

Logics for processes (I)

Luís S. Barbosa

DI-CCTC
Universidade do Minho
Braga, Portugal

April, 2011

Motivation

System's correctness wrt a specification

- equivalence checking (between two designs), through \sim and $=$
- unsuitable to check properties such as

can the system perform action α followed by β ?

which are best answered by exploring the process state space

Motivation

The taxi network example

- $\phi_0 =$ *In a taxi network, a car can collect a passenger or be allocated by the Central to a pending service*
- $\phi_1 =$ *This applies only to cars already on service*
- $\phi_2 =$ *If a car is allocated to a service, it must first collect the passenger and then plan the route*
- $\phi_3 =$ *On detecting an emergence the taxi becomes inactive*
- $\phi_4 =$ *A car on service is not inactive*

Motivation

The taxi network example

- $\phi_0 = \langle rec, alo \rangle \text{true}$
- $\phi_1 = [onservice] \langle rec, alo \rangle \text{true}$ or
 $\phi_1 = [onservice] \phi_0$
- $\phi_2 = [alo] \langle rec \rangle \langle plan \rangle \text{true}$
- $\phi_3 = [sos] [-] \text{false}$
- $\phi_4 = [onservice] \langle - \rangle \text{true}$

Notes

- Modalities: $\langle K \rangle \phi$, $[L] \psi$ for $K, L \subset Act$
- Valuations in non modal logics are based on valuations
 $V : \text{Variables} \rightarrow \mathbf{2}$: propositions are true or false depending on the unique referential provided by V
- Valuations in a modal logic also depends on the **current state** of computation: $V : \text{Variables} \times \mathbb{P} \rightarrow \mathbf{2}$ or, equivalently, ,
 $V : \text{Variables} \rightarrow \mathcal{P}\mathbb{P}$: each variable is associated to the set of processes in which its value is fixed as true
- In our case, models for such a logic are defined over the universe of processes \mathbb{P} (i.e., **terms** of our process language) equipped with relations $\{\overset{x}{\rightarrow} \mid x \in Act\}$ defined by the **operational semantics** of the language.
- ... but the topic **modal logics** has a longer story and a broad spectrum of applications ...

The language

Syntax

$$\phi ::= \text{true} \mid \text{false} \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \langle K \rangle \phi \mid [K] \phi$$

The language

Semantics: $E \models \phi$

$E \models \text{true}$

$E \not\models \text{false}$

$E \models \phi_1 \wedge \phi_2$ iff $E \models \phi_1 \wedge E \models \phi_2$

$E \models \phi_1 \vee \phi_2$ iff $E \models \phi_1 \vee E \models \phi_2$

$E \models \langle K \rangle \phi$ iff $\exists_{F \in \{E' \mid E \xrightarrow{a} E' \wedge a \in K\}} \cdot F \models \phi$

$E \models [K] \phi$ iff $\forall_{F \in \{E' \mid E \xrightarrow{a} E' \wedge a \in K\}} \cdot F \models \phi$

Example

$$Sem \triangleq get.put.Sem$$

$$P_i \triangleq \overline{get}.c_i.\overline{put}.P_i$$

$$S \triangleq new \{get, put\} (Sem \mid (\mid_{i \in I} P_i))$$

- $Sem \models \langle get \rangle true$ holds because

$$\exists_{F \in \{Sem' \mid Sem \xrightarrow{get} Sem'\}} . F \models true$$

with $F = put.Sem$.

- However, $Sem \models [put] false$ also holds, because

$$T = \{Sem' \mid Sem \xrightarrow{put} Sem'\} = \emptyset.$$

Hence $\forall_{F \in T} . F \models false$ becomes trivially true.

- The only action initially permited to S is τ : $\models [-\tau] false$.

Example

$$Sem \triangleq get.put.Sem$$

$$P_i \triangleq \overline{get}.c_i.\overline{put}.P_i$$

$$S \triangleq new \{get, put\} (Sem \mid (\prod_{i \in I} P_i))$$

- Afterwards, S can engage in any of the critical events c_1, c_2, \dots, c_i :
 $[\tau]\langle c_1, c_2, \dots, c_i \rangle true$
- After the semaphore initial synchronization and the occurrence of c_j in P_j , a new synchronization becomes inevitable:
 $S \models [\tau][c_j](\langle - \rangle true \wedge [-\tau] false)$

Notes

- inevitability of a : $\langle - \rangle \text{true} \wedge [-a] \text{false}$
- progress: $\langle - \rangle \text{true}$
- deadlock or termination: $[-] \text{false}$
- what about

$\langle - \rangle \text{false}$ and $[-] \text{true}$?

