[e]_s$ 以外的值都没有定义，其中 $[e]_s$ 表示 e 在 s 上求值的结果
 - 如 $x \mapsto 1$ 表示指针 x 所指向的地址保存了值1，且堆中没有其他地址
 - 如 $x \mapsto n + 1$ 表示指针 x 所指向的地址保存的值等于变量 n 保存的值加上1，且堆中没有其他地址



$Ass_1 * Ass_2$ (分离合取、星号)

- $s, h \models P * Q$
 - $\exists h_1, h_2$, 使得 $h = h_1 \cup h_2 \wedge h_1 \perp h_2$
 - $s, h_1 \models P$
 - $s, h_2 \models Q$
 - 即 P 和 Q 都成立, 但其中地址不相交
 - 分离逻辑中最重要的逻辑符号
 - 如 $x \mapsto a * y \mapsto b$ 说明 x 和 y 不是别名
 - 如: 下面递归定义了判断 p 是否为链表的谓词, $*$ 用来保证链表不会出现环
 - $list(p) = p = null \rightarrow \mathbf{emp}$
 $\wedge p \neq null$
 $\rightarrow \exists a, p', p \mapsto \{value: a, next: p'\} * list(p')$
 - 练习: 用分离合取定义二叉树的谓词
 - 性质: $P = \mathbf{emp} * P = P * \mathbf{emp}$



$Ass_1 - * Ass_2$ (分离蕴含、魔法棒)

- $s, h \models P - * Q$
 - $\forall h', (h' \perp h) \wedge ((s, h' \models P) \rightarrow (s, h' \cup h \models Q))$
 - 即给h补上一块满足P的堆就能满足Q
 - 如 $(x \mapsto 1) - * Q$ 表示当前堆不包括x，但加上一个x指向1的堆就满足Q
- 分离合取和分离蕴含具有一些普通合取和蕴含的性质

$$\bullet \frac{\models P \wedge (P \rightarrow Q)}{\models Q}$$

$$\frac{s, h \models P * (P - * Q)}{s, h \models Q}$$



扩展霍尔三元组

- $\{P\}c\{Q\}$

- 表示对任意满足P的s和h，如果执行语句c之后得到了s'和h'，那么s'和h'满足Q

- 如：

- $\{x \mapsto a * y \mapsto b\} \text{swap}(x,y) \{x \mapsto b * y \mapsto a\}$



推导规则-新增语句相关

- Store: $\{x \mapsto -\} * x = v \{x \mapsto v\}$
- Load: $\{x \mapsto v\} a = *x \{a = v \wedge x \mapsto v\}$
- Alloc: $\{\mathbf{emp}\} x = \text{malloc}() \{x \mapsto -\}$
- AllocFail: $\{\mathbf{emp}\} x = \text{malloc}() \{x \mapsto - \vee x = 0\}$
- DeAlloc: $\{x \mapsto -\} \text{free}(x) \{\mathbf{emp}\}$

- “ $x \mapsto -$ ”表示 “ $\exists v, x \mapsto v$ ”



推导规则-Frame规则

• Frame :

$$\bullet \quad \frac{\{P\}c\{Q\}}{\{P*R\}c\{Q*R\}}$$



完整性

- 以上推导规则，加上断言上的公理（本课程跳过）、部分霍尔逻辑规则和一阶逻辑规则，可以推导出所有的为真的三元组



证明举例1

- 证明定理: $\{x \mapsto a * y \mapsto b\}t=*x;*x=*y;*y=t;\{x \mapsto b * y \mapsto a\}$
- 根据Load可得
 - $\{x \mapsto a\}t=*x\{t = a \wedge x \mapsto a\}$
- 注意 $t = a$ 不涉及到堆, 根据Consequence可得
 - $\{x \mapsto a\}t=*x\{(t = a \wedge \mathbf{emp}) * x \mapsto a\}$
- 再根据Frame可得
 - $\{x \mapsto a * y \mapsto b\}t=*x\{(t = a \wedge \mathbf{emp}) * x \mapsto a * y \mapsto b\}$
- 根据Load和Store可得
 - $\{x \mapsto - * y \mapsto b\}*x=*y\{x \mapsto b * y \mapsto b\}$
- 根据Frame和Consequence可得
 - $\{(t = a \wedge \mathbf{emp}) * x \mapsto a * y \mapsto b\}*x=*y\{(t = a \wedge \mathbf{emp}) * x \mapsto b * y \mapsto b\}$
- 类似可得
 - $\{(t = a \wedge \mathbf{emp}) * x \mapsto b * y \mapsto b\}*y=t\{x \mapsto b * y \mapsto a\}$
- 根据Seq, 原命题得证



