Concurrency Theory

Winter Semester 2019/20

Lecture 15: Hennessy-Milner Logic and Bisimilarity

Joost-Pieter Katoen and Thomas Noll
Software Modeling and Verification Group
RWTH Aachen University

https://moves.rwth-aachen.de/teaching/ws-19-20/ct/

Software Modeling and Verification Chair | RWTH AACHEN UNIVERSITY

Outline of Lecture 15

Recap: Weak Bisimilarity

Weak Bisimilarity

1. Weak bisimilarity is based on mutually mimicking processes.
2. But: τ-actions do not need to be mimicked, as they are internal.
3. Weak bisimilarity is (non-τ) deadlock sensitive.
4. Divergence is weakly bisimilar to a deadlock process.
5. Weak bisimilarity is not a congruence for choice ($+$).
6. Observation congruence remedies this by forcing initial τ-actions to be mimicked.
7. Checking weak (non-)bisimilarity can be done using a two-player game.

Outline of Lecture 15

Deciding Weak Bisimilarity

- Checking whether $P \approx Q$ (or $P \approx^c Q$) over finite-state processes can be reduced to checking strong bisimilarity $\sim$, using a technique called saturation.

Deciding Weak Bisimilarity

- Checking whether $P \approx Q$ (or $P \approx^c Q$) over finite-state processes can be reduced to checking strong bisimilarity $\sim$, using a technique called saturation.
- Intuitively, saturation amounts to:
  1. pre-computing the weak transition relation $\stackrel{\alpha}{\Longrightarrow}$ (for $\alpha \neq \tau$) based on given relation $\longrightarrow$
  2. constructing a new finite-state process by replacing original transitions with weak transitions

  (similarly to elimination of ε-transitions in ε-NFA; see following example)

Deciding Weak Bisimilarity

- Checking whether $P \approx Q$ (or $P \approx^c Q$) over finite-state processes can be reduced to checking strong bisimilarity $\sim$, using a technique called saturation.
- Intuitively, saturation amounts to:
  1. pre-computing the weak transition relation $\stackrel{\alpha}{\Longrightarrow}$ (for $\alpha \neq \tau$) based on given relation $\longrightarrow$
  2. constructing a new finite-state process by replacing original transitions with weak transitions

  (similarly to elimination of ε-transitions in ε-NFA; see following example)
- The question whether $P \approx Q$ now boils down to checking $\sim$ on the saturated processes.

Deciding Weak Bisimilarity

- Checking whether $P \approx Q$ (or $P \approx^c Q$) over finite-state processes can be reduced to checking strong bisimilarity $\sim$, using a technique called saturation.
- Intuitively, saturation amounts to:
  1. pre-computing the weak transition relation $\stackrel{\alpha}{\Longrightarrow}$ (for $\alpha \neq \tau$) based on given relation $\longrightarrow$
  2. constructing a new finite-state process by replacing original transitions with weak transitions

  (similarly to elimination of ε-transitions in ε-NFA; see following example)
- The question whether $P \approx Q$ now boils down to checking $\sim$ on the saturated processes.
- As computing $\Longrightarrow$ and $\sim$ can be done in polynomial time, $P \approx Q$ can also be checked in polynomial time.

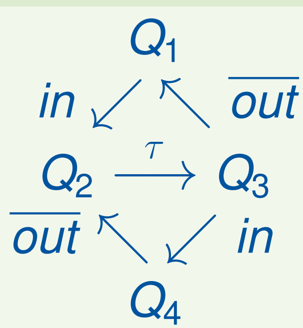
Deciding Weak Bisimilarity

Deciding Weak Bisimilarity

- Checking whether $P \approx Q$ (or $P \approx^c Q$) over finite-state processes can be reduced to checking strong bisimilarity $\sim$, using a technique called saturation.
- Intuitively, saturation amounts to:
  1. pre-computing the weak transition relation $\stackrel{\alpha}{\Longrightarrow}$ (for $\alpha \neq \tau$) based on given relation $\longrightarrow$
  2. constructing a new finite-state process by replacing original transitions with weak transitions

  (similarly to elimination of ε-transitions in ε-NFA; see following example)
- The question whether $P \approx Q$ now boils down to checking $\sim$ on the saturated processes.
- As computing $\Longrightarrow$ and $\sim$ can be done in polynomial time, $P \approx Q$ can also be checked in polynomial time.

