的性质一定是正确的,我们的工具在合成性质的数量和时间上都明显优于 CppInv,并且能够处理多维数组.

Table 5 Synthesis or verification results of CppInv, BOOSTER, ESBMC, SMACK+Corral and our tool

表 5 CppInv,BOOSTER,ESBMC,SMACK+Corral 及我们的工具合成或验证结果

工具	合成或验证的数量	时间(s)	是否需要提供断言	是否能处理多维数组	是否 sound	是否直接支持量词性质
CppInv	27	15 163	否	否	是	是
BOOSTER	36	1 855	是	否	否	是
ESBMC	39	10 989	是	是	否	否
SMACK+Corral	44	10 310	是	是	否	否
我们的工具	44	703	否	是	是	是

6 相关工作

合成数组程序不变式的相关工作可分为如下几类:数组展开(array expansion)方法、Array smashing 方法、数组划分方法、谓词抽象方法、基于模板的方法、基于定理证明的方法.

- 数组展开方法[23].这个方法展开循环和数组单元来处理程序.数组展开方法优点是精确,缺点是只能处理小尺寸的数组,不能处理无界数组;
- Array smashing 方法[5,24].这个方法将整个数组 A 看作一个变量 a.初始时,a 被赋予一个整个数组单元都满足的已知最强的性质.$A[i]:=e$ 按照弱赋值 $a⊔:=e$ 来处理.Array smashing 方法的缺点是弱赋值(weak assignment⊔:=)会丢失信息,并且无法处理关于数组单元的条件测试语句.因此,这个方法获得的结果常常不精确;
- 数组划分方法[7,8,17].数组划分方法将索引域($[1...n]$)划分为若干符号区间(e.g.,$I_1=[1...i-1]$,$I_2=[i,i]$,$I_3=[i+1...n]$),并且给每个子数组 $A[I_k]$关联一个辅助摘要变量(summary auxiliary variable)a_k.该方法基于语法启发式[7,8]或者预分析[17]的方法对索引域进行划分.这类方法与本文的方法类似.区别在于:本文的方法不是给每个子数组关联一个摘要变量,而是直接将一个数组性质表示为一个或多个全称量词性质.相对于数组划分方法,本文的方法具有更高的表达能力和灵活性.因此,本文的方法可以非常容易地处理多维数组性质,但是数组划分方法难以处理多维数组.此外,数组划分方法无法处理在循环中首先进行索引更新的程序(例如图 1(b)所示的 another arrayPartCopy 程序),然而在循环中首先进行索引更新的写法在实际编码中经常使用(例如 $A[++i]$).本文的方法可以处理这类程序;
- 谓词抽象方法[1,25,26].谓词抽象方法会使用一些语法启发式策略推导用于抽象的谓词,或者由用户提供谓词.它将程序抽象到谓词程序,然后基于谓词程序合成数组性质.反例制导的精化[3]和 Craig 插值[4]提供了获取谓词的其他方法,提高了谓词抽象方法的效果.谓词抽象方法中,因为缺少有效的选择算法,因此一些原子谓词常常必须手动提供.本文的方法不需要用户提供任何输入.此外,谓词抽象方法需要使用约束求解器检查猜出来的量词性质是否真的成立,而目前约束求解器对量词性质的支持有限,因此难以处理较复杂的量词性质.本文的方法只使用到了约束求解器的线性约束求解功能,而约束求解器(如 Z3)对线性求解功能支持比较好.因此相对于这类方法,本文的方法能够自动地分析得到更复杂的量词性质;
- 基于模板的方法[9,27].基于模板的方法非常有用但是代价昂贵,其基本思想是:用户提供所需要不变式的模板,然后分析过程去搜索实例化模板参数.这个方法的过程需要用户的参与,本文的方法是完全自动化的;
- 基于定理证明的方法[18,28].这类方法使用定理证明器去生成循环不变式.文献[18]方法的基本思想是:将数组第 i 次迭代中的修改编码为一个量化事实(quantified fact),然后运用定理证明器去推导一个封闭形式的公式.这类方法的缺点是受限于使用的定理证明器,同时可证明的性质比较有局限性.

7 总结和未来工作

本文提出了一个使用抽象解释框架自动合成数组性质的方法.该方法的特点在于抽象域是性质集合(性质包括全称量词性质和原子性质),这种抽象域灵活并且具有较高的表达能力.该方法通过前向迭代数据流分析合成数组性质.本文证明了该方法是收敛的,并且获得的不变式是正确的.本文也展示了方法具有一定的灵活性,可以通过增加抽象域和传播规则来增加分析能力.本文所述分析方法的工具原型基于 clang 和 Z3 实现.该工具被用于分析一些常见程序以及 SV-COMP 的 array-examples benchmark.array-examples benchmark 共有 88 个文件中,我们的工具在 74 个文件中发现了全称量词性质.

