Important note
Corrections have been made to the CTL part.
An important hypothesis was missing (see slide 15)

Crash course – Verification of Finite Automata
CTL model-checking

Exercise session - 08.12.2016
Romain Jacob
Reminders – Big picture

**Objective**
- Verify properties over DES models
- Formal method $\Rightarrow$ Absolute guarantee!

**Problem**
- Combinatorial explosion
  $\rightarrow$ Huge amount of states, computationally intractable

**Solution**
- Work with sets of states
  $\rightarrow$ Symbolic Model-Checking
  $\rightarrow$ (O)BDDs
Reminders – First exercise session

Sets

- $\sigma(s) = x_1 \bar{x}_0 = (1,0)$ and $\psi_A = x_1 + x_0$
- $s \vDash \psi_A$

\[ \psi_E = 1 \]
\[ \psi_A = f \]
\[ \psi_B = g \]
\[ \psi_{AB} = f \cdot g \]

Examples:

Reads “$s$ satisfies $\psi_A$”

Equivalence between sets and Boolean equations

BBD representation of Boolean functions

\[ f(x_1, x_2, x_3) = x_1 + \bar{x}_1 \bar{x}_2 + \bar{x}_2 \bar{x}_3 \]

Fall $x_1 = 0$

- $f_{|x_1=0, x_2=0} = \bar{x}_3$
- $f_{|x_1=0, x_2=1} = 1$

Fall $x_1 = 1$

- $f_{|x_1=1} = 1$

0

1
Let see what you remember!
Discrete Event Systems

Roger P. Wattenhofer, Lothar Thiele, Laurent Vanbever

Learning objective:
Over the past few decades the rapid evolution of computing, communication, and information technologies has brought about the proliferation of new dynamic systems. A significant part of activity in these systems is governed by operational rules designed by humans. The dynamics of these systems are characterized by asynchronous occurrences of discrete events, some controlled (e.g. hitting a keyboard key, sending a message), some not (e.g. spontaneous failure, packet loss).

The mathematical arsenal centered around differential equations that has been employed in systems engineering to model and study processes governed by the laws of nature is often inadequate or inappropriate for discrete event systems. The challenge is to develop new modeling frameworks, analysis techniques, design...
Today’s menu

1. Reachability of states
2. Comparison of automata
3. Formulation and verification of CTL properties

Can be formulated as reachability problems
Reachability of states

Fairly simple

1. Start from the initial set of states,
2. Compute all states you can transition to in one hop (one transition),
   \[ \text{→ The successor states,} \]
3. Join the two sets,
4. Iterate from 2. until you reach a fix point.
5. Done !

*Is this guarantee to terminate?*
Reachability of states

Fairly simple

1. Start from the initial set of states,
2. Compute all states you can transition to in one hop (one transition), \rightarrow The successor states,
3. Join the two sets,
4. Iterate from 2. until you reach a fix point.
5. Done!

*Is this guarantee to terminate?*

\rightarrow Only if you have a finite model!!

*How can we formalize this problem?*
Formalization of reachable states

\[ \delta : X \subseteq E \rightarrow X' \subseteq E \]

\[ q \iff q' \]

\[ q \in X \iff \exists q' \in X', \quad \delta(q, q') \text{ is defined} \]

\[ \psi_\delta(q, q') = 1 \]

\[ \overline{q} \notin X \iff \nexists q' \in X', \delta(q, q') \text{ is defined} \]

\[ \forall q' \in X, \psi_\delta(q, q') = 0 \]
Formalization of reachable states

$$\delta : X \subseteq E \rightarrow X' \subseteq E$$

$$q \mapsto q'$$

What is Q’?

$$q' \in Q' \Rightarrow q' \in X' \Rightarrow \exists q \in X, \psi_\delta(q, q') = 1$$

Not sufficient!

We also need that q belongs to Q:

$$q \in Q$$

or equivalently

$$\psi_Q(q) = 1$$
Formalization of reachable states

\[ \delta : X \subseteq E \rightarrow X' \subseteq E \]
\[ q \mapsto q' \]

What is Q’?

\[ q' \in Q' \iff \exists q \in X, \quad \psi_Q(q) = 1 \quad \text{and} \quad \psi_\delta(q, q') = 1 \]

\[ Q' = \text{Suc}(Q, \delta) = \{ q' \mid \exists q \in X, \psi_Q(q) \cdot \psi_\delta(q, q') = 1 \} \]
Formalization of reachable states

