Algorithms for Model Checking (2IW55)

Lecture 3

Symbolic Model Checking: Fairness and Counterexamples

Chapter 6.3, 6.4.

Tim Willemse
(timw@win.tue.nl)
http://www.win.tue.nl/∼timw
MF 7.073

Technische Universiteit Eindhoven University of Technology

Outline

In summary, symbolic model checking:

- Recursively processes subformulae
- Represent the set of states satisfying a subformula by OBDDs
- Treats temporal operators by fixed point computations
- Relies on efficient implementation of equivalence test, and $\wedge, \vee, \neg$ and $\exists$ connectives on OBDDs.

Fix a Kripke Structure $M = \langle S, R, L \rangle$.

The temporal operators of CTL are characterised by fixed points:

- $\mathsf{E}\ \mathsf{F}\ g = \mu Z. g \lor \mathsf{E}\ \mathsf{X}\ Z$
- $\mathsf{E}\ \mathsf{G}\ f = \nu Z. f \land \mathsf{E}\ \mathsf{X}\ Z$
- $\mathsf{E}\ [f\ \mathsf{U}\ g] = \mu Z. g \lor (f \land \mathsf{E}\ \mathsf{X}\ Z)$

- Least Fixed Points: start iteration at false ($\emptyset$)
- Greatest Fixed Points: start iteration at true (S)

Intuition:

- Eventually . . . . least fixed points
- Globally . . . . greatest fixed points

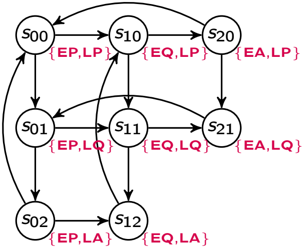
CTL model checking with Fixed Points

Function check(f) takes a formula f and returns the set of states where f holds: $\{s \mid s \models f\}$ (given a fixed Kripke Structure $M = \langle S, R, L \rangle$).

check(p)	$\{s \mid p \in L(s)\}$
check($\neg f$)	$S \setminus$ check(f)
check($f \vee g$)	check(f)$\cup$ check(g)
check(E X f)	Pre_R(check(f))
check(E [f U g])	lfp$\big(Z \mapsto$ check(g) $\cup$ (check(f) $\cap$ $Pre_R(Z)$)$\big)$
check(E G f)	gfp$\big(Z \mapsto$ check(f) $\cap$ $Pre_R(Z)\big)$

Recall: $Pre_R(Z) = \{s \in S \mid \exists t \in Z . s\ R\ t\}$

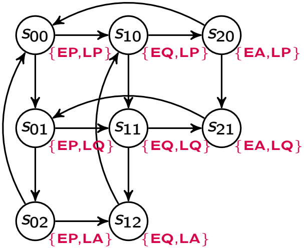
Example: demanding children



- Atomic Propositions: EP, EQ, EA, LP, LQ, LA
- Intended meaning: Linus or Emma is either Playing, posing Questions, getting Answers

Requirement: Whenever Linus asks a question, he eventually gets an answer
Formula: A G ($LQ \rightarrow$ A F LA)

Example: demanding children

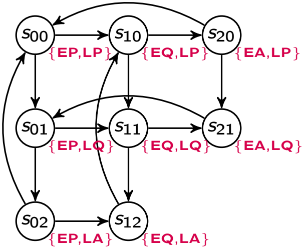


- Atomic Propositions: EP, EQ, EA, LP, LQ, LA
- Intended meaning: Linus or Emma is either Playing, posing Questions, getting Answers

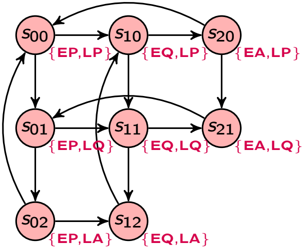
- *Step 1: express using basic operators*

$$\mathsf{A\ G}\ (LQ \rightarrow \mathsf{A\ F}\ LA)$$
$$\equiv$$
$$\neg \mathsf{E}\ [\mathsf{true}\ \mathsf{U}\ \neg(\neg LQ \vee \neg \mathsf{E\ G}\ \neg LA)]$$
$$\equiv$$
$$\neg \mathsf{E}\ [\mathsf{true}\ \mathsf{U}\ (LQ \wedge \mathsf{E\ G}\ \neg LA)]$$
$$\equiv$$
$$\neg \mu Y.((LQ \wedge \mathsf{E\ G}\ \neg LA) \cup \mathsf{E\ X}\ Y)$$

Example: demanding children

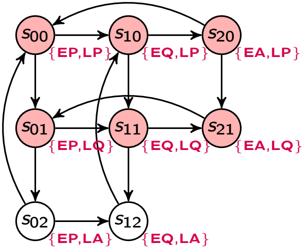


- *Step 2: compute* check(E G $\neg LA$)*, i.e., compute* $\nu Z.(\neg LA \wedge \text{E X } Z)$.