在未来工作中,我们计划处理全称量词性质分裂和合成的情况(一个全称量词性质分为若干全称量词性质,或者将若干全称量词性质合成一个全称量词性质).一个更长远的目标是处理数组和链表的全称量词和存在量词性质.

References:

[1] Flanagan C, Qadeer S. Predicate abstraction for software verification. ACM SIGPLAN Notices, 2002,37(1):191−202. [doi: 10.1145/565816.503291]

[2] Lahiri SK, Bryant RE. Indexed predicate discovery for unbounded system verification. In: Alur R, Peled DA, eds. Proc. of the Computer Aided Verification. Berlin: SpringerVerlag, 2004. 135−147. [doi: 10.1007/978-3-540-27813-9_11]

[3] Beyer D, Henzinger TA, Majumdar R, Rybalchenko A. Path invariants. ACM SIGPLAN Notices, 2007,42(6):300−309. [doi: 10.1145/1273442.1250769]

[4] Jhala R, McMillan KL. Array abstractions from proofs. In: Damm W, Hermanns H, eds. Proc. of the Computer Aided Verification. Berlin: Springer-Verlag, 2007. 193−206. [doi: 10.1007/978-3-540-73368-3_23]

[5] Blanchet B, Cousot P, Cousot R, Feret J, Mauborgne L, Miné A, Monniaux D, Rival X. A static analyzer for large safety-critical software. ACM SIGPLAN Notices, 2003,38(5):196−207. [doi: 10.1145/780822.781153]

[6] Gopan D. Numeric program analysis techniques with applications to array analysis and library summarization [Ph.D. Thesis]. Madison: University of Wisconsin, 2007.

[7] Gopan D, Reps T, Sagiv M. A framework for numeric analysis of array operations. ACM SIGPLAN Notices, 2005,40(1):338−350. [doi: 10.1145/1047659.1040333]

[8] Halbwachs N, Péron M. Discovering properties about arrays in simple programs. ACM SIGPLAN Notices, 2008,43(6):339−348. [doi: 10.1145/1379022.1375623]

[9] Gulwani S, McCloskey B, Tiwari A. Lifting abstract interpreters to quantified logical domains. ACM SIGPLAN Notices, 2008, 43(1):235−246. [doi: 10.1145/1328897.1328468]

[10] Kam JB, Ullman JD. Monotone data flow analysis frameworks. Acta Informatica, 1977,7(3):305−317. [doi: 10.1007/BF00290339]

[11] Zhao JH, Li XD. Scope logic: An extension to hoare logic for pointers and recursive data structures. In: Liu ZM, ed. Proc. of the Theoretical Aspects of Computing (CICTAC 2013). Shanghai: Springer-Verlag, 2013. 409−426. [doi: 10.1007/978-3-642-39718-9_24]

[12] http://clang.llvm.org/

[13] De Moura L, Bjrner N. Z3: An efficient smt solver. In: Ramakrishnan CR, Rehof J, eds. Proc. of the Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-Verlag, 2008. 337−340. [doi: 10.1007/978-3-540-78800-3_24]

[14] http://sv-comp.sosy-lab.org/2017/index.php

[15] Hoare CA. Proof of a program: Find. Communications of the ACM, 1971,14(1):39−45. [doi: 10.1145/362452.362489]

[16] https://github.com/sosy-lab/sv-benchmarks/releases/tag/svcomp17

[17] Cousot P, Cousot R, Logozzo F. A parametric segmentation functor for fully automatic and scalable array content analysis. ACM SIGPLAN Notices, 2011,46(1):105−118. [doi: 10.1145/1925844.1926399]

[18] Kovács L, Voronkov A. Finding loop invariants for programs over arrays using a theorem prover. In: Chechik M, Wirsing M, eds. Proc. of the Fundamental Approaches to Software Engineering. Berlin: Springer-Verlag, 2009. 470−485. [doi: 10.1007/978-3-642-00593-0_33]

[19] Daniel L, Rodríguez-Carbonell E, Rubio A. SMT-Based array invariant generation. In: Giacobazzi R, Berdine J, Mastroeni I, eds. Proc. of the Verification, Model Checking, and Abstract Interpretation. Berlin: Springer-Verlag, 2013. 169−188. [doi: 10.1007/978-3-642-35873-9_12]

[20] Francesco A, Ghilardi S, Sharygina N. Decision procedures for flat array properties. In: Ábrahám E, Havelund K, eds. Proc. of the Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-averlag, 2014. 15−30. [doi: 10.1007/978-3-642-54862-8_2]

[21] Cordeiro L, Fischer B, Marques-Silva J. SMT-Based bounded model checking for embedded ANSI-C software. IEEE Trans. on Software Engineering, 2012,38(4):957−974. [doi: 10.1109/TSE.2011.59]

