

Axiomatic Program Semantics and Theories of Consistency Enforcement

Sebastian Link

Information Systems, Massey University, Palmerston North, New Zealand

1. Program Semantics based on $\mathcal{L}_{\infty\omega}^\omega$ and \mathcal{L}_{ar}
2. GCS-Consistency Enforcement
3. Effective Computation
4. Summary and Outlook

1.1. Logics, State Concept and Specifications

- $\mathcal{L}_{\infty\omega}^{\omega}$ or \mathcal{L}_{ar} wrt. arithmetic $(\mathbb{N}, 0, S, +, *, =)$
- **state space**: finite set X of variables with associated types $\#x$
- **state**: type-compatible variable assignment $x \mapsto \sigma(x) \in \#x$
- **static invariants**: formulae \mathcal{I} with $fr(\mathcal{I}) \subseteq X$ in one of our logics
- **guarded commands on X** :
 - skip, fail, loop and $\mathbf{x}_{i_1} := \mathbf{t}_{i_1} \parallel \dots \parallel \mathbf{x}_{i_k} := \mathbf{t}_{i_k}$
 - sequences $S;T$, choice $S \square T$, restricted choice $S \boxtimes T$ in $\mathcal{S}(X \cup Y)$, precondition $\mathcal{P} \rightarrow S$, unbounded choice $@y \bullet S$ and least fixed points $\mu S.f(S)$

1.2. Relational Semantics

- $\mathcal{L}_{\infty\omega}^\omega$:
 - $\Sigma = \Sigma(X)$ denotes set of all states on X
 - $\Delta(S) \subseteq \Sigma \times (\Sigma \cup \{\infty\})$ set of state transitions of S
- \mathcal{L}_{ar} :
 - take any pair of formulae $(\Delta(S), \Sigma_0(S))$ with $2k$ and k free variables
 - interpret $\Delta(S)(\mathbf{x}, \mathbf{y})$ by state pairs and $\Sigma_0(S)(\mathbf{x})$ by states
 - (σ, τ) with $\models_{(\sigma, \tau)} \Delta(S)$ is interpreted as an **execution** of S with start state σ and final state τ
 - a state σ satisfying $\Sigma_0(S)$ is considered as a start state for S , in which a **non-terminating execution** of S exists

1.3. Predicate Transformers

- associate with $S \in \mathcal{S}(X)$ two **predicate transformers** $wlp(S)$ and $wp(S)$ —functions from (equivalence classes) of formulae to (equivalence classes) of formulae
- $wlp(S)(\varphi)$ characterizes those initial states σ such that each terminating execution of S starting in σ results in a state τ satisfying φ
- $wp(S)(\varphi)$ characterizes those initial states σ such that each execution of S starting in σ terminates and results in a state τ satisfying φ

1.4. Existence

- $\mathcal{L}_{\infty\omega}^w$: states by formulae, i.e., $\models_{\tau} \varphi_{\sigma}$ iff $\sigma = \tau$
 - $wlp(S)(\varphi) \Leftrightarrow \bigvee_{\sigma \in \Sigma_1} \varphi_{\sigma}$ with
 $\Sigma_1 = \{\sigma \in \Sigma \mid \models_{\tau} \varphi \text{ for all } \tau \in \Sigma \text{ with } (\sigma, \tau) \in \Delta(S)\}$
 - $wp(S)(\varphi) \Leftrightarrow \bigvee_{\sigma \in \Sigma_2} \varphi_{\sigma}$ with
 $\Sigma_2 = \{\sigma \in \Sigma \mid \models_{\tau} \varphi \text{ and } \tau \neq \infty \text{ for all } \tau \in \Sigma \cup \{\infty\} \text{ with } (\sigma, \tau) \in \Delta(S)\}$
- \mathcal{L}_{ar} : S given by relational semantics $(\Delta(S), \Sigma_0(S))$
 - $wlp(S)(\varphi(\mathbf{x})) \Leftrightarrow \forall \mathbf{y}. \Delta(S)(\mathbf{x}, \mathbf{y}) \Rightarrow \varphi(\mathbf{y})$
 - $wp(S)(\varphi(\mathbf{x})) \Leftrightarrow (\forall \mathbf{y}. \Delta(S)(\mathbf{x}, \mathbf{y}) \Rightarrow \varphi(\mathbf{y})) \wedge \neg \Sigma_0(S)(\mathbf{x})$

