Modelling and Validation of Concurrent System

Hennessy-Milner Logic

António Ravara

May 9, 2024

Motivation

Properties of Concurrent/Reactive Systems

We have proposed:

- a language to define concurrent systems;
- an equivalence notion to equate systems with the same behaviour.

Properties of Concurrent/Reactive Systems

We have proposed:

- a language to define concurrent systems;
- an equivalence notion to equate systems with the same behaviour.

We still need a logic to specify behavioural *properties*.

Example

Consider a shared one-place buffer: $\mathit{Buf}\,1 = \mathit{in}(x).\overline{\mathit{out}}\langle x\rangle.\mathit{Buf}\,1$.

Properties of Concurrent/Reactive Systems

We have proposed:

- a language to define concurrent systems;
- an equivalence notion to equate systems with the same behaviour.

We still need a logic to specify behavioural *properties*.

Example

Consider a shared one-place buffer: $Buf1 = in(x).\overline{out}\langle x \rangle.Buf1$.

How can one guarantee that:

- after an *in* there is always an *out*;
- no *in* (or *out*) follows an *in* (or *out*).

Temporal Logics - behaviour over time

- In ABP, no message is undelivered.
  It is never the case that something bad happens.
  Safety.

Temporal Properties of Concurrent/Reactive Systems

Temporal Logics - behaviour over time

- In ABP, no message is undelivered.
  It is never the case that something bad happens.
  Safety.
- In ABP, every message will be delivered.
  Eventually, something good will happen.
  Liveness.

Temporal Properties of Concurrent/Reactive Systems

Temporal Logics - behaviour over time

- In ABP, no message is undelivered.
  It is never the case that something bad happens.
  Safety.
- In ABP, every message will be delivered.
  Eventually, something good will happen.
  Liveness.

How can one talk about "simpler" properties?

- The coffee machine requires a coin before selecting the beverage.
- After putting a coin, one can choose between tea and coffee.

Modal Properties of Concurrent/Reactive Systems

How to talk (sequences of) actions?

- Put a coin before selecting tea or coffee.
- Have either coffee or tea (but not both) after putting a coin.

What we want to express

In a given moment, a system:

- *may* do something
  (and then continue with some other behaviour);

Modal Properties of Concurrent/Reactive Systems

How to talk (sequences of) actions?

- Put a coin before selecting tea or coffee.
- Have either coffee or tea (but not both) after putting a coin.

What we want to express

In a given moment, a system:

- *may* do something
  (and then continue with some other behaviour);
- *must* do something
  (and then continue with some other behaviour).

We want to talk about (sequences of) *possible* and *necessary* actions.

Modal Logics

From Wikipedia

A modal – a word that expresses a modality – qualifies a statement.

Modalities of truth

Possibility and necessity.

Modal Logics

From Wikipedia

A modal – a word that expresses a modality – qualifies a statement.

Modalities of truth

Possibility and necessity.

What may, must, and cannot happen.

Modal Logics

From Wikipedia

A modal – a word that expresses a modality – qualifies a statement.

Modalities of truth

Possibility and necessity.

What may, must, and cannot happen.

Ingredients

1. Considers *actions*, Instead of propositional variables.
2. Uses *propositional logic* connectives.
3. Uses *modalities of truth*.

Hennessy-Milner Logic

Hennessy-Milner Logic - syntax

Consider $a \in \mathrm{Act}$.

Syntax

The set $\mathcal{F}$ of modal formulæ is inductively defined by the following grammar.

$$\varphi ::= \bot \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle a \rangle \varphi$$

Hennessy-Milner Logic - syntax

Consider $a \in \mathrm{Act}$.

Syntax

The set $\mathcal{F}$ of modal formulæ is inductively defined by the following grammar.

$$\varphi ::= \bot \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle a \rangle \varphi$$

Intuitive meaning

$\bot$ denotes the absurdity

$\neg$ **and** $\wedge$ denote the usual propositional connectives

$\langle a \rangle$ denotes the possibility of performing action a

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.

