Concurrency Theory

Winter 2025/26

Lecture 10: Hennessy-Milner Logic with Recursion

Thomas Noll, Peter Thiemann
Programming Languages Group
University of Freiburg

https://proglang.github.io/teaching/25ws/ct.html

Thomas Noll, Peter Thiemann

Winter 2025/26

Outline of Lecture 10

1. Recap: Hennessy-Milner Logic
2. Adding Recursion to HML
3. HML with One Recursive Variable
4. Re-Applying Fixed-Point Theory

Semantics of HML

Definition (Semantics of HML)

Let $(S, Act, \longrightarrow)$ be an LTS and $F \in HMF$.
The set of processes in S that satisfy F, $[\![F]\!] \subseteq S$, is defined by:

$$
\begin{aligned}
[\![\mathrm{tt}]\!] &:= S & [\![\mathrm{ff}]\!] &:= \emptyset \\
[\![F_1 \wedge F_2]\!] &:= [\![F_1]\!] \cap [\![F_2]\!] & [\![F_1 \vee F_2]\!] &:= [\![F_1]\!] \cup [\![F_2]\!] \\
[\![\langle\alpha\rangle F]\!] &:= \langle\cdot\alpha\cdot\rangle([\![F]\!]) & [\![[\alpha]F]\!] &:= [\cdot\alpha\cdot]([\![F]\!])
\end{aligned}
$$

where $\langle\cdot\alpha\cdot\rangle, [\cdot\alpha\cdot] : 2^S \to 2^S$ are given by

$$
\begin{aligned}
\langle\cdot\alpha\cdot\rangle(T) &:= \{s \in S \mid \exists s' \in T : s \xrightarrow{\alpha} s'\} \\
[\cdot\alpha\cdot](T) &:= \{s \in S \mid \forall s' \in S : s \xrightarrow{\alpha} s' \Rightarrow s' \in T\}
\end{aligned}
$$

We write $s \models F$ iff $s \in [\![F]\!]$. Two HML formulae are equivalent (written $F \equiv G$) iff they are satisfied by the same processes in every LTS.

Observation: Negation is *not* one of the HML constructs.
Reason: HML is closed under complement.

Lemma

For every $F \in HMF$ *there exists* $F^c \in HMF$ *such that* $[\![F^c]\!] = S \setminus [\![F]\!]$ *for every LTS* $(S, Act, \longrightarrow)$.

Proof.

Definition of F^c:

$$
\begin{aligned}
tt^c &:= ff & ff^c &:= tt \\
(F_1 \wedge F_2)^c &:= F_1^c \vee F_2^c & (F_1 \vee F_2)^c &:= F_1^c \wedge F_2^c \\
(\langle\alpha\rangle F)^c &:= [\alpha]F^c & ([\alpha]F)^c &:= \langle\alpha\rangle F^c
\end{aligned}
$$

HML and Process Traces

Lemma (HML and process traces)

Let $(S, Act, \longrightarrow)$ be an LTS, and let $s, t \in S$ satisfy the same HMF (i.e., for all $F \in HMF$: $s \models F \iff t \models F$). Then $Tr(s) = Tr(t)$.

Proof.

Let $s, t \in S$ such that for all $F \in HMF$: $s \models F \iff t \models F$.

Assumption: $Tr(s) \neq Tr(t)$.
Then there exists $n \geq 1$ and $w = \alpha_1 \ldots \alpha_n \in Act^+$ with $w \in Tr(s) \setminus Tr(t)$ (or vice versa).
Hence, for $F := \langle \alpha_1 \rangle \ldots \langle \alpha_n \rangle \mathrm{tt} \in HMF$: $s \models F$ but $t \not\models F$. $\lightning$ □

Relationship Between HML and Strong Bisimilarity

Theorem (Hennessy-Milner Theorem)

Let $(S, Act, \{\xrightarrow{a} \mid a \in Act\})$ *be a finitely branching LTS and* $s, t \in S$. *Then:*

$$s \sim t \quad \textit{iff} \quad \textit{for every } F \in HMF : (s \models F \iff t \models F).$$

Outline of Lecture 10

Finiteness of HML

Observation: HML formulae only describe finite part of process behaviour

- each modal operator ($[.]$, $\langle . \rangle$) talks about one step
- only finite nesting of operators (modal depth)

Finiteness of HML

Observation: HML formulae only describe finite part of process behaviour

- each modal operator ($[.]$, $\langle . \rangle$) talks about one step
- only finite nesting of operators (modal depth)

Example 10.1

- $F := (\langle a \rangle [a] \mathrm{ff}) \vee \langle b \rangle \mathrm{tt} \in \mathit{HMF}$ has modal depth 2.
- Checking F involves analysis of all behaviours of length ≤ 2.

Finiteness of HML

Observation: HML formulae only describe finite part of process behaviour

- each modal operator ($[.]$, $\langle . \rangle$) talks about one step
- only finite nesting of operators (modal depth)

Example 10.1

- $F := (\langle a \rangle [a] \mathrm{ff}) \vee \langle b \rangle \mathrm{tt} \in HMF$ has modal depth 2.
- Checking F involves analysis of all behaviours of length ≤ 2.