1.5. Inversion Theorems

- **dual predicate transformers:** $w(l)p(S)^*(\varphi) \Leftrightarrow \neg w(l)p(S)(\neg\varphi)$
- $\mathcal{L}_{\infty\omega}^\omega$:
 - $wp(S)(\varphi) \Leftrightarrow wlp(S)(\varphi) \wedge wp(S)(true)$
 - $wlp(S)(\bigwedge_{i \in I} \varphi_i) \Leftrightarrow \bigwedge_{i \in I} wlp(S)(\varphi_i)$
 - $\Delta(S) = \{(\sigma, \infty) \models_{\sigma} wp(S)^*(false)\} \cup \{(\sigma, \tau) \models_{\sigma} wlp(S)^*(\varphi_{\tau})\}$
- \mathcal{L}_{ar} :
 - $wp(S)(\varphi) \Leftrightarrow wlp(S)(\varphi) \wedge wp(S)(true)$
 - $wlp(S)(\forall \mathbf{y}. Q(\mathbf{y}) \Rightarrow \varphi(\mathbf{x}, \mathbf{y})) \Leftrightarrow \forall \mathbf{y}. Q(\mathbf{y}) \Rightarrow wlp(S)(\varphi(\mathbf{x}, \mathbf{y}))$
 - $\Delta(S)(\mathbf{x}, \mathbf{y}) \Leftrightarrow wlp(S)^*(\mathbf{x} = \mathbf{y})$ and $\Sigma_0(S)(\mathbf{x}) \Leftrightarrow wp(S)^*(false)$

1.6. Dijkstra's Guarded Commands - Semantics

- obtain semantics by assigning predicate transformers and verifying the properties above

$$w(l)p(\textit{skip})(\varphi) \Leftrightarrow \varphi ,$$

$$w(l)p(\textit{fail})(\varphi) \Leftrightarrow \textit{true} ,$$

$$w(l)p(\textit{loop})(\varphi) \Leftrightarrow \textit{false}(\vee \textit{true}) ,$$

$$w(l)p(x_{i_1} := t_{i_1} \parallel \dots \parallel x_{i_k} := t_{i_k})(\varphi) \Leftrightarrow \{x_{i_1}/t_{i_1}, \dots, x_{i_k}/t_{i_k}\}.\varphi ,$$

$$w(l)p(S; T)(\varphi) \Leftrightarrow w(l)p(S)(w(l)p(T)(\varphi)) ,$$

$$w(l)p(\mathcal{P} \rightarrow S)(\varphi) \Leftrightarrow \mathcal{P} \Rightarrow w(l)p(S)(\varphi) ,$$

$$w(l)p(S \square T)(\varphi) \Leftrightarrow w(l)p(S)(\varphi) \wedge w(l)p(T)(\varphi) ,$$

$$w(l)p(S \boxtimes T)(\varphi) \Leftrightarrow w(l)p(S)(\varphi) \wedge \\ (wp(S)^*(\textit{true}) \vee w(l)p(T)(\varphi)) ,$$

$$w(l)p(@y \bullet S)(\varphi) \Leftrightarrow \forall y. w(l)p(S)(\varphi)$$

1.7. Recursive Program Specifications

- so far, straightline non-deterministic partial programs with unrestricted choice are covered
- from the simple recursive definition $S = f(S)$, where $f(S)$ consists of basic operations, constructors and the program variable S , we obtain a mapping f on programs and a **fixpoint equation**
- $\mathcal{L}_{\infty\omega}^{\omega}$ allows to study recursion in very general form, to show existence of least fixpoints $\mu S.f(S)$ on top of \mathcal{L}_{ar} we investigate **simple WHILE-loops** $f(S) = \mathcal{P} \rightarrow T; S \square \neg \mathcal{P} \rightarrow skip$ only
- fundamental is the **Nelson-order** \preceq on program specifications on a state space X , i.e., we have $S \preceq T$ iff

$$\models wlp(T)(\varphi) \Rightarrow wlp(S)(\varphi) \quad \text{and} \quad \models wp(S)(\varphi) \Rightarrow wp(T)(\varphi)$$