Let $\models \subseteq \mathrm{Proc} \times \mathcal{F}$ be a *satisfaction relation*, *i.e.*, a set of pairs.

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.
Let $\models \subseteq \mathrm{Proc} \times \mathcal{F}$ be a *satisfaction relation*, *i.e.*, a set of pairs.
We write $P \models \varphi$ if $(P, \varphi) \in \models$
(and $P \not\models \varphi$ if $(P, \varphi) \notin \models$).

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.

Let $\models \subseteq \mathrm{Proc} \times \mathcal{F}$ be a *satisfaction relation*, *i.e.*, a set of pairs.

We write $P \models \varphi$ if $(P, \varphi) \in \models$

(and $P \not\models \varphi$ if $(P, \varphi) \notin \models$).

Satisfaction Semantics

The relation $\models$ is inductively defined by the rules below.

$$P \not\models \bot \;\; \textit{i.e.}, \forall P \in \mathrm{Proc}.\, (P, \bot) \notin \models$$

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.
Let $\models \subseteq \mathrm{Proc} \times \mathcal{F}$ be a *satisfaction relation*, *i.e.*, a set of pairs.
We write $P \models \varphi$ if $(P, \varphi) \in \models$
(and $P \not\models \varphi$ if $(P, \varphi) \notin \models$).

Satisfaction Semantics

The relation $\models$ is inductively defined by the rules below.

$$P \not\models \bot \quad \textit{i.e.}, \forall P \in \mathrm{Proc}.\, (P, \bot) \notin \models$$

$$P \models \neg\varphi \ \text{ if } P \not\models \varphi$$

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.
Let $\models \subseteq \mathrm{Proc} \times \mathcal{F}$ be a *satisfaction relation*, *i.e.*, a set of pairs.
We write $P \models \varphi$ if $(P, \varphi) \in \models$
(and $P \not\models \varphi$ if $(P, \varphi) \notin \models$).

Satisfaction Semantics

The relation $\models$ is inductively defined by the rules below.

$$
\begin{aligned}
P \not\models \bot \quad & \textit{i.e.}, \forall P \in \mathrm{Proc}.\, (P, \bot) \notin \models \\
P \models \neg\varphi \quad & \text{if } P \not\models \varphi \\
P \models \varphi \wedge \psi \quad & \text{if } P \models \varphi \text{ and } P \models \psi
\end{aligned}
$$

Hennessy-Milner Logic - semantics

Consider $a \in \mathrm{Act}$.
Let $\models\ \subseteq \mathrm{Proc} \times \mathcal{F}$ be a *satisfaction relation*, *i.e.*, a set of pairs.
We write $P \models \varphi$ if $(P, \varphi) \in\ \models$
(and $P \not\models \varphi$ if $(P, \varphi) \notin\ \models$).

Satisfaction Semantics

The relation $\models$ is inductively defined by the rules below.

$$\begin{aligned}
P &\not\models \bot \ \ \textit{i.e.}, \forall P \in \mathrm{Proc}.\,(P, \bot) \notin\ \models \\
P &\models \neg\varphi \ \text{ if } P \not\models \varphi \\
P &\models \varphi \land \psi \ \text{ if } P \models \varphi \text{ and } P \models \psi \\
P &\models \langle a \rangle \varphi \ \text{ if } \exists\, Q. P \xrightarrow{a} Q \text{ and } Q \models \varphi
\end{aligned}$$

Hennessy-Milner Logic - semantics

Satisfaction Semantics

The relation $\models$ is inductively defined by the rules below.

