System Specification, Verification and Synthesis (SSVS) – CS 4830/7485, Fall 2019

15: Formal Verification: CTL Model Checking

Stavros Tripakis

Northeastern University
Khoury College of Computer Sciences

Solving the general model-checking problem

We know how to model check LTL and CTL formulas of the form

$$\mathbf{G}\psi \qquad \text{or} \qquad \mathbf{AG}\psi$$

where ψ is a propositional formula: we do this by reachability analysis.

But how can we model-check **arbitrary** LTL and CTL formulas?

We first look at CTL. Then at LTL.

CTL Model-Checking

Recall: the model-checking problem for CTL

Given:

- the implementation: a transition system (Kripke structure) $M = (\mathsf{AP}, S, S_0, L, R)$
- the specification: a CTL formula ϕ

check where M satisfies ϕ:

$$M \overset{?}{\models} \phi$$

i.e., check whether for **every initial state of** M **satisfies** ϕ:

$$\forall s \in S_0 : s \models \phi \quad ?$$

Recall: the model-checking problem for CTL

Given:

- the implementation: a transition system (Kripke structure) $M = (\mathsf{AP}, S, S_0, L, R)$
- the specification: a CTL formula ϕ

check where M satisfies ϕ:

$$M \overset{?}{\models} \phi$$

i.e., check whether for **every initial state of M satisfies ϕ**:

$$\forall s \in S_0 : s \models \phi \quad ?$$

We will assume that M is finite and has no deadlock states.
What if M has deadlocks? $\to$ **Homework.**

CTL model-checking: basic idea

1. Compute $[\![\phi]\!]$: the set of all states satisfying ϕ.
   (Note that $[\![\phi]\!]$ may contain unreachable states. That's OK.)
2. Check that $S_0 \subseteq [\![\phi]\!]$: every initial state satisfies ϕ.

CTL model-checking: basic idea

1. Compute $[\![\phi]\!]$: the set of all states satisfying ϕ.
   (Note that $[\![\phi]\!]$ may contain unreachable states. That's OK.)

2. Check that $S_0 \subseteq [\![\phi]\!]$: every initial state satisfies ϕ.
   How can we implement this test symbolically?
   E.g., if S_0 and $[\![\phi]\!]$ are implemented as BDDs B_{S_0} and $B_{[\![\phi]\!]}$.

CTL model-checking: basic idea

1. Compute $[\![\phi]\!]$: the set of all states satisfying ϕ.
   (Note that $[\![\phi]\!]$ may contain unreachable states. That's OK.)
2. Check that $S_0 \subseteq [\![\phi]\!]$: every initial state satisfies ϕ.
   How can we implement this test symbolically?
   E.g., if S_0 and $[\![\phi]\!]$ are implemented as BDDs B_{S_0} and $B_{[\![\phi]\!]}$.
   Check whether $B_{S_0} \Rightarrow B_{[\![\phi]\!]}$ is valid, i.e., whether $B_{S_0} \wedge \neg B_{[\![\phi]\!]}$ is unsatisfiable. Amounts to checking that $S_0 \cap \overline{[\![\phi]\!]} = \emptyset$.

CTL model-checking: basic idea

1. Compute $[\![\phi]\!]$: the set of all states satisfying ϕ.
   (Note that $[\![\phi]\!]$ may contain unreachable states. That's OK.)
2. Check that $S_0 \subseteq [\![\phi]\!]$: every initial state satisfies ϕ.
   How can we implement this test symbolically?
   E.g., if S_0 and $[\![\phi]\!]$ are implemented as BDDs B_{S_0} and $B_{[\![\phi]\!]}$.
   Check whether $B_{S_0} \Rightarrow B_{[\![\phi]\!]}$ is valid, i.e., whether $B_{S_0} \land \neg B_{[\![\phi]\!]}$ is unsatisfiable. Amounts to checking that $S_0 \cap \overline{[\![\phi]\!]} = \emptyset$.

We will compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. Compute $[\![\psi]\!]$ for every **subformula** ψ of ϕ: **bottom-up** on the syntax tree of ϕ.
2. Combine the results to obtain $[\![\phi]\!]$.