Example 15.1 (Parallel two-place buffer; cf. Example 14.10)



(on the board)

Outline of Lecture 15

Bisimilarity and HML

- Weak and strong bisimilarity are based on mutually mimicking of processes.
- They possess the required properties of behavioural equivalences.[1]
- In particular, $\sim$ and $\approx^c$ are deadlock-sensitive CCS congruences.
- Hennessy-Milner Logic (HML) is a logic for expressing properties of processes.

[1] For weak bisimilarity, the notion of observation congruence was needed.

Bisimilarity and HML

- Weak and strong bisimilarity are based on mutually mimicking of processes.
- They possess the required properties of behavioural equivalences.[1]
- In particular, $\sim$ and $\approx^c$ are deadlock-sensitive CCS congruences.
- Hennessy-Milner Logic (HML) is a logic for expressing properties of processes.

Aim of this lecture

Study the connection between strong bisimilarity and satisfaction of HML formulae

[1] For weak bisimilarity, the notion of observation congruence was needed.

Verifying Correctness of Reactive Systems

Equivalence-checking approach

$$\mathit{Impl} \equiv \mathit{Spec}$$

- $\equiv$ is some equivalence, e.g., $\sim$ or $\approx^c$.
- *Spec* is often expressed in the same language as *Impl*, e.g., CCS.
- *Spec* provides the full specification of the intended behaviour.

Verifying Correctness of Reactive Systems

Equivalence-checking approach

$$\mathit{Impl} \equiv \mathit{Spec}$$

- $\equiv$ is some equivalence, e.g., $\sim$ or $\approx^c$.
- *Spec* is often expressed in the same language as *Impl*, e.g., CCS.
- *Spec* provides the full specification of the intended behaviour.

Model-checking approach

$$\mathit{Impl} \models \mathit{Prop}$$

- $\models$ is the satisfaction relation.
- *Prop* is a particular feature, often expressed via a logic, e.g., HML.
- *Prop* is a partial specification of the intended behaviour.

Outline of Lecture 15

Recap: HML

Definition (Syntax of HML; Definition 3.4) (Hennessy & Milner 1985)

The set *HMF* of Hennessy-Milner formulae over a set of actions *Act* is defined by the following syntax:

$$F ::= \mathrm{tt} \mid \mathrm{ff} \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle\alpha\rangle F \mid [\alpha]F$$

where $\alpha \in Act$.

Intuitive interpretation

- tt: all processes satisfy this property
- ff: no process satisfies this property
- $\wedge$, $\vee$: logical conjunction and disjunction
- $\langle\alpha\rangle F$: there is at least one α-successor that satisfies F
- $[\alpha]$: all α-successors have to satisfy F

Hennessy-Milner Logic

Recap: HML

Definition (Syntax of HML; Definition 3.4) (Hennessy & Milner 1985)

The set *HMF* of Hennessy-Milner formulae over a set of actions *Act* is defined by the following syntax:

$$F ::= \mathrm{tt} \mid \mathrm{ff} \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle\alpha\rangle F \mid [\alpha]F$$

where $\alpha \in Act$.

Intuitive interpretation

- tt: all processes satisfy this property
- ff: no process satisfies this property
- $\wedge$, $\vee$: logical conjunction and disjunction
- $\langle\alpha\rangle F$: there is at least one α-successor that satisfies F
- $[\alpha]$: all α-successors have to satisfy F

Lemma (HML and process traces; Lemma 3.9)

Let $(Prc, Act, \longrightarrow)$ be an LTS, and let $P, Q \in Prc$ satisfy the same HMF (i.e., $\forall F \in HMF : P \models F \iff Q \models F$). Then $Tr(P) = Tr(Q)$.

Outline of Lecture 15

Image-Finite Transition Systems

Definition 15.1 (Image-finiteness)

- A process $P \in \mathit{Prc}$ is image-finite iff the set $\{P' \in \mathit{Prc} \mid P \stackrel{\alpha}{\longrightarrow} P'\}$ is finite for every $\alpha \in \mathit{Act}$.
- A labelled transition system is image-finite if so is each of its states.

Image-Finite Transition Systems

Definition 15.1 (Image-finiteness)

- A process $P \in Prc$ is image-finite iff the set $\{P' \in Prc \mid P \xrightarrow{\alpha} P'\}$ is finite for every $\alpha \in Act$.
- A labelled transition system is image-finite if so is each of its states.