holds for each state formula φ on X

1.8. Least Fixpoint in the $\mathcal{L}_{\infty\omega}^\omega$ -Case

- **fixpoint theorem from Knaster-Tarski**: if (A, \leq) poset such that every \leq -chain $K \subseteq A$ has a \leq -l.u.b. and every $\emptyset \neq B \subseteq A$ has a \leq -g.l.b., then every \leq -preserving $f : A \rightarrow A$ has least fixpoint $fix(f) = lub\{f^\alpha(min(A)) \mid \alpha \in Ord\} = glb\{a \mid f(a) \leq a\}$
- it can be shown that
 - \preceq -chains of guarded commands have a l.u.b.
 - non-empty sets of guarded-commands have a \preceq -g.l.b. and
 - guarded-commands are order-preserving with respect to \preceq
- consequently, $\mu S.f(S)$ exists and
 - $wlp(\mu S.f(S))(\varphi) \Leftrightarrow \bigwedge_{\alpha \in Ord} wlp(f^\alpha(loop))(\varphi)$
 - $wp(\mu S.f(S))(\varphi) \Leftrightarrow \bigvee_{\alpha \in Ord} wp(f^\alpha(loop))(\varphi)$

1.9. Gödel Numbering of Terms, Formulae and Guarded Commands in Case of \mathcal{L}_{ar}

$$\begin{aligned}
 h(0) &= 1, & h(x_i) &= 3^i, & h(s(t)) &= 2 \cdot 3^{h(t)}, & h(t_1 + t_2) &= 4 \cdot 3^{h(t_1)} \cdot 5^{h(t_2)}, \\
 h(t_1 * t_2) &= 8 \cdot 3^{h(t_1)} \cdot 5^{h(t_2)}, & h(t_1 = t_2) &= 16 \cdot 3^{h(t_1)} \cdot 5^{h(t_2)}, & h(\neg\varphi) &= 32 \cdot 3^{h(\varphi)}, \\
 h(\varphi_1 \Rightarrow \varphi_2) &= 64 \cdot 3^{h(\varphi_1)} \cdot 5^{h(\varphi_2)} & \text{and} & & h(\forall x_i. \varphi) &= 2^{6+i} \cdot 3^{h(\varphi)}
 \end{aligned}$$

$$\begin{aligned}
 g(\text{fail}) &= 1, & g(\text{loop}) &= 2, & g(\text{skip}) &= 4, \\
 g(x_{i_1} := t_{i_1} \parallel \dots \parallel x_{i_k} := t_{i_k}) &= 8 \cdot \prod_{j=1}^k \text{prim}(i_j)^{h(t_{i_j})}, \\
 g(S_1; S_2) &= 16 \cdot 3^{g(S_1)} \cdot 5^{g(S_2)}, & g(S_1 \square S_2) &= 32 \cdot 3^{g(S_1)} \cdot 5^{g(S_2)}, \\
 g(S_1 \boxtimes S_2) &= 64 \cdot 3^{g(S_1)} \cdot 5^{g(S_2)}, \\
 g(\mathcal{P} \rightarrow S) &= 128 \cdot 3^{h(\mathcal{P})} \cdot 5^{g(S)}, & g(@x_j \bullet S) &= 256 \cdot 3^j \cdot 5^{g(S)} \\
 g(T_j) &= 512 \cdot 3^j & \text{and} & & g(\mu T_j. f(T_j)) &= 1024 \cdot 3^j \cdot 5^{g(f(T_j))}
 \end{aligned}$$