$$
\begin{aligned}
P &\not\models \bot \;\; \textit{i.e.}, \forall P \in \mathrm{Proc.}\,(P, \bot) \notin \models \\
P &\models \neg\varphi \;\text{ if } P \not\models \varphi \\
P &\models \varphi \wedge \psi \;\text{ if } P \models \varphi \text{ and } P \models \psi \\
P &\models \langle a \rangle \varphi \;\text{ if } \exists\, Q.P \xrightarrow{a} Q \text{ and } Q \models \varphi
\end{aligned}
$$

Abbreviations

$$
\begin{aligned}
\top &\;\text{ abbreviates } \neg\bot \\
\varphi \vee \psi &\;\text{ abbreviates } \neg(\neg\varphi \wedge \neg\psi) \\
\varphi \rightarrow \psi &\;\text{ abbreviates } \neg\varphi \vee \psi \\
[a]\varphi &\;\text{ abbreviates } \neg\langle a \rangle \neg\varphi
\end{aligned}
$$

Specifying the intended behaviour of a semaphor

Consider a Semaphor controlling the access in mutual exclusion to a resource *crit*.

Let the system include three processes whishing access to the resource.

Specifying the intended behaviour of a semaphor

Consider a Semaphor controlling the access in mutual exclusion to a resource *crit*.

Let the system include three processes whishing access to the resource.

$$\begin{aligned} Sem &= get.crit.put.Sem \\ Prc_i &= \overline{get}.\overline{crit}.\overline{put} \\ System &= (\mathbf{new}\, get, put)(Sem \mid Prc_1 \mid Prc_2 \mid Prc_3) \end{aligned}$$

Specifying the intended behaviour of a semaphor

Consider a Semaphor controlling the access in mutual exclusion to a resource *crit*.

Let the system include three processes whishing access to the resource.

$$Sem = get.crit.put.Sem$$
$$Prc_i = \overline{get}.\overline{crit}.\overline{put}$$
$$System = (\mathbf{new}\, get, put)(Sem \mid Prc_1 \mid Prc_2 \mid Prc_3)$$

Properties, and their meaning

$Sem \models \langle get \rangle \top$ says *Sem* may do *get*

$Sem \models [put] \bot$ says *Sem* cannot do *put*

$System \models [\tau]\langle crit \rangle \top$ says *System* must do an internal action to release *crit*

Proving $\mathit{System} \models [\tau]\langle \mathit{crit}\rangle \top$

Recall that

$$\mathit{Sem} = \mathit{get}.\mathit{crit}.\mathit{put}.\mathit{Sem} \qquad \mathit{Prc}_i = \overline{\mathit{get}}.\overline{\mathit{crit}}.\overline{\mathit{put}}.\mathit{Prc}_i$$

$$\mathit{System} = (\mathbf{new}\, \mathit{get}, \mathit{put})(\mathit{Sem} \mid \mathit{Prc}_1 \mid \mathit{Prc}_2 \mid \mathit{Prc}_3)$$

Consider $\mathit{Sys}_1 = (\mathbf{new}\, \mathit{get}, \mathit{put})(\mathit{Locked} \mid \overline{\mathit{crit}}.\overline{\mathit{put}}.\mathit{Prc}_1 \mid \mathit{Prc}_2 \mid \mathit{Prc}_3)$
where $\mathit{Locked} = \mathit{crit}.\mathit{put}.\mathit{Sem}$ (Sys_2 and Sys_3 are similar).

Proving $System \models [\tau]\langle crit\rangle\top$

Recall that

$$Sem = get.crit.put.Sem \qquad Prc_i = \overline{get}.\overline{crit}.\overline{put}.Prc_i$$

$$System = (\textbf{new}\, get, put)(Sem \mid Prc_1 \mid Prc_2 \mid Prc_3)$$

Consider $Sys_1 = (\textbf{new}\, get, put)(Locked \mid \overline{crit}.\overline{put}.Prc_1 \mid Prc_2 \mid Prc_3)$ where $Locked = crit.put.Sem$ (Sys_2 and Sys_3 are similar).

