Semantics and Verification

Lecture 7

12 March 2010

Overview

Last lecture:

- Tarski's fixed-point theorem

This lecture:

- Bisimilarity as a fixed point
- Hennessy-Milner logic with recursion

Next lecture:

- Mini project!

Hennessy-Milner logic with recursion

1. Strong bisimilarity as a greatest fixed point
2. Hennessy-Milner logic with recursion
3. Game characterization of HML with recursion
4. Advanced recursive formulae
5. Characteristic property

Tarski's Fixed Point Theorem

Let $(D, \sqsubseteq)$ be a complete lattice and let $f : D \to D$ be a monotonic function.

Tarski's Fixed Point Theorem

Then f has a unique largest fixed point z_{max} and a unique least fixed point z_{min} given by:

$$z_{max} \stackrel{\text{def}}{=} \sqcup\{x \in D \mid x \sqsubseteq f(x)\}$$

$$z_{min} \stackrel{\text{def}}{=} \sqcap\{x \in D \mid f(x) \sqsubseteq x\}$$

Computing Fixed Points in Finite Lattices

If D is a finite set then there exist integers $M, m > 0$ such that

- $z_{max} = f^M(\top)$
- $z_{min} = f^m(\bot)$

Recall: Strong Bisimilarity

Let $(Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ be an LTS.

Strong Bisimulation

A binary relation $R \subseteq Proc \times Proc$ is a strong bisimulation iff whenever $(s, t) \in R$ then for each $a \in Act$:

- if $s \xrightarrow{a} s'$ then $t \xrightarrow{a} t'$ for some t' such that $(s', t') \in R$
- if $t \xrightarrow{a} t'$ then $s \xrightarrow{a} s'$ for some s' such that $(s', t') \in R$.

Two processes $p, q \in Proc$ are strongly bisimilar ($p \sim q$) iff there exists a strong bisimulation R such that $(p, q) \in R$.

$$\sim \; = \bigcup \{R \mid R \text{ is a strong bisimulation}\}$$

Strong Bisimilarity as a Greatest Fixed Point

Function $\mathcal{F} : 2^{(Proc \times Proc)} \to 2^{(Proc \times Proc)}$

Let $S \subseteq Proc \times Proc$. Then we define $\mathcal{F}(S)$ as follows:

$(s, t) \in \mathcal{F}(S)$ if and only if for each $a \in Act$:

- if $s \xrightarrow{a} s'$ then $t \xrightarrow{a} t'$ for some t' such that $(s', t') \in S$
- if $t \xrightarrow{a} t'$ then $s \xrightarrow{a} s'$ for some s' such that $(s', t') \in S$.

Strong Bisimilarity as a Greatest Fixed Point

Function $\mathcal{F} : 2^{(Proc \times Proc)} \to 2^{(Proc \times Proc)}$

Let $S \subseteq Proc \times Proc$. Then we define $\mathcal{F}(S)$ as follows:

$(s, t) \in \mathcal{F}(S)$ if and only if for each $a \in Act$:

- if $s \xrightarrow{a} s'$ then $t \xrightarrow{a} t'$ for some t' such that $(s', t') \in S$
- if $t \xrightarrow{a} t'$ then $s \xrightarrow{a} s'$ for some s' such that $(s', t') \in S$.

Observations

- $(2^{(Proc \times Proc)}, \subseteq)$ is a complete lattice and $\mathcal{F}$ is monotonic
- S is a strong bisimulation if and only if $S \subseteq \mathcal{F}(S)$

Strong Bisimilarity is the Greatest Fixed Point of $\mathcal{F}$

$$\sim \; = \bigcup \{ S \in 2^{(Proc \times Proc)} \mid S \subseteq \mathcal{F}(S) \}$$

HML with One Recursively Defined Variable

Syntax of Formulae

Formulae are given by the following abstract syntax

$$F ::= X \mid tt \mid ff \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle a \rangle F \mid [a]F$$

where $a \in Act$ and X is a variable with a definition

- $X \overset{\min}{=} F_X$, or $X \overset{\max}{=} F_X$

such that F_X is a formula of the logic which can contain X.

