Syntax and semantics of Hennessy-Milner logic (reprise)
Correspondence with strong bisimilarity
Examples in CWB
Hennessy-Milner logic and temporal properties
Syntax of the Formulae ($a \in \text{Act}$)

\[
F, G ::= \text{tt} \mid \text{ff} \mid F \land G \mid F \lor G \mid \langle a \rangle F \mid [a]F
\]

Intuition:
- \text{tt} all processes satisfy this property
- \text{ff} no process satisfies this property
- \text{\land}, \text{\lor} usual logical AND and OR
- \langle a \rangle F there is at least one $a$-successor that satisfies $F$
- $[a]F$ all $a$-successors have to satisfy $F$
Let \((\text{Proc}, \text{Act}, \{\xrightarrow{a} | a \in \text{Act}\})\) be an LTS.

Validity of the logical triple \(p \models F\) \((p \in \text{Proc}, F\) a HM formula\)

- \(p \models \text{tt}\) for each \(p \in \text{Proc}\)
- \(p \models \text{ff}\) for no \(p\) (we also write \(p \not\models \text{ff}\))
- \(p \models F \land G\) iff \(p \models F\) and \(p \models G\)
- \(p \models F \lor G\) iff \(p \models F\) or \(p \models G\)
- \(p \models \langle a \rangle F\) iff \(p \xrightarrow{a} p'\) for some \(p' \in \text{Proc}\) such that \(p' \models F\)
- \(p \models [a] F\) iff \(p' \models F,\) for all \(p' \in \text{Proc}\) such that \(p \xrightarrow{a} p'\)

We write \(p \not\models F\) whenever \(p\) does not satisfy \(F\).
What about Negation?

For every formula $F$ we define the formula $F^c$ as follows:

- $tt^c = ff$
- $ff^c = tt$
- $(F \land G)^c = F^c \lor G^c$
- $(F \lor G)^c = F^c \land G^c$
- $(\langle a \rangle F)^c = [a]F^c$
- $([a]F)^c = \langle a \rangle F^c$

Theorem ($F^c$ is equivalent to the negation of $F$)

For any $p \in Proc$ and any HM formula $F$

1. $p \models F \implies p \not\models F^c$
2. $p \not\models F \implies p \models F^c$
What about Negation?

For every formula $F$ we define the formula $F^c$ as follows:

- $tt^c = ff$
- $ff^c = tt$
- $(F \land G)^c = F^c \lor G^c$
- $(F \lor G)^c = F^c \land G^c$
- $(\langle a \rangle F)^c = [a]F^c$
- $(\lbrack a \rbrack F)^c = \langle a \rangle F^c$

**Theorem** ($F^c$ is equivalent to the negation of $F$)

For any $p \in Proc$ and any HM formula $F$

1. $p \models F \implies p \not\models F^c$
2. $p \not\models F \implies p \models F^c$
Idea: $[[F]]$ is the set of all states that satisfy $F$

- $[[tt]] = Proc$
- $[[ff]] = \emptyset$
- $[[F \lor G]] = [[F]] \cup [[G]]$
- $[[F \land G]] = [[F]] \cap [[G]]$
- $[[\langle a \rangle F]] = \langle \cdot a \cdot \rangle [[F]]$
- $[[[a]F]] = [\cdot a \cdot][[F]]$

where $\langle \cdot a \cdot \rangle, [\cdot a \cdot] : 2^{(Proc)} \rightarrow 2^{(Proc)}$ are defined by

\[
\langle \cdot a \cdot \rangle S = \{ p \in Proc | \exists p'. p \xrightarrow{a} p' \text{ and } p' \in S \}\n\]
\[
[\cdot a \cdot] S = \{ p \in Proc | \forall p'. p \xrightarrow{a} p' \implies p' \in S \}.\n\]
The Correspondence Theorem