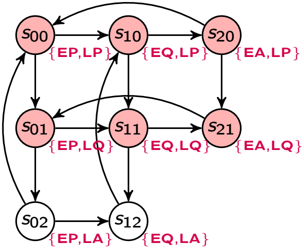
- *Step 2: compute* check(E G $\neg LA$)*, i.e., compute* $\nu Z.(\neg LA \wedge \text{E X } Z)$*.*
  - Greatest fixpoint, so start with approximating from true (i.e. all states)

Example: demanding children

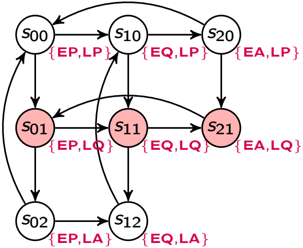


- *Step 2: compute* check(E G $\neg LA$)*, i.e., compute* $\nu Z.(\neg LA \wedge \text{E X } Z)$.
  - Greatest fixpoint, so start with approximating from true (i.e. all states)
  - Stable at $\{s_{00}, s_{10}, s_{20}, s_{01}, s_{11}, s_{21}\}$

Example: demanding children

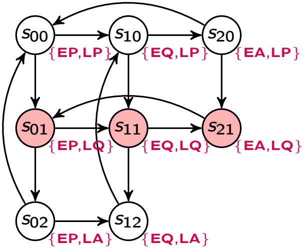


- *Step 3: compute* $LQ \wedge \mathsf{E}\ \mathsf{G}\ \neg LA$



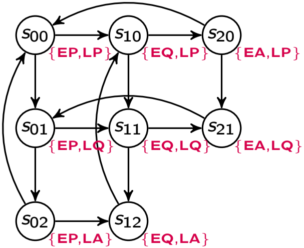
- *Step 3: compute* $LQ \wedge \mathsf{E}\ \mathsf{G}\ \neg LA$
  - $LQ \wedge \mathsf{E}\ \mathsf{G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$

Example: demanding children



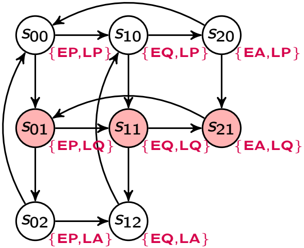
- *Step 3: compute* $LQ \wedge \mathrm{E\ G}\ \neg LA$
  - $LQ \wedge \mathrm{E\ G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$

Example: demanding children



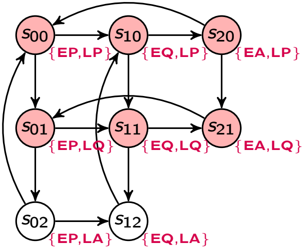
- *Step 3: compute* $LQ \wedge \mathrm{E}\ \mathrm{G}\ \neg LA$
  - $LQ \wedge \mathrm{E}\ \mathrm{G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \wedge \mathrm{E}\ \mathrm{G}\ \neg LA) \cup \mathrm{E}\ \mathrm{X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)

Example: demanding children



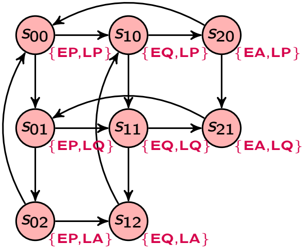
- *Step 3: compute* $LQ \wedge \mathrm{E\ G} \neg LA$
  - $LQ \wedge \mathrm{E\ G} \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \wedge \mathrm{E\ G} \neg LA) \cup \mathrm{E\ X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)
  - Add states that satisfy $LQ \wedge \mathrm{E\ G} \neg LA$

Example: demanding children



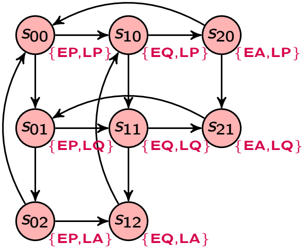
- *Step 3: compute* $LQ \wedge \mathrm{E\ G}\ \neg LA$
  - $LQ \wedge \mathrm{E\ G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)
  - Add states that satisfy $LQ \wedge \mathrm{E\ G}\ \neg LA$ and states that go there...