But: sometimes necessary to refer to arbitrarily long computations
(e.g., "no deadlock state reachable")

- possible solution: support infinite conjunctions and disjunctions

Infinite Conjunctions

Example 10.2

- Let $C = a.C$, $D = a.D + a.\text{nil}$.
- Then $C \models [a]\langle a\rangle\text{tt}$ but $D \not\models [a]\langle a\rangle\text{tt}$ (i.e., C and D distinguishable by formula of depth 2). ✓

Infinite Conjunctions

Example 10.2

- Let $C = a.C$, $D = a.D + a.\text{nil}$.
- Then $C \models [a]\langle a\rangle\text{tt}$ but $D \not\models [a]\langle a\rangle\text{tt}$ (i.e., C and D distinguishable by formula of depth 2). ✓
- Now define $D_n = a.D_n + a.E_n$ where $n \in \mathbb{N}$, $E_n = a.E_{n-1}$ ($n \geq 1$), $E_0 = \text{nil}$.
- Then (for $[\alpha]^k F := \underbrace{[\alpha] \ldots [\alpha]}_{k \text{ times}} F$ where $F \in HMF$):
  - $C \models [a]^k\langle a\rangle\text{tt}$ for all $k \in \mathbb{N}$
  - $D_n \models [a]^k\langle a\rangle\text{tt}$ for all $0 \leq k \leq n$
  - $D_n \not\models [a]^k\langle a\rangle\text{tt}$ for all $k > n$

Infinite Conjunctions

Example 10.2

- Let $C = a.C$, $D = a.D + a.\text{nil}$.
- Then $C \models [a]\langle a\rangle\text{tt}$ but $D \not\models [a]\langle a\rangle\text{tt}$ (i.e., C and D distinguishable by formula of depth 2). ✓
- Now define $D_n = a.D_n + a.E_n$ where $n \in \mathbb{N}$, $E_n = a.E_{n-1}$ $(n \geq 1)$, $E_0 = \text{nil}$.
- Then (for $[\alpha]^k F := \underbrace{[\alpha]\ldots[\alpha]}_{k \text{ times}} F$ where $F \in HMF$):
  - $C \models [a]^k\langle a\rangle\text{tt}$ for all $k \in \mathbb{N}$
  - $D_n \models [a]^k\langle a\rangle\text{tt}$ for all $0 \leq k \leq n$
  - $D_n \not\models [a]^k\langle a\rangle\text{tt}$ for all $k > n$
- Conclusion: No single HML formula can distinguish C from all D_n. ↯
  - unsatisfactory as behaviour clearly different

Infinite Conjunctions

Example 10.2

- Let $C = a.C$, $D = a.D + a.\text{nil}$.
- Then $C \models [a]\langle a\rangle\text{tt}$ but $D \not\models [a]\langle a\rangle\text{tt}$ (i.e., C and D distinguishable by formula of depth 2). ✓
- Now define $D_n = a.D_n + a.E_n$ where $n \in \mathbb{N}$, $E_n = a.E_{n-1}$ $(n \geq 1)$, $E_0 = \text{nil}$.
- Then (for $[\alpha]^k F := \underbrace{[\alpha]\ldots[\alpha]}_{k \text{ times}} F$ where $F \in HMF$):
  - $C \models [a]^k\langle a\rangle\text{tt}$ for all $k \in \mathbb{N}$
  - $D_n \models [a]^k\langle a\rangle\text{tt}$ for all $0 \leq k \leq n$
  - $D_n \not\models [a]^k\langle a\rangle\text{tt}$ for all $k > n$
- Conclusion: No single HML formula can distinguish C from all D_n. ↯
  - unsatisfactory as behaviour clearly different
- Generally: invariant property "always $\langle a\rangle\text{tt}$" not expressible.
- Requires infinite conjunction:

$$Inv(\langle a\rangle\text{tt}) = \langle a\rangle\text{tt} \wedge [a]\langle a\rangle\text{tt} \wedge [a][a]\langle a\rangle\text{tt} \wedge \ldots = \bigwedge_{k \in \mathbb{N}} [a]^k\langle a\rangle\text{tt}$$

Infinite Disjunctions

Dually: possibility properties expressible by infinite disjunctions

Example 10.3

- Let $C = a.C$, $D = a.D + a.\text{nil}$ as before.
- C has no possibility to terminate.
- D has the option to terminate (i.e., to eventually satisfy $[a]\text{ff}$) at any time by choosing the $a.\text{nil}$ branch).

Infinite Disjunctions

Dually: possibility properties expressible by infinite disjunctions

Example 10.3

- Let $C = a.C$, $D = a.D + a.\mathrm{nil}$ as before.
- C has no possibility to terminate.
- D has the option to terminate (i.e., to eventually satisfy $[a]\mathrm{ff}$) at any time by choosing the $a.\mathrm{nil}$ branch).
- Expressible by infinite disjunction:

$$Pos([a]\mathrm{ff}) = [a]\mathrm{ff} \vee \langle a\rangle[a]\mathrm{ff} \vee \langle a\rangle\langle a\rangle[a]\mathrm{ff} \vee \ldots = \bigvee_{k\in\mathbb{N}} \langle a\rangle^k[a]\mathrm{ff}$$

Dually: possibility properties expressible by infinite disjunctions

Example 10.3

- Let $C = a.C$, $D = a.D + a.\text{nil}$ as before.
- C has no possibility to terminate.
- D has the option to terminate (i.e., to eventually satisfy $[a]\text{ff}$) at any time by choosing the $a.\text{nil}$ branch).
- Expressible by infinite disjunction:

$$Pos([a]\text{ff}) = [a]\text{ff} \vee \langle a\rangle[a]\text{ff} \vee \langle a\rangle\langle a\rangle[a]\text{ff} \vee \ldots = \bigvee_{k\in\mathbb{N}} \langle a\rangle^k[a]\text{ff}$$

Problem: infinite formulae are not easy to handle...

Introducing Recursion

Solution: employ recursion!

- $Inv(\langle a\rangle \mathrm{tt}) = \langle a\rangle \mathrm{tt} \wedge [Act]\; Inv(\langle a\rangle \mathrm{tt})$
- $Pos([Act]\mathrm{ff}) = [Act]\mathrm{ff} \vee \langle Act\rangle\; Pos([Act]\mathrm{ff})$

Introducing Recursion

Solution: employ recursion!