证明举例2

- ```
void delete(p) {
 if (p = null) return;
 else {
 delete(p->next);
 free(p);
 }
}
```
- 证明:  $\{list(p)\}delete(p)\{\mathbf{emp}\}$



# 证明举例2

- 先考虑then分支，根据Skip可得
  - $\{\mathbf{emp}\}skip\{\mathbf{emp}\}$
- 再根据Consequence可得
  - $\{list(p) \wedge p = null\}skip\{\mathbf{emp}\}$
- 再考虑else分支，根据DeAlloc可得
  - $\{p \mapsto -\}free(p)\{\mathbf{emp}\}$
- 根据归纳假设可得
  - $\{list(p \rightarrow next)\}delete(p \rightarrow next)\{\mathbf{emp}\}$
- 从上面两项，根据Frame可得
  - $\{list(p \rightarrow next) * p \mapsto -\}delete(p \rightarrow next)\{p \mapsto - * \mathbf{emp}\}$
  - $\{p \mapsto - * \mathbf{emp}\}free(p)\{\mathbf{emp} * \mathbf{emp}\}$
- 再根据Consequence可得
  - $\{(list(p) \wedge p \neq null)\}delete(p \rightarrow next)\{p \mapsto - * \mathbf{emp}\}$
  - $\{p \mapsto - * \mathbf{emp}\}free(p)\{\mathbf{emp}\}$
- 最后根据If，原命题得证



# 教材和分离逻辑相关内容

- 教材第5卷：一套用分离逻辑验证C语言的证明系统VST
- 教材第6卷：一套基于Ocaml程序的验证介绍分离逻辑的教程



# 分离逻辑证明工具推荐

- C\*
  - <https://cstarlang.org/>
  - 北京大学程序设计语言研究室领导开发
  - 在C语言中添加分离逻辑断言和证明
  - 断言是C语言数据，证明语句是函数，对断言进行转换。证明策略可以直接用C语言编写。
- QCP
  - <https://www.qua.codes/index.html>
  - 上海交大曹钦翔老师团队开发，VST的后续
  - 在C中用注释方式加入分离逻辑标注，导出的命题用Rocq证明



# Incorrectness Logic

- Peter W. O'Hearn指示团队用分离逻辑开发Infer
- 但发现团队不听他的，反而写了很多启发式规则查错
- 这些启发式规则没有理论基础，但居然很有效
- 逻辑学背景的O'Hearn不能容忍这件事，决定要做一套新的逻辑
- Peter W. O'Hearn: Incorrectness logic. Proc. ACM Program. Lang. 4(POPL): 10:1-10:32 (2020)



# 下近似三元组

- [presumption]code[result]
- 对任意满足result的状态s，一定存在一个满足presumption的状态，使得执行code之后到达s
  - 即：后条件覆盖状态是后状态的子集
- 对比：
  - 霍尔三元组：上近似三元组
  - 后条件覆盖状态是后状态的超集



# 例子

```
1 /* presumes: [z==11] */
2 if (x is even) {
3 if (y is odd) {
4 z=42;
5 } }
6 /* achieves: [z==42 && (x is even) && (y is odd)] */
```



# 带返回类型的下近似三元组

- [presumption]code[re:result]
  - re可以是ok或er或者其他类型
  - ok表示正常返回
  - er表示程序执行了error()

```
1 void foo(char* str)
2 /* presumes: [*str[]]==s]
3 achieves: [er: *str[]]==s && length(s) > 16] */
4 { char buf[16];
5 strcpy(buf, str);
6 }
7
8 int main(int argc, char *argv[])
9 { foo(argv[1]); }
```



# 一般性规则

*Empty under-approximates*

$$[p]C[\epsilon: false]$$

*Consequence*

$$\frac{p' \Leftarrow p \quad [p]C[\epsilon: q] \quad q \Leftarrow q'}{[p']C[\epsilon: q']}$$