Computing $[\![\phi]\!]$

Assume the transition system is $(\mathsf{AP}, S, S_0, R, L)$.

Compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] =$

Computing $[\![\phi]\!]$

Assume the transition system is $(\mathsf{AP}, S, S_0, R, L)$.

Compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] =$

Computing $[\![\phi]\!]$

Assume the transition system is $(\mathsf{AP}, S, S_0, R, L)$.

Compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$

Computing $[\![\phi]\!]$

Assume the transition system is $(\mathsf{AP}, S, S_0, R, L)$.

Compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] =$

Computing $[\![\phi]\!]$

Assume the transition system is $(\mathsf{AP}, S, S_0, R, L)$.

Compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$

Recall that:

$$\mathbf{pre}(X) = \{s \in S \mid \exists s' \in X : s \longrightarrow s'\}$$

that is, $\mathbf{pre}(X)$ is the set of 1-step **predecessors** of states in X.

Computing $[\![\phi]\!]$

Assume the transition system is $(\mathsf{AP}, S, S_0, R, L)$.

Compute $[\![\phi]\!]$ recursively based on the syntax of ϕ:

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$

Recall that:

$$\mathbf{pre}(X) = \{s \in S \mid \exists s' \in X : s \longrightarrow s'\}$$

that is, $\mathbf{pre}(X)$ is the set of 1-step **predecessors** of states in X.
We will see later how to compute $\mathbf{pre}(X)$ symbolically.

Computing $[\![\mathbf{EF}\phi]\!]$

How to compute $[\![\mathbf{EF}\phi]\!]$?

Computing $[\![\mathbf{EF}\phi]\!]$

How to compute $[\![\mathbf{EF}\phi]\!]$?

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

Computing $[\![\mathbf{EF}\phi]\!]$

How to compute $[\![\mathbf{EF}\phi]\!]$?

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

Computing $[\![\mathbf{EF}\phi]\!]$

How to compute $[\![\mathbf{EF}\phi]\!]$?

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

This is like the symbolic reachability algorithm, except we are going backwards.

Will the iteration terminate? Why?

Computing $[\![\mathbf{EF}\phi]\!]$

How to compute $[\![\mathbf{EF}\phi]\!]$?

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

This is like the symbolic reachability algorithm, except we are going backwards.

Will the iteration terminate? Why?

Because the state-space is finite.

Fixpoints

Fixpoints on monotonic functions on powersets

Let $F : 2^S \to 2^S$ be a function from sets of states to sets of states.

A **fixpoint** of F is a set of states $X \subseteq S$, such that

$$F(X) = X$$

Fixpoints on monotonic functions on powersets

Let $F : 2^S \rightarrow 2^S$ be a function from sets of states to sets of states.

A **fixpoint** of F is a set of states $X \subseteq S$, such that

$$F(X) = X$$

Suppose F is **monotonic**, i.e.,

$$X_1 \subseteq X_2 \Rightarrow F(X_1) \subseteq F(X_2)$$

for any X_1, X_2.

Fixpoints on monotonic functions on powersets

Let $F : 2^S \to 2^S$ be a function from sets of states to sets of states.

A **fixpoint** of F is a set of states $X \subseteq S$, such that

$$F(X) = X$$

Suppose F is **monotonic**, i.e.,

$$X_1 \subseteq X_2 \Rightarrow F(X_1) \subseteq F(X_2)$$

for any X_1, X_2.

Then F has a **least fixpoint** X^*, meaning that:

- X^* is a fixpoint of F: $F(X^*) = X^*$
- X^* is the least fixpoint of F: for any X, if $F(X) = X$ then $X^* \subseteq X$.

Fixpoints on monotonic functions on powersets

Let $F : 2^S \to 2^S$ be a function from sets of states to sets of states.

A **fixpoint** of F is a set of states $X \subseteq S$, such that

$$F(X) = X$$

Suppose F is **monotonic**, i.e.,

$$X_1 \subseteq X_2 \Rightarrow F(X_1) \subseteq F(X_2)$$

for any X_1, X_2.

Then F has a **least fixpoint** X^*, meaning that:

- X^* is a fixpoint of F: $F(X^*) = X^*$
- X^* is the least fixpoint of F: for any X, if $F(X) = X$ then $X^* \subseteq X$.