$$\begin{aligned}
& System \models [\tau]\langle crit\rangle\top \\
\textit{iff} \quad & \forall\, P \in \{Q \,.\, System \xrightarrow{\tau} Q\} \,.\, P \models \langle crit\rangle\top \\
\textit{iff} \quad & \forall\, P \in \{Sys_1, Sys_2, Sys_3\} \,.\, P \models \langle crit\rangle\top \\
\textit{iff} \quad & \forall\, i \in \{1,2,3\} \exists\, P \in \{Q \,.\, Sys_i \xrightarrow{crit} Q\} \,.\, P \models \top \\
\textit{iff} \quad & \forall\, i \in \{1,2,3\} Sys_i \xrightarrow{crit} Sys_i' \wedge Sys_i' \models \top, \text{ which holds making} \\
& Sys_i' = (\textbf{new}\, get, put)(put.Sem \mid \overline{crit}.\overline{put}.Prc_1 \mid Prc_2 \mid Prc_3)
\end{aligned}$$

Useful formulæ

Let $a, b \in \alpha$ and $\mathcal{A} \subseteq \alpha$.

$$
\begin{aligned}
\langle a, b\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi \wedge \langle b\rangle\varphi \\
\langle \mathcal{A}\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi, \text{ for any } a \in \mathcal{A} \\
\langle -a\rangle\varphi &\text{ abbreviates } \langle c\rangle\varphi, \text{ for any } c \in \alpha \setminus \{a\} \\
\langle -\mathcal{A}\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi, \text{ for any } a \in \alpha \setminus \mathcal{A} \\
\langle -\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi, \text{ for any } a \in \alpha \setminus \emptyset.
\end{aligned}
$$

Useful formulæ

Let $a, b \in \alpha$ and $\mathcal{A} \subseteq \alpha$.

$$\begin{aligned}
\langle a, b\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi \wedge \langle b\rangle\varphi \\
\langle \mathcal{A}\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi, \text{ for any } a \in \mathcal{A} \\
\langle -a\rangle\varphi &\text{ abbreviates } \langle c\rangle\varphi, \text{ for any } c \in \alpha \setminus \{a\} \\
\langle -\mathcal{A}\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi, \text{ for any } a \in \alpha \setminus \mathcal{A} \\
\langle -\rangle\varphi &\text{ abbreviates } \langle a\rangle\varphi, \text{ for any } a \in \alpha \setminus \emptyset.
\end{aligned}$$

Patterns

$\langle -\rangle\top$ means *some action may happen*.

$[-]\bot$ means *no action can happen*.

$\langle -\rangle\top \wedge [-a]\bot$ means *only action a can happen*.

$\langle -\rangle\top \wedge [-]\varphi$ means φ *holds after one step*.

Properties: logical equivalence

Terminated processes behave like deadlocks

$$P \models [-]\bot \text{ if } P \equiv 0$$

Properties: logical equivalence

Terminated processes behave like deadlocks

$$P \models [-]\bot \text{ if } P \equiv 0$$

Processes that satisfy the same formulæ are equivalent.

Logical Equivalence of Processes

$P \sim_l Q$, if $\forall \varphi \in \mathcal{F} . \mathcal{P} \models \varphi$ if and only if $Q \models \varphi$

Properties: logical equivalence

Terminated processes behave like deadlocks

$$P \models [-]\bot \text{ if } P \equiv 0$$

Processes that satisfy the same formulæ are equivalent.

Logical Equivalence of Processes

$P \sim_l Q$, if $\forall \varphi \in \mathcal{F} \,.\, \mathcal{P} \models \varphi$ if and only if $Q \models \varphi$

In turn, formulæ that satisfy the same processes are equivalent.