HML with One Recursively Defined Variable

Syntax of Formulae

Formulae are given by the following abstract syntax

$$F ::= X \mid tt \mid ff \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle a \rangle F \mid [a]F$$

where $a \in Act$ and X is a variable with a definition

- $X \stackrel{\min}{=} F_X$, or $X \stackrel{\max}{=} F_X$

such that F_X is a formula of the logic which can contain X.

How to Define Semantics?

For every formula F we define a function $O_F : 2^{Proc} \to 2^{Proc}$ s.t.

- if S is the set of processes that satisfy X then
- $O_F(S)$ is the set of processes that satisfy F.

Definition of $O_F : 2^{Proc} \to 2^{Proc}$ (let $S \subseteq Proc$)

$$
\begin{aligned}
O_X(S) &= S \\
O_{tt}(S) &= Proc \\
O_{ff}(S) &= \emptyset \\
O_{F_1 \wedge F_2}(S) &= O_{F_1}(S) \cap O_{F_2}(S) \\
O_{F_1 \vee F_2}(S) &= O_{F_1}(S) \cup O_{F_2}(S) \\
O_{\langle a \rangle F}(S) &= \langle \cdot a \cdot \rangle O_F(S) \\
O_{[a]F}(S) &= [\cdot a \cdot] O_F(S)
\end{aligned}
$$

O_F is monotonic for every formula F

$$S_1 \subseteq S_2 \Rightarrow O_F(S_1) \subseteq O_F(S_2)$$

Proof: easy (structural induction on the structure of F).

Semantics

Observation

We know that $(2^{Proc}, \subseteq)$ is a complete lattice and O_F is monotonic, so O_F has unique greatest and least fixed points.

Semantics of the Variable X

- If $X \stackrel{\max}{=} F_X$ then

$$[\![X]\!] = \bigcup \{S \subseteq Proc \mid S \subseteq O_{F_X}(S)\}.$$

- If $X \stackrel{\min}{=} F_X$ then

$$[\![X]\!] = \bigcap \{S \subseteq Proc \mid O_{F_X}(S) \subseteq S\}.$$

Examples

- $\mathit{Inv}(F)$: $X \overset{\max}{=} F \wedge [\mathit{Act}]X$
- $\mathit{Pos}(F)$: $X \overset{\min}{=} F \vee \langle \mathit{Act} \rangle X$

- $\mathit{Safe}(F)$: $X \overset{\max}{=} F \wedge ([\mathit{Act}]\mathit{ff} \vee \langle \mathit{Act} \rangle X)$
- $\mathit{Even}(F)$: $X \overset{\min}{=} F \vee (\langle \mathit{Act} \rangle \mathit{tt} \wedge [\mathit{Act}]X)$

- $F \,\mathcal{U}^w\, G$: $X \overset{\max}{=} G \vee (F \wedge [\mathit{Act}]X)$
- $F \,\mathcal{U}^s\, G$: $X \overset{\min}{=} G \vee (F \wedge \langle \mathit{Act} \rangle \mathit{tt} \wedge [\mathit{Act}]X)$

Game Characterization of HML with Recursion

Intuition: the attacker claims $s \not\models F$, the defender claims $s \models F$.

Configurations of the game are of the form (s, F)

- (s, tt) and (s, ff) have no successors
- (s, X) has one successor (s, F_X)
- $(s, F_1 \wedge F_2)$ has two successors (s, F_1) and (s, F_2) (selected by the attacker)
- $(s, F_1 \vee F_2)$ has two successors (s, F_1) and (s, F_2) (selected by the defender)
- $(s, [a]F)$ has successors (s', F) for every s' s.t. $s \xrightarrow{a} s'$ (selected by the attacker)
- $(s, \langle a \rangle F)$ has successors (s', F) for every s' s.t. $s \xrightarrow{a} s'$ (selected by the defender)

Who is the Winner?