Example 15.2

1. The process $A_{rep} = a.\mathrm{nil} \parallel A_{rep}$ is not image-finite.
   By induction on n, one can prove that for each $n \in \mathbb{N}$:

$$A_{rep} \xrightarrow{a} \underbrace{a.\mathrm{nil} \parallel \cdots \parallel a.\mathrm{nil}}_{n \text{ times}} \parallel \mathrm{nil} \parallel A_{rep}$$

Image-Finite Transition Systems

Definition 15.1 (Image-finiteness)

- A process $P \in Prc$ is image-finite iff the set $\{P' \in Prc \mid P \xrightarrow{\alpha} P'\}$ is finite for every $\alpha \in Act$.
- A labelled transition system is image-finite if so is each of its states.

Example 15.2

1. The process $A_{rep} = a.\mathrm{nil} \parallel A_{rep}$ is not image-finite.
   By induction on n, one can prove that for each $n \in \mathbb{N}$:
   $$A_{rep} \xrightarrow{a} \underbrace{a.\mathrm{nil} \parallel \cdots \parallel a.\mathrm{nil}}_{n \text{ times}} \parallel \mathrm{nil} \parallel A_{rep}$$
2. Also the process $A^{<\omega} = \sum_{i \geq 0} a^i$ with $a^0 = \mathrm{nil}$ and $a^{i+1} = a.a^i$ is not image-finite:
   for each $i \geq 0$, $A^{<\omega} \xrightarrow{a} a^i$.

Relationship Between HML and Strong Bisimilarity

Theorem 15.3 (Hennessy-Milner Theorem)

Let $(Prc, Act, \{\xrightarrow{a} \mid a \in Act\})$ *be an image-finite LTS and* $P, Q \in Prc$. *Then:*

$$P \sim Q$$

if and only if

for every $F \in HMF : (P \models F \text{ iff } Q \models F)$.

Relationship Between HML and Strong Bisimilarity

Theorem 15.3 (Hennessy-Milner Theorem)

Let $(Prc, Act, \{\xrightarrow{a} \mid a \in Act\})$ *be an image-finite LTS and* $P, Q \in Prc$. *Then:*

$$P \sim Q$$

if and only if

$$\text{for every } F \in HMF : (P \models F \text{ iff } Q \models F)\,.$$

Proof.

on the board □

Correspondence of HML and Strong Bisimilarity

Relationship Between HML and Strong Bisimilarity

Theorem 15.3 (Hennessy-Milner Theorem)

Let $(Prc, Act, \{\xrightarrow{a} \mid a \in Act\})$ be an image-finite LTS and $P, Q \in Prc$. Then:

$$P \sim Q$$

if and only if

$$\text{for every } F \in HMF : (P \models F \text{ iff } Q \models F).$$

Proof.

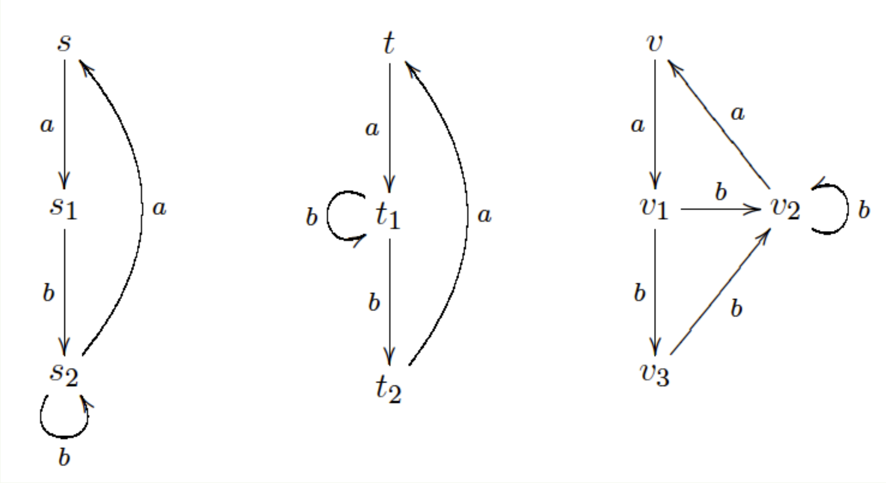
on the board □

Proving non-bisimilarity

Showing $P \not\sim Q$ thus amounts to finding a single HML-formula F with $P \models F$ and $Q \not\models F$.

