Lecture 7

- bisimulation as a fixed point
- Hennessy-Milner logic with recursively defined variables
- game semantics and temporal properties of reactive systems

Tarski's Fixed Point Theorem – Summary

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 greatest 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

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

(Recalling of) Definition of Strong Bisimulation

Let $(Proc, Act, (\xrightarrow{a})_{a \in Act})$ 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 Bisimulation as a Greatest Fixed Point

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

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

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

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

Observations

- $\mathcal{F}$ 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 distinguished 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 $O_F : 2^{Proc} \rightarrow 2^{Proc}$ 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 $O_F : 2^{Proc} \rightarrow 2^{Proc}$ (let $S \subseteq Proc$)

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

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

O_F is monotonic on $(2^{Proc}, \subseteq)$, so O_F has the (unique) greatest fixed point and the (unique) least fixed point.

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

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

Game Characterization

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?

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

Infinite Play

- The attacker is the winner of an infinite play if X is defined as $X \stackrel{\min}{=} F_X$.
- The defender is the winner of an infinite play if X is defined as $X \stackrel{\max}{=} F_X$.

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)

Selection of Temporal Properties

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

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

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

Using until we can express e.g. $Inv(F)$ and $Even(F)$:

$$Inv(F) \equiv F\ \mathcal{U}^w\ \mathit{ff} \qquad\qquad Even(F) \equiv \mathit{tt}\ \mathcal{U}^s\ F$$

Examples of More Advanced Recursive Formulae

Nested Definitions of Recursive Variables

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

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

Mutually Recursive Definitions

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

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 $S_2 \subseteq S_2'$.

Theorem (Characteristic Property for Finite-State Processes)

Let s be a process with finitely many reachable states. There exists a property X_s s.t. for all processes t: $s \sim t$ if and only if $t \in [\![X_s]\!]$.