Let \((\text{Proc}, \text{Act}, \{\xrightarrow{a} | a \in \text{Act}\})\) be an LTS, \(p \in \text{Proc}\) and \(F\) a formula of Hennessy-Milner logic. Then

\[ p \models F \iff p \in \llbracket F \rrbracket. \]

Proof: by structural induction on the structure of the formula \(F\).
The Correspondence Theorem

Theorem

Let \((\text{Proc}, \text{Act}, \{\xrightarrow{a} | a \in \text{Act}\})\) be an LTS, \(p \in \text{Proc}\) and \(F\) a formula of Hennessy-Milner logic. Then

\[ p \models F \quad \text{if and only if} \quad p \in \llbracket F \rrbracket. \]

Proof: by structural induction on the structure of the formula \(F\).
Let \((\mathit{Proc}, \mathit{Act}, \{ \xrightarrow{a} | a \in \mathit{Act} \})\) be an LTS. We call it \textit{image-finite} iff for every \(p \in \mathit{Proc}\) and every \(a \in \mathit{Act}\) the set 

\[
\{ p' \in \mathit{Proc} | p \xrightarrow{a} p' \}
\]

is finite.
Theorem (Hennessy-Milner)

Let \((\text{Proc}, \text{Act}, \{a \xrightarrow{a} | a \in \text{Act}\})\) be an image-finite LTS and \(p, q \in \text{St}\). Then

\[ p \sim q \]

if and only if

for every HM formula \(F\): \((p \models F \iff q \models F)\).
CWB Session

hm.cwb

agent S = a.S1;
agent S1 = b.0 + c.0;
agent T = a.T1 + a.T2;
agent T1 = b.0;
agent T2 = c.0;

[luca@ve15638 CWB]$ 
./xccscwb.x86-linux

> input "hm.cwb";
> print;
> help logic;
> checkprop(S,<a>(<b>T & <c>T));
  true
> checkprop(T,<a>(<b>T & <c>T));
  false
> help dfstrong;
> dfstrong(S,T);
  [a]<b>T
> exit;
Modal depth (nesting degree) for Hennessy-Milner formulae:

- $md(tt) = md(ff) = 0$
- $md(F \land G) = md(F \lor G) = \max\{md(F), md(G)\}$
- $md([a]F) = md(\langle a \rangle F) = md(F) + 1$

Idea: a formula $F$ can “see” only up to depth $md(F)$.

**Theorem (let $F$ be a HM formula and $k = md(F)$)**

If the defender has a defending strategy in the strong bisimulation game from $s$ and $t$ up to $k$ rounds then $s \models F$ if and only if $t \models F$.

**Conclusion**

There is no Hennessy-Milner formula $F$ that can detect a deadlock in an arbitrary LTS.
Modal depth (nesting degree) for Hennessy-Milner formulae:

- $md(tt) = md(ff) = 0$
- $md(F \land G) = md(F \lor G) = \max\{md(F), md(G)\}$
- $md([a]F) = md(\langle a \rangle F) = md(F) + 1$

Idea: a formula $F$ can “see” only up to depth $md(F)$.

**Theorem (let $F$ be a HM formula and $k = md(F)$)**

If the defender has a defending strategy in the strong bisimulation game from $s$ and $t$ up to $k$ rounds then $s \models F$ if and only if $t \models F$.

**Conclusion**

There is no Hennessy-Milner formula $F$ that can detect a deadlock in an arbitrary LTS.
Modal depth (nesting degree) for Hennessy-Milner formulae:

- \( md(tt) = md(ff) = 0 \)
- \( md(F \land G) = md(F \lor G) = \max\{md(F), md(G)\} \)
- \( md([a]F) = md(\langle a \rangle F) = md(F) + 1 \)

Idea: a formula \( F \) can “see” only up to depth \( md(F) \).

**Theorem (let \( F \) be a HM formula and \( k = md(F) \))**

If the defender has a defending strategy in the strong bisimulation game from \( s \) and \( t \) up to \( k \) rounds then \( s \models F \) if and only if \( t \models F \).