Example

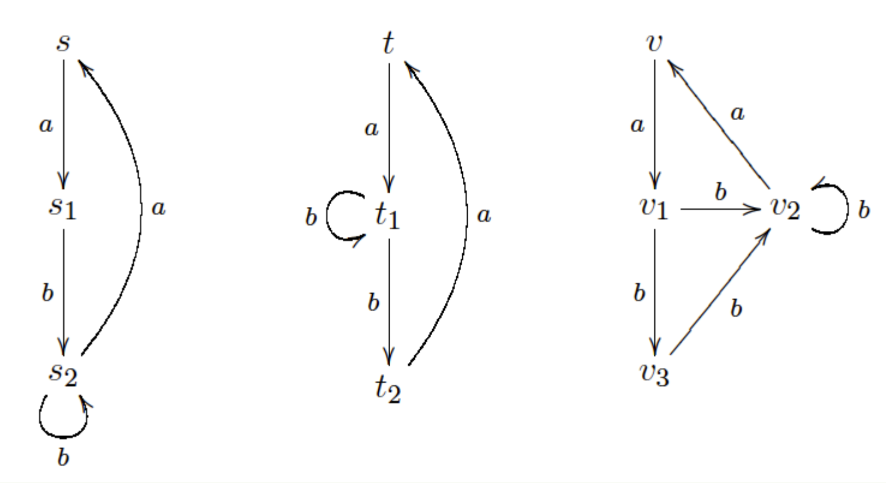
Example 15.4



It follows $s \not\sim t$ and $s \not\sim v$ and $t \not\sim v$.

Example

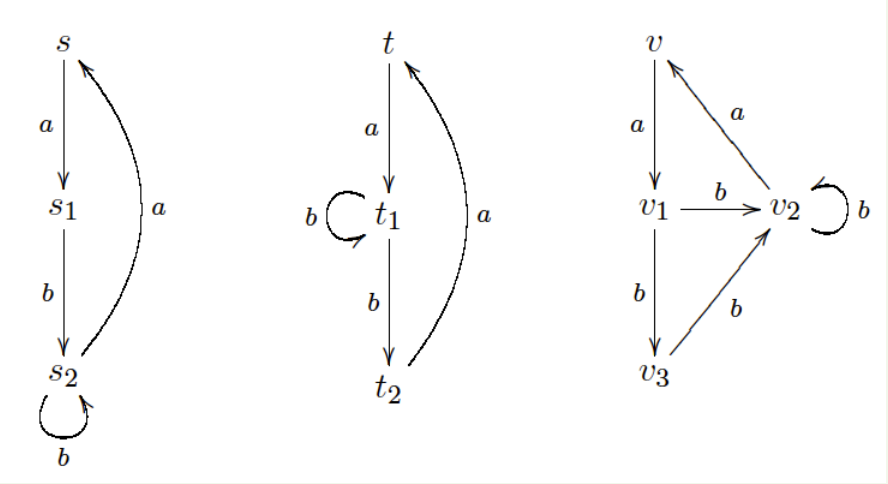
Example 15.4



It follows $s \not\sim t$ and $s \not\sim v$ and $t \not\sim v$. Distinguishing HML-formulas for:

Example

Example 15.4

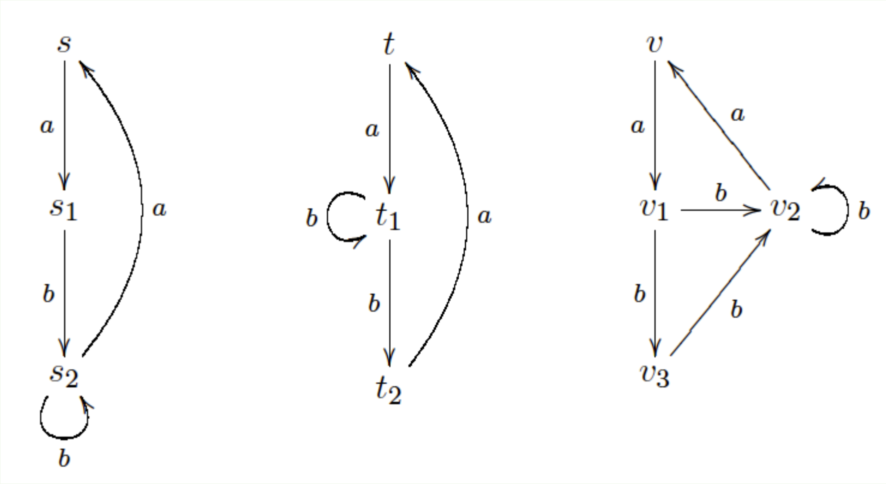


It follows $s \not\sim t$ and $s \not\sim v$ and $t \not\sim v$. Distinguishing HML-formulas for:

- $s \not\sim t$: $F = \langle a \rangle \langle b \rangle [b] \mathrm{ff}$ (as $t \models F$ and $s \not\models F$)

Example

Example 15.4



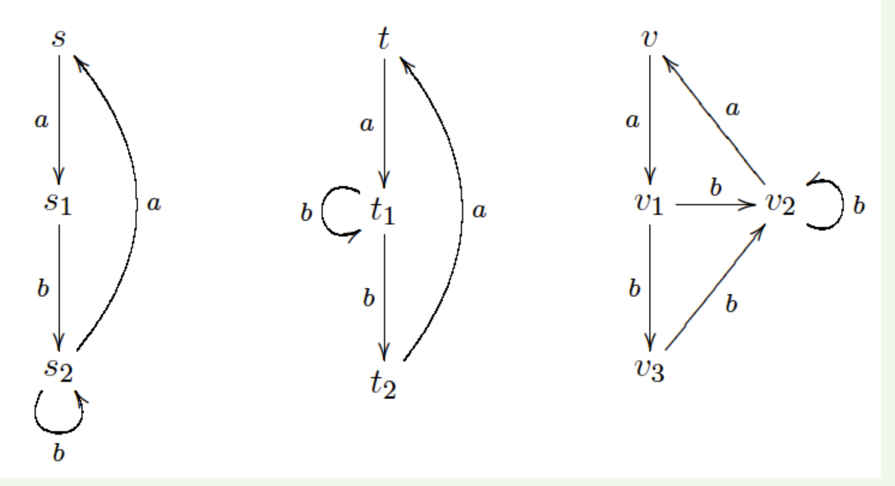
It follows $s \not\sim t$ and $s \not\sim v$ and $t \not\sim v$. Distinguishing HML-formulas for:

- $s \not\sim t$: $F = \langle a \rangle \langle b \rangle [b]\mathrm{ff}$ (as $t \models F$ and $s \not\models F$)
- $s \not\sim v$: $F = \langle a \rangle \langle b \rangle [a]\mathrm{ff}$ (as $v \models F$ and $s \not\models F$)

Correspondence of HML and Strong Bisimilarity

Example

Example 15.4



It follows $s \not\sim t$ and $s \not\sim v$ and $t \not\sim v$. Distinguishing HML-formulas for:

- $s \not\sim t$: $F = \langle a \rangle \langle b \rangle [b]\mathrm{ff}$ (as $t \models F$ and $s \not\models F$)
- $s \not\sim v$: $F = \langle a \rangle \langle b \rangle [a]\mathrm{ff}$ (as $v \models F$ and $s \not\models F$)
- $t \not\sim v$: $F = \langle a \rangle \langle b \rangle (\langle a \rangle \mathrm{tt} \wedge \langle b \rangle \mathrm{tt})$ (as $v \models F$ and $t \not\models F$)

Counterexample for Non Image-Finite Processes

Lemma 15.5

Let $A^{<\omega} = \sum_{i\geq 0} a^i$ *and* $A^\omega = a.A^\omega$. *Then* $A^{<\omega}$ *and* $A^{<\omega} + A^\omega$

1. *are not strongly bisimilar, but*
2. *satisfy the same HML formulae.*

Counterexample for Non Image-Finite Processes

Lemma 15.5

Let $A^{<\omega} = \sum_{i \geq 0} a^i$ *and* $A^\omega = a.A^\omega$. *Then* $A^{<\omega}$ *and* $A^{<\omega} + A^\omega$

1. *are not strongly bisimilar, but*
2. *satisfy the same HML formulae.*

Proof.

1. Assume that $A^{<\omega} \sim A^{<\omega} + A^\omega$. Then $A^{<\omega} + A^\omega \xrightarrow{a} A^\omega$ must be mimicked by $A^{<\omega} \xrightarrow{a} a^{i-1}$ for some $i \geq 1$. But obviously $A^\omega \not\sim a^{i-1}$.

Software Modeling and Verification Chair
RWTH AACHEN UNIVERSITY

Counterexample for Non Image-Finite Processes

Lemma 15.5

Let $A^{<\omega} = \sum_{i \geq 0} a^i$ and $A^\omega = a.A^\omega$. Then $A^{<\omega}$ and $A^{<\omega} + A^\omega$

1. *are not strongly bisimilar, but*
2. *satisfy the same HML formulae.*

Proof.