[22] Haran A, Carter M, Emmi M, Lal A, Qadeer S, Rakamarić Z. Smack+ Corral: A modular verifier. In: Baier C, Tinelli C, eds. Proc. of the Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-Verlag, 2015. 451−454. [doi: 10.1007/978-3-662-46681-0_42]

[23] Blanchet B, Cousot P, Cousot R, Feret J, Mauborgne L, Miné A, Monniaux D, Rival X. Design and implementation of aspecial-purpose static program analyzer for safety-critical real-time embedded software. In: Mogensen TÆ, Schmidt DA, Sudborough IH, eds. Proc. of the Essence of Computation. Berlin: Springer-Verlag, 2002. 85−108. [doi: 10.1007/3-540-36377-7_5]

[24] Gopan D, DiMaio F, Dor N, Reps T, Sagiv M. Numeric domains with summarized dimensions. In: Jensen K, Podelski A, eds. Proc. of the Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-Verlag, 2004. 512−529. [doi: 10.1007/978-3-540-24730-2_38]

[25] Lahiri SK, Bryant RE. Constructing quantified invariants via predicate abstraction. In: Steffen B, Levi G, eds. Proc. of the Verification, Model Checking, and Abstract Interpretation. Berlin: Springer-Verlag, 2004. 267−281. [doi: 10.1007/978-3-540-24622-0_22]

[26] Lahiri SK, Bryant RE, Cook B. A symbolic approach to predicate abstraction. In: Hunt WA, Somenzi F, eds. Proc. of the Computer Aided Verification. Berlin: Springer-Verlag, 2003. 141−153. [doi: 10.1007/978-3-540-45069-6_15]

[27] Srivastava S, Gulwani S. Program verification using templates over predicate abstraction. ACM SIGPLAN Notices, 2009,44(6): 223−234. [doi: 10.1145/1543135.1542501]

[28] McMillan KL. Quantified invariant generation using an interpolating saturation prover. In: Ramakrishnan CR, Rehof J, eds. Proc. of the Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-Verlag, 2008. 413−427. [doi: 10.1007/978-3-540-78800-3_31]

附录:证明

定理1的证明

证明文中的定理 1 之前,首先给出下面的推论及引理.

推论 1. *ProperP* 的势(大小)是有穷的.

推论 2. 对任何性质 p_1,p_2 和 p_3:

(1) 如果 $p_1 \preceq p_2$ 并且 $p_2 \preceq p_3$,那么 $p_1 \preceq p_3$;

(2) 如果 $p_1 \preceq p_2$ 并且 $p_2 \preceq p_1$,那么 $p_1 = p_2$;

(3) 如果 $p_1 \preceq p_2$,那么 $M(p_1) \subseteq M(p_2)$(注意,M(false)是$\varnothing$).

推论 3. 对于任何性质 p_1,p_2 和 p_3:

(1) 如果 $p_1 \sqcap p_2 \neq \bot$:

$$\begin{matrix} p_1 \sqcap p_2 \preceq p_1 \wedge p_1 \sqcap p_2 \preceq p_2 \\ p_1 \vee p_2 \Rightarrow p_1 \sqcap p_2 \end{matrix};$$

(2) $p_1 \preceq p_2 \Leftrightarrow p_1 = p_1 \sqcap p_2$;

(3) $p_1 \preceq p_2 \wedge p_1 \preceq p_3 \Rightarrow p_1 \preceq p_2 \sqcap p_3$.

推论 4. 对任何性质 p_1,p_2 和 p_3,$\sqcap$ 满足下面的性质:

(1) $p_1 \sqcap p_1 = p_1$;

(2) $p_1 \sqcap p_2 = p_2 \sqcap p_1$;

(3) $(p_1 \sqcap p_2) \sqcap p_3 = p_1 \sqcap (p_2 \sqcap p_3)$.

推论 5. 对任何 $S_1,S_2 \in 2^{ProperP}$:

(1) $Reduce(S_1) \in L_G$;

(2) $S_1 \sqsubseteq Reduce(S_1) \wedge Reduce(S_1) \sqsubseteq S_1$;

(3) $S_1 \sqsubseteq S_2 \Rightarrow Reduce(S_1) \sqsubseteq Reduce(S_2)$.

引理 1. 对任何 $L_1,L_2 \in L_G, L_1 \sqcap_G L_2 \in L_G$.

证明:结论直接来自 $\sqcap_G$ 的定义. □

引理 2. 对任何 $S_1,S_2,S_3 \in 2^{ProperP}$:

(1) $S_1 \sqcap_G S_2 \sqsubseteq S_1 \wedge S_1 \sqcap_G S_2 \sqsubseteq S_2$;

(2) $S_1 \sqsubseteq S_2 \wedge S_1 \sqsubseteq S_3 \Rightarrow S_1 \sqsubseteq S_2 \sqcap_G S_3$.

