Software Verification

(2021/22, Static Program Analysis)

prof. FRANCESCO RANZATO

Homework: choose between options A, B and C.

Exam: Contact by email the teacher for agreeing on a date for an oral exam, possibly via Zoom; please contact me about 10 days in advance of your preferred date. The oral exam on Homework A will consist in solving a couple of exercises selected by the teacher (for a Zoom exam please be prepared to use some tool for sharing your screen where you will dynamically write your solutions, such as a tablet for free hand-writing or some text editor supporting Latex-like extensions such as Atom). The oral exam on Homeworks B and C will consist of a 40 minutes oral presentation with slides.

1 Homework A: Solve all the Exercises

Exercise 1

Let C and A be complete lattices and let (α, C, A, γ) be a Galois connection. Prove the following properties:

1. for all $c \in C$, $\alpha(c) = \wedge_A\{a \in A \mid c \leq_C \gamma(a)\}$
2. for all $S \subseteq C$, $\alpha(\vee_C S) = \vee_A\{\alpha(c) \in A \mid c \in S\}$
3. $\gamma \circ \alpha : C \to C$ is idempotent (i.e., $(\gamma \circ \alpha) \circ (\gamma \circ \alpha) = \gamma \circ \alpha$)

Exercise 2

Let X be any nonempty set and $S \subsetneq X$ be any proper (i.e., $S \neq X$) subset. Define $\alpha : \wp(X) \to \wp(S)$ by $\alpha(Y) \stackrel{\text{def}}{=} Y \cap S$. Prove that there exists $\gamma : \wp(S) \to \wp(X)$ such that $(\alpha, \langle \wp(X), \subseteq\rangle, \langle \wp(S), \subseteq\rangle, \gamma)$ is a Galois insertion.

Exercise 3

Let $\langle C, \leq_C, \wedge_C, \vee_C, \top_C, \bot_C\rangle$ be a complete lattice and let $S \subseteq C$ be a subset of C which is *meet-closed*, that is:

$$\forall Y \subseteq S.\ \wedge_C Y \in S$$

where, as usual, $\wedge_C \varnothing = \top_C$. Prove that the poset $\langle S, \leq_C\rangle$ can be viewed as an abstract domain of C where the concretization map $\gamma : S \to C$ is the identity.

Exercise 4

Let C and A be complete lattices, (α, C, A, γ) be a Galois insertion, $f : C \to C$ be a monotone concrete operation and $f^\sharp : A \to A$ be a monotone abstract operation.

1. Assume that $f \circ \gamma = \gamma \circ f^\sharp$ (in this case, $f^\sharp$ is called a γ-complete approximation of f).
   (a) Prove that $\alpha(\mathrm{gfp}(f)) = \mathrm{gfp}(f^\sharp)$.
   (b) Give a counterexample to the equality $\alpha(\mathrm{lfp}(f)) = \mathrm{lfp}(f^\sharp)$.
2. Assume that $\alpha \circ f = f^\sharp \circ \alpha$ (in this case, $f^\sharp$ is called a α-complete approximation of f)
   (a) Prove that $\alpha(\mathrm{lfp}(f)) = \mathrm{lfp}(f^\sharp)$.
   (b) Give a counterexample to the equality $\mathrm{lfp}(f) = \gamma(\mathrm{lfp}(f^\sharp))$.

Exercise 5

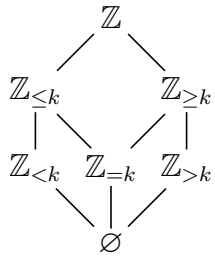
Let C and A be complete lattices, (α, C, A, γ) be a Galois insertion, $f : C \to C$ be a monotone concrete operation. Prove that there exists a monotone abstract operation $f^\sharp : A \to A$ such that $f \circ \gamma = \gamma \circ f^\sharp$ holds if and only if for all $c \in \gamma(A)$, there exists some $a \in A$ such that $f(c) = \gamma(a)$.

Exercise 6

Let C and A be complete lattices, (α, C, A, γ) be a Galois insertion, $f : C \to C$ be a monotone concrete operation. Prove the following:
Fact: there exists a monotone abstract operation $f^\sharp : A \to A$ such that $\alpha \circ f = f^\sharp \circ \alpha$ holds $\Leftrightarrow \gamma \circ \alpha \circ f = \gamma \circ \alpha \circ f \circ \gamma \circ \alpha$.

Exercise 7

Consider the following abstract domain Sign_k of $\langle \wp(\mathbb{Z}, \subseteq) \rangle$ where $k \in \mathbb{Z}$ is any given integer:



Hence, Sign_k is a parametric abstract domain of "signs", where the constant 0 is replaced by a parameter $k \in \mathbb{Z}$. Provide sound definitions for the following abstract transfer functions: $\mathcal{B}^\sharp[\![x \leq k]\!]$, $\mathcal{B}^\sharp[\![x < y]\!]$, $\mathcal{A}^\sharp[\![x * x]\!]$, $\mathcal{A}^\sharp[\![k/x]\!]$, $\mathcal{A}^\sharp[\![x - y]\!]$ which are **as precise as possible**, ideally the best correct approximations.