**Conclusion**

There is no Hennessy-Milner formula \( F \) that can detect a deadlock in an arbitrary LTS.
Temporal Properties not Expressible in HM Logic

\[ s \models Inv(F) \text{ iff all states reachable from } s \text{ satisfy } F \]
\[ s \models Pos(F) \text{ iff there is a reachable state which satisfies } F \]

Fact

Properties \( Inv(F) \) and \( Pos(F) \) are not expressible in HM logic.

Let \( Act = \{a_1, a_2, \ldots, a_n\} \) be a finite set of actions. We define

- \( \langle Act \rangle F \stackrel{\text{def}}{=} \langle a_1 \rangle F \lor \langle a_2 \rangle F \lor \ldots \lor \langle a_n \rangle F \)
- \( [Act]F \stackrel{\text{def}}{=} [a_1]F \land [a_2]F \land \ldots \land [a_n]F \)

\( Inv(F) \equiv F \land [Act]F \land [Act][Act]F \land [Act][Act][Act]F \land \ldots \)
\( Pos(F) \equiv F \lor \langle Act \rangle F \lor \langle Act \rangle \langle Act \rangle F \lor \langle Act \rangle \langle Act \rangle \langle Act \rangle F \lor \ldots \)
s \models \text{Inv}(F) \iff \text{all states reachable from } s \text{ satisfy } F
s \models \text{Pos}(F) \iff \text{there is a reachable state which satisfies } F

\textbf{Fact}

Properties \text{Inv}(F) \text{ and } \text{Pos}(F) \text{ are not expressible in } \text{HM} \text{ logic.}

Let \text{Act} = \{a_1, a_2, \ldots, a_n\} \text{ be a finite set of actions. We define}

\begin{itemize}
  \item \langle \text{Act} \rangle F \overset{\text{def}}{=} \langle a_1 \rangle F \lor \langle a_2 \rangle F \lor \ldots \lor \langle a_n \rangle F
  \item [\text{Act}] F \overset{\text{def}}{=} [a_1] F \land [a_2] F \land \ldots \land [a_n] F
\end{itemize}

\text{Inv}(F) \equiv F \land [\text{Act}]F \land [\text{Act}][\text{Act}]F \land [\text{Act}][\text{Act}][\text{Act}]F \land \ldots
\text{Pos}(F) \equiv F \lor \langle \text{Act} \rangle F \lor \langle \text{Act} \rangle \langle \text{Act} \rangle F \lor \langle \text{Act} \rangle \langle \text{Act} \rangle \langle \text{Act} \rangle F \lor \ldots
Temporal Properties not Expressible in HM Logic

$s \models Inv(F)$ iff all states reachable from $s$ satisfy $F$

$s \models Pos(F)$ iff there is a reachable state which satisfies $F$

**Fact**

Properties $Inv(F)$ and $Pos(F)$ are not expressible in HM logic.

Let $Act = \{a_1, a_2, \ldots, a_n\}$ be a finite set of actions. We define

- $\langle Act \rangle F \overset{\text{def}}{=} \langle a_1 \rangle F \lor \langle a_2 \rangle F \lor \ldots \lor \langle a_n \rangle F$
- $[Act]F \overset{\text{def}}{=} [a_1]F \land [a_2]F \land \ldots \land [a_n]F$

$Inv(F) \equiv F \land [Act]F \land [Act][Act]F \land [Act][Act][Act]F \land \ldots$

$Pos(F) \equiv F \lor \langle Act \rangle F \lor \langle Act \rangle \langle Act \rangle F \lor \langle Act \rangle \langle Act \rangle \langle Act \rangle F \lor \ldots$
Problems

- Infinite formulae are not allowed in HM logic
- Infinite formulae are difficult to handle

Why not to use recursion?