$\delta : X \subseteq E \rightarrow X' \subseteq E$
$q \mapsto q'$

$Q' = \text{Suc}(Q, \delta) = \{ q' | \exists q \in X, \psi_Q(q) \cdot \psi_\delta(q, q') = 1 \}$
$\Leftrightarrow \psi_{Q'} = \psi_Q \cdot \psi_\delta$

$Q_R$: set of reachable states

$Q_R = Q_0 \cup \bigcup_{i \geq 0} \text{Suc}(Q_i, \delta)$
$\Leftrightarrow \psi_{Q_R} = \psi_{Q_0} \sum_{i \geq 0} \psi_{Q_i} \cdot \psi_\delta$

Again, finite union if finite model
Comparison of automata

Two automata are equivalent if the following term is true,

\[ \exists y_1, y_2 : \psi_Y(y_1, y_2) \cdot (y_1 \neq y_2) \]

- Computation of the joint transition function,
  \[ \psi_\delta(q_1, q_2, q'_1, q'_2) = (\exists u : \psi_{\omega_1}(u, q_1, q'_1) \cdot \psi_{\omega_2}(u, q_2, q'_2)) \]

- Computation of the reachable states (method according to previous slides),
  \[ \psi_Q(q_1, q_2) \]

- Computation of the reachable output values,
  \[ \psi_Y(y_1, y_2) = (\exists q_1, q_2 : \psi_Q(q_1, q_2) \cdot \psi_{\omega_1}(q_1, y_1) \cdot \psi_{\omega_2}(q_2, y_2)) \]

The automata are not equivalent if the following term is true,

\[ \psi_\delta(q_1, q_2, q'_1, q'_2) = (\exists u : \psi_{\omega_1}(u, q_1, q'_1) \cdot \psi_{\omega_2}(u, q_2, q'_2)) \]

Don’t compare states!

- Get rid of the input
- Compute \( Q_R \)
- Deduce reachable outputs
- Test for equivalence
Formulation of CTL properties

Based on atomic propositions ($\phi$) and quantifiers

$A\phi \rightarrow \langle \text{All } \phi \rangle$, $\phi$ holds on all paths

$E\phi \rightarrow \langle \text{Exists } \phi \rangle$, $\phi$ holds on at least one path

$X\phi \rightarrow \langle \text{Next } \phi \rangle$, $\phi$ holds on the next state

$F\phi \rightarrow \langle \text{Finally } \phi \rangle$, $\phi$ holds at some state along the path

$G\phi \rightarrow \langle \text{Globally } \phi \rangle$, $\phi$ holds on all states along the path

$\phi_1 U \phi_2 \rightarrow \langle \phi_1 \text{ Until } \phi_2 \rangle$, $\phi_1$ holds until $\phi_2$ holds

Quantifiers over paths

Path-specific quantifiers
Formulation of CTL properties

Proper CTL formula: \{A,E\} \{X,F,G,U\} \phi

→ Quantifiers **go by pairs**, you need one of each.

**Missing Hypothesis**

Interpretation on CTL formula

→ Transition functions are **assumed to be serial**
  (i.e. every state has at least one successor)

Automaton of interest

→ Automaton to work with

Simple “means” that we get rid of leaf nodes...
→ They transition to themselves
Formulation of CTL properties

\( \text{EF } \phi \): “There exists a path along which at some state \( \phi \) holds.”

\[ q \models \text{EF } \phi \]

\[ r \models ? \]

\[ s \models ? \]
Formulation of CTL properties

$$EF \, \phi : \text{“There exists a path along which at some state } \phi \text{ holds.”}$$

$q \models EF \, \phi$

$r \not\models EF \, \phi$

$s \not\models EF \, \phi$
Formulation of CTL properties

AF $\phi$ : “On all paths, at some state $\phi$ holds.”

$q \models \phi$

$q \models AF \phi$

$r \models ?$

$s \models ?$
Formulation of CTL properties

AF $\phi$ : “On all paths, at some state $\phi$ holds.”

$q \models \phi$

$r \models \text{AF } \phi$

$s \not\models \text{AF } \phi$
Formulation of CTL properties

\( \text{AG } \phi : \text{"On all paths, for all states } \phi \text{ holds."} \)

\( q \models \phi \)

\( r \not\models \phi \)

\( s \not\models \phi \)
Formulation of CTL properties

AG $\phi$: “On all paths, for all states $\phi$ holds.”