Logical Equivalence of Formulæ

$\varphi \sim_l \psi$, if $\forall P \,.\, P \models \varphi$ if and only if $P \models \psi$

Properties: relationship between logical equivalence and bisimulation

Finitely branching processes

P is finitely branching, if $\forall\, a \in \mathrm{Act} \,.\{Q \,.\, P \xrightarrow{a} Q\}$ is finite

Properties: relationship between logical equivalence and bisimulation

Finitely branching processes

P is finitely branching, if $\forall\, a \in \mathrm{Act}\,.\{Q\,.\,P \xrightarrow{a} Q\}$ is finite

Proposition If P and Q are finitely branching and $P \sim_l Q$ then $P \sim Q$

Properties: relationship between logical equivalence and bisimulation

Finitely branching processes

P is finitely branching, if $\forall\, a \in \mathrm{Act}\,.\{Q\,.\,P \xrightarrow{a} Q\}$ is finite

Proposition If P and Q are finitely branching and $P \sim_l Q$ then $P \sim Q$

Proposition If $P \sim Q$ then $P \sim_l Q$

Observational equivalence in HML

How to express observational properties?

one needs to abstract away silent actions.

Observational equivalence in HML

How to express observational properties?

one needs to abstract away silent actions.

Eventual possibility and necessity, after idle activity

$$P \models \langle\langle\rangle\rangle\varphi \quad , \text{ if } \exists\, Q \,.\, P \stackrel{\tau}{\Longrightarrow} Q \text{ and } Q \models \varphi$$

$$P \models [[\,]]\varphi \quad , \text{ if } \forall\, Q \in \{P' \,.\, P \stackrel{\tau}{\Longrightarrow} P'\} \,.\, Q \models \varphi$$

Observational equivalence in HML

How to express observational properties?

one needs to abstract away silent actions.

Eventual possibility and necessity, after idle activity

$$P \models \langle\!\langle\rangle\!\rangle\varphi \quad , \text{ if } \exists\, Q \,.\, P \stackrel{\tau}{\Longrightarrow} Q \text{ and } Q \models \varphi$$

$$P \models [\![]\!]\varphi \quad , \text{ if } \forall\, Q \in \{P' \,.\, P \stackrel{\tau}{\Longrightarrow} P'\} \,.\, Q \models \varphi$$

Consider $\mathcal{A} \subseteq \alpha$.

$\langle\!\langle\mathcal{A}\rangle\!\rangle\varphi$ abbreviates $\langle\!\langle\rangle\!\rangle\langle\mathcal{A}\rangle\langle\!\langle\rangle\!\rangle\varphi$

$[\![\mathcal{A}]\!]\varphi$ abbreviates $[\![]\!][\mathcal{A}][\![]\!]\varphi$

Observational equivalence in HML

How to express observational properties?

one needs to abstract away silent actions.

Eventual possibility and necessity, after idle activity

$$P \models \langle\langle\rangle\rangle\varphi \quad , \text{ if } \exists\, Q \,.\, P \overset{\tau}{\Longrightarrow} Q \text{ and } Q \models \varphi$$

$$P \models [\![]\!]\varphi \quad , \text{ if } \forall\, Q \in \{P' \,.\, P \overset{\tau}{\Longrightarrow} P'\} \,.\, Q \models \varphi$$

Consider $\mathcal{A} \subseteq \alpha$.

$\langle\langle\mathcal{A}\rangle\rangle\varphi$ abbreviates $\langle\langle\rangle\rangle\langle\mathcal{A}\rangle\langle\langle\rangle\rangle\varphi$

$[\![\mathcal{A}]\!]\varphi$ abbreviates $[\![]\!][\mathcal{A}][\![]\!]\varphi$

Examples

$\langle\langle a_1\rangle\rangle \cdots \langle\langle a_n\rangle\rangle\top$ represents the possibility of performing the sequence of observable actions $a_1 \cdots a_n$

$[\![-]\!]\bot$ represents the absence of observable behaviour.

Convergence in HML

How to express the necessity of observing an action?

$\langle\!\langle - \rangle\!\rangle \top \wedge [\![-a]\!] \bot$ is not exactly what one wants, as it is satisfiable by $A = a.A + \tau.0$

Convergence in HML

How to express the necessity of observing an action?

