Channels

Example: Leader election

We have \( n \) processes on a directed ring, each having a unique \( id \in \{1, \ldots, n\} \).

\[
\text{send}(id)
\]

\[
\text{loop forever}
\]

\[
\text{receive}(x)
\]

\[
\text{if } (x = id) \text{ then STOP fi}
\]

\[
\text{if } (x > id) \text{ then send}(x)
\]

Channels

Definition: Channels

- **Declaration:**
  \[
  c : \text{channel} [k] \text{ of bool size } k
  \]
  \[
  c : \text{channel} [\infty] \text{ of int unbounded}
  \]
  \[
  c : \text{channel} [0] \text{ of colors Rendez-vous}
  \]
- **Primitives:**
  \[
  \text{empty}(c)
  \]
  \[
  c ! e \quad \text{add the value of expression } e \text{ to channel } c
  \]
  \[
  c ? x \quad \text{read a value from } c \text{ and assign it to variable } x
  \]
- **Domain:**
  Let \( D_m \) be the domain for a single message.
  \[
  D_c = D_{k,m} \text{ size } k
  \]
  \[
  D_c = D_{\infty,m} \text{ unbounded}
  \]
  \[
  D_c = \{\varepsilon\} \text{ Rendez-vous}
  \]
- **Politics:** FIFO, LIFO, BAG, …

Channels

Semantics: (lossy) FIFO

- **Send**
  \[
  s_i \xrightarrow{c!e} s'_i \land \nu(c) = \nu(e) \cdot \nu(c)
  \]
  \[
  (s, \nu) \xrightarrow{c!e} (s', \nu')
  \]
- **Receive**
  \[
  s_i \xrightarrow{c?x} s'_i \land \nu(v) = \nu'(c) \cdot \nu'(x)
  \]
  \[
  (s, \nu) \xrightarrow{c?x} (s', \nu')
  \]
- **Lossy send**
  \[
  s_i \xrightarrow{c!e} s'_i
  \]
  \[
  (s, \nu) \xrightarrow{c!e} (s', \nu)
  \]

Implicit assumption: all variables that do not occur in the premise are not modified.

Exercises:

1. Implement a FIFO channel using rendez-vous with an intermediary process.
2. Give the semantics of a LIFO channel.
3. Model the alternating bit protocol (ABP) using a lossy FIFO channel.

Fairness assumption: For each channel, if infinitely many messages are sent, then infinitely many messages are delivered.

High-level descriptions

Summary

- Sequential program = transition system with variables
- Concurrent program with shared variables
- Concurrent program with Rendez-vous
- Concurrent program with FIFO communication
- Petri net
- …
Models: expressivity versus decidability

Definition: (Un)decidability
- Automata with 2 integer variables = Turing powerful
- Restriction to variables taking values in finite sets
- Asynchronous communication: unbounded fifo channels = Turing powerful
- Restriction to bounded channels

Definition: Some infinite state models are decidable
- Petri nets. Several unbounded integer variables but no zero-test.
- Pushdown automata. Model for recursive procedure calls.
- Timed automata.
- ...

Static and dynamic properties

Definition: Static properties
Example: Mutual exclusion
Safety properties are often static.
They can be reduced to reachability.

Definition: Dynamic properties
Example: Every request should be eventually granted.
\( \forall t, (Call_i(t) \rightarrow \exists t' \geq t, (atLevel_i(t') \land openDoor_i(t')))) \)

The elevator should not cross a level for which a call is pending without stopping.
\( \forall i \forall t', (Call_i(t) \land t \leq t' \land atLevel_i(t')) \rightarrow \exists t \leq t'' \leq t', (atLevel_i(t'') \land openDoor_i(t''))) \)

First Order specifications

First order logic
- These specifications can be written in FO(<).
- FO(<) has a good expressive power.
- ...but FO(<)-formulae are not easy to write and to understand.
- FO(<) is decidable.
- ...but satisfiability and model checking are non elementary.

Definition: Temporal logics
- no variables: time is implicit.
- quantifications and variables are replaced by modalities.
- Usual specifications are easy to write and read.
- Good complexity for satisfiability and model checking problems.
Linear versus Branching

Let $M = (S, T, I, AP, \ell)$ be a Kripke structure.