X^* is often denoted $\mathbf{lfp}F$ or μF.

Computing least fixpoints iteratively

X^* can be computed by starting from the empty set and applying F repeatedly:

$$\emptyset \subseteq F(\emptyset) \subseteq F(F(\emptyset)) \subseteq F^3(\emptyset) \subseteq \cdots \subseteq F^n(\emptyset) \subseteq \cdots$$

(When) does this iteration terminate?

Computing least fixpoints iteratively

X^* can be computed by starting from the empty set and applying F repeatedly:

$$\emptyset \subseteq F(\emptyset) \subseteq F(F(\emptyset)) \subseteq F^3(\emptyset) \subseteq \cdots \subseteq F^n(\emptyset) \subseteq \cdots$$

(When) does this iteration terminate?

It terminates when $F^{n+1}(\emptyset) = F^n(\emptyset)$.

When S is finite, this is bound to happen.
In fact, $F^{n+1}(\emptyset) = F^n(\emptyset)$ for some $n \leq |S|$.

Theorem

Let S be a finite set. Let $n = |S|$. Let $F : 2^S \to 2^S$ be a monotonic function on the powerset of S (i.e., $X_1 \subseteq X_2 \Rightarrow F(X_1) \subseteq F(X_2)$). Then:

1. *F has a least fixpoint X^*.*
2. *$X^* = F^n(\emptyset)$.*

Proof.

$\emptyset \subseteq F(\emptyset)$, since the emptyset is a subset of any other set. By monotonicity of F, $F(\emptyset) \subseteq F(F(\emptyset)) = F^2(\emptyset)$. Continuing the same way, we can prove by induction that $F^i(\emptyset) \subseteq F^{i+1}(\emptyset)$ for all $i = 0, 1, 2, \ldots$. Since S is finite, for some $i \leq n$, it must be that $F^i(\emptyset) = F^{i+1}(\emptyset)$. Therefore, for all $i \leq j \leq n$, it must be that $F^j(\emptyset) = F^{j+1}(\emptyset)$. Thus, it must also be that $F^n(\emptyset) = F^{n+1}(\emptyset)$. Let $X^* = F^n(\emptyset)$. By construction, $F(X^*) = F^{n+1}(\emptyset) = F^n(\emptyset) = X^*$, i.e., X^* is a fixpoint of F.

We next show that X^* is the least fixpoint of F. Suppose X is another fixpoint of F, i.e., $F(X) = X$. $\emptyset \subseteq X$, since the emptyset is a subset of any set. By monotonicity of F, $F(\emptyset) \subseteq F(X)$, and since $F(X) = X$, $F(\emptyset) \subseteq X$. Continuing the same way, we can prove by induction that $F^i(\emptyset) \subseteq X$ for all $i = 0, 1, 2, \ldots$. Thus, $X^* = F^n(\emptyset) \subseteq X$. □

CTL Model-Checking continued

Computing $[\![\mathbf{EF}\phi]\!]$

Recall:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

Computing $[\![\mathbf{EF}\phi]\!]$

Recall:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

But also:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\,\mathbf{EF}\phi$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\mathbf{EF}\phi]\!])$$

This looks like a fixpoint equation! What is the function F?

Computing $[\![\mathbf{EF}\phi]\!]$

Recall:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

But also:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\,\mathbf{EF}\phi$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\mathbf{EF}\phi]\!])$$

This looks like a fixpoint equation! What is the function F?

$$F(X) = [\![\phi]\!] \cup \mathbf{pre}(X)$$

Computing $[\![\mathbf{EF}\phi]\!]$

Recall:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

But also:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\,\mathbf{EF}\phi$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\mathbf{EF}\phi]\!])$$

This looks like a fixpoint equation! What is the function F?

$$F(X) = [\![\phi]\!] \cup \mathbf{pre}(X)$$

Is F monotonic?