Example: demanding children



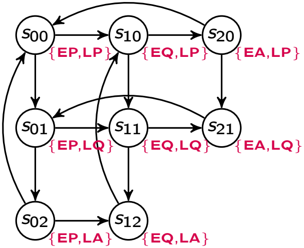
- *Step 3: compute* $LQ \land \mathrm{E\ G}\ \neg LA$
  - $LQ \land \mathrm{E\ G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \land \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)
  - Add states that satisfy $LQ \land \mathrm{E\ G}\ \neg LA$ and states that go there...and again...

Example: demanding children



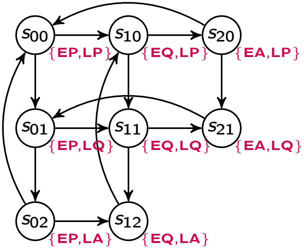
- *Step 3: compute* $LQ \land \mathrm{E\ G} \neg LA$
  - $LQ \land \mathrm{E\ G} \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \land \mathrm{E\ G} \neg LA) \cup \mathrm{E\ X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)
  - Add states that satisfy $LQ \land \mathrm{E\ G} \neg LA$ and states that go there...and again...
- *Step 5: compute negation of* $\mu Y.((LQ \land \mathrm{E\ G} \neg LA) \cup \mathrm{E\ X}\ Y)$
  - $\mu Y.((LQ \land \mathrm{E\ G} \neg LA) \cup \mathrm{E\ X}\ Y)$ holds everywhere

Example: demanding children



- *Step 3: compute* $LQ \wedge \mathrm{E\ G}\ \neg LA$
  - $LQ \wedge \mathrm{E\ G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)
  - Add states that satisfy $LQ \wedge \mathrm{E\ G}\ \neg LA$ and states that go there...and again...
- *Step 5: compute negation of* $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$
  - $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$ holds everywhere
  - $\neg\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$ holds nowhere

Example: demanding children



- *Step 3: compute* $LQ \wedge \mathrm{E\ G}\ \neg LA$
  - $LQ \wedge \mathrm{E\ G}\ \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- *Step 4: compute* $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$
  - Least fixpoint, so start with approximating from false (i.e. no states)
  - Add states that satisfy $LQ \wedge \mathrm{E\ G}\ \neg LA$ and states that go there...and again...
- *Step 5: compute negation of* $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$
  - $\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$ holds everywhere
  - $\neg\mu Y.((LQ \wedge \mathrm{E\ G}\ \neg LA) \cup \mathrm{E\ X}\ Y)$ holds nowhere ← ANSWER

Conclusion:

- So, $\mathsf{A\ G}\ (LQ \rightarrow \mathsf{A\ F}\ LA)$ holds in no state
- The requirement does not hold for the full Kripke Structure
- Why? Because in this case, there is a path in which only Emma progresses while Linus is not being served.
- Next, we look at the Kripke Structure with Fairness Constraints

Outline

Temporal Logics: Fairness

Sometimes properties are violated by "unrealistic" paths only, for instance due to a scheduler. In this case, one may wish to restrict to fair paths.

A Kripke Structure over AP with fairness constraints is a structure $M = \langle S, R, L, F \rangle$, where:

- $\langle S, R, L \rangle$ is an "ordinary" Kripke Structure as before
- $F \subseteq 2^S$ is a set of fairness constraints

A path is fair if it "hits" each fairness constraint infinitely often:

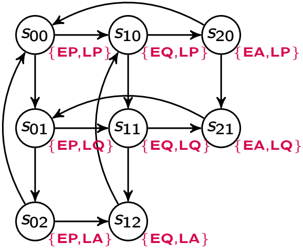
$$\text{fair}(\pi) \ \ \textit{iff} \ \ \forall C \in F.\ \{i \mid \pi(i) \in C\} \ \text{ is an infinite set}$$

In CTL* with fairness semantics ($\models_F$), only fair paths will be considered.

Given a fixed Kripke Structure with fairness constraints $M = \langle S, R, L, F \rangle$, $s \models_F f$ means: formula f holds in state s in the fair CTL* semantics.

The definition of $\models_F$ coincides with $\models$ except for the following four clauses:

$s \models_F \text{true}$	iff	there is some fair path starting in s
$s \models_F p$	iff	$p \in L(s)$ and there is some fair path starting in s
$s \models_F \text{A } f$	iff	for all fair paths π starting in s, we have $\pi \models_F f$
$s \models_F \text{E } f$	iff	for some fair path π starting in s, we have $\pi \models_F f$



- To exclude runs in which one child gets all attention, we want that both $\neg EQ$ as well as $\neg LQ$ hold infinitely often
- fairness constraints ensuring this: $F = \{\{s_{00}, s_{01}, s_{02}, s_{20}, s_{21}\}, \{s_{00}, s_{10}, s_{20}, s_{02}, s_{12}\}\}$
- Check whether A G ($LQ \to$ A F LA) holds fairly!