- $\text{Inv}(F)$ expressed by $X \overset{\text{def}}{=} F \land [\text{Act}]X$
- $\text{Pos}(F)$ expressed by $X \overset{\text{def}}{=} F \lor \langle \text{Act} \rangle X$

Question: How to define the semantics of such equations?
Problems

- infinite formulae are not allowed in HM logic
- infinite formulae are difficult to handle

Why not to use recursion?

- $\text{Inv}(F)$ expressed by $X \overset{\text{def}}{=} F \land [\text{Act}]X$
- $\text{Pos}(F)$ expressed by $X \overset{\text{def}}{=} F \lor \langle \text{Act} \rangle X$

Question: How to define the semantics of such equations?
Problems

- infinite formulae are not allowed in HM logic
- infinite formulae are difficult to handle

Why not to use recursion?

- $\text{Inv}(F)$ expressed by $X \overset{\text{def}}{=} F \land [\text{Act}]X$
- $\text{Pos}(F)$ expressed by $X \overset{\text{def}}{=} F \lor \langle \text{Act} \rangle X$

Question: How to define the semantics of such equations?
**Solving Equations is Tricky**

### Equations over Natural Numbers ($n \in \mathbb{N}$)

- $n = 2 \times n$  one solution $n = 0$
- $n = n + 1$  no solution
- $n = 1 \times n$  many solutions (every $n \in \text{Nat}$ is a solution)

### Equations over Sets of Integers ($M \in 2^{\mathbb{N}}$)

- $M = \{7\} \cap M$  one solution $M = \{7\}$
- $M = \mathbb{N} \setminus M$  no solution
- $M = \{3\} \cup M$  many solutions (every $M \supseteq \{3\}$ is a solution)

### What about Equations over Processes?

\[ X \overset{\text{def}}{=} [a]ff \lor \langle a \rangle X \quad \Rightarrow \quad \text{find } S \subseteq 2^{\text{Proc}} \text{ s.t. } S = [\cdot a \cdot]\emptyset \cup \langle a \rangle S \]
Solving Equations is Tricky

Equations over Natural Numbers \((n \in \mathbb{N})\)

\[
\begin{align*}
n &= 2 \times n & \text{one solution } n = 0 \\
n &= n + 1 & \text{no solution} \\
n &= 1 \times n & \text{many solutions (every } n \in Nat \text{ is a solution)}
\end{align*}
\]

Equations over Sets of Integers \((M \in 2^\mathbb{N})\)

\[
\begin{align*}
M &= \{7\} \cap M & \text{one solution } M = \{7\} \\
M &= \mathbb{N} \setminus M & \text{no solution} \\
M &= \{3\} \cup M & \text{many solutions (every } M \supseteq \{3\} \text{ is a solution)}
\end{align*}
\]

What about Equations over Processes?

\[
X \overset{\text{def}}{=} [a]f \lor \langle a \rangle X \quad \Rightarrow \quad \text{find } S \subseteq 2^{Proc} \text{ s.t. } S = [\cdot a \cdot] \emptyset \cup \langle \cdot a \cdot \rangle S
\]
### Equations over Natural Numbers ($n \in \mathbb{N}$)

- $n = 2 \times n$  
  - one solution $n = 0$
- $n = n + 1$  
  - no solution
- $n = 1 \times n$  
  - many solutions (every $n \in \text{Nat}$ is a solution)

### Equations over Sets of Integers ($M \in 2^{\mathbb{N}}$)

- $M = \{7\} \cap M$  
  - one solution $M = \{7\}$
- $M = \mathbb{N} \setminus M$  
  - no solution
- $M = \{3\} \cup M$  
  - many solutions (every $M \supseteq \{3\}$ is a solution)

### What about Equations over Processes?

$$X \overset{\text{def}}{=} [a]ff \lor \langle a \rangle X \quad \Rightarrow \quad \text{find } S \subseteq 2^{\text{Proc}} \text{ s.t. } S = [\cdot a\cdot]\emptyset \cup \langle \cdot a\cdot \rangle S$$