Axiomatic Program Semantics and Theories of Consistency Enforcement

Sebastian Link

Information Systems, Massey University, Palmerston North, New Zealand

1. Program Semantics based on $\mathcal{L}_\omega$ and $\mathcal{L}_{ar}$

2. GCS-Consistency Enforcement

3. Effective Computation

4. Summary and Outlook
1.1. Logics, State Concept and Specifications

- $\mathcal{L}_\omega^\omega$ or $\mathcal{L}_{ar}$ wrt. arithmetic ($\mathbb{N}, 0, S, +, \times, =$)

- **state space**: finite set $X$ of variables with associated types $\mathbb{X}x$

- **state**: type-compatible variable assignment $x \mapsto \sigma(x) \in \mathbb{X}x$

- **static invariants**: formulae $\mathcal{I}$ with $fr(\mathcal{I}) \subseteq X$ in one of our logics

- **guarded commands on $X$**:
  - skip, fail, loop and $x_{i_1} := t_{i_1} \parallel \ldots \parallel x_{i_k} := t_{i_k}$
  
  - sequences $S; T$, choice $S \sqcup T$, restricted choice $S \boxtimes T$ in $S(X \cup Y)$, precondition $\mathcal{P} \rightarrow S$, unbounded choice $\@y \bullet S$ and least fixed points $\mu S.f(S)$
1.2. Relational Semantics

- \( \mathcal{L}_\omega^{\omega} \):
  - \( \Sigma = \Sigma(X) \) 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 \( \Delta(S)(x, y) \) by state pairs and \( \Sigma_0(S)(x) \) by states
  - \( (\sigma, \tau) \) with \( \models_{(\sigma, \tau)} \Delta(S) \) is interpreted as an execution of \( S \) with
    start state \( \sigma \) and final state \( \tau \)
  - a state \( \sigma \) 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 $S \in S(X)$ 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 $\sigma$ such that each terminating execution of $S$ starting in $\sigma$ results in a state $\tau$ satisfying $\varphi$

- $wp(S)(\varphi)$ characterizes those initial states $\sigma$ such that each execution of $S$ starting in $\sigma$ terminates and results in a state $\tau$ satisfying $\varphi$
1.4. Existence

- $\mathcal{L}_\omega^\omega$: states by formulae, i.e., $\models_\tau \varphi_\sigma$ iff $\sigma = \tau$
  
  - $wlp(S)(\varphi) \iff \bigvee_{\sigma \in \Sigma_1} \varphi_\sigma$ with
    
    $\Sigma_1 = \{ \sigma \in \Sigma \mid \models_\tau \varphi \text{ for all } \tau \in \Sigma \text{ with } (\sigma, \tau) \in \Delta(S') \}$
  
  - $wp(S)(\varphi) \iff \bigvee_{\sigma \in \Sigma_2} \varphi_\sigma$ 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 $(\Delta(S), \Sigma_0(S))$
  
  - $wlp(S)(\varphi(x)) \iff \forall y. \Delta(S)(x, y) \Rightarrow \varphi(y)$
  
  - $wp(S)(\varphi(x)) \iff (\forall y. \Delta(S)(x, y) \Rightarrow \varphi(y)) \land \neg \Sigma_0(S)(x)$
1.5. Inversion Theorems

- **dual predicate transformers**: \( w(l)p(S)^*(\varphi) \iff \neg w(l)p(S)(\neg \varphi) \)

- **\( L^\omega_\omega \)**:
  - \( wp(S)(\varphi) \iff wlp(S)(\varphi) \land wp(S)(true) \)
  - \( wlp(S)(\bigwedge \varphi_i) \iff \bigwedge wlp(S)(\varphi_i) \quad \forall i \in I \)
  - \( \Delta(S) = \{ (\sigma, \infty) \models_\sigma wp(S)^*(false) \} \cup \{ (\sigma, \tau) \models_\sigma wlp(S)^*(\varphi_\tau) \} \)