证明:根据推论 3,对于任何性质 $p_1 \in S_1, p_2 \in S_2, p_1 \sqcap p_2 \in S_2 \sqcap_G S_2$,并且 $p_1 \sqcap p_2 \preceq p_1 \wedge p_1 \sqcap p_2 \preceq p_2$.因此,$S_1 \sqcap_G S_2 \sqsubseteq S_1 \wedge S_1 \sqcap_G S_2 \sqsubseteq S_2$ 成立.$S_1 \sqsubseteq S_2 \wedge S_1 \sqsubseteq S_3$ 推出 $\forall p_1 \in S_1, \exists p_2 \in S_2, \exists p_3 \in S_3 \Rightarrow p_1 \preceq p_2 \wedge p_1 \preceq p_3$.根据推论 3,$S_1 \sqsubseteq S_2 \sqcap_G S_3$ 成立. □

引理 3. 对任何 $L_1,L_2 \in L_G, L_1 \sqsubseteq L_2 \wedge L_2 \sqsubseteq L_1 \Leftrightarrow L_1 = L_2$.

证明:$L_1 \sqsubseteq L_2 \wedge L_2 \sqsubseteq L_1$ 推出 $\forall p_1 \in L_1, \exists p_2 \in L_2, \exists p_3 \in L_1 \Rightarrow p_1 \preceq p_2 \wedge p_2 \preceq p_3$.$p_1 \preceq p_2 \wedge p_2 \preceq p_3 \wedge L_1 \in L_G \Rightarrow p_1 = p_3 \Rightarrow p_1 \preceq p_2 \wedge p_2 \preceq p_1 \Rightarrow p_1 = p_2$.因此 $\forall p_1 \in L_1, \exists p_2 \in L_2 \Rightarrow p_1 = p_2 \Rightarrow p_1 \in L_2$.同理可证,$\forall p_2 \in L_2, \exists p_1 \in L_1 \Rightarrow p_1 = p_2 \Rightarrow p_2 \in L_1$.因此,$L_1 = L_2$. □

引理 4. 对任何 $L_1,L_2 \in L_G, L_1 \sqsubseteq L_2 \Leftrightarrow L_1 = L_1 \sqcap_G L_2$.

证明:要证明 $L_1 \sqsubseteq L_2 \Leftrightarrow L_1 = L_1 \sqcap_G L_2$,只需证明:

1) $L_1 \sqsubseteq L_2 \Rightarrow L_1 = L_1 \sqcap_G L_2$;

2) $L_1 = L_1 \sqcap_G L_2 \Rightarrow L_1 \sqsubseteq L_2$.

根据引理 2,结论 1)可证.令 $S(L_1,L_2)=\{p_1 \sqcap p_2 | p_1 \in L_1 \wedge p_2 \in L_2 \wedge p_1 \sqcap p_2 \neq \bot\}$.根据推论 5,$L_1 \sqsubseteq L_2 \Rightarrow L_1 \sqsubseteq S(L_1,L_2) \Rightarrow L_1 \sqsubseteq Reduce(S(L_1,L_2)) \Rightarrow L_1 \sqsubseteq L_1 \sqcap_G L_2$.根据引理 2,$L_1 \sqcap_G L_2 \sqsubseteq L_1$.根据引理 3,$L_1 \sqsubseteq L_1 \sqcap_G L_2 \wedge L_1 \sqcap_G L_2 \sqsubseteq L_1 \wedge L_1 \in L_G \wedge L_1 \sqcap_G L_2 \in L_G$ 推出 $L_1 = L_1 \sqcap_G L_2$.结论 2)成立. □

定理 1. $(L_G, \sqcap_G)$是关于 CFG G 的一个交半格(meet semi-lattice),并且格的高度有穷.

证明:$(L_G, \sqcap_G)$是交半格当且仅当,对所有 $L_1,L_2,L_3 \in L_G$:

1) $L_1 \sqcap_G L_1 = L_1$;

2) $L_1 \sqcap_G L_2 = L_2 \sqcap_G L_1$;

3) $(L_1 \sqcap_G L_2) \sqcap_G L_3 = L_1 \sqcap_G (L_2 \sqcap_G L_3)$;

4) $L_1 \sqsubseteq L_2 \Leftrightarrow L_1 = L_1 \sqcap_G L_2$.

结论 1)、结论 2)和结论 4)通过引理 4 和 $\sqcap_G$ 定义可证.令 $S(L_1,L_2)=\{p_1 \sqcap p_2 | p_1 \in L_1 \wedge p_2 \in L_2 \wedge p_1 \sqcap p_2 \neq \bot\}$.要证明结论 3),只需要证明:

a) $(L_1 \sqcap_G L_2) \sqcap_G L_3 = Reduce(S(S(L_1,L_2),L_3))$;

b) $Reduce(S(S(L_1,L_2),L_3)) = Reduce(S(L_1,S(L_2,L_3)))$;

c) $Reduce(S(L_1,S(L_2,L_3))) = L_1 \sqcap_G (L_2 \sqcap_G L_3)$.