1. Assume that $A^{<\omega} \sim A^{<\omega} + A^\omega$. Then $A^{<\omega} + A^\omega \xrightarrow{a} A^\omega$ must be mimicked by $A^{<\omega} \xrightarrow{a} a^{i-1}$ for some $i \geq 1$. But obviously $A^\omega \not\sim a^{i-1}$.
2. by structural induction on $F \in HMF$, using the following lemma □

Counterexample for Non Image-Finite Processes

Lemma 15.5

Let $A^{<\omega} = \sum_{i \geq 0} a^i$ and $A^\omega = a.A^\omega$. Then $A^{<\omega}$ and $A^{<\omega} + A^\omega$

1. *are not strongly bisimilar, but*
2. *satisfy the same HML formulae.*

Proof.

1. Assume that $A^{<\omega} \sim A^{<\omega} + A^\omega$. Then $A^{<\omega} + A^\omega \xrightarrow{a} A^\omega$ must be mimicked by $A^{<\omega} \xrightarrow{a} a^{i-1}$ for some $i \geq 1$. But obviously $A^\omega \not\sim a^{i-1}$.
2. by structural induction on $F \in HMF$, using the following lemma □

Lemma 15.6

For every $F \in HMF$, $A^\omega \models F$ iff $a^k \models F$, where k is the modal depth[2] of F.

[2] the maximal number of nested occurrences of modal operators in F

Counterexample for Non Image-Finite Processes

Lemma 15.5

Let $A^{<\omega} = \sum_{i \geq 0} a^i$ and $A^\omega = a.A^\omega$. Then $A^{<\omega}$ and $A^{<\omega} + A^\omega$

1. *are not strongly bisimilar, but*
2. *satisfy the same HML formulae.*

Proof.

1. Assume that $A^{<\omega} \sim A^{<\omega} + A^\omega$. Then $A^{<\omega} + A^\omega \xrightarrow{a} A^\omega$ must be mimicked by $A^{<\omega} \xrightarrow{a} a^{i-1}$ for some $i \geq 1$. But obviously $A^\omega \not\sim a^{i-1}$.
2. by structural induction on $F \in HMF$, using the following lemma □

Lemma 15.6

For every $F \in HMF$, $A^\omega \models F$ iff $a^k \models F$, where k is the modal depth[2] of F.

Proof.

by structural induction on F □

[2]the maximal number of nested occurrences of modal operators in F

Outline of Lecture 15

Characteristic Formulae

- The Hennessy-Milner theorem asserts that for image-finite processes, strong bisimilarity and HML-equivalence coincide.

Characteristic Formulae

- The Hennessy-Milner theorem asserts that for image-finite processes, strong bisimilarity and HML-equivalence coincide.
- As a next step, we show that for finite transition systems, the equivalence classes under $\sim$ can be characterised by a system of formulae in HML extended with recursion – one for each state.

Characteristic Formulae

- The Hennessy-Milner theorem asserts that for image-finite processes, strong bisimilarity and HML-equivalence coincide.
- As a next step, we show that for finite transition systems, the equivalence classes under $\sim$ can be characterised by a system of formulae in HML extended with recursion – one for each state.
- For a finite process P, this HML-formula is called P's characteristic formula.

The Need for Recursion

Lemma 15.7

There is no recursion-free formula $F \in HMF$ that can characterise the process $A^\omega = a.A^\omega$ up to strong bisimilarity.

The Need for Recursion

Lemma 15.7

There is no recursion-free formula $F \in HMF$ that can characterise the process $A^\omega = a.A^\omega$ up to strong bisimilarity.

Proof.

- Assume $F \in HMF$ with $[\![F]\!] = \{P \mid P \sim A^\omega\}$.

The Need for Recursion

Lemma 15.7

There is no recursion-free formula $F \in HMF$ that can characterise the process $A^\omega = a.A^\omega$ up to strong bisimilarity.

Proof.

- Assume $F \in HMF$ with $[\![F]\!] = \{P \mid P \sim A^\omega\}$.
- Obviously $a^i \not\sim A^\omega$ for every $i \geq 0$.

The Need for Recursion

Lemma 15.7

There is no recursion-free formula $F \in HMF$ *that can characterise the process* $A^\omega = a.A^\omega$ *up to strong bisimilarity.*

Proof.