- $Inv(\langle a \rangle tt) = \langle a \rangle tt \wedge [Act]\, Inv(\langle a \rangle tt)$
- $Pos([Act]ff) = [Act]ff \vee \langle Act \rangle\, Pos([Act]ff)$

Interpretation: the sets of states $X, Y \subseteq S$ satisfying the respective formula should solve the corresponding semantic equations, i.e.,

- $X = \langle \cdot a \cdot \rangle(S) \cap [\cdot Act \cdot](X)$
- $Y = [\cdot Act \cdot](\emptyset) \cup \langle \cdot Act \cdot \rangle(Y)$

Introducing Recursion

Solution: employ recursion!

- $Inv(\langle a\rangle \mathrm{tt}) = \langle a\rangle \mathrm{tt} \wedge [Act]\, Inv(\langle a\rangle \mathrm{tt})$
- $Pos([Act]\mathrm{ff}) = [Act]\mathrm{ff} \vee \langle Act\rangle\, Pos([Act]\mathrm{ff})$

Interpretation: the sets of states $X, Y \subseteq S$ satisfying the respective formula should solve the corresponding semantic equations, i.e.,

- $X = \langle\cdot a\cdot\rangle(S) \cap [\cdot Act\cdot](X)$
- $Y = [\cdot Act\cdot](\emptyset) \cup \langle\cdot Act\cdot\rangle(Y)$

Open questions

- Do such recursive equations (always) have solutions?
- If so, are these unique?
- How can we decide whether a process satisfies a recursive formula (“model checking”)?

Introducing Recursion

Solution: employ recursion!

- $Inv(\langle a \rangle \mathrm{tt}) = \langle a \rangle \mathrm{tt} \wedge [Act]\, Inv(\langle a \rangle \mathrm{tt})$
- $Pos([Act]\mathrm{ff}) = [Act]\mathrm{ff} \vee \langle Act \rangle\, Pos([Act]\mathrm{ff})$

Interpretation: the sets of states $X, Y \subseteq S$ satisfying the respective formula should solve the corresponding semantic equations, i.e.,

- $X = \langle \cdot a \cdot \rangle(S) \cap [\cdot Act \cdot](X)$
- $Y = [\cdot Act \cdot](\emptyset) \cup \langle \cdot Act \cdot \rangle(Y)$

Open questions

- Do such recursive equations (always) have solutions?
  Yes, they do.
- If so, are these unique?
  Not necessarily.
- How can we decide whether a process satisfies a recursive formula (“model checking”)?
  Employ fixed-point iteration.

Example 10.4

- Consider again $C = a.C$, $D = a.D + a.\mathrm{nil}$

Example 10.4

- Consider again $C = a.C$, $D = a.D + a.\mathrm{nil}$
- Invariant: $X \equiv \langle a \rangle \mathrm{tt} \wedge [a]X$
  - $X = \emptyset$ is a solution (as no process can satisfy both $\langle a \rangle \mathrm{tt}$ and $[a]\mathrm{ff}$)
  - but we expect $C \in X$ (as C can perform a invariantly)
  - in fact, $X = \{C\}$ also solves the equation (and is the greatest solution w.r.t. $\subseteq$)

$\Rightarrow$ write $X \stackrel{max}{=} \langle a \rangle \mathrm{tt} \wedge [a]X$

Existence of Solutions

Example 10.4

- Consider again $C = a.C$, $D = a.D + a.\mathrm{nil}$
- Invariant: $X \equiv \langle a \rangle \mathrm{tt} \wedge [a]X$
  - $X = \emptyset$ is a solution (as no process can satisfy both $\langle a \rangle \mathrm{tt}$ and $[a]\mathrm{ff}$)
  - but we expect $C \in X$ (as C can perform a invariantly)
  - in fact, $X = \{C\}$ also solves the equation (and is the greatest solution w.r.t. $\subseteq$)

  $\Rightarrow$ write $X \overset{max}{=} \langle a \rangle \mathrm{tt} \wedge [a]X$
- Possibility: $Y \equiv [a]\mathrm{ff} \vee \langle a \rangle Y$
  - greatest solution: $Y = \{C, D, \mathrm{nil}\}$
  - but we expect $C \notin Y$ (as C cannot terminate at all)
  - here: least solution with respect to $\subseteq$: $Y = \{D, \mathrm{nil}\}$

  $\Rightarrow$ write $Y \overset{min}{=} [a]\mathrm{ff} \vee \langle a \rangle Y$

Uniqueness of Solutions

Uniqueness of solutions

- Use greatest solutions for properties that hold unless the process has a finite computation that disproves it.
- Use least solutions for properties that hold if the process has a finite computation that proves it.

Uniqueness of Solutions

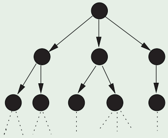
Uniqueness of solutions

- Use greatest solutions for properties that hold unless the process has a finite computation that disproves it.
- Use least solutions for properties that hold if the process has a finite computation that proves it.

Example 10.5

Let $(S, Act, \longrightarrow)$ be an LTS, $s \in S$, and $F \in HMF$.

- Invariant: $Inv(F) \equiv X$ for $X \stackrel{max}{=} F \wedge [Act]X$
  - $s \models Inv(F)$ if all states reachable from s satisfy F.



Uniqueness of Solutions

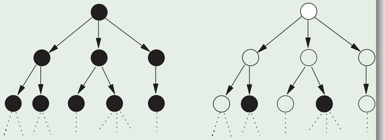
Uniqueness of solutions

- Use greatest solutions for properties that hold unless the process has a finite computation that disproves it.
- Use least solutions for properties that hold if the process has a finite computation that proves it.

Example 10.5

Let $(S, Act, \longrightarrow)$ be an LTS, $s \in S$, and $F \in HMF$.