$(L_1\sqcap_G L_2)\sqcap_G L_3\sqsubseteq L_1\sqcap_G L_2\sqsubseteq S(L_1,L_2)$.根据引理 2,$(L_1\sqcap_G L_2)\sqcap_G L_3\sqsubseteq S(L_1,L_2)\sqcap_G L_3$.同理可以证明 $S(L_1,L_2)\sqcap_G L_3\sqsubseteq(L_1\sqcap_G L_2)\sqcap_G L_3$.因此,$(L_1\sqcap_G L_2)\sqcap_G L_3=S(L_1,L_2)\sqcap_G L_3=Reduce(S(S(L_1,L_2),L_3))$.同理可以证明 $Reduce(S(L_1,S(L_2,L_3)))=L_1\sqcap_G(L_2\sqcap_G L_3)$.根据函数 S 的定义,$S(S(L_1,L_2),L_3)=S(L_1,S(L_2,L_3))$可证.因此,结论 3)可证.$L_G=\{S|S\subseteq ProperP\wedge\forall p_1,p_2\in S, p_1\preceq p_2\Rightarrow p_1=p_2\}$.因此 $L_G\subseteq 2^{ProperP}$.根据推论 1,$ProperP$ 是有穷的,因此$(L_G,\sqcap_G)$的高度有穷. □

定理2的证明

证明文中的定理 2 之前,首先证明下面的引理:

引理 5. 对任何 $L_1,L_2\in L_G$,如果 $L_1\sqsubseteq L_2$,那么 $Semantics(n,L_1)\sqsubseteq Semantics(n,L_2)$.

证明:根据 *Semantics* 的定义:

(1) 如果 n 是 *cond*,那么结论显然成立;

(2) 如果 n 是 $lh:=e$,对任何 $lh_i\in LH$,如果$\wedge L_1\Rightarrow\&lh\neq\&lh_i$,那么$\wedge L_2\Rightarrow\&lh\neq\&lh_i$.因此:

$$Semantics(n,L_1)\sqsubseteq Semantics(n,L_2).$$

综上,$Semantics(n,L_1)\sqsubseteq Semantics(n,L_2)$. □

引理 6. 对任何性质集合 S_1 和 S_2,如果 $S_1\sqsubseteq S_2$,那么 $Propagated(S_1)\sqsubseteq Propagated(S_2)$.

证明:对任何 $ins_1\subseteq S_1$,如果规则 r 可以应用 ins_1,并且产生输出性质 out_1,根据 $S_1\sqsubseteq S_2$ 和传播规则约束,一定存在 $ins_2\subseteq S_2$ 和一个对应的规则 r',使得:(a) $ins_1\sqsubseteq ins_2$;(b) r'可以应用到 ins_2;(c) r'和 ins_2 产生性质 out_2 并且 $out_1\preceq out_2$.因此,$Propagated(S_1)\sqsubseteq Propagated(S_2)$. □

引理 7. 对任何 $x,y\in L_G$,如果 $x\sqsubseteq y$,$Transfer(e_1[e_2]:=e_3,x)\sqsubseteq Transfer(e_1[e_2]:=e_3,y)$.

证明:令 $TS(x)=\{p|p\in x\wedge p$ 是全称量词$\wedge\&e_1[e_2]\notin M(p)\}$.根据 *Transfer* 函数的定义,只需要证明 $TS(x)\sqsubseteq TS(y)$.对任何 $p_1\in TS(x)$,肯定存在 $p_2\in y$ 并且 $p_1\preceq p_2$.根据推论 2,$M(p_1)\subseteq M(p_2)$.$\wedge x\Rightarrow\&e_1[e_2]\notin M(p_1)$推出$\wedge y\Rightarrow\&e_1[e_2]\notin M(p_2)$,因此,$p_2\in TS(y)$.综上,$TS(x)\sqsubseteq TS(y)$. □

引理 8. 对任何 $x,y\in L_G$,如果 n 是 $i:=i+c$,$x\sqsubseteq y$,那么 $HandleInterval(n,x)\sqsubseteq HandleInterval(n,y)$.