Computing $[\![\mathbf{EF}\phi]\!]$

Recall:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\phi \vee \mathbf{EX}\,\mathbf{EX}\phi \vee \cdots$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots$$

But also:

$$\mathbf{EF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{EX}\,\mathbf{EF}\phi$$

therefore

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\mathbf{EF}\phi]\!])$$

This looks like a fixpoint equation! What is the function F?

$$F(X) = [\![\phi]\!] \cup \mathbf{pre}(X)$$

Is F monotonic? Yes: follows from the fact that $\mathbf{pre}$ is monotonic.

Computing $[\![\mathbf{EF}\phi]\!]$

$$F(X) = [\![\phi]\!] \cup \mathbf{pre}(X)$$

To compute the least fixpoint of F, we need to compute the sequence:

$$\begin{array}{lllll}
X_0 & = & \emptyset & & \\
X_1 & = & F(X_0) & = & [\![\phi]\!] \cup \mathbf{pre}(\emptyset) \\
 & & & = & [\![\phi]\!] \qquad \text{Why?} \\
X_2 & = & F(X_1) & = & [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \\
X_3 & = & F(X_2) & = & [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!])) \\
 & & & = & [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \qquad \text{Why?} \\
\ldots & & & &
\end{array}$$

so that

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots = \mathbf{lfp} F$$

Computing $[\![\mathbf{EF}\phi]\!]$

$$F(X) = [\![\phi]\!] \cup \mathbf{pre}(X)$$

To compute the least fixpoint of F, we need to compute the sequence:

$$\begin{array}{lllll}
X_0 & = & \emptyset & & \\
X_1 & = & F(X_0) & = & [\![\phi]\!] \cup \mathbf{pre}(\emptyset) \\
 & & & = & [\![\phi]\!] \qquad \text{Why?} \\
X_2 & = & F(X_1) & = & [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \\
X_3 & = & F(X_2) & = & [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!])) \\
 & & & = & [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \qquad \text{Why?} \\
\ldots & & & &
\end{array}$$

so that

$$[\![\mathbf{EF}\phi]\!] = [\![\phi]\!] \cup \mathbf{pre}([\![\phi]\!]) \cup \mathbf{pre}(\mathbf{pre}([\![\phi]\!])) \cup \cdots = \mathbf{lfp}F$$

Lambda notation for the fixpoint: $\mathbf{lfp}X.[\![\phi]\!] \cup \mathbf{pre}(X)$

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in$ AP: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
6. $[\![\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2)]\!] = ???$

Computing $[\![\mathbf{E}(\phi_1 \, \mathbf{U} \, \phi_2)]\!]$

$$\mathbf{E}(\phi_1 \, \mathbf{U} \, \phi_2) \quad \Leftrightarrow \quad \phi_2 \vee (\phi_1 \wedge \mathbf{EX} \, \mathbf{E}(\phi_1 \, \mathbf{U} \, \phi_2))$$

therefore

$$[\![\mathbf{E}(\phi_1 \, \mathbf{U} \, \phi_2)]\!] = \mathbf{lfp} X . [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$$

Computing $[\![\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2)]\!]$

$$\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2) \quad \Leftrightarrow \quad \phi_2 \vee (\phi_1 \wedge \mathbf{EX}\,\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2))$$

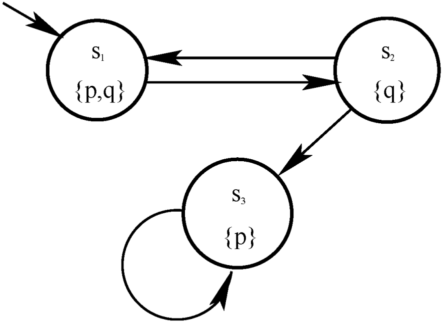
therefore

$$[\![\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$$

Iterative computation:

$$\begin{array}{lclcl} X_0 & = & \emptyset & & \\ X_1 & = & F(X_0) & = & [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(\emptyset)) \\ & & & = & [\![\phi_2]\!] \\ X_2 & = & F(X_1) & = & [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}([\![\phi_2]\!])) \\ \ldots & & & & \end{array}$$

Computing $[\![\mathbf{E}(\phi_1 \, \mathbf{U} \, \phi_2)]\!]$: example



- **Homework**: model-check $\mathbf{E}(p \, \mathbf{U} \, q)$.