- Invariant: $Inv(F) \equiv X$ for $X \overset{max}{=} F \wedge [Act]X$
  - $s \models Inv(F)$ if all states reachable from s satisfy F.
- Possibility: $Pos(F) \equiv Y$ for $Y \overset{min}{=} F \vee \langle Act \rangle Y$
  - $s \models Pos(F)$ if a state satisfying F is reachable from s.



$Inv(black)$ $\quad$ $Pos(black)$

Uniqueness of Solutions

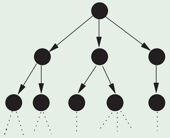
Uniqueness of solutions

- Use greatest solutions for properties that hold unless the process has a finite computation that disproves it.
- Use least solutions for properties that hold if the process has a finite computation that proves it.

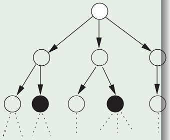
Example 10.5

Let $(S, Act, \longrightarrow)$ be an LTS, $s \in S$, and $F \in HMF$.

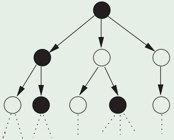
- Invariant: $Inv(F) \equiv X$ for $X \stackrel{max}{=} F \wedge [Act]X$
  - $s \models Inv(F)$ if all states reachable from s satisfy F.
- Possibility: $Pos(F) \equiv Y$ for $Y \stackrel{min}{=} F \vee \langle Act \rangle Y$
  - $s \models Pos(F)$ if a state satisfying F is reachable from s.
- Safety: $Safe(F) \equiv X$ for $X \stackrel{max}{=} F \wedge ([Act]\mathrm{ff} \vee \langle Act \rangle X)$
  - $s \models Safe(F)$ if s has a complete (i.e., infinite or terminating) transition sequence where each state satisfies F.



$Inv(black)$



$Pos(black)$



$Safe(black)$

Uniqueness of Solutions

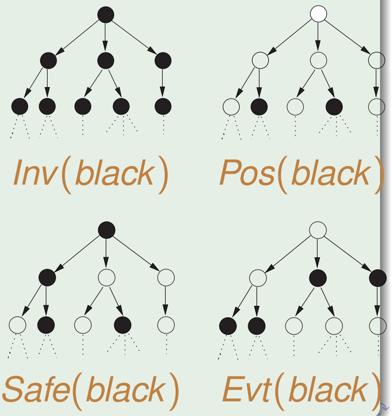
Uniqueness of solutions

- Use greatest solutions for properties that hold unless the process has a finite computation that disproves it.
- Use least solutions for properties that hold if the process has a finite computation that proves it.

Example 10.5

Let $(S, Act, \longrightarrow)$ be an LTS, $s \in S$, and $F \in HMF$.

- Invariant: $Inv(F) \equiv X$ for $X \overset{max}{=} F \wedge [Act]X$
  - $s \models Inv(F)$ if all states reachable from s satisfy F.
- Possibility: $Pos(F) \equiv Y$ for $Y \overset{min}{=} F \vee \langle Act \rangle Y$
  - $s \models Pos(F)$ if a state satisfying F is reachable from s.
- Safety: $Safe(F) \equiv X$ for $X \overset{max}{=} F \wedge ([Act]\mathrm{ff} \vee \langle Act \rangle X)$
  - $s \models Safe(F)$ if s has a complete (i.e., infinite or terminating) transition sequence where each state satisfies F.
- Eventuality: $Evt(F) \equiv Y$ for $Y \overset{min}{=} F \vee (\langle Act \rangle \mathrm{tt} \wedge [Act]Y)$
  - $s \models Evt(F)$ if each complete transition sequence starting in s contains a state satisfying F.



Outline of Lecture 10

Syntax of HML with One Recursive Variable

Initially: only one variable (for simplicity; later: mutual recursion)

Syntax of HML with One Recursive Variable

Initially: only one variable (for simplicity; later: mutual recursion)

Definition 10.6 (Syntax of HML with one variable)

The set HMF_X of Hennessy-Milner formulae with one variable X over a set of actions Act is defined by the following syntax:

$$
\begin{array}{rll}
F ::= & X & \text{(variable)} \\
| & \mathrm{tt} & \text{(true)} \\
| & \mathrm{ff} & \text{(false)} \\
| & F_1 \wedge F_2 & \text{(conjunction)} \\
| & F_1 \vee F_2 & \text{(disjunction)} \\
| & \langle \alpha \rangle F & \text{(diamond)} \\
| & [\alpha] F & \text{(box)}
\end{array}
$$

where $\alpha \in Act$.

Semantics of HML with One Recursive Variable I

So far: $\llbracket F \rrbracket \subseteq S$ for $F \in HMF$ and LTS $(S, Act, \longrightarrow)$.

Now: Semantics of formula depends on states that (are assumed to) satisfy X ("predicate transformer").

Semantics of HML with One Recursive Variable I

So far: $[\![F]\!] \subseteq S$ for $F \in HMF$ and LTS $(S, Act, \longrightarrow)$.

Now: Semantics of formula depends on states that (are assumed to) satisfy X ("predicate transformer").

