

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 6.073

Symbolic Model Checking

Fairness for CTL

Fair Symbolic Model Checking

Counterexamples and Witnesses

Witnesses for $E [U]$

Witnesses for fair $E G$

Exercise

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:

- ▶ $E F g = \mu Z. g \vee E X Z$
- ▶ $E G f = \nu Z. f \wedge E X Z$
- ▶ $E [f U g] = \mu Z. g \vee (f \wedge E X Z)$

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

Intuition:

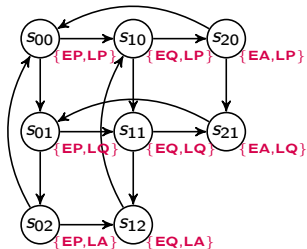
- ▶ Eventually least fixed points
- ▶ Globally greatest fixed points

CTL model checking with Fixed Points

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

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

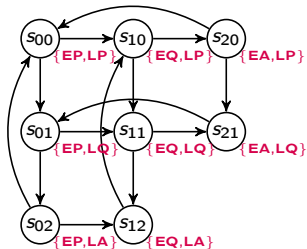
Recall: $\text{Pre}_R(Z) = \{s \in S \mid \exists t \in Z. s R t\}$



- ▶ 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: $A G (LQ \rightarrow A F LA)$



- ▶ 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

$$A G (LQ \rightarrow A F LA)$$

$$\equiv$$

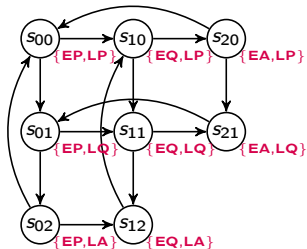
$$\neg E [\text{true} U \neg(\neg LQ \vee \neg E G \neg LA)]$$

$$\equiv$$

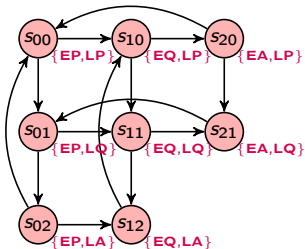
$$\neg E [\text{true} U (LQ \wedge E G \neg LA)]$$

$$\equiv$$

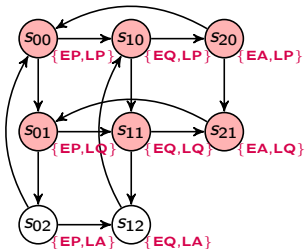
$$\neg \mu Y. ((LQ \wedge E G \neg LA) \cup E X Y)$$



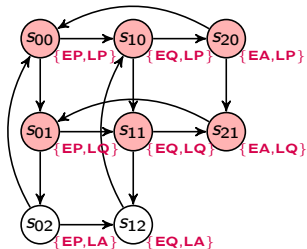
- ▶ Step 2: compute $\text{check}(E \text{ G } \neg LA)$, i.e., compute $\nu Z.(\neg LA \wedge E X Z)$.



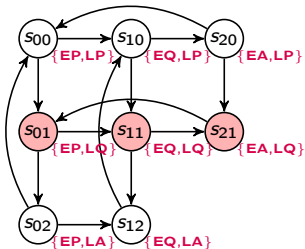
- ▶ Step 2: compute $\text{check}(E \text{ G } \neg LA)$, i.e., compute $\nu Z.(\neg LA \wedge E X Z)$.
 - Greatest fixpoint, so start with approximating from true (i.e. all states)



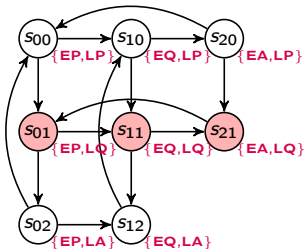
- ▶ Step 2: compute $\text{check}(E \text{ G } \neg LA)$, i.e., compute $\nu Z.(\neg LA \wedge E X Z)$.
 - Greatest fixpoint, so start with approximating from true (i.e. all states)
 - Stable at $\{s_{00}, s_{10}, s_{20}, s_{01}, s_{11}, s_{21}\}$



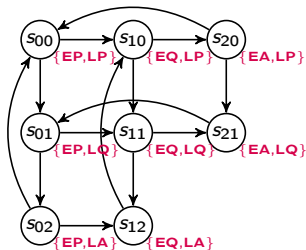
- ▶ Step 3: compute $LQ \wedge EG \neg LA$



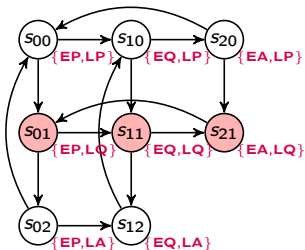
- ▶ Step 3: compute $LQ \wedge EG \neg LA$
 - $LQ \wedge EG \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$