- satisfaction decided by unfolding the definition of \models : no need to compute the transition graph

A denotational semantics

Idea: associate to each formula ϕ the set of processes that make it true

$$\phi \text{ vs } \|\phi\| = \{E \in \mathbb{P} \mid E \models \phi\}$$

$$\|\text{true}\| = \mathbb{P}$$

$$\|\text{false}\| = \emptyset$$

$$\|\phi_1 \wedge \phi_2\| = \|\phi_1\| \cap \|\phi_2\|$$

$$\|\phi_1 \vee \phi_2\| = \|\phi_1\| \cup \|\phi_2\|$$

$$\|[K]\phi\| = \|[K]\|(\|\phi\|)$$

$$\|\langle K \rangle \phi\| = \|\langle K \rangle\|(\|\phi\|)$$

$\| [K] \|$ and $\| \langle K \rangle \|$

Just as \wedge corresponds to \cap and \vee to \cup , modal logic combinators correspond to **unary functions** on sets of processes:

$$\| [K] \| = \lambda_{X \subseteq \mathbb{P}} . \{ F \in \mathbb{P} \mid \text{if } F \xrightarrow{a} F' \wedge a \in K \text{ then } F' \in X \}$$

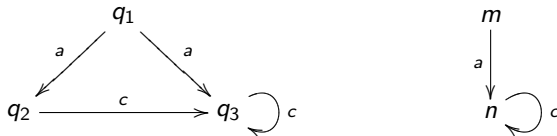
$$\| \langle K \rangle \| = \lambda_{X \subseteq \mathbb{P}} . \{ F \in \mathbb{P} \mid \exists_{F' \in X, a \in K} . F \xrightarrow{a} F' \}$$

Note

These combinators perform a **reduction to the previous state** indexed by actions in K

$$\| [K] \| \text{ and } \| \langle K \rangle \|$$

Example



$$\| \langle a \rangle \| \{q_2, n\} = \{q_1, m\}$$

$$\| [a] \| \{q_2, n\} = \{q_2, q_3, m, n\}$$

A denotational semantics

$$E \models \phi \text{ iif } E \in \|\phi\|$$

Example: $\mathbf{0} \models [-]\text{false}$

because

$$\begin{aligned}\|[-]\text{false}\| &= \|[-]\|(\|\text{false}\|) \\ &= \|[-]\|(\emptyset) \\ &= \{F \in \mathbb{P} \mid \text{if } F \xrightarrow{x} F' \wedge x \in \text{Act} \text{ then } F' \in \emptyset\} \\ &= \{\mathbf{0}\}\end{aligned}$$

A denotational semantics

$$E \models \phi \text{ iif } E \in \|\phi\|$$

Example: $?? \models \langle - \rangle \text{true}$

because

$$\begin{aligned} \|\langle - \rangle \text{true}\| &= \|\langle - \rangle\|(\|\text{true}\|) \\ &= \|\langle - \rangle\|(\mathbb{P}) \\ &= \{F \in \mathbb{P} \mid \exists_{F' \in \mathbb{P}, a \in K} . F \xrightarrow{a} F'\} \\ &= \mathbb{P} \setminus \{\mathbf{0}\} \end{aligned}$$

A denotational semantics

Complement

Any property ϕ divides \mathbb{P} into two disjoint sets:

$$\|\phi\| \text{ and } \mathbb{P} - \|\phi\|$$

The **characteristic formula** of the complement of $\|\phi\|$ is ϕ^c :

$$\|\phi^c\| = \mathbb{P} - \|\phi\|$$

where ϕ^c is defined inductively on the formulae structure:

$$\text{true}^c = \text{false} \quad \text{false}^c = \text{true}$$

$$(\phi_1 \wedge \phi_2)^c = \phi_1^c \vee \phi_2^c$$

$$(\phi_1 \vee \phi_2)^c = \phi_1^c \wedge \phi_2^c$$

$$(\langle a \rangle \phi)^c = [a] \phi^c$$

... but **negation** is not explicitly introduced in the logic.

Modal Equivalence

For each (finite or infinite) set Γ of formulae,

$$E \simeq_{\Gamma} F \Leftrightarrow \forall \phi \in \Gamma . E \models \phi \Leftrightarrow F \models \phi$$

Examples

$$a.b.\mathbf{0} + a.c.\mathbf{0} \simeq_{\Gamma} a.(b.\mathbf{0} + c.\mathbf{0})$$

for $\Gamma = \{\langle x_1 \rangle \langle x_2 \rangle \dots \langle x_n \rangle \text{true} \mid x_i \in \text{Act}\}$

(what about \simeq_{Γ} for $\Gamma = \{\langle x_1 \rangle \langle x_2 \rangle \langle x_3 \rangle \dots \langle x_n \rangle [-] \text{false} \mid x_i \in \text{Act}\}$?)

Modal Equivalence

For each (finite or infinite) set Γ of formulae,

$$E \simeq F \iff E \simeq_{\Gamma} F \text{ for every set } \Gamma \text{ of well-formed formulae}$$

Lemma

$$E \sim F \Rightarrow E \simeq F$$

Note

the converse of this lemma does not hold, e.g. let

- $A \triangleq \sum_{i \geq 0} A_i$, where $A_0 \triangleq \mathbf{0}$ and $A_{i+1} \triangleq a.A_i$
- $A' \triangleq A + \underline{\text{fix}}(X = a.X)$

$$A \approx A' \text{ but } A \not\simeq A'$$

Modal Equivalence

Theorem [Hennessy-Milner, 1985]

$$E \sim F \Leftrightarrow E \simeq F$$

for **image-finite** processes.

Image-finite processes

E is **image-finite** iff $\{F \mid F \xrightarrow{a} E\}$ is **finite** for every action $a \in Act$

Modal Equivalence

Theorem [Hennessy-Milner, 1985]

$$E \sim F \Leftrightarrow E \simeq F$$

for **image-finite** processes.

proof

\Rightarrow : by induction of the formula structure

\Leftarrow : show that \simeq is itself a bisimulation, by contradiction