$\langle\langle - \rangle\rangle \top \wedge [\![-a]\!] \bot$ is not exactly what one wants, as it is satisfiable by $A = a.A + \tau.0$

$[\![-]\!] \langle\langle - \rangle\rangle \top \wedge [\![-a]\!] \bot$ requires observable transactions to happen. Now the process A above does not satisfy it.

Convergence in HML

How to express the necessity of observing an action?

$\langle\langle - \rangle\rangle \top \wedge [\![-a]\!] \bot$ is not exactly what one wants, as it is satisfiable by $A = a.A + \tau.0$

$[\![-]\!] \langle\langle - \rangle\rangle \top \wedge [\![-a]\!] \bot$ requires observable transactions to happen. Now the process A above does not satisfy it. However, is is still not exactly what one wants, as it is satisfiable by $S = a.S + \tau.S$

Convergence in HML

How to express the necessity of observing an action?

$\langle\!\langle - \rangle\!\rangle \top \wedge [\![-a]\!] \bot$ is not exactly what one wants, as it is satisfiable by $A = a.A + \tau.0$

$[\![-]\!] \langle\!\langle - \rangle\!\rangle \top \wedge [\![-a]\!] \bot$ requires observable transactions to happen.
Now the process A above does not satisfy it.
However, is is still not exactly what one wants, as it is satisfiable by $S = a.S + \tau.S$

The problem is $[\![-]\!] \langle\!\langle - \rangle\!\rangle \top$ is satisfied by a *divergent* process.

Convergence in HML

Convergent processes

A process P that cannot perform an infinite sequence of silent actions is said *convergent*, denoted $P\downarrow$.

Convergence in HML

Convergent processes

A process P that cannot perform an infinite sequence of silent actions is said *convergent*, denoted $P\downarrow$.

Convergence Modality

$P \models [\![\downarrow]\!]\varphi$ if $P\downarrow$ and $\forall\, Q \in \{P' \,.\, P \stackrel{\tau}{\Longrightarrow} P'\} \,.\, Q \models \varphi$.

Convergence in HML

Convergent processes

A process P that cannot perform an infinite sequence of silent actions is said *convergent*, denoted $P{\downarrow}$.

Convergence Modality

$P \models [\![{\downarrow}]\!]\varphi$ if $P{\downarrow}$ and $\forall\, Q \in \{P' \,.\, P \overset{\tau}{\Longrightarrow} P'\} \,.\, Q \models \varphi$.

Action a must happen

$[\![{\downarrow}]\!]\langle\!\langle - \rangle\!\rangle\top \wedge [\![-a]\!]\bot$

Convergence in HML

Convergent processes

A process P that cannot perform an infinite sequence of silent actions is said *convergent*, denoted $P\downarrow$.

Convergence Modality

$P \models [\![\downarrow]\!]\varphi$ if $P\downarrow$ and $\forall\, Q \in \{P' \,.\, P \overset{\tau}{\Longrightarrow} P'\} \,.\, Q \models \varphi$.

Action a must happen

$[\![\downarrow]\!]\langle\!\langle -\rangle\!\rangle\top \wedge [\![-a]\!]\bot$

$$\begin{aligned}
[\![\downarrow\mathcal{A}]\!]\varphi &\text{ abbreviates } [\![\downarrow]\!]\langle\mathcal{A}\rangle[\![]\!]\varphi \\
[\![\mathcal{A}\downarrow]\!]\varphi &\text{ abbreviates } [\![]\!]\langle\mathcal{A}\rangle[\![\downarrow]\!]\varphi \\
[\![\downarrow\mathcal{A}\downarrow]\!]\varphi &\text{ abbreviates } [\![\downarrow]\!]\langle\mathcal{A}\rangle[\![\downarrow]\!]\varphi
\end{aligned}$$