Exercise 8

Use the web Interproc static analyzer with Octagons and Linear Equalities (a.k.a. Karr domain) on programs that include at least a loop whose body includes at least two nontrivial assignments at different variables. For each abstract domain $A \in \{\text{Octagons}, \text{Linear Equalities}\}$ exhibit two programs P_A and Q_A such that:

- the abstract loop invariants in A computed by Interproc for P_A **are the most precise**, i.e., for each loop invariant program point p of P_A, if $\mathcal{S}[\![P_A]\!](p)$ is the concrete program invariant at the loop invariant program point p then Interproc infers precisely $\alpha_A(\mathcal{S}[\![P_A]\!](p))$.
- the abstract loop invariants in A computed by Interproc for Q_A **are not the most precise**, i.e., for each loop invariant program point p of Q_A, if $\mathcal{S}[\![Q_A]\!](p)$ is the concrete program invariant at the loop invariant program point p then Interproc is able to infer a sound but not precise invariant $a_p \in A$ such that $\alpha_A(\mathcal{S}[\![Q_A]\!](p)) <_A a_p$.
- For the abstract domain Octagons, the concrete loop invariant must be a relation which simultaneously involves **two** different numerical variables (e.g., $x - y \leq 0$), while for the abstract domain Linear Equalities, the concrete loop invariant must be a relation involving **at least three** different numerical variables (e.g., $x + y - z = 0$).

The student should be able to explain why and how Interproc computes the results of these program analyses.

Exercise 9 (Optional, not easy)

Let C and A be complete lattices, (α, C, A, γ) be a Galois insertion, $f : C \to C$ be an **additive** concrete operation (i.e., for all $S \subseteq C$, $f(\vee_C S) = \vee_C f(S)$). Prove that there exists a monotone abstract operation $f^\sharp : A \to A$ such that $\alpha \circ f = f^\sharp \circ \alpha$ holds if and only if for all $a \in A$ there exists some $a' \in A$ such that $\vee_C\{c \in C \mid f(c) \leq \gamma(a)\} = \gamma(a')$.

2 Homework B: Read, understand and then present in 40 minutes a research paper

Some examples are listed below, different research papers can be proposed and discussed with the teacher:

1. Xavier Rival and Laurent Mauborgne. "The trace partitioning abstract domain". ACM Trans. Program. Lang. Syst. 29, 5, Article 26 (August 2007).
2. Vijay D'Silva, Daniel Kroening: "Abstraction of Syntax". VMCAI 2013: 396-413 (2013)
3. Roberto Giacobazzi, Isabella Mastroeni: "Abstract non-interference: parameterizing non-interference by abstract interpretation". POPL 2004: 186-197 (2004)
4. Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, Martin T. Vechev: "AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation". In IEEE Symposium on Security and Privacy 2018: 3-18
5. Singh, G.; Gehr, T.; Puschel, M.; and Vechev, M. 2019. "An abstract domain for certifying neural networks". Proc. ACM Program. Lang. 3(POPL 2019):41:1–41:30.
6. Gagandeep Singh, Markus Püschel, Martin T. Vechev: "Fast polyhedra abstract domain". In ACM POPL 2017: 46-59
7. Section 5.2 "The affine equalities domain (Karr's domain)" of: Antoine Miné: Tutorial on static inference of numeric invariants by abstract interpretation. In Foundations and Trends in Programming Languages (FnTPL), 4(3-4), 120-372, 2017.

3 Homework C (possibly 2 students together)

Design an abstract interpreter for the abstract denotational semantics $\mathcal{D}^\sharp$ of **While**. This means to write a program $\mathcal{AI}$ (in a suitable programming language) such that:

- the abstract interpreter $\mathcal{AI}$ can be instantianted to a numerical abstract domain A which abstracts $\wp(\mathbb{Z})$ and to a state abstract domain $\mathbb{S}$ which abstracts $\wp(\textbf{State})$. This instantantiation is denoted by $\mathcal{AI}_{A,\mathbb{S}}$. As an option, the state abstract domain $\mathbb{S}$ can be automatically derived from A as a nonrelational abstract domain, so that the instantiation is simply denoted by $\mathcal{AI}_A$. One can assume that the abstract domains are complete lattices so that they are defined together with their partial order relations, bottom and top elements, lub's, glb's, widening operations if needed.
- $\mathcal{AI}_{A,\mathbb{S}}$ takes as input any program $P \in$ **While** and abstract state $s^\sharp \in \mathbb{S}$ (for the variables occurring in P).
- $\mathcal{AI}_{A,\mathbb{S}}(P, s^\sharp)$ provides as output $\mathcal{D}^\sharp[\![P]\!]s^\sharp$. As an **optional task**, $\mathcal{AI}_{A,\mathbb{S}}(P, s^\sharp)$ may also provide as output the abstract loop invariants for any loop occurring in P, which are computed by $\mathcal{AI}_{A,\mathbb{S}}(P, s^\sharp)$ in order to compute its output $\mathcal{D}^\sharp[\![P]\!]s^\sharp$ (this is a matter of storing the needed information in $\mathcal{AI}_{A,\mathbb{S}}$).

- To test $\mathcal{AI}$, instantiate $\mathcal{AI}_A$ to a simple numerical abstract domain A (such as extended Signs or, more challenging, Intervals).

For parsing programs $S \in$ **While**, one can use automatic parser generators such as GNU Bison, Yacc and JavaCC or pattern matching in functional languages.