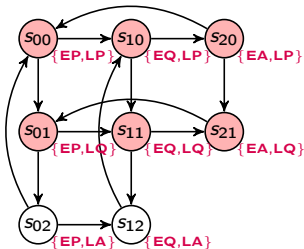
- ▶ Step 3: compute $LQ \wedge E G \neg LA$
 - $LQ \wedge E G \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- ▶ Step 4: compute $\mu Y.((LQ \wedge E G \neg LA) \cup E X Y)$



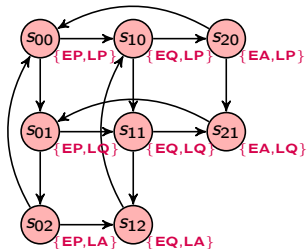
- ▶ Step 3: compute $LQ \wedge E G \neg LA$
 - $LQ \wedge E G \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- ▶ Step 4: compute $\mu Y. ((LQ \wedge E G \neg LA) \cup E X Y)$
 - Least fixpoint, so start with approximating from false (i.e. no states)



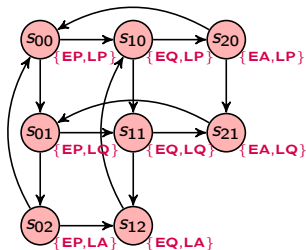
- ▶ Step 3: compute $LQ \wedge E G \neg LA$
 - $LQ \wedge E G \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- ▶ Step 4: compute $\mu Y.((LQ \wedge E G \neg LA) \cup E X Y)$
 - Least fixpoint, so start with approximating from false (i.e. no states)
 - Add states that satisfy $LQ \wedge E G \neg LA$



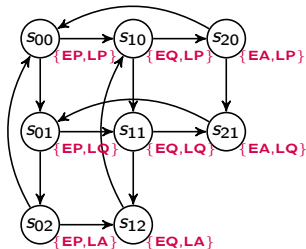
- ▶ Step 3: compute $LQ \wedge E G \neg LA$
 - $LQ \wedge E G \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- ▶ Step 4: compute $\mu Y.((LQ \wedge E G \neg LA) \cup E X Y)$
 - Least fixpoint, so start with approximating from false (i.e. no states)
 - Add states that satisfy $LQ \wedge E G \neg LA$ and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



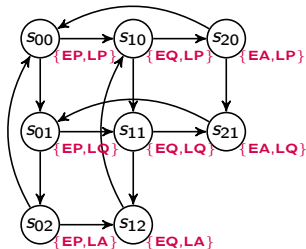
- ▶ Step 3: compute $LQ \wedge E G \neg LA$
 - $LQ \wedge E G \neg LA$ holds in $\{s_{01}, s_{11}, s_{21}\}$
- ▶ Step 4: compute $\mu Y.((LQ \wedge E G \neg LA) \cup E X Y)$
 - Least fixpoint, so start with approximating from false (i.e. no states)
 - Add states that satisfy $LQ \wedge E G \neg LA$ and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



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



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



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

Conclusion:

- ▶ So, $A G (LQ \rightarrow A F LA)$ 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

Symbolic Model Checking

Fairness for CTL

Fair Symbolic Model Checking

Counterexamples and Witnesses

Witnesses for $E [U]$

Witnesses for fair $E G$

Exercise

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:

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

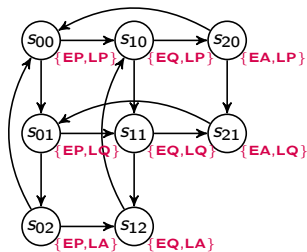
In CTL* with fairness semantics (\models_F), only fair paths will be considered.

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
- $s \models_F A f$ iff for all **fair** paths π starting in s , we have $\pi \models_F f$
- $s \models_F E f$ iff for some **fair** path π starting in s , we have $\pi \models_F f$

Write \bar{f} if we mean f /wish to compute f under fairness constraints



- ▶ 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 $A G (LQ \rightarrow A F LA)$ holds fairly!