- Assume $F \in HMF$ with $[\![F]\!] = \{P \mid P \sim A^\omega\}$.
- Obviously $a^i \not\sim A^\omega$ for every $i \geq 0$.
- On the other hand, $A^\omega \models F$ implies (by Lemma 15.6) that $a^k \models F$ where k is the modal depth of F.

The Need for Recursion

Lemma 15.7

There is no recursion-free formula $F \in HMF$ *that can characterise the process* $A^\omega = a.A^\omega$ *up to strong bisimilarity.*

Proof.

- Assume $F \in HMF$ with $[\![F]\!] = \{P \mid P \sim A^\omega\}$.
- Obviously $a^i \not\sim A^\omega$ for every $i \geq 0$.
- On the other hand, $A^\omega \models F$ implies (by Lemma 15.6) that $a^k \models F$ where k is the modal depth of F.
- Thus, $a^k \sim A^\omega$, which contradicts $a^i \not\sim A^\omega$. □

Recap: HML with Recursion

Definition (Syntax of mutually recursive equational systems; Def. 6.4)

Let $\mathcal{X} = \{X_1, \ldots, X_n\}$ be a set of variables. The set $HMF_{\mathcal{X}}$ of Hennessy-Milner formulae over $\mathcal{X}$ is defined by the following syntax:

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

where $1 \leq i \leq n$ and $\alpha \in Act$. A mutually recursive equational system has the form

$$(X_i = F_{X_i} \mid 1 \leq i \leq n)$$

where $F_{X_i} \in HMF_{\mathcal{X}}$ for every $1 \leq i \leq n$.

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.
- Intuitively, X_P is the syntactic symbol for the characteristic formula of process $P \in S$.

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.
- Intuitively, X_P is the syntactic symbol for the characteristic formula of process $P \in S$.
- A characteristic formula for P has to describe
  - which actions P can perform,
  - which actions it cannot perform and
  - what happens after performing an action.

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.
- Intuitively, X_P is the syntactic symbol for the characteristic formula of process $P \in S$.
- A characteristic formula for P has to describe
  - which actions P can perform,
  - which actions it cannot perform and
  - what happens after performing an action.

Example 15.8 (Coffee/tea machine; cf. Example 11.10)

$$M = m.M' \qquad M' = \overline{c}.M + \overline{t}.M$$

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.
- Intuitively, X_P is the syntactic symbol for the characteristic formula of process $P \in S$.
- A characteristic formula for P has to describe
  - which actions P can perform,
  - which actions it cannot perform and
  - what happens after performing an action.

Example 15.8 (Coffee/tea machine; cf. Example 11.10)

$$M = m.M' \qquad M' = \overline{c}.M + \overline{t}.M$$

Observations:

1. M can perform m and become M'
2. Performing m, M necessarily becomes M'
3. M cannot perform any other action than m

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.
- Intuitively, X_P is the syntactic symbol for the characteristic formula of process $P \in S$.
- A characteristic formula for P has to describe
  - which actions P can perform,
  - which actions it cannot perform and
  - what happens after performing an action.

Example 15.8 (Coffee/tea machine; cf. Example 11.10)

$$M = m.M' \qquad M' = \overline{c}.M + \overline{t}.M$$

Observations:

1. M can perform m and become M'
2. Performing m, M necessarily becomes M'
3. M cannot perform any other action than m

Therefore:

$$X_M = \langle m \rangle X_{M'} \wedge [m] X_{M'} \wedge [\{\overline{c}, \overline{t}\}]\mathrm{ff}$$

Characteristic Formulae by Example

- Consider the finite LTS $(S, Act, \longrightarrow)$, and let $\mathcal{X}$ contain (at least) $|S|$ variables.
- Intuitively, X_P is the syntactic symbol for the characteristic formula of process $P \in S$.
- A characteristic formula for P has to describe
  - which actions P can perform,
  - which actions it cannot perform and
  - what happens after performing an action.