*Disjunction*

$$\frac{[p_1]C[\epsilon: q_1] \quad [p_2]C[\epsilon: q_2]}{[p_1 \vee p_2]C[\epsilon: q_1 \vee q_2]}$$

*Unit*

$$[p]skip[ok: p][er: false]$$

*Sequencing (short-circuit)*

$$\frac{[p]C_1[er: r]}{[p]C_1; C_2[er: r]}$$

*Sequencing (normal)*

$$\frac{[p]C_1[ok: q] \quad [q]C_2[\epsilon: r]}{[p]C_1; C_2[\epsilon: r]}$$

*Iterate zero*

$$[p]C^*[ok: p]$$

*Iterate non-zero*

$$\frac{[p]C^*; C[\epsilon: q]}{[p]C^*[\epsilon: q]}$$

*Backwards Variant (where  $n$  fresh)*

$$\frac{[p(n) \wedge nat(n)]C[ok: p(n+1) \wedge nat(n)]}{[p(0)]C^*[ok: \exists n. p(n) \wedge nat(n)]}$$

*Choice (where  $i = 1$  or  $2$ )*

$$\frac{[p]C_i[\epsilon: q]}{[p]C_1 + C_2[\epsilon: q]}$$

*Error*

$$[p]error()[ok: false][er: p]$$

*Assume*

$$[p]assume B[ok: p \wedge B][er: false]$$

$$\text{while } B \text{ do } C \quad =_{def} \quad (\text{assume}(B); C)^*; \text{assume}(\neg B)$$

$$\text{if } B \text{ then } C \text{ else } C' \quad =_{def} \quad (\text{assume}(B); C) + (\text{assume}(\neg B); C')$$

$$\text{assert}(B) \quad =_{def} \quad \text{assume}(B) + (\text{assume}(\neg B); \text{error}())$$



# 变量规则

*Assignment*

$$[p]x = e [ok: \exists x'. p[x'/x] \wedge x = e[x'/x]] [er: false]$$

Mod(C): 语句C可以修改的变量集合  
Free(f): f中的自由变量

*Constancy*

$$\frac{[p]C[\epsilon: q]}{[p \wedge f]C[\epsilon: q \wedge f]} \text{Mod}(C) \cap \text{Free}(f) = \emptyset$$

*Substitution I*

$$\frac{[p]C[\epsilon: q]}{([p]C[\epsilon: q])(e/x)} (\text{Free}(e) \cup \{x\}) \cap \text{Free}(C) = \emptyset$$

*Substitution II*

$$\frac{[p]C[\epsilon: q]}{([p]C[\epsilon: q])(y/x)} y \notin \text{Free}(p, C, q)$$

原霍尔逻辑规则不保证下近似  
比如:  $\{42=y\}x=42\{x=y\}$   
但 $x=y=3$ 也满足后条件



# 例子

```
1 /* presumes: [z==11] */
2 if (x is even) {
3 if (y is odd) {
4 z=42;
5 } }
6 /* achieves: [z==42 && (x is even) && (y is odd)] */
```

- 展开if定义，得到
  - $\text{assume}(x \text{ is even}); (\text{assume}(y \text{ is odd}); z=42 + \text{assume}(\neg y \text{ is odd}); \text{skip}) + \text{assume}(\neg x \text{ is even}); \text{skip}$
- 根据choice，选择
  - $\text{assume}(x \text{ is even}); \text{assume}(y \text{ is odd}); z=42$
- 根据assume，得到：
  - $[\text{True}]\text{assume}(x \text{ is even})[\text{ok}: x \text{ is even}]$
  - $[x \text{ is even}]\text{assume}(y \text{ is odd})[\text{ok}: x \text{ is even} \wedge y \text{ is odd}]$
- 根据assignment，得到：
  - $[x \text{ is even} \wedge y \text{ is odd}]z=42[x \text{ is even} \wedge y \text{ is odd} \wedge z==42]$
- 根据sequencing，原命题得证



# 性质

- 正确性：所有推出的三元组都是下近似三元组
- 完整性：所有有限支撑（finitely-supported）下近似三元组都能推出
  - 有限支撑：一个谓词只依赖于有限个变量