Outline

Technische Universiteit Eindhoven University of Technology

Fair Symbolic Model Checking

Fix a fair Kripke Structure $M = \langle S, R, L, \{F_1, \ldots, F_n\}\rangle$

Recall that a fair path infinitely often hits some state from each fairness constraint F_i

- First, note that in fair CTL (with $\models_F$),

$$\mathsf{E\,G}\, f \equiv f \wedge \bigwedge_{k=1}^{n} \mathsf{E\,X\,E}\,[f \;\mathsf{U}\; (F_k \wedge \mathsf{E\,G}\, f)] \qquad \text{(prove } \subseteq \text{ and } \supseteq\text{)}$$

- Next, if

$$Z \equiv f \wedge \bigwedge_{k=1}^{n} \mathsf{E\,X\,E}\,[f \;\mathsf{U}\; (F_k \wedge Z)]$$

  Then $Z \subseteq \mathsf{E\,G}\, f$ (construct a path cycling through $F_1, \ldots, F_n$)

- Hence, we found:

$$\mathsf{E\,G}\, f \equiv \nu Z. f \wedge \bigwedge_{k=1}^{n} \mathsf{E\,X\,E}\,[f \;\mathsf{U}\; (F_k \wedge Z)]$$

The equivalence

$$\mathrm{E\ G}\ f \equiv \nu Z. f \wedge \bigwedge_{k=1}^{n} \mathrm{E\ X\ E}\,[f\ \mathrm{U}\ (F_k \wedge Z)]$$

leads to the following algorithm:

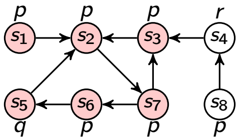
$\mathrm{check}_F(\mathrm{E\ G}\ f)$	$\mathrm{gfp}\big(Z \mapsto \mathrm{check}(f \wedge \bigwedge_{k=1}^{n} \mathrm{E\ X\ (E}\,[f\ \mathrm{U}\ (F_k \wedge Z)])\big)$

So, in the greatest fixed point computation for E G , we perform nested least fixed point computations to compute E [U].

Next, we can compute an OBDD *fair* $:= \mathrm{check}_F(\mathrm{E\ G\ true})$. The remaining temporal operators can then be encoded as follows:

$\mathrm{check}_F(\mathrm{E\ X}\ f)$	$\mathrm{check}(\mathrm{E\ X}\ (f \wedge \mathit{fair}))$
$\mathrm{check}_F(\mathrm{E}\,[f\ \mathrm{U}\ g])$	$\mathrm{check}(\mathrm{E}\,[f\ \mathrm{U}\ (g \wedge \mathit{fair})])$

Example



- To check: $E\,[p \mathbin{U} q]$
- Fairness constraint: $\{\neg r\}$
- Compute $\mathit{fair} := \text{check}_F(\text{E G true})$ $(= S)$
- Compute: $\mu Z.(q \wedge \mathit{fair}) \vee (p \wedge \text{E X } Z)$ (with lfp)

$$
\begin{aligned}
Z_0 &= \text{false} = \emptyset \\
Z_1 &= q \vee (p \wedge \text{E X } Z_0) = \{s_5\} \\
Z_2 &= q \vee (p \wedge \text{E X } Z_1) = \{s_5, s_6\} \\
Z_3 &= q \vee (p \wedge \text{E X } Z_2) = \{s_5, s_6, s_7\} \\
Z_4 &= q \vee (p \wedge \text{E X } Z_3) = \{s_2, s_5, s_6, s_7\} \\
Z_5 &= q \vee (p \wedge \text{E X } Z_4) = \{s_1, s_2, s_3, s_5, s_6, s_7\} \\
Z_6 &= q \vee (p \wedge \text{E X } Z_5) = \{s_1, s_2, s_3, s_5, s_6, s_7\}
\end{aligned}
$$

$Z_5 = Z_6$, so this is the least fixed point.