Example 15.8 (Coffee/tea machine; cf. Example 11.10)

$$M = m.M' \qquad M' = \overline{c}.M + \overline{t}.M$$

Observations:

1. M can perform m and become M'
2. Performing m, M necessarily becomes M'
3. M cannot perform any other action than m

Therefore:

$$X_M = \langle m \rangle X_{M'} \wedge [m] X_{M'} \wedge [\{\overline{c}, \overline{t}\}]\mathrm{ff}$$

Similarly:

$$X_{M'} = \langle \overline{c} \rangle X_M \wedge \langle \overline{t} \rangle X_M \wedge [\{\overline{c}, \overline{t}\}] X_M \wedge [m]\mathrm{ff}$$

Characteristic Formulae

The General Case

Enabled actions: $P \models \bigwedge_{\{\alpha, P' \mid P \xrightarrow{\alpha} P'\}} \langle\alpha\rangle X_{P'}$

Resulting states: $P \models \bigwedge_{\{\alpha \mid P \xrightarrow{\alpha}\}} [\alpha] \left(\bigvee_{\{P' \mid P \xrightarrow{\alpha} P'\}} X_{P'}\right)$

Disabled actions: $P \models \bigwedge_{\{\alpha \mid P \not\xrightarrow{\alpha}\}} [\alpha]\mathrm{ff}$

The General Case

Enabled actions: $P \models \bigwedge_{\{\alpha, P' \mid P \xrightarrow{\alpha} P'\}} \langle\alpha\rangle X_{P'}$

Resulting states: $P \models \bigwedge_{\{\alpha \mid P \xrightarrow{\alpha}\}} [\alpha] \left(\bigvee_{\{P' \mid P \xrightarrow{\alpha} P'\}} X_{P'} \right)$

Disabled actions: $P \models \bigwedge_{\{\alpha \mid P \not\xrightarrow{\alpha}\}} [\alpha]\mathrm{ff}$

can be combined!

The General Case

Enabled actions: $P \models \bigwedge_{\{\alpha,P' \mid P \xrightarrow{\alpha} P'\}} \langle\alpha\rangle X_{P'}$

Resulting states: $P \models \bigwedge_{\{\alpha \mid P \xrightarrow{\alpha}\}} [\alpha] \left(\bigvee_{\{P' \mid P \xrightarrow{\alpha} P'\}} X_{P'} \right)$

Disabled actions: $P \models \bigwedge_{\{\alpha \mid P \not\xrightarrow{\alpha}\}} [\alpha]\mathrm{ff}$

can be combined!

Theorem 15.9 (Characteristic Formula) (Ingolfsdottir et al. 1987)

For a finite-state process $P \in Prc$, let the characteristic formula $X_P \in HMF_{\mathcal{X}}$ be defined by:

$$X_P \overset{max}{=} \bigwedge_{\{\alpha,P' \mid P \xrightarrow{\alpha} P'\}} \langle\alpha\rangle X_{P'} \wedge \bigwedge_{\alpha \in Act} [\alpha] \left(\bigvee_{\{P' \mid P \xrightarrow{\alpha} P'\}} X_{P'} \right)$$

Then, for every $Q \in Prc$: $Q \models X_P$ iff $P \sim Q$.

The General Case

Enabled actions: $P \models \bigwedge_{\{\alpha, P' \mid P \xrightarrow{\alpha} P'\}} \langle\alpha\rangle X_{P'}$

Resulting states: $P \models \bigwedge_{\{\alpha \mid P \xrightarrow{\alpha}\}} [\alpha] \left(\bigvee_{\{P' \mid P \xrightarrow{\alpha} P'\}} X_{P'} \right)$

Disabled actions: $P \models \bigwedge_{\{\alpha \mid P \not\xrightarrow{\alpha}\}} [\alpha]\mathrm{ff}$

can be combined!

Theorem 15.9 (Characteristic Formula) (Ingolfsdottir et al. 1987)

For a finite-state process $P \in Prc$, let the characteristic formula $X_P \in HMF_{\mathcal{X}}$ be defined by:

$$X_P \stackrel{max}{=} \bigwedge_{\{\alpha, P' \mid P \xrightarrow{\alpha} P'\}} \langle\alpha\rangle X_{P'} \wedge \bigwedge_{\alpha \in Act} [\alpha] \left(\bigvee_{\{P' \mid P \xrightarrow{\alpha} P'\}} X_{P'} \right)$$

Then, for every $Q \in Prc$: $Q \models X_P$ iff $P \sim Q$.

Proof.

omitted □

Outline of Lecture 15

Recap: Weak Bisimilarity

Deciding Weak Bisimilarity

Bisimilarity and HML

Hennessy-Milner Logic

Correspondence of HML and Strong Bisimilarity

Characteristic Formulae

Summary

Summary

- Strong bisimilarity and HML-equivalence coincide for image-finite processes.
- This result does not hold for processes that are not image-finite.
- For each finite-state process, a recursive HML-formula does exist that precisely characterises the strongly bisimilar processes.