Definition 10.7 (Semantics of HML with one variable)

Let $(S, Act, \longrightarrow)$ be an LTS and $F \in HMF_X$. The semantics of F,

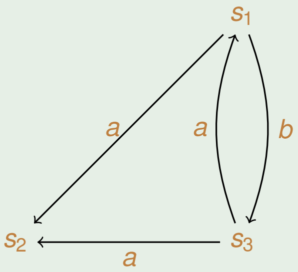
$$[\![F]\!] : 2^S \to 2^S,$$

is defined by

$$\begin{aligned}
[\![X]\!](T) &:= T \\
[\![\mathrm{tt}]\!](T) &:= S \\
[\![\mathrm{ff}]\!](T) &:= \emptyset \\
[\![F_1 \wedge F_2]\!](T) &:= [\![F_1]\!](T) \cap [\![F_2]\!](T) \\
[\![F_1 \vee F_2]\!](T) &:= [\![F_1]\!](T) \cup [\![F_2]\!](T) \\
[\![\langle\alpha\rangle F]\!](T) &:= \langle\cdot\alpha\cdot\rangle([\![F]\!](T)) \\
[\![[\alpha]F]\!](T) &:= [\cdot\alpha\cdot]([\![F]\!](T))
\end{aligned}$$

Semantics of HML with One Recursive Variable II

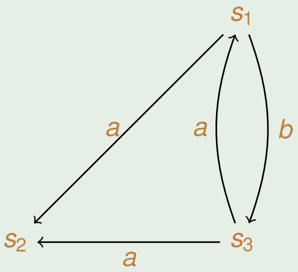
Example 10.8



Let $S := \{s_1, s_2, s_3\}$.

Semantics of HML with One Recursive Variable II

Example 10.8

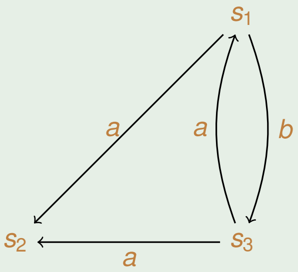


Let $S := \{s_1, s_2, s_3\}$.

- $[\![\langle a \rangle X]\!](\{s_1\}) = \{s_3\}$

Semantics of HML with One Recursive Variable II

Example 10.8

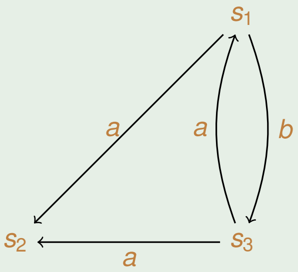


Let $S := \{s_1, s_2, s_3\}$.

- $[\![\langle a\rangle X]\!](\{s_1\}) = \{s_3\}$
- $[\![\langle a\rangle X]\!](\{s_1, s_2\}) = \{s_1, s_3\}$

Semantics of HML with One Recursive Variable II

Example 10.8



Let $S := \{s_1, s_2, s_3\}$.

- $[\![\langle a \rangle X]\!](\{s_1\}) = \{s_3\}$
- $[\![\langle a \rangle X]\!](\{s_1, s_2\}) = \{s_1, s_3\}$
- $[\![[b] X]\!](\{s_2\}) = \{s_2, s_3\}$

Semantics of HML with One Recursive Variable III

- Idea underlying the definition of

$$[\![.]\!] : HMF_X \to (2^S \to 2^S) :$$

If $T \subseteq S$ is the set of states that satisfy X, then $[\![F]\!](T)$ will be the set of states that satisfy F.

Semantics of HML with One Recursive Variable III

- Idea underlying the definition of

$$[\![.]\!] : HMF_X \to (2^S \to 2^S) :$$

  If $T \subseteq S$ is the set of states that satisfy X, then $[\![F]\!](T)$ will be the set of states that satisfy F.
- How to determine this T?
- According to previous discussion: as solution of recursive equation of the form $X \equiv F_X$ where $F_X \in HMF_X$.

Semantics of HML with One Recursive Variable III

- Idea underlying the definition of

$$[\![.]\!] : HMF_X \to (2^S \to 2^S) :$$

  If $T \subseteq S$ is the set of states that satisfy X, then $[\![F]\!](T)$ will be the set of states that satisfy F.
- How to determine this T?
- According to previous discussion: as solution of recursive equation of the form $X \equiv F_X$ where $F_X \in HMF_X$.
- But: solution not unique; therefore write:

$$X \stackrel{min}{=} F_X \qquad \text{or} \qquad X \stackrel{max}{=} F_X$$

Semantics of HML with One Recursive Variable III

- Idea underlying the definition of

$$[\![.]\!] : HMF_X \to (2^S \to 2^S) :$$

If $T \subseteq S$ is the set of states that satisfy X, then $[\![F]\!](T)$ will be the set of states that satisfy F.

- How to determine this T?
- According to previous discussion: as solution of recursive equation of the form $X \equiv F_X$ where $F_X \in HMF_X$.
- But: solution not unique; therefore write:

$$X \overset{min}{=} F_X \qquad \text{or} \qquad X \overset{max}{=} F_X$$

- In the following we will see:
  (1) Equation $X \equiv F_X$ is always solvable.
  (2) Least and greatest solutions are unique and can be obtained by fixed-point iteration.

Outline of Lecture 10

Partial Orders

Definition (Partial order; cf. Definition 7.1)

A partial order (PO) $(D, \sqsubseteq)$ consists of a set D, called domain, and of a relation $\sqsubseteq \subseteq D \times D$ such that, for every $d_1, d_2, d_3 \in D$,

reflexivity: $d_1 \sqsubseteq d_1$

transitivity: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_3 \Rightarrow d_1 \sqsubseteq d_3$

antisymmetry: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_1 \Rightarrow d_1 = d_2$

It is called total if, in addition, always $d_1 \sqsubseteq d_2$ or $d_2 \sqsubseteq d_1$.

Partial Orders

Definition (Partial order; cf. Definition 7.1)

A partial order (PO) $(D, \sqsubseteq)$ consists of a set D, called domain, and of a relation $\sqsubseteq \subseteq D \times D$ such that, for every $d_1, d_2, d_3 \in D$,

- reflexivity: $d_1 \sqsubseteq d_1$
- transitivity: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_3 \Rightarrow d_1 \sqsubseteq d_3$
- antisymmetry: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_1 \Rightarrow d_1 = d_2$

It is called total if, in addition, always $d_1 \sqsubseteq d_2$ or $d_2 \sqsubseteq d_1$.