- **\( L_{ar} \)**:
  - \( wp(S)(\varphi) \iff wlp(S)(\varphi) \land wp(S)(true) \)
  - \( wlp(S)(\forall y.Q(y) \Rightarrow \varphi(x, y)) \iff \forall y.Q(y) \Rightarrow wlp(S)(\varphi(x, y)) \)
  - \( \Delta(S)(x, y) \iff wlp(S)^*(x = y) \) and \( \Sigma_0(S)(x) \iff wp(S)^*(false) \)
1.6. Dijkstra’s Guarded Commands - Semantics

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

\[ w(l)p(skip)(\varphi) \iff \varphi , \]
\[ w(l)p(fail)(\varphi) \iff true , \]
\[ w(l)p(loop)(\varphi) \iff false(\neg true) , \]
\[ w(l)p(x_{i_1} := t_{i_1} \parallel \ldots \parallel x_{i_k} := t_{i_k})(\varphi) \iff \{x_{i_1}/t_{i_1}, \ldots, x_{i_k}/t_{i_k}\}.\varphi , \]
\[ w(l)p(S;T)(\varphi) \iff w(l)p(S)(w(l)p(T)(\varphi)) , \]
\[ w(l)p(P \rightarrow S)(\varphi) \iff P \Rightarrow w(l)p(S)(\varphi) , \]
\[ w(l)p(S \square T)(\varphi) \iff w(l)p(S)(\varphi) \land w(l)p(T)(\varphi) , \]
\[ w(l)p(S \square T)(\varphi) \iff w(l)p(S)(\varphi) \land (wp(S)^*\neg true) \lor w(l)p(T)(\varphi) , \]
\[ w(l)p(@y \bullet S)(\varphi) \iff \forall y.w(l)p(S)(\varphi) \]
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

- $\mathcal{L}_\omega^\omega$ 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 $f(S) = \mathcal{P} \rightarrow T; S\square\neg\mathcal{P} \rightarrow \text{skip}$ 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 $\varphi$ on $X$
1.8. Least Fixpoint in the $L^\omega_\infty$-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 $fix(f) = \text{lub}\{f^\alpha(\text{min}(A)) \mid \alpha \in \text{Ord}\} = \text{glb}\{a \mid f(a) \leq a\}$

- it can be shown that
  - $\leq$-chains of guarded commands have a l.u.b.
  - non-empty sets of guarded-commands have a $\leq$-g.l.b. and
  - guarded-commands are order-preserving with respect to $\leq$

- consequently, $\mu S.f(S)$ exists and
  - $wp(\mu S.f(S))(\varphi) \Leftrightarrow \bigvee_{\alpha \in \text{Ord}} wp(f^\alpha(\text{loop}))(\varphi)$
  - $wp(\mu S.f(S))(\varphi) \Leftrightarrow \bigvee_{\alpha \in \text{Ord}} wp(f^\alpha(\text{loop}))(\varphi)$
1.9. Gödel Numbering of Terms, Formulae and Guarded Commands in Case of $\mathcal{L}_{ar}$

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

\[ g(\text{fail}) = 1, \quad g(\text{loop}) = 2, \quad g(\text{skip}) = 4, \]
\[ g(x_{i_1} := t_{i_1} \| \ldots \| x_{i_k} := t_{i_k}) = 8 \cdot \prod_{j=1}^{k} \text{prim}(i_j)^{h(t_{i_j})}, \]
\[ g(S_1; S_2) = 16 \cdot 3^{g(S_1)} \cdot 5^{g(S_2)}, \quad g(S_1 \Box S_2) = 32 \cdot 3^{g(S_1)} \cdot 5^{g(S_2)}, \]
\[ g(S_1 \boxtimes S_2) = 64 \cdot 3^{g(S_1)} \cdot 5^{g(S_2)}, \]
\[ g(\mathcal{P} \rightarrow S) = 128 \cdot 3^{h(\mathcal{P})} \cdot 5^{g(S)}, \quad g(\exists x_j \in S) = 256 \cdot 3^j \cdot 5^{g(S)} \]
\[ g(T_j) = 512 \cdot 3^j \quad \text{and} \quad g(\mu T_j.f(T_j)) = 1024 \cdot 3^j \cdot 5^{g(f(T_j))} \]
1.10. Least Fixpoints in the $L_{ar}$ case