证明:下面分情况讨论:

(1) 如果$\forall k(k\in[init_i,c,i+c)\Rightarrow p)$在 x 中,那么肯定存在$\forall k(k\in[init_i,c,i+c)\Rightarrow p')$在 y 中,因为$\{\forall k(k\in[init_i,c,i)\Rightarrow p)\}\sqsubseteq\{\forall k(k\in[init_i,c,i)\Rightarrow p')\}$,因此 $HandleInterval(n,x)\sqsubseteq HandleInterval(n,y)$;

(2) 如果$\forall k(k\in[init_i,c,i)\Rightarrow p)$在 x 中,那么肯定存在$\forall k(k\in[init_i,c,i)\Rightarrow p')$在 y 中,因为$\{\forall k(k\in[init_i,c,i-c)\Rightarrow p)\}\sqsubseteq\{\forall k(k\in[init_i,c,i-c)\Rightarrow p')\}$,因此 $HandleInterval(n,x)\sqsubseteq HandleInterval(n,y)$;

(3) 如果 $i=init_i$ 在 x 中,那么肯定存在 $i=init_i$ 在 y 中,因此 $HandleInterval(n,x)\sqsubseteq HandleInterval(n,y)$;

(4) 其他情况,$HandleInterval(n,x)\sqsubseteq HandleInterval(n,y)$.

综上,$HandleInterval(n,x)\sqsubseteq HandleInterval(n,y)$. □

引理 9. 对任何 $S_1,S_2\in 2^{ProperP}$,如果 $S_1\sqsubseteq S_2$,那么 $GenAQ(S_1)\sqsubseteq GenAQ(S_2)$.

证明:令 $\psi(k)$表示 $\psi(\ldots,e_1[f_1(k)],\ldots)$的缩写:

(1) 如果在 x 中条件(1)成立,$GenAQ(S_1)=S_1\cup\{\forall k(k\in[init_i,c,i+c)\Rightarrow\psi(k))\}$.$S_1\sqsubseteq S_2$ 推出 $i=init_i\in S_2$ 并且$\forall k(k\in[init_i,c,i)\Rightarrow$false)在 S_2 中并且 $\psi'(i)\in S_2$,其中,$\psi(i)\preceq\psi'(i)$.$GenAQ(S_2)=S_2\cup\{\forall k(k\in[init_i,c,i+c)\Rightarrow\psi'(k))\}$.因此,$GenAQ(S_1)\sqsubseteq GenAQ(S_2)$;

(2) 如果在 x 中条件(2)成立,$GenAQ(S_1)=S_1\cup\{\forall k(k\in[init_i,c,i+c),\psi(k)\}$.$S_1\sqsubseteq S_2$ 推出$\forall k(k\in[init_i,c,i)\Rightarrow\psi_1'(k))$在

S_2 中并且 $\psi'_2(i)\in S_2$,其中,$\psi_1(k)\preceq\psi'_1(k)\wedge\psi_2(i)\preceq\psi'_2(i).\psi(k)=\psi_1(k)\sqcap\psi_2(k)\Rightarrow\psi(k)\preceq\psi_1(k)\wedge\psi(k)\preceq\psi_2(k)\Rightarrow\psi(k)\preceq\psi'_1(k)\wedge\psi(k)\preceq\psi'_2(k)\Rightarrow\psi(k)\preceq\psi'_1(k)\sqcap\psi'_2(k).$令 $\psi'(k)=\psi'_1(k)\sqcap\psi'_2(k).GenAQ(S_2)=S_2\cup\{\forall k(k\in[init_i,c,i+c)\Rightarrow\psi'(k))\}$.因此,$GenAQ(S_1)\sqsubseteq GenAQ(S_2)$;

(3) 如果在 x 中条件(3)和条件(4)成立,证明过程类似条件(1)和条件(2);

(4) 其他情况,$GenAQ(S_1)\sqsubseteq GenAQ(S_2)$显然成立.

综上,$GenAQ(S_1)\sqsubseteq GenAQ(S_2)$. □

推论 6. 如果 n 是循环控制变量初始化语句,$GenSpecial(n)\sqsubseteq GenSpecial(n)$.

推论 7. 令 G 表示 CFG.令 a 表示语句 n 之前的数据流值.如果 $a\in L_G$,那么 $F_n(a)\in L_G$.

定理 2. 令 n 表示 CFG G 的语句.F_n 是单调的.

证明:F_n 是单调的当且仅当$\forall x,y\in L_G:x\sqsubseteq y\Rightarrow F_n(x)\sqsubseteq F_n(y)$.根据 F_n 的定义,只需要证明 F_n 中出现的函数都是单调的.根据引理 5~引理 9 和推论 6、推论 7,F_n 是单调的. □

定理4的证明

引理 10. 令 c_i 表示语句 n 之前的状态,令 a_i 表示语句 n 之前的数据流值.如果 $G\vdash c_i\rightsquigarrow c_{i+1}$ 并且$[\![a_i]\!](c_i)$成立,那么下面的条件成立:

1) $[\![Semantics(n,a_i)]\!](c_{i+1})$成立;