Lemma 10.9 (Application to HML with recursion)

Let $(S, Act, \longrightarrow)$ be an LTS. Then $(2^S, \subseteq)$ is a PO.

Complete Lattices

Definition (Complete lattice; cf. Definition 7.5)

A complete lattice is a partial order $(D, \sqsubseteq)$ such that all subsets of D have LUBs and GLBs. In this case,

$$\bot := \bigsqcup \emptyset \; (= \bigsqcap D) \qquad \text{and} \qquad \top := \bigsqcap \emptyset \; (= \bigsqcup D)$$

respectively denote the least and greatest element of D.

Lemma (cf. Lemma 7.7)

Let S be some (finite or infinite) set. Then $(2^S, \subseteq)$ is a complete lattice with

- $\bigsqcup \mathcal{T} = \bigcup \mathcal{T} = \bigcup_{T \in \mathcal{T}} T$ *for all* $\mathcal{T} \subseteq 2^S$
- $\bigsqcap \mathcal{T} = \bigcap \mathcal{T} = \bigcap_{T \in \mathcal{T}} T$ *for all* $\mathcal{T} \subseteq 2^S$
- $\bot = \bigsqcup \emptyset = \bigsqcap 2^S = \emptyset$
- $\top = \bigsqcap \emptyset = \bigsqcup 2^S = S$

Complete Lattices

Definition (Complete lattice; cf. Definition 7.5)

A complete lattice is a partial order $(D, \sqsubseteq)$ such that all subsets of D have LUBs and GLBs. In this case,

$$\bot := \bigsqcup \emptyset \; (= \bigsqcap D) \qquad \text{and} \qquad \top := \bigsqcap \emptyset \; (= \bigsqcup D)$$

respectively denote the least and greatest element of D.

Lemma (cf. Lemma 7.7)

Let S be some (finite or infinite) set. Then $(2^S, \subseteq)$ is a complete lattice with

- $\bigsqcup \mathcal{T} = \bigcup \mathcal{T} = \bigcup_{T \in \mathcal{T}} T$ *for all* $\mathcal{T} \subseteq 2^S$
- $\bigsqcap \mathcal{T} = \bigcap \mathcal{T} = \bigcap_{T \in \mathcal{T}} T$ *for all* $\mathcal{T} \subseteq 2^S$
- $\bot = \bigsqcup \emptyset = \bigsqcap 2^S = \emptyset$
- $\top = \bigsqcap \emptyset = \bigsqcup 2^S = S$

Corollary 10.10 (Application to HML with recursion)

Let $(S, Act, \longrightarrow)$ be an LTS. Then $(2^S, \subseteq)$ is a complete lattice.

The Fixed-Point Theorems

Theorem (Tarski's fixed-point theorem; cf. Theorem 7.12)

Let $(D, \sqsubseteq)$ be a complete lattice and $f : D \to D$ monotonic. Then f has a least fixed point $\mathrm{lfp}(f)$ and a greatest fixed point $\mathrm{gfp}(f)$, which are given by

$$\mathrm{lfp}(f) := \bigsqcap\{d \in D \mid f(d) \sqsubseteq d\} \qquad \text{(GLB of all pre-fixed points of } f\text{)}$$

$$\mathrm{gfp}(f) := \bigsqcup\{d \in D \mid d \sqsubseteq f(d)\} \qquad \text{(LUB of all post-fixed points of } f\text{)}$$

Theorem (Fixed-point theorem for finite lattices; cf. Theorem 7.14)

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

$$\mathrm{lfp}(f) = f^m(\bot) \quad \text{and} \quad \mathrm{gfp}(f) = f^M(\top)$$

for some $m, M \in \mathbb{N}$ where $f^0(d) := d$ and $f^{k+1}(d) := f(f^k(d))$.

Lemma 10.11

Let $(S, Act, \longrightarrow)$ *be an LTS and* $F \in HMF_X$. *Then*

(1) $[\![F]\!] : 2^S \to 2^S$ *is monotonic w.r.t.* $(2^S, \subseteq)$

Application to HML with Recursion

Lemma 10.11

Let $(S, Act, \longrightarrow)$ *be an LTS and* $F \in HMF_X$. *Then*

(1) $[\![F]\!] : 2^S \to 2^S$ *is monotonic w.r.t.* $(2^S, \subseteq)$

(2) $\text{lfp}([\![F]\!]) = \bigcap\{T \subseteq S \mid [\![F]\!](T) \subseteq T\}$

(3) $\text{gfp}([\![F]\!]) = \bigcup\{T \subseteq S \mid T \subseteq [\![F]\!](T)\}$

Application to HML with Recursion

Lemma 10.11

Let $(S, Act, \longrightarrow)$ *be an LTS and* $F \in HMF_X$*. Then*

(1) $[\![F]\!] : 2^S \to 2^S$ *is monotonic w.r.t.* $(2^S, \subseteq)$

(2) $\text{lfp}([\![F]\!]) = \bigcap\{T \subseteq S \mid [\![F]\!](T) \subseteq T\}$

(3) $\text{gfp}([\![F]\!]) = \bigcup\{T \subseteq S \mid T \subseteq [\![F]\!](T)\}$

If, in addition, S *is finite, then*

(4) $\text{lfp}([\![F]\!]) = [\![F]\!]^m(\emptyset)$ *for some* $m \in \mathbb{N}$

(5) $\text{gfp}([\![F]\!]) = [\![F]\!]^M(S)$ *for some* $M \in \mathbb{N}$

Application to HML with Recursion

Lemma 10.11

Let $(S, Act, \longrightarrow)$ *be an LTS and* $F \in HMF_X$. *Then*

(1) $[\![F]\!] : 2^S \to 2^S$ *is monotonic w.r.t.* $(2^S, \subseteq)$

(2) $\mathrm{lfp}([\![F]\!]) = \bigcap\{T \subseteq S \mid [\![F]\!](T) \subseteq T\}$

(3) $\mathrm{gfp}([\![F]\!]) = \bigcup\{T \subseteq S \mid T \subseteq [\![F]\!](T)\}$

If, in addition, S *is finite, then*

(4) $\mathrm{lfp}([\![F]\!]) = [\![F]\!]^m(\emptyset)$ *for some* $m \in \mathbb{N}$

(5) $\mathrm{gfp}([\![F]\!]) = [\![F]\!]^M(S)$ *for some* $M \in \mathbb{N}$

Proof.