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in$ AP: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \land \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
6. $[\![\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$
7. $[\![\mathbf{AX}\phi_1]\!] = ???$

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
6. $[\![\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$
7. $[\![\mathbf{AX}\phi_1]\!] = [\![\neg\mathbf{EX}\neg\phi_1]\!] = \overline{[\![\mathbf{EX}\neg\phi_1]\!]} = \overline{\mathbf{pre}([\![\neg\phi_1]\!])} = \overline{\mathbf{pre}(\overline{[\![\phi_1]\!]})}$

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in \mathsf{AP}$: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
6. $[\![\mathbf{E}(\phi_1 \,\mathbf{U}\, \phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$
7. $[\![\mathbf{AX}\phi_1]\!] = [\![\neg\mathbf{EX}\neg\phi_1]\!] = \overline{\mathbf{pre}(\overline{[\![\phi_1]\!]})}$
8. $[\![\mathbf{AG}\phi_1]\!] = [\![\neg\mathbf{EF}\neg\phi_1]\!] = \overline{\mathbf{lfp}X.\overline{[\![\phi_1]\!]} \cup \mathbf{pre}(X)}$

Computing $\llbracket \phi \rrbracket$ (continued)

1. For atomic proposition $p \in \mathsf{AP}$: $\llbracket p \rrbracket = \{s \in S \mid p \in L(s)\}$
2. $\llbracket \phi_1 \wedge \phi_2 \rrbracket = \llbracket \phi_1 \rrbracket \cap \llbracket \phi_2 \rrbracket$
3. $\llbracket \neg\phi_1 \rrbracket = \overline{\llbracket \phi_1 \rrbracket} = S - \llbracket \phi_1 \rrbracket$
4. $\llbracket \mathbf{EX}\phi_1 \rrbracket = \mathbf{pre}(\llbracket \phi_1 \rrbracket)$
5. $\llbracket \mathbf{EF}\phi_1 \rrbracket = \mathbf{lfp}X.\llbracket \phi_1 \rrbracket \cup \mathbf{pre}(X)$
6. $\llbracket \mathbf{E}(\phi_1 \,\mathbf{U}\, \phi_2) \rrbracket = \mathbf{lfp}X.\llbracket \phi_2 \rrbracket \cup (\llbracket \phi_1 \rrbracket \cap \mathbf{pre}(X))$
7. $\llbracket \mathbf{AX}\phi_1 \rrbracket = \llbracket \neg\mathbf{EX}\neg\phi_1 \rrbracket = \overline{\mathbf{pre}(\overline{\llbracket \phi_1 \rrbracket})}$
8. $\llbracket \mathbf{AG}\phi_1 \rrbracket = \llbracket \neg\mathbf{EF}\neg\phi_1 \rrbracket = \overline{\mathbf{lfp}X.\overline{\llbracket \phi_1 \rrbracket} \cup \mathbf{pre}(X)}$

Is there a more direct way to compute $\llbracket \mathbf{AX}\phi_1 \rrbracket$ and $\llbracket \mathbf{AG}\phi_1 \rrbracket$?

Computing $[\![\mathbf{AG}\phi]\!]$

$$\mathbf{AG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{AX}\,\mathbf{AG}\phi$$

therefore (?)

$$[\![\mathbf{AG}\phi]\!] = \mathbf{lfp}X.[\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{X})}$$

Computing $[\![\mathbf{AG}\phi]\!]$

$$\mathbf{AG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{AX}\,\mathbf{AG}\phi$$

therefore (?)

$$[\![\mathbf{AG}\phi]\!] = \mathbf{lfp}X.[\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{X})}$$

What is the least fixpoint here?

Computing $[\![\mathbf{AG}\phi]\!]$

$$\mathbf{AG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{AX}\,\mathbf{AG}\phi$$

therefore (?)

$$[\![\mathbf{AG}\phi]\!] = \mathbf{lfp}X.[\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{X})}$$

What is the least fixpoint here?

Iterative computation:

$$\begin{aligned}
X_0 &= \emptyset \\
X_1 &= F(X_0) = [\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{\emptyset})} \\
&= [\![\phi]\!] \cap \overline{\mathbf{pre}(S)} \\
&= [\![\phi]\!] \cap \overline{S} \qquad \text{Why?} \\
&= [\![\phi]\!] \cap \emptyset \\
&= \emptyset
\end{aligned}$$

Oops. What has gone wrong?