$q \models \phi$

$r \models AG \phi$

$s \not\models AG \phi$
Formulation of CTL properties

**EG \( \phi \):** “There exists a path along which for all states \( \phi \) holds.”

\[ q \models EG \phi \]

\[ r \models ? \]

\[ s \models ? \]
Formulation of CTL properties

\[ \text{EG } \phi : \text{"There exists a path along which for all states } \phi \text{ holds."} \]
Formulation of CTL properties

$\mathcal{E} \phi U \Psi$ : “There exists a path along which $\phi$ holds until $\Psi$ holds.”
Formulation of CTL properties

$E\phi U\psi$ : “There exists a path along which $\phi$ holds until $\psi$ holds.”
Formulation of CTL properties

AφUΨ : “On all paths, φ holds until Ψ holds.”
Formulation of CTL properties

$A\phi U \Psi$ : “On all paths, $\phi$ holds until $\Psi$ holds.”
Formulation of CTL properties

**AXφ**: “On all paths, the next state satisfies φ.”

**EXφ**: “There exists a path along which the next state satisfies φ.”
Formulation of CTL properties

$AX\phi$: “On all paths, the next state satisfies $\phi$.”

$EX\phi$: “There exists a path along which the next state satisfies $\phi$.”

$q \models EX\phi$

$r \models EX\phi$

$s \not\models EX\phi$
Formulation of CTL properties

**AG EF φ**: “On all paths and for all states, there exists a path along which at some state φ holds.”

```
q ≅ AG EF \phi
r \not\models \phi
s \models \phi
```
Formulation of CTL properties

\[ \text{AG EF } \phi : \text{ “On all paths and for all states, there exists a path along which at some state } \phi \text{ holds.”} \]

\[ q \models \text{AG EF } \phi \]
\[ r \models \text{AG EF } \phi \]
\[ s \models \text{AG EF } \phi \]
Inverting properties is sometimes useful!

\[
\begin{align*}
AG \, \phi & \equiv \neg EF \, \neg \phi \\
AF \, \phi & \equiv \neg EG \, \neg \phi \\
EF \, \phi & \equiv \neg AG \, \neg \phi \\
EG \, \phi & \equiv \neg AF \, \neg \phi
\end{align*}
\]

Remark: There exists other temporal logics

→ LTL (Linear Tree Logic)
→ CTL* = {CTL,LTL}
→ ...

“On all paths, for all states \(\phi\) holds.”

≡

“There exists no path along which at some state \(\phi\) doesn’t hold.”
How to verify CTL properties?

*Convert the property verification into a reachability problem*

1. Start from states in which the property holds;
2. Compute all predecessor states for which the property still holds true; (same as for computing successor, with the inverse the transition function)
3. If initial states set is a subset, the property is satisfied by the model.

*Computation specifics are described in the lecture slides.*
So... what is Model-Checking exactly?

An algorithm

**Input**
- A DES model, $M$
  - Finite automata,
  - Petri nets,
  - Kripke machine, ...

**Output**
- $M \models \phi$ ?
- A trace for which the property does not hold!

- A logic property, $\phi$
  - CTL,
  - LTL, ...
Crash course – Verification of Finite Automata
CTL model-checking

Your turn to work!

Slides online on my webpage:
http://people.ee.ethz.ch/~jacobr/
Comparison of Finite Automata

a) Express the characteristic function of the transition relation for both automaton, $\psi_r(x, x', u)$.

$$\psi_A(x_A, x_A', u) = \overline{x_A}x_A' \overline{u} + \overline{x_A}x_A' u$$
$$+ x_A x_A' u + x_A x_A' \overline{u}$$

$$\psi_B(x_B, x_B', u) = \overline{x_B}x_B' \overline{u} + \overline{x_B}x_B' u$$
$$+ x_B x_B' u + x_B x_B' \overline{u}$$
Comparison of Finite Automata

b) Express the joint transition function, $\psi_f$.

$$\psi_f(x_A, x'_A, x_B, x'_B) = (\exists u : \psi_A(x_A, x'_A, u) \cdot \psi_B(x_B, x'_B, u))$$

$$\psi_f(x_A, x'_A, x_B, x'_B)$$

$$= (\overline{x_A}x'_A + x_Ax'_A) \cdot (\overline{x_B}x'_B + x_Bx'_B) +$$

$$= (\overline{x_A}x'_A + x_Ax'_A) \cdot (\overline{x_B}x'_B + x_Bx'_B) +$$

$$= \overline{x_A}x'_A \overline{x_B}x'_B + \overline{x_A}x'_A x_Bx'_B + x_Ax'_A \overline{x_B}x'_B + x_Ax'_A x_Bx'_B +$$

$$= \overline{x_A}x'_A \overline{x_B}x'_B + \overline{x_A}x'_A x_Bx'_B + x_Ax'_A \overline{x_B}x'_B + x_Ax'_A x_Bx'_B$$
Comparison of Finite Automata

c) Express the characteristic function of the reachable states, $\psi_X(x_A, x_B)$.