(1) by induction on the structure of F (important: HMF_X does not support negation!)

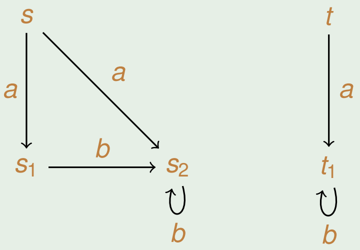
(2) by Corollary 10.10 and Theorem 7.12

(3) by Corollary 10.10 and Theorem 7.12

(4) by Corollary 10.10 and Theorem 7.14

A Greatest Fixed Point

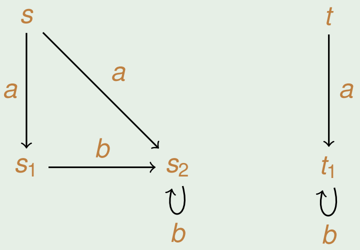
Example 10.12



Let $S := \{s, s_1, s_2, t, t_1\}$.

A Greatest Fixed Point

Example 10.12



Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

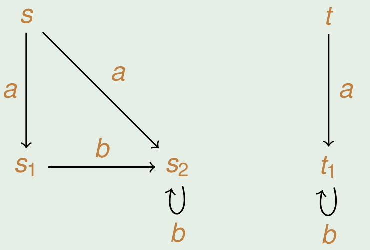
$$X \overset{max}{=} \langle b \rangle \text{tt} \wedge [b]X$$

(invariant: "all b^*-successors have a b-successor") equals $\text{gfp}(f)$ for

$$f : 2^S \to 2^S : T \mapsto \langle \cdot b \cdot \rangle(S) \cap [\cdot b \cdot](T)$$

A Greatest Fixed Point

Example 10.12



Application of Lemma 10.11(5):

$$f(S) = \langle \cdot b \cdot \rangle(S) \cap [\cdot b \cdot](S)$$
$$= \{s_1, s_2, t_1\} \cap S$$
$$= \{s_1, s_2, t_1\}$$

Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

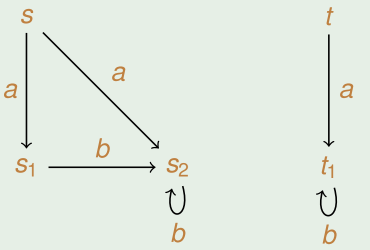
$$X \stackrel{max}{=} \langle b \rangle \mathrm{tt} \wedge [b]X$$

(invariant: “all b^*-successors have a b-successor”) equals $\mathrm{gfp}(f)$ for

$$f : 2^S \to 2^S : T \mapsto \langle \cdot b \cdot \rangle(S) \cap [\cdot b \cdot](T)$$

A Greatest Fixed Point

Example 10.12



Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

$$X \overset{max}{=} \langle b\rangle \mathrm{tt} \wedge [b]X$$

(invariant: "all b^*-successors have a b-successor") equals $\mathrm{gfp}(f)$ for

$$f : 2^S \to 2^S : T \mapsto \langle\cdot b\cdot\rangle(S) \cap [\cdot b\cdot](T)$$

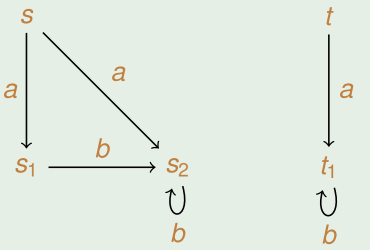
Application of Lemma 10.11(5):

$$\begin{aligned} f(S) &= \langle\cdot b\cdot\rangle(S) \cap [\cdot b\cdot](S) \\ &= \{s_1, s_2, t_1\} \cap S \\ &= \{s_1, s_2, t_1\} \end{aligned}$$

$$\begin{aligned} f^2(S) &= \langle\cdot b\cdot\rangle(S) \cap [\cdot b\cdot](\{s_1, s_2, t_1\}) \\ &= \{s_1, s_2, t_1\} \cap \{s, s_1, s_2, t, t_1\} \\ &= \{s_1, s_2, t_1\} \\ &= f(S) \end{aligned}$$

A Greatest Fixed Point

Example 10.12



Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

$$X \overset{max}{=} \langle b \rangle \mathrm{tt} \wedge [b]X$$

(invariant: "all b^*-successors have a b-successor") equals $\mathrm{gfp}(f)$ for

$$f : 2^S \to 2^S : T \mapsto \langle \cdot b \cdot \rangle(S) \;\cap\; [\cdot b \cdot](T)$$

Application of Lemma 10.11(5):

$$\begin{aligned} f(S) &= \langle \cdot b \cdot \rangle(S) \cap [\cdot b \cdot](S) \\ &= \{s_1, s_2, t_1\} \cap S \\ &= \{s_1, s_2, t_1\} \end{aligned}$$