The play is a maximal sequence of configurations formed according to the rules given on the previous slide.

Finite Play

- The attacker is the winner of a finite play if the defender gets stuck or the players reach a configuration (s', ff).
- The defender is the winner of a finite play if the attacker gets stuck or the players reach a configuration (s', tt).

Infinite Play

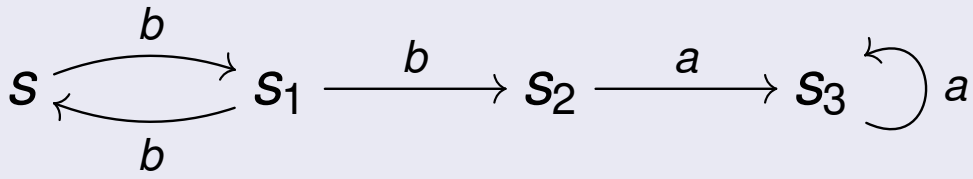
- If X is defined by $X \stackrel{\min}{=} F_X$, then the attacker wins an infinite play.
- If X is defined by $X \stackrel{\max}{=} F_X$, then the defender wins an infinite play.

Game Characterization

Theorem

- $s \models F$ if and only if the defender has a universal winning strategy from (s, F).
- $s \not\models F$ if and only if the attacker has a universal winning strategy from (s, F).

Examples



- $s \models [b](\langle b\rangle [b] ff \wedge \langle b\rangle [a] ff)$
- $s \models X$ with $X \stackrel{\min}{=} \langle a\rangle tt \vee \langle b\rangle X$
- $s \models X$ with $X \stackrel{\max}{=} \langle a\rangle tt \vee [a] X$

Examples of More Advanced Recursive Formulae

Nested Definitions of Recursive Variables

$$X \overset{\min}{=} Y \vee \langle Act \rangle X \qquad\qquad Y \overset{\max}{=} \langle a \rangle tt \wedge \langle Act \rangle Y$$

Solution: compute first $[\![Y]\!]$ and then $[\![X]\!]$.

Examples of More Advanced Recursive Formulae

Nested Definitions of Recursive Variables

$$X \overset{\min}{=} Y \vee \langle Act \rangle X \qquad\qquad Y \overset{\max}{=} \langle a \rangle tt \wedge \langle Act \rangle Y$$

Solution: compute first $[\![Y]\!]$ and then $[\![X]\!]$.

Mutually Recursive Definitions

$$X \overset{\max}{=} [a] Y \qquad\qquad Y \overset{\max}{=} \langle a \rangle X$$

Solution: consider a complete lattice $(2^{Proc} \times 2^{Proc}, \sqsubseteq)$ where $(S_1, S_2) \sqsubseteq (S_1', S_2')$ iff $S_1 \subseteq S_1'$ and $S_2 \subseteq S_2'$.

Characteristic Property

Theorem (Characteristic Property for Finite-State Processes)

For any process p in a finite LTS there exists a HML-Rec formula X_p such that for all processes q: $q \sim p \iff q \models X_p$.

- One says that HML-Rec is expressive with respect to strong bisimilarity for finite LTS.
- Adequacy: $\forall p, q : (p \sim q \Leftrightarrow \forall F : (p \models F \Leftrightarrow q \models F))$
- Expressivity: $\forall p : \exists X_p : \forall q : (q \sim p \Leftrightarrow q \models X_p)$

Here's the formula:

$$X_p \overset{\max}{=} \bigwedge_{p \xrightarrow{a} p'} \langle a \rangle X_{p'} \quad \wedge \quad \bigwedge_{a} [a] \Big(\bigvee_{p \xrightarrow{a} p'} X_{p'} \Big)$$