\[
\psi_X(x_A, x_B) = \overline{x_A} x_B \\
\psi_{X_1} = \overline{x_A} x_B + \overline{x_A} x_B + x_A x_B \\
\psi_{X_2} = \overline{x_A} x_B + \overline{x_A} x_B + x_A x_B \\
= \psi_{X_1} \\
\rightarrow \text{ the fix-point is reached!}
\]

\[
\psi_X = \overline{x_A} x_B + \overline{x_A} x_B + x_A x_B
\]
Comparison of Finite Automata

d) Express the characteristic function of the reachable output, $\psi_Y(x_A, x_B)$. 

\[
\psi_{g_A} = \overline{x_A y_A} + x_A y_A \\
\psi_{g_B} = \overline{x_B y_B} + x_B y_B \\
\psi_X = \overline{x_A x_B} + x_A x_B + x_A x_B \\

\psi_Y(y_A, y_B) = (\exists (x_A, x_B) : \psi_X \cdot \psi_{g_A} \cdot \psi_{g_B}) \\
= y_A y_B + \overline{y_A y_B} + \overline{y A y_B}
\]
Comparison of Finite Automata

e) Are the automata equivalent? **Hint**: Evaluate, for example, $\psi_Y(0,1)$.

$$\psi_Y((y_A, y_B) = (0, 1)) = 1$$

Or, in a more general way,

$$\psi_Y(y_A, y_B) = y_A y_B + \overline{y_A} y_B + y_A \overline{y_B}$$

and $(y_A \neq y_B) = \overline{y_A} y_B + y_A \overline{y_B}$

implies $\psi_Y \cdot (y_A \neq y_B) \neq 0$

→ Automata are not equivalent.
Temporal Logic

i. $\text{EF } a$

ii. $\text{EG } a$

iii. $\text{EX AX } a$

iv. $\text{EF } (a \land \text{EX NOT}(a))$
Temporal Logic

i. \( \text{EF } a \)
\[ Q = \{0, 1, 2, 3\} \]

ii. \( \text{EG } a \)

iii. \( \text{EX AX } a \)

iv. \( \text{EF ( a AND EX NOT(a) )} \)
Temporal Logic

i. $EF \ a$
   
   $Q = \{0, 1, 2, 3\}$

ii. $EG \ a$
   
   $Q = \{0, 3\}$

iii. $EX \ AX \ a$

iv. $EF (a \ AND \ EX \ NOT(a))$
Temporal Logic

i. EF a
   \[ Q = \{ 0, 1, 2, 3 \} \]

ii. EG a
   \[ Q = \{ 0, 3 \} \]

iii. EX AX a
    \[ Q = \{ 1, 2 \} \]

iv. EF ( a AND EX NOT(a) )
Temporal Logic

i. EF a
   \[ Q = \{0, 1, 2, 3\} \]

ii. EG a
   \[ Q = \{0, 3\} \]

iii. EX AX a
    \[ Q = \{1, 2\} \]

iv. EF ( a AND EX NOT(a) )
    \[ Q = \{0, 1, 2, 3\} \]
Temporal Logic

**Trick** \( \text{AF } Z \land \neg(\text{EG } \neg Z) \)

Require: \( \psi_{Z}, \psi_{f} \)

\[
\begin{align*}
\text{current} &= \text{NOT}(\psi_{Z}); \\
\text{next} &= \text{current AND } \psi_{\text{PRE}}(\text{current}, f); \\
\text{while next } \neq \text{current do} \\
\quad &\text{current} = \text{next}; \\
\quad &\text{next} = \text{current AND } \psi_{\text{PRE}}(\text{current}, f); \\
\text{return } \psi_{\text{AF } Z} &= \text{NOT}(\text{current}); \\
\end{align*}
\]

\( \triangleright \) Equivalence in term of sets:

\( \triangleright X_{0} \)

\[
\begin{align*}
\triangleright X_{1} &= X_{0} \cap \text{Pre}(X_{0}, f) \\
\triangleright X_{i}! &= X_{i-1} \\
\triangleright X_{f} \models \text{EG NOT}(Z) \\
\triangleright \overline{X_{f}} &= \text{AF } Z = \text{NOT}(\text{EG NOT}(Z))
\end{align*}
\]
Crash course – Verification of Finite Automata

CTL model-checking

See you next week!

Slides online on my webpage:
http://people.ee.ethz.ch/~jacobr/