Computing $[\![\mathbf{AG}\phi]\!]$

$$\mathbf{AG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{AX}\,\mathbf{AG}\phi$$

tells us that $[\![\mathbf{AG}\phi]\!]$ is **a** fixpoint of the function

$$F(X) = [\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{X})}$$

but it does not tell us **which one**.

F may have **more than one fixpoints**: e.g., S or $\emptyset$.

Computing $[\![\mathbf{AG}\phi]\!]$

$$\mathbf{AG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{AX}\,\mathbf{AG}\phi$$

tells us that $[\![\mathbf{AG}\phi]\!]$ is **a** fixpoint of the function

$$F(X) = [\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{X})}$$

but it does not tell us **which one**.

F may have **more than one fixpoints**: e.g., S or $\emptyset$.

In this case the least fixpoint is $\emptyset$.
What we want instead is the greatest fixpoint.

Greatest Fixpoints

Greatest fixpoints

Let $F : 2^S \rightarrow 2^S$ be a monotonic function from sets of states to sets of states.

Then F has a **greatest fixpoint** X^*:

- X^* is a fixpoint of F: $F(X^*) = X^*$
- X^* is the greatest fixpoint of F: for any X, if $F(X) = X$ then $X^* \supseteq X$.

Greatest fixpoints

Let $F : 2^S \to 2^S$ be a monotonic function from sets of states to sets of states.

Then F has a **greatest fixpoint** X^*:

- X^* is a fixpoint of F: $F(X^*) = X^*$
- X^* is the greatest fixpoint of F: for any X, if $F(X) = X$ then $X^* \supseteq X$.

X^* is often denoted $\mathbf{gfp}F$ or νF.

Greatest fixpoints

Let $F : 2^S \to 2^S$ be a monotonic function from sets of states to sets of states.

Then F has a **greatest fixpoint** X^*:

- X^* is a fixpoint of F: $F(X^*) = X^*$
- X^* is the greatest fixpoint of F: for any X, if $F(X) = X$ then $X^* \supseteq X$.

X^* is often denoted $\mathbf{gfp}F$ or νF.

X^* can be computed by starting from the set S and applying F repeatedly:

$$S \supseteq F(S) \supseteq F(F(S)) \supseteq F^3(S) \supseteq \cdots \supseteq F^n(S) \supseteq \cdots$$

As with least fixpoints, for finite S the above terminates after at most $n = |S|$ steps.

CTL Model-Checking continued

Computing $[\![\mathbf{AG}\phi]\!]$

$$[\![\mathbf{AG}\phi]\!] = \mathbf{gfp}X.[\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{X})}$$

Iterative computation:

$$\begin{array}{lllll}
X_0 & = & S & & \\
X_1 & = & F(X_0) & = & [\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{S})} \\
 & & & = & [\![\phi]\!] \cap \overline{\mathbf{pre}(\emptyset)} \\
 & & & = & [\![\phi]\!] \cap \overline{\emptyset} \\
 & & & = & [\![\phi]\!] \cap S \\
 & & & = & [\![\phi]\!] \\
X_2 & = & F(X_1) & = & [\![\phi]\!] \cap \overline{\mathbf{pre}(\overline{[\![\phi]\!]})} \\
\ldots & & & &
\end{array}$$

The $\overline{\mathbf{pre}(\bar{\cdot})}$ operator

What does $\overline{\mathbf{pre}(\overline{X})}$ really compute?

The $\overline{\mathbf{pre}(\bar{\cdot})}$ operator

What does $\overline{\mathbf{pre}(\overline{X})}$ really compute?

The set of all states which are **not** 1-step predecessors of $\overline{X}$.

The $\overline{\mathbf{pre}(\bar{\cdot})}$ operator

What does $\overline{\mathbf{pre}(\overline{X})}$ really compute?

The set of all states which are **not** 1-step predecessors of $\overline{X}$.

I.e., the set of all states which **don't** have 1-step successors in $\overline{X}$.

The $\overline{\mathbf{pre}(\bar{\cdot})}$ operator

What does $\overline{\mathbf{pre}(\overline{X})}$ really compute?