1.10. Least Fixpoints in the \mathcal{L}_{ar} case

- $f(T) = \mathcal{P} \rightarrow S; T \square \neg\mathcal{P} \rightarrow skip: \tau_l(j), \tau(j)$ for every j st
 $\chi_j^1(i, \mathbf{x}) = \tau_l(j)(\varphi(\mathbf{x}))$ and $\chi_j^2(i, \mathbf{x}) = \tau(j)(\varphi(\mathbf{x}))$ for $\varphi(\mathbf{x})$
 $\forall \mathbf{x}. \forall i. (\chi_{h(\varphi)}^1(i, \mathbf{x}) \Leftrightarrow wlp(f^i(loop))(\varphi(\mathbf{x})))$ and
 $\forall \mathbf{x}. \forall i. (\chi_{h(\varphi)}^2(i, \mathbf{x}) \Leftrightarrow wp(f^i(loop))(\varphi(\mathbf{x})))$
- define $\mathcal{S} = \lim_{k \in \mathbb{N}} f^k(loop)$ via $wlp(\mathcal{S})(\varphi(\mathbf{x})) \Leftrightarrow \forall k. \chi_{h(\varphi)}^1(k, \mathbf{x})$ and
 $wlp(\mathcal{S})(\varphi(\mathbf{x})) \Leftrightarrow \exists k. \chi_{h(\varphi)}^2(k, \mathbf{x})$
- $\lim_{i \in \mathbb{N}} f^i(loop)$ is the l.u.b. of \preceq -chain $\{f^i(loop)\}_{i \in \mathbb{N}}$
- f has a least \preceq -fixpoint $\mu T. f(T) = \lim_{i \in \mathbb{N}} f^i(loop)$

2.1. Consistency and Preservation of Effects

- S is called **consistent w.r.t. a static invariant** \mathcal{I} iff it transfers \mathcal{I} -satisfying states into exactly such ones
- this leads to proof obligation $\mathcal{I} \Rightarrow wlp(S)(\mathcal{I})$
- for specifications S on X and T on Y with $X \subseteq Y$, S is specialized by T ($T \sqsubseteq S$) iff $wlp(S)(\varphi) \Rightarrow wlp(T)(\varphi)$ and $wp(S)(\varphi) \Rightarrow wp(T)(\varphi)$ hold for all state formulae φ on X

2.2. Definition of Greatest Consistent Specializations

- Let \mathcal{I} be a static invariant on X and S be a specification on $Y \subseteq X$. A specification $S_{\mathcal{I}}$ is called **greatest consistent specialization** (GCS) of S w.r.t. \mathcal{I} iff the following properties hold
 - (i) $S_{\mathcal{I}} \sqsubseteq S$,
 - (ii) $S_{\mathcal{I}}$ is consistent w.r.t. \mathcal{I} and
 - (iii) \mathcal{I} -consistent specifications T on X with $T \sqsubseteq S$ satisfy $T \sqsubseteq S_{\mathcal{I}}$.

2.3. Fundamental Properties concerning GCSs

- GCS $S_{\mathcal{I}}$ for \mathcal{I} on X and S on $Y \subseteq X$ have always the form

$$(\mathcal{I} \rightarrow S; @\xi \bullet \mathbf{z} := \xi; \mathcal{I} \rightarrow skip) \boxtimes (\neg\mathcal{I} \rightarrow S; @\xi \bullet \mathbf{z} := \xi) \quad ,$$

where \mathbf{z} is an abbreviation for sequence of state variables in $X - Y$ and ξ an abbreviation for a disjoint copy of these

- GCSs **always exist** and are **uniquely determined** up to equivalence
- $\mathcal{I}_1 \wedge \mathcal{I}_2 \rightarrow S_{\mathcal{I}_1 \wedge \mathcal{I}_2} \equiv \mathcal{I}_1 \wedge \mathcal{I}_2 \rightarrow (S_{\mathcal{I}_1})_{\mathcal{I}_2}$
- for $T \sqsubseteq S$ we always receive $T_{\mathcal{I}} \sqsubseteq S_{\mathcal{I}}$

2.4. The Upper Bound Theorem

- \mathcal{I} -reducedness is a property of sequences $S_1; S_2$ which does not allow interim states causing wrongly a consistency enforcement in a branch but is irrelevant for the entire specification
- given a static invariant \mathcal{I} and an \mathcal{I} -reduced complex program specification S , we obtain a new program specification $S'_{\mathcal{I}}$ by replacing all involved basic operations in S by their respective GCSs
- the **upper bound theorem** guarantees that $S_{\mathcal{I}} \sqsubseteq S'_{\mathcal{I}}$ holds
- proof is done by structural induction on S using the constructors of guarded commands, where the tricky cases are the ones for sequences and recursion