2) 如果 n 是 $e_1[e_2]:=e_3$,那么$[\![Transfer(n,a_i)]\!](c_{i+1})$成立;

3) 如果 n 是循环控制变量初始化语句 $i:=init$,那么$[\![GenSpecial(n)]\!](c_{i+1})$成立;

4) 如果 n 是循环控制变量初始化语句 $i:=i+c$,那么$[\![HandleInterval(n,a_i)]\!](c_{i+1})$成立;

5) 对任何 $S\in 2^{ProperP}$,如果$[\![S]\!](c_{i+1})$成立,那么$[\![GenAQ(S)]\!](c_{i+1})$成立;

6) 对任何 $S\in 2^{ProperP}$,如果$[\![S]\!](c_{i+1})$成立,那么$[\![Recude(S)]\!](c_{i+1})$.

证明:

(1) 要证明结论 1),需要证明:

a) 如果 n 是 $lh:=e$,那么 $[\![lh]\!](c_{i+1})=[\![e]\!](c_i)\wedge\underset{[\![\&lh\neq\&lh_i]\!](c_i)\wedge lh_i\in LH}{\wedge}[\![lh_i]\!](c_{i+1})=[\![lh_i]\!](c_i)$ 成立;

b) 如果 n 是 $cond$,并且:

i c_{i+1} 是 true 分支语句之前的状态,那么$[\![cond]\!](c_{i+1})\wedge\underset{lh_i\in LH}{\wedge}[\![lh_i]\!](c_{i+1})=[\![lh_i]\!](c_i)$;

ii c_{i+1} 是 false 分支语句之前的状态,那么$[\![\neg cond]\!](c_{i+1})\wedge\underset{lh_i\in LH}{\wedge}[\![lh_i]\!](c_{i+1})=[\![lh_i]\!](c_i)$.

根据语句语义的定义,结论 a)、结论 b)成立.

(2) 要证明结论 2),需要证明:

$$[\![\{p|p\in a\wedge p\text{ 是全称量词性质}\wedge\&e_1[e_2]\notin M(p)\}]\!](c_{i+1}).$$

$[\![a_i]\!](c_i)\wedge p\in a_i\Rightarrow[\![p]\!](c_i).(\wedge a_i\Rightarrow\&e_1[e_2]\notin M(p))\wedge[\![a_i]\!](c_i)$推出$[\![\&e_1[e_2]\notin M(p)]\!](c_i)$.因为$[\![\&e_1[e_2]\notin M(p)]\!](c_i),p$ 中所有内存单元的值保持不变,因此$[\![p]\!](c_{i+1})$成立.

(3) 为了证明结论 3),需要证明$[\![\{\forall k(k\in[init,c,i)\Rightarrow\text{false}\}]\!](c_{i+1})$.因为 n 是 $i:=init$ 并且 i 不出现在 $init$ 中,所以$[\![i=init]\!](c_{i+1})$成立.$[\![i=init]\!](c_{i+1})\Rightarrow[\![[init,step,i)=\varnothing]\!](c_{i+1})$,因此$[\![\forall k(k\in[init,c,i)\Rightarrow\text{false}]\!](c_{i+1})$成立.

(4) 为了证明结论 4):

a) 如果$\forall x(x\in[init_i,c,i+c)\Rightarrow p)$在 a_i 中,那么只需要证明$[\![\forall x(x\in[init_i,c,i)\Rightarrow p)]\!](c_{i+1}).(\forall x(x\in[init_i,c,i+c)\Rightarrow p)$在 a_i 中并且$[\![a_i]\!](c_i)$推出$[\![\forall x(x\in[init_i,c,i+c)\Rightarrow p]\!](c_i)$.因为 n 是 $i:=i+c,p$ 不包含 $i,init_i$ 不包含 i,并且$[\![i]\!](c_{i+1})=[\![i+c]\!](c_i)$,所以$[\![\forall x(x\in[init_i,c,i)\Rightarrow p]\!](c_{i+1})$;

b) 如果$(\forall x(x\in[init_i,c,i)\Rightarrow p)$在 a_i 中,那么只需要证明$\forall x(x\in[init_i,c,i-c)\Rightarrow p)(c_{i+1}).\forall x(x\in[init_i,c,i)\Rightarrow p)$在 a_i 中

$\wedge[\![a_i]\!](c_i)$推出$[\![\forall x(x\in[init_i,c,i)\Rightarrow p]\!](c_i)$.因为 n 是 $i:=i+c$,p 不包含 i,$init_i$ 不包含 i,并且$[\![i-c]\!](c_{i+1})=[\![i]\!](c_i)$,所以$[\![\forall x(x\in[init_i,c,i-c)\Rightarrow p)]\!](c_{i+1})$;

c) 如果$(i=init_i)$在 a_i 中,那么只需要证明$[\![i-c=init_i]\!](c_{i+1})$.$(i=init_i)\in a_i\wedge[\![a_i]\!](c_i)$推出$[\![i=init_i]\!](c_i)$.因为 n 是 $i:=i+c$,$init_i$ 不包含 i,并且$[\![i-c]\!](c_{i+1})=[\![i]\!](c_i)$,所以$[\![i-c=init_i]\!](c_{i+1})$;

d) 其他情况,结论 4)显然成立.