- $f(T) = \mathcal{P} \rightarrow S; T \square \neg \mathcal{P} \rightarrow \text{skip}$: $\tau_l(j), \tau(j)$ for every $j$ st
  \[
  \chi_j^1(i, x) = \tau_l(j)(\varphi(x)) \text{ and } \chi_j^2(i, x) = \tau(j)(\varphi(x)) \text{ for } \varphi(x)
  \]
  \[\forall x. \forall i. (\chi_j^1(i, x) \iff \text{wp}(f^i(\text{loop}))(\varphi(x))) \text{ and } \forall x. \forall i. (\chi_j^2(i, x) \iff \text{wp}(f^i(\text{loop}))(\varphi(x)))\]

- define $S = \lim_{k \in \mathbb{N}} f^k(\text{loop})$ via $\text{wp}(S)(\varphi(x)) \iff \forall k. \chi_{h(\varphi)}^1(k, x)$ and $\text{wp}(S)(\varphi(x)) \iff \exists k. \chi_{h(\varphi)}^2(k, x)$

- $\lim_{i \in \mathbb{N}} f^i(\text{loop})$ is the l.u.b. of $\leq$ -chain $\{f^i(\text{loop})\}_{i \in \mathbb{N}}$

- $f$ has a least $\leq$-fixpoint $\mu T.f(T) = \lim_{i \in \mathbb{N}} f^i(\text{loop})$
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 $\mathcal{I} \Rightarrow \text{wp}(S)(\mathcal{I})$.

- For specifications $S$ on $X$ and $T$ on $Y$ with $X \subseteq Y$, $S$ is specialized by $T$ $(T \sqsubseteq S)$ iff $\text{wp}(S)(\varphi) \Rightarrow \text{wp}(T)(\varphi)$ and $\text{wp}(S)(\varphi) \Rightarrow \text{wp}(T)(\varphi)$ hold for all state formulae $\varphi$ 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
  \[(\mathcal{I} \rightarrow S; \@\xi \bullet z := \xi; \mathcal{I} \rightarrow \text{skip}) \boxtimes (\neg \mathcal{I} \rightarrow S; \@\xi \bullet z := \xi)\],
  where $z$ is an abbreviation for sequence of state variables in $X - Y$ and $\xi$ 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 $T \subseteq S$ we always receive $T_\mathcal{I} \subseteq S_\mathcal{I}$
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 $S_{\mathcal{I}} \subseteq S'_{\mathcal{I}}$ 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 $S'_I$ those executions that are not allowed to occur in a specialization of $S$

- Let $I$, $S$ and $S'_I$ be as in the upper bound theorem. Let $Z$ be a disjoint copy of the state space $Y$. With the formulae

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

where $S''_I$ results from $S'_I$ by renaming the $Y$ to $Z$, the GCS $S'_I$ is semantically equivalent to

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

- 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

- $(S, I) \mapsto S'_{I}$ is computable as building $S'_{I}$ is syntactical replacement

- considering the precondition $\mathcal{P}(S, I, x')$ for arbitrary $x'$ is sufficient

- building involved predicate transformers $wlp(S)$ and $wlp(S'_{I})$ 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 $(S, I, x') \mapsto \mathcal{P}(S, I, x')$—and hence $(S, I) \mapsto S'_{I}$, too—is computable
3.2. The Computability Result

- Occurrence of a loop in $S$ means that a loop also occurs within $S'_{I}$

- Limit operator must be used for determining $wlp(S)$ and $wlp(S'_{I})$, 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 $(S, I) \mapsto S_I$ is computable. In general, however, the GCS cannot be computed.
3.3. Effective Computation

- upper bound theorem shows \((P \rightarrow S)_I = P \rightarrow S_I\), \((S_1 \square S_2)_I = (S_1)_I \square (S_2)_I\) and \((@y \cdot S')_I = @y \cdot S_I\)

- \((S_1)_I; (S_2)_I\) 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 \(\varphi\) 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 \(@x_1 \cdot \ldots @x_n \cdot S'\) such that \(S'\) does not contain @

- \(S\) such that every loop is bounded and all preconditions are decidable, \(I\) decidable static constraint: \(S_I\) can be computed in form \(S_I = @y_1 \cdot \ldots @y_n \cdot T_I\) with all preconditions \(P(T', I, x)\) 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