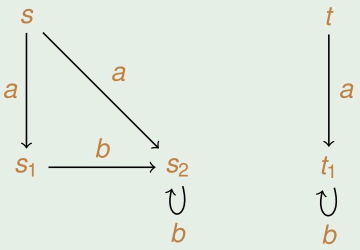
$$\begin{aligned} f^2(S) &= \langle \cdot b \cdot \rangle(S) \cap [\cdot b \cdot](\{s_1, s_2, t_1\}) \\ &= \{s_1, s_2, t_1\} \cap \{s, s_1, s_2, t, t_1\} \\ &= \{s_1, s_2, t_1\} \\ &= f(S) \end{aligned}$$

$$\Rightarrow \mathrm{gfp}(f) = \{s_1, s_2, t_1\}$$

(verify using CAAL)

A Least Fixed Point

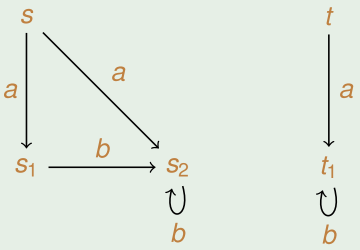
Example 10.13



Let $S := \{s, s_1, s_2, t, t_1\}$.

A Least Fixed Point

Example 10.13



Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

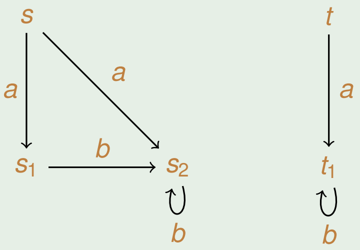
$$Y \overset{min}{=} \langle b\rangle \mathrm{tt} \vee \langle\{a, b\}\rangle Y$$

(possibility: “a b-transition is reachable”)
equals $\mathrm{lfp}(g)$ for

$$g : 2^S \to 2^S : T \mapsto \langle\cdot b\cdot\rangle(S) \cup \langle\cdot\{a, b\}\cdot\rangle(T)$$

A Least Fixed Point

Example 10.13



Application of Lemma 10.11(4):

$$\begin{aligned} g(\emptyset) &= \langle\cdot b\cdot\rangle(S) \cup \langle\cdot\{a,b\}\cdot\rangle(\emptyset) \\ &= \{s_1, s_2, t_1\} \cup \emptyset \\ &= \{s_1, s_2, t_1\} \end{aligned}$$

Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

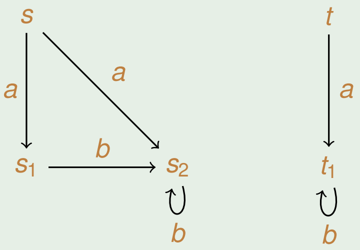
$$Y \overset{min}{=} \langle b\rangle \mathrm{tt} \vee \langle\{a,b\}\rangle Y$$

(possibility: “a b-transition is reachable”)
equals $\mathrm{lfp}(g)$ for

$$g : 2^S \to 2^S : T \mapsto \langle\cdot b\cdot\rangle(S) \cup \langle\cdot\{a,b\}\cdot\rangle(T)$$

A Least Fixed Point

Example 10.13



Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

$$Y \stackrel{min}{=} \langle b \rangle \mathrm{tt} \vee \langle \{a, b\} \rangle Y$$

(possibility: "a b-transition is reachable")
equals $\mathrm{lfp}(g)$ for

$$g : 2^S \to 2^S : T \mapsto \langle \cdot b \cdot \rangle(S) \cup \langle \cdot \{a, b\} \cdot \rangle(T)$$

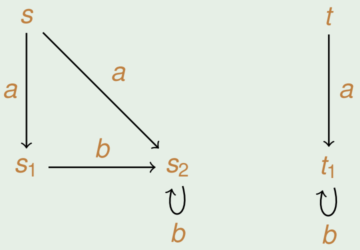
Application of Lemma 10.11(4):

$$\begin{aligned} g(\emptyset) &= \langle \cdot b \cdot \rangle(S) \cup \langle \cdot \{a, b\} \cdot \rangle(\emptyset) \\ &= \{s_1, s_2, t_1\} \cup \emptyset \\ &= \{s_1, s_2, t_1\} \end{aligned}$$

$$\begin{aligned} g^2(\emptyset) &= \langle \cdot b \cdot \rangle(S) \cup \langle \cdot \{a, b\} \cdot \rangle(\{s_1, s_2, t_1\}) \\ &= \{s_1, s_2, t_1\} \cup \{s, s_1, s_2, t, t_1\} \\ &= \{s, s_1, s_2, t, t_1\} \\ &= S \end{aligned}$$

A Least Fixed Point

Example 10.13



Let $S := \{s, s_1, s_2, t, t_1\}$.

Solution of

$$Y \overset{min}{=} \langle b \rangle \mathrm{tt} \vee \langle \{a, b\} \rangle Y$$

(possibility: "a b-transition is reachable")
equals $\mathrm{lfp}(g)$ for

$$g : 2^S \to 2^S : T \mapsto \langle \cdot b \cdot \rangle(S) \cup \langle \cdot \{a, b\} \cdot \rangle(T)$$

Application of Lemma 10.11(4):

$$\begin{aligned} g(\emptyset) &= \langle \cdot b \cdot \rangle(S) \cup \langle \cdot \{a, b\} \cdot \rangle(\emptyset) \\ &= \{s_1, s_2, t_1\} \cup \emptyset \\ &= \{s_1, s_2, t_1\} \end{aligned}$$

$$\begin{aligned} g^2(\emptyset) &= \langle \cdot b \cdot \rangle(S) \cup \langle \cdot \{a, b\} \cdot \rangle(\{s_1, s_2, t_1\}) \\ &= \{s_1, s_2, t_1\} \cup \{s, s_1, s_2, t, t_1\} \\ &= \{s, s_1, s_2, t, t_1\} \\ &= S \end{aligned}$$

$$\Rightarrow \mathrm{lfp}(f) = S$$

(verify using CAAL)