The set of all states which are **not** 1-step predecessors of $\overline{X}$.

I.e., the set of all states which **don't** have 1-step successors in $\overline{X}$.

In other words, the set of all states whose 1-step successors are all in X.

The $\overline{\mathbf{pre}(\bar{\cdot})}$ operator

What does $\overline{\mathbf{pre}(\overline{X})}$ really compute?

The set of all states which are **not** 1-step predecessors of $\overline{X}$.

I.e., the set of all states which **don't** have 1-step successors in $\overline{X}$.

In other words, the set of all states whose 1-step successors are all in X.

In other words, since we assumed no deadlocks, the set of all states which will **inevitably** move into X in 1-step.

We define:

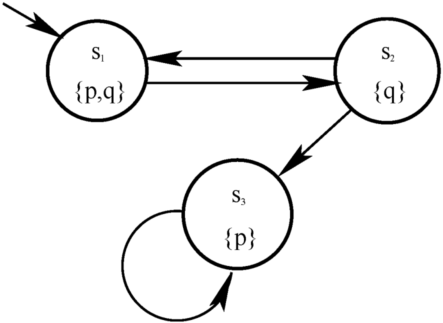
$$\mathbf{inev}(X) = \overline{\mathbf{pre}(\overline{X})}$$

$\mathbf{inev}(X)$ is often denoted $\widetilde{\mathbf{pre}}(X)$.

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in$ AP: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
6. $[\![\mathbf{E}(\phi_1\,\mathbf{U}\,\phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$
7. $[\![\mathbf{AX}\phi_1]\!] = \mathbf{inev}([\![\phi_1]\!])$
8. $[\![\mathbf{AG}\phi_1]\!] = \mathbf{gfp}X.[\![\phi_1]\!] \cap \mathbf{inev}(X)$

More CTL model-checking examples



Homework:

- Let's model-check $\mathbf{E}(q\,\mathbf{U}\,\mathbf{AG}p)$.
- What about $\mathbf{E}(p\,\mathbf{U}\,\mathbf{AG}q)$?

Computing $[\![\phi]\!]$ (continued)

1. For atomic proposition $p \in$ AP: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
6. $[\![\mathbf{E}(\phi_1\, \mathbf{U}\, \phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$
7. $[\![\mathbf{AX}\phi_1]\!] = \mathbf{inev}([\![\phi_1]\!])$
8. $[\![\mathbf{AG}\phi_1]\!] = \mathbf{gfp}X.[\![\phi_1]\!] \cap \mathbf{inev}(X)$
9. $[\![\mathbf{EG}\phi_1]\!] = ???$
10. $[\![\mathbf{AF}\phi_1]\!] = ???$
11. $[\![\mathbf{A}(\phi_1\, \mathbf{U}\, \phi_2)]\!] = ???$

Computing $[\![\mathbf{EG}\phi]\!]$

$$\mathbf{EG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{EX}\,\mathbf{EG}\phi$$

Fixpoint equation:

$$F(X) =$$

Computing $[\![\mathbf{EG}\phi]\!]$

$$\mathbf{EG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{EX}\,\mathbf{EG}\phi$$

Fixpoint equation:

$$F(X) = [\![\phi]\!] \cap \mathbf{pre}(X)$$

Least or greatest fixpoint?

Computing $[\![\mathbf{EG}\phi]\!]$

$$\mathbf{EG}\phi \quad \Leftrightarrow \quad \phi \wedge \mathbf{EX}\,\mathbf{EG}\phi$$

Fixpoint equation:

$$F(X) = [\![\phi]\!] \cap \mathbf{pre}(X)$$

Least or greatest fixpoint?

$$[\![\mathbf{EG}\phi]\!] = \mathbf{gfp}X.[\![\phi]\!] \cap \mathbf{pre}(X)$$

Computing $[\![\mathbf{AF}\phi]\!]$

$$\mathbf{AF}\phi \quad \Leftrightarrow$$

Computing $[\![\mathbf{AF}\phi]\!]$

$$\mathbf{AF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{AX}\,\mathbf{AF}\phi$$

Fixpoint equation:

$$F(X) =$$