综上,结论 4)成立.

(5) 为了证明结论 5):

a) 如果第 4.3.5 节中的条件(1)或者条件(2)成立,那么只需要证明$[\![\{\forall x(x\in[init_i,c,i+c)\Rightarrow\psi(...,e_1[f_1(x)],...)\}]\!](c_{i+1})$成立,其中,$(...,e_1[f_1(x)],...)$不包含 i.令 $\psi(k)$表示 $\psi(...,e_1[f_1(x)],...)$的缩写:

1) 如果条件(1)成立,那么$(\wedge S\Rightarrow(1))\wedge[\![S]\!](c_{i+1})$推出$[\![(1)]\!](c_{i+1})$.$[\![(1)]\!](c_{i+1})\Rightarrow[\![i=init_i\wedge\psi(i)]\!](c_{i+1})$.因为$[\![i=init_i]\!](c_{i+1})$成立,那么$[\![[init_i,c,i+c)]\!](c_{i+1})=[\![\{i\}]\!](c_{i+1})$.因此,$[\![\forall x(x\in[init_i,c,i+c)\Rightarrow\psi(x)]\!](c_{i+1})$成立;

2) 如果条件(2)成立,那么$(\wedge S\Rightarrow(2))\wedge[\![S]\!](c_{i+1})$推出$[\![(2)]\!](c_{i+1})$.因为$[\![(2)]\!](c_{i+1})$,因此$[\![\forall x(x\in[init_i,c,i)\Rightarrow\psi_1(x)\wedge\psi_2(i)]\!](c_{i+1})$成立.因为 $\psi(x)=\psi_1(x)\sqcap\psi_2(x)$,所以 $\psi_2(x)\Rightarrow\psi(x)\wedge\psi_1(x)\Rightarrow\psi(x)$成立.因此,$\forall x(x\in[init_i,c,i)\Rightarrow\psi_1(x))$推出$\forall x(x\in[init_i,c,i)\Rightarrow\psi(x)$.$[\![\forall x(x\in[init_i,c,i)\Rightarrow\psi_1(x)\wedge\psi_2(i)]\!](c_{i+1})$推出$[\![\forall x(x\in[init_i,c,i)\Rightarrow\psi(x)\wedge\psi(i)]\!](c_{i+1})$.因为$[\![[init_i,c,i+c)]\!](c_{i+1})=[\![[init_i,c,i)\cup\{i\}]\!](c_{i+1})$,故$[\![\forall x(x\in[init_i,c,i+c)\Rightarrow\psi(x)]\!](c_{i+1})$成立;

b) 如果第 3.3.6 节中的条件(3)或者条件(4)成立,那么只需要证明$[\![\{\forall x(x\in[init_i,c,i)\Rightarrow\psi(...,e_1[f_1(x)],...)\}]\!](c_{i+1})$成立,其中,$(...,e_1[f_1(x)],...)$不包含 i.它的证明过程类似证明条件(1)和条件(2).

综上,结论 5)成立.

(6) $[\![S]\!](c_{i+1})\Rightarrow[\![Reduce(S)]\!](c_{i+1})$,因此结论 6)成立. □

定理 4. 令 c_i 表示语句 n 之前的状态,a_i 表示语句 n 之前的数据流值.如果 $G\vdash c_i\rightsquigarrow c_{i+1}$ 并且$[\![a_i]\!](c_i)$成立,那么$[\![F_n(a_i)]\!](c_{i+1})$成立.

证明:传播规则正确,所以$[\![Semantics(n,a_i)]\!](c_i\cup c_{i+1})$成立,因此$[\![Propagated(a_i\cup Semantics(n,a_i)]\!](c_{i+1})$成立.根据引理 10 以及 F_n 的定义,$[\![F_n(a_i)]\!](c_{i+1})$成立. □

李彬(1988－),男,河北邯郸人,博士生,主要研究领域为软件工程,程序分析,程序验证.

翟娟(1988－),女,博士,CCF 专业会员,主要研究领域为软件工程,程序分析,程序验证,程序合成.

汤震浩(1989－),男,博士生,主要研究领域为软件工程,程序分析,程序验证.

汤恩义(1982－),男,博士,助理研究员,CCF 专业会员,主要研究领域为软件工程,新型软件测试方法与程序分析方法.

赵建华(1971－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为形式化方法,软件工程,程序设计语言.