Outline

- Motivation:
  - In practice, a model checker is often used as an extended debugger
  - If a bug is found, the model checker should provide a particular trace, which shows it
- A formula with a universal path quantifier has a counterexample consisting of one trace
- A formula with an existential path quantifier has a witness consisting of one trace
- Due to the dualities in CTL, we only have to consider:
  - a finite trace witnessing $\mathsf{E}\,[f\ \mathsf{U}\ g]$
  - an infinite trace witnessing $\mathsf{E}\,\mathsf{G}\,f$; for finite systems, the latter is a so-called lasso, consisting of a prefix and a loop
- For fair counter examples we require that the loop contains a state from each fairness constraint

- $\mathsf{E}\,[f \;\mathsf{U}\; g] = \mu Z.\; g \vee (f \wedge \mathsf{E}\;\mathsf{X}\; Z)$
- Unfolding the recursion, we get:

$$\begin{array}{lcl} Z_0 & = & \text{false} \\ Z_1 & = & g \\ Z_2 & = & g \vee (f \wedge \mathsf{E}\;\mathsf{X}\; g) \\ Z_3 & = & g \vee (f \wedge \mathsf{E}\;\mathsf{X}\; (g \vee (f \wedge \mathsf{E}\;\mathsf{X}\; g))) \end{array}$$

- So, the fixed point computation corresponds to a backward reachability analysis
- Z_i contains those states that can reach g in at most $i-1$ steps (and f holds in between).
- Assume $s_0 \models \mathsf{E}\,[f \;\mathsf{U}\; g]$. To find a minimal witness from state s_0, we start in the smallest N such that $s_0 \in Z_N$.
- For $i \in 1, \ldots, N{-}1$, we define s_i to be a state in Z_{N-i} satisfying $s_{i-1}\; R\; s_i$.

- We want an initial path to a cycle on which each fairness constraint $\{F_1, \dots, F_n\}$ occurs (i.e. the cycle must contain at least one state from all F_i).
- $\text{E G } f = \nu Z. f \wedge \bigwedge_{k=1}^{n} \text{E X } \text{E } [f \text{ U } (F_k \wedge Z)]$
- Unfolding the recursion, we get:

$$\begin{aligned} Z_0 &= \text{true} \\ &\dots \\ Z_L &= f \wedge \bigwedge_{k=1}^{n} \text{E X } \text{E } [f \text{ U } (F_k \wedge Z_{L-1})] \end{aligned}$$

- Let $Z := Z_L = Z_{L-1} = \text{E G } f$ be the fixed point
- To compute Z, we compute for each k $(1 \leq k \leq n)$, $\text{E } [f \text{ U } (F_k \wedge Z)]$ using backward reachability. So, we have for each k the approximations: $Q_0^k \subseteq Q_1^k \subseteq Q_2^k \subseteq \dots \subseteq Q_{j_k}^k$
- From the E [U] case, recall that Q_i^k contains those states that can reach $F_k \wedge Z$ in at most i steps

- Assume $s_0 \models_F \text{E G } f$, hence, $s_0 \in Z$
- We will now inductively construct a path $s_0 \to^* s_1 \to^* \ldots \to^* s_n$, such that:
  - f holds along the whole path
  - $s_k \in Z \wedge F_k$ (for $1 \leq k \leq n$)
- Observe: by induction $s_{k-1} \models Z$, so, by definition of Z: $s_{k-1} \models \text{E X } \text{E} [f \text{ U } (Z \wedge F_k)]$
- For $1 \leq k \leq n$ do:
  1. Determine the minimal M such that s_{k-1} has a successor $t_0^k \in Q_M^k$.
  2. Construct (as the witness for E [U]):

     $$s_{k-1} \to t_0^k \to \cdots \to t_M^k \in Z \wedge F_k$$

  3. Define $s_k := t_M^k$.
- heuristic improvement: Visit the F_k in a different order: continue with the closest F_k that has not yet been visited.

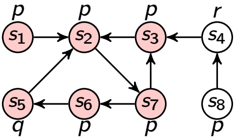
- Finally, we must close the loop, but this is not always possible: Check if $s_n \models \text{E X } \text{E} [f \text{ U } \{s_1\}]$.
- If so: the E [U]-witness closes the loop
- If not: the cycle cannot be closed. Hence:
  - The sequence so far $s_0 \to \cdots \to s_n$ is in the prefix of the lasso, not yet on the loop.
  - Restart the whole procedure of the previous slide, now starting in $s_n \in Z$.
- Eventually, this process must terminate:
  - We only restart if s_n cannot reach s_1
  - so we moved to the next Strongly Connected Component
  - The SCC graph cannot contain cycles
- Optimisation: By precomputing $\text{E} [f \text{ U } \{s_1\}]$, one can detect earlier that closing the cycle will not be possible.

Outline

Technische Universiteit Eindhoven University of Technology

Exercise

Example



- Check that $s_1 \models_F \mathrm{E\,G}\,(p \vee q)$
- Fairness constraint: $\neg r$ and q
- Construct a witness for $s_1 \models_F \mathrm{E\,G}\,(p \vee q)$