2.5. General Form

- idea is now to cut out from $S'_{\mathcal{I}}$ those executions that are not allowed to occur in a specialization of S
- Let \mathcal{I} , S and $S'_{\mathcal{I}}$ be as in the upper bound theorem. Let Z be a disjoint copy of the state space Y . With the formulae

$$\mathcal{P}(S, \mathcal{I}, \mathbf{x}') \equiv \{\mathbf{z}/\mathbf{y}\}.wlp(S''_{\mathcal{I}}; \mathbf{z} = \mathbf{x}' \rightarrow skip)(wlp(S)^*(\mathbf{z} = \mathbf{y})) \quad ,$$

where $S''_{\mathcal{I}}$ results from $S'_{\mathcal{I}}$ by renaming the Y to Z , the GCS $S_{\mathcal{I}}$ is semantically equivalent to

$$@\mathbf{x}' \bullet \mathcal{P}(S, \mathcal{I}, \mathbf{x}') \rightarrow S'_{\mathcal{I}}; \mathbf{y} = \mathbf{x}' \rightarrow skip \quad .$$

- taking the form claimed in the theorem as a definition and verifying the conditions in the definition of the GCS gives the proof

3.1. Investigation of Computability

- $(S, \mathcal{I}) \mapsto S'_{\mathcal{I}}$ is computable as building $S'_{\mathcal{I}}$ is syntactical replacement
- considering the precondition $\mathcal{P}(S, \mathcal{I}, \mathbf{x}')$ for arbitrary \mathbf{x}' is sufficient
- building involved predicate transformers $wlp(S)$ and $wlp(S'_{\mathcal{I}})$ is done by syntactic replacement operations according to definition of axiomatic semantics for commands
- by application of our Gödel numbering h for recursion-free S , the mapping $(S, \mathcal{I}, \mathbf{x}') \mapsto \mathcal{P}(S, \mathcal{I}, \mathbf{x}')$ —and hence $(S, \mathcal{I}) \mapsto S_{\mathcal{I}}$, too—is computable

3.2. The Computability Result

- occurrence of a loop in S means that a loop also occurs within $S'_{\mathcal{I}}$
- limit operator must be used for determining $wlp(S)$ and $wlp(S'_{\mathcal{I}})$, which means $wlp(f^i(loop))$ must be build for all $i \in \mathbb{N}$
- only possible for case of bounded loop, i.e., if $wlp(f^n(loop)) = wlp(f^m(loop))$ holds for all $m \geq n$ and some $n \in \mathbb{N}$
- If recursive guarded commands are restricted to bounded loops, then GCSs are computable, i.e., the function $(S, \mathcal{I}) \mapsto S'_{\mathcal{I}}$ is computable. In general, however, the GCS cannot be computed.

3.3. Effective Computation

- upper bound theorem shows $(\mathcal{P} \rightarrow S)_{\mathcal{I}} = \mathcal{P} \rightarrow S_{\mathcal{I}}$, $(S_1 \square S_2)_{\mathcal{I}} = (S_1)_{\mathcal{I}} \square (S_2)_{\mathcal{I}}$ and $(@y \bullet S)_{\mathcal{I}} = @y \bullet S_{\mathcal{I}}$
- $(S_1)_{\mathcal{I}}; (S_2)_{\mathcal{I}}$ is not necessarily specialization of $S_1; S_2$ since $wlp(S_2)(\varphi)$ is not a state formula of underlying S_1 -state space
- S built of basic commands, choices, guards with decidable preconditions and sequences: if φ is decidable, then $wlp(S)(\varphi)$ and $wlp(S)^*(\varphi)$ are decidable as well
- each S whose occurrences of loops are all bounded can be written in the form $@x_1 \bullet \dots \bullet @x_n \bullet S'$ such that S' does not contain $@$
- S such that every loop is bounded and all preconditions are decidable, \mathcal{I} decidable static constraint: $S_{\mathcal{I}}$ can be computed in form $S_{\mathcal{I}} = @y_1 \bullet \dots \bullet @y_n \bullet T_{\mathcal{I}}$ with all preconditions $\mathcal{P}(T', \mathcal{I}, \mathbf{x})$ being decidable

4.1. Summary

- the theory provides
 - Commutativity
 - Compositionality
 - Effective Computation for a large class of specifications

4.2. Questions

- axiomatic program semantics on top of **linear logic**
- relationship of predicate transformers and **modal logic**
- **weakened approaches** to consistency enforcement together with a discussion of computability and decidability
- GCSs for basic commands and various classes of relational constraints