Symbolic Model Checking

Fairness for CTL

Fair Symbolic Model Checking

Counterexamples and Witnesses

Witnesses for $E [U]$

Witnesses for fair $E G$

Exercise

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

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),

$$\overline{E G f} \equiv \bar{f} \wedge \bigwedge_{k=1}^n E X E [\bar{f} U (F_k \wedge \overline{E G f})] \quad (\text{prove } \subseteq \text{ and } \supseteq)$$

- ▶ Next, if

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

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

- ▶ Hence, we found:

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

The equivalence

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

leads to the following algorithm:

$$\text{check}_F(E G f) \quad \text{gfp}(Z \mapsto \text{check}(\bar{f} \cap \bigwedge_{k=1}^n E X (E [\bar{f} U (F_k \wedge Z)])))$$

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

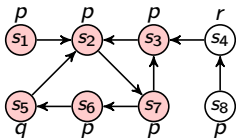
Next, we can compute $\text{fair} := \text{gfp}(Z \mapsto \text{check}(\bigwedge_{k=1}^n E X (E [true U (F_k \wedge Z)])))$.

The remaining temporal operators can then be encoded as follows:

$\text{check}_F(p)$	$\text{check}(p) \cap \text{fair}$
$\text{check}_F(E X f)$	$\text{check}(E X (\bar{f} \wedge \text{fair}))$
$\text{check}_F(E [f U g])$	$\text{check}(E [\bar{f} U (\bar{g} \wedge \text{fair})])$

Example

- ▶ To check: $E [p U q]$
- ▶ Fairness constraint: $\{\neg r\}$
- ▶ Compute *fair* ($= S$)
- ▶ Compute: $\mu Z. (\bar{q} \wedge \text{fair}) \vee (\bar{p} \wedge E X Z)$ (with lfp)



$$Z_0 = \text{false} = \emptyset$$

$$Z_1 = q \vee (p \wedge E X Z_0) = \{s_5\}$$

$$Z_2 = q \vee (p \wedge E X Z_1) = \{s_5, s_6\}$$

$$Z_3 = q \vee (p \wedge E X Z_2) = \{s_5, s_6, s_7\}$$

$$Z_4 = q \vee (p \wedge E X Z_3) = \{s_2, s_5, s_6, s_7\}$$

$$Z_5 = q \vee (p \wedge E X Z_4) = \{s_1, s_2, s_3, s_5, s_6, s_7\}$$

$$Z_6 = q \vee (p \wedge E X Z_5) = \{s_1, s_2, s_3, s_5, s_6, s_7\}$$

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

Symbolic Model Checking

Fairness for CTL

Fair Symbolic Model Checking

Counterexamples and Witnesses

Witnesses for $E [U]$

Witnesses for fair $E G$

Exercise

- ▶ 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 $E [f U g]$
 - an infinite trace witnessing $E G f$; 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

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

$$Z_0 = \text{false}$$

$$Z_1 = g$$

$$Z_2 = g \vee (f \wedge E X g)$$

$$Z_3 = g \vee (f \wedge E X (g \vee (f \wedge E X g)))$$

- ▶ 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 $s_0 \models E [f U g]$. To find a minimal witness from state s_0 , we start in the smallest N such that $s_0 \in Z_N$.
- ▶ For $i \in 1, \dots, N-1$, 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).

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

- ▶ Unfolding the recursion, we get:

$$Z_0 = \text{true}$$

...

$$Z_L = f \wedge \bigwedge_{k=1}^n E X E [f U (F_k \wedge Z_{L-1})]$$

- ▶ Let $Z := Z_L = Z_{L-1} = E G f$ be the fixed point
- ▶ To compute Z , we compute for each k ($1 \leq k \leq n$), $E [f U (F_k \wedge Z)]$ 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 $s_0 \models_F E G f$, hence, $s_0 \in Z$
- ▶ We will now inductively construct a path $s_0 \rightarrow^* s_1 \rightarrow^* \dots \rightarrow^* s_n$, 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 : $s_{k-1} \models E X E [f U (Z \wedge F_k)]$
- ▶ 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 $E [U]$):

$$s_{k-1} \rightarrow t_0^k \rightarrow \dots \rightarrow t_M^k \in Z \wedge F_k$$
 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 $s_n \models E X E [f U \{s_1\}]$.
- ▶ If so: the E [U]-witness closes the loop
- ▶ If not: the cycle cannot be closed. Hence:
 - The sequence so far $s_0 \rightarrow \dots \rightarrow s_n$ 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 $E [f U \{s_1\}]$, one can detect **earlier** that closing the cycle will not be possible.

Symbolic Model Checking

Fairness for CTL

Fair Symbolic Model Checking

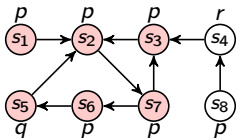
Counterexamples and Witnesses

Witnesses for $E [U]$

Witnesses for fair $E G$

Exercise

Example



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