**Definition: Linear specifications**

Example: The printer manager is *fair*.

On each run, whenever some process requests the printer, it eventually gets it.

**Execution sequences (runs):**

$\sigma = s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \cdots$ with $s_i \rightarrow s_{i+1} \in T$

Two Kripke structures having the same execution sequences satisfy the same linear specifications.

Actually, linear specifications only depend on the *label* of the execution sequence

$\ell(\sigma) = \ell(s_0) \rightarrow \ell(s_1) \rightarrow \ell(s_2) \rightarrow \cdots$

Models are words in $\Sigma^\omega$ with $\Sigma = 2^{AP}$.

**Definition: Branching specifications**

Example: Each process has the *possibility* to print first.

Such properties depend on the execution tree.

**Execution tree** = unfolding of the transition system

---

**Some original References**

*Tense Logic and the Theory of Linear Order.*

*Fast LTL to Büchi automata translation.*
http://www.lsv.ens-cachan.fr/~gastin/mes-publis.php

The tableau method for temporal logic: An overview,

The complexity of propositional linear temporal logic.

---

**References**

**Bibliography**

*Specification and Verification using Temporal Logics.*
In Modern applications of automata theory, IISc Research Monographs 2.
http://www.lsv.ens-cachan.fr/~gastin/mes-publis.php

A large list of references is given in this paper.

**Bibliography**

First-order definable languages.
http://www.lsv.ens-cachan.fr/~gastin/mes-publis.php

A large overview of formalisms expressively equivalent to First-Order.

---

**Some original References**

Checking that finite state concurrent programs satisfy their linear specification.

On the temporal analysis of fairness.

The declarative past and imperative future: Executable temporal logics for interactive systems.
Outline

Introduction
Models
Specifications
  • Linear Time Specifications
    • Definitions
    • Main results
    • B"uchi automata
    • From LTL to BA
    • Hardness results
Branching Time Specifications

Linear Temporal Logic (Pnueli 1977)

Definition: Syntax: LTL(\mathcal{AP}, X, U)
\varphi ::= \bot \mid p \ (p \in \mathcal{AP}) \mid \neg \varphi \mid \varphi \lor \psi \mid X \varphi \mid \varphi \lor \psi

Definition: Semantics: w = a_0a_1a_2\cdots \in \Sigma^\omega with \Sigma = 2^{\mathcal{AP}} and i \in \mathbb{N}
\begin{align*}
w, i = p & \quad \text{if} \quad p \in a_i \\
w, i = \neg \varphi & \quad \text{if} \quad w, i \not\models \varphi \\
w, i = \varphi \lor \psi & \quad \text{if} \quad w, i \models \varphi \lor w, i \models \psi \\
w, i = X \varphi & \quad \text{if} \quad w, i+1 \models \varphi \\
w, i = \varphi \lor \psi & \quad \text{if} \quad \exists k. \ i \leq k \ \text{and} \ w, k \models \psi \ \text{and} \ \forall j. \ (i \leq j < k) \ \rightarrow \ w, j \models \varphi \\
w, i = \varphi \lor \psi & \quad \text{if} \quad \exists k. \ i \leq k \ \text{and} \ w, k \models \psi \ \text{and} \ \forall j. \ (i \leq j < k) \ \rightarrow \ w, j \models \varphi
\end{align*}

Example:
\[ X \varphi \quad \varphi \quad \varphi \quad \varphi \quad \varphi \quad \cdots \]
Linear Temporal Logic (Pnueli 1977)

Definition: Macros

- **Eventually:** $\text{F } \varphi = \top U \varphi$

- **Always:** $\text{G } \varphi = \neg \text{F } \neg \varphi$

- **Weak until:** $\varphi \text{ W } \psi = \text{G } \varphi \lor \varphi \text{ U } \psi$

- **Release:** $\varphi \text{ R } \psi = \psi \text{ W } (\varphi \land \psi)$

- **Next until:** $\varphi \text{ XU } \psi = X (\varphi \text{ U } \psi)$

- $X \psi = \bot \text{ XU } \psi$ and $\varphi \text{ U } \psi = \psi \lor (\varphi \land \varphi \text{ XU } \psi)$.

Definition: Specifications:

- **Safety:** $\text{G good}$

- **MutEx:** $\neg \text{F } (\text{crit}_1 \land \text{crit}_2)$

- **Liveness:** $\text{G F } \text{active}$

- **Response:** $\text{G (request } \rightarrow \text{ F grant)}$

- **Response’:** $\text{G (request } \rightarrow \text{ X } (\neg \text{ request U } \text{ grant}))$

- **Release:** reset $\text{R alarm}$

- **Strong fairness:** $\text{G F request } \rightarrow \text{ G F grant}$

- **Weak fairness:** $\text{F G request } \rightarrow \text{ G F grant}$

Examples:

Every elevator request should be eventually satisfied.

$$\bigwedge_i \text{G (Call}_i \rightarrow \text{F (atLevel}_i \land \text{openDoor}_i))$$

The elevator should not cross a level for which a call is pending without stopping.

$$\bigwedge_i \text{G (Call}_i \rightarrow \neg \text{atLevel}_i \text{ W (atLevel}_i \land \text{openDoor}_i))$$

Linear Temporal Logic (Pnueli 1977)

Past LTL

Definition: Semantics:

$$w = a_0 a_1 a_2 \cdots \in \Sigma^\omega \text{ with } \Sigma = 2^{\mathbb{AP}} \text{ and } i \in \mathbb{N}$$

- $w, i \models Y \varphi$ if $i > 0$ and $w, i-1 \models \varphi$

- $w, i \models \varphi S \psi$ if $\exists k. k \leq i \text{ and } w, k \models \psi \text{ and } \forall j. (k < j \leq i) \rightarrow w, j \models \varphi$

Example:

Example: LTL versus PLTL

$\text{G(grant } \rightarrow \text{ Y (} \neg \text{grant S request)}))$

Theorem (Laroussinie & Markey & Schnoebelen 2002)

PLTL may be exponentially more succinct than LTL.
Past LTL

Definition: Semantics: \( w = a_0 a_1 a_2 \cdots \in \Sigma^\omega \) with \( \Sigma = 2^{AP} \) and \( i \in \mathbb{N} \)

- \( w, i \models Y \varphi \) if \( i > 0 \) and \( w, i-1 \models \varphi \)
- \( w, i \models \varphi S \psi \) if \( \exists k. k \leq i \) and \( w, k \models \psi \) and \( \forall j. (k < j \leq i) \rightarrow w, j \models \varphi \)

Example:

Example: LTL versus PLTL

\[ G(\text{grant} \rightarrow Y(\neg \text{grant} S \text{request})) = (\text{request} R \neg \text{grant}) \land G(\text{grant} \rightarrow (\text{request} \lor X(\text{request} R \neg \text{grant}))) \]

Theorem (Laroussinie & Markey & Schnoebelen 2002)

PLTL may be exponentially more succinct than LTL.

Expressivity

Theorem [8, Kamp 68]

\[ \text{LTL}(Y, S, X, U) = \text{FO}_\Sigma(\leq) \]

Separation Theorem [13, Gabbay, Pnueli, Shelah & Stavi 80]

For all \( \varphi \in \text{LTL}(Y, S, X, U) \) there exist \( \varphi_i \in \text{LTL}(Y, S) \) and \( \varphi_i' \in \text{LTL}(X, U) \) such that for all \( w \in \Sigma^\omega \) and \( k \geq 0 \),

\[ w, k \models \varphi \iff w, k \models \bigvee_i \varphi_i \land \varphi_i' \]

Corollary: \( \text{LTL}(Y, S, X, U) = \text{LTL}(X, U) \)

For all \( \varphi \in \text{LTL}(Y, S, X, U) \) there exist \( \varphi' \in \text{LTL}(X, U) \) such that for all \( w \in \Sigma^\omega \),

\[ w, 0 \models \varphi \iff w, 0 \models \varphi' \]

Elegant algebraic proof of \( \text{LTL}(X, U) = \text{FO}_\Sigma(\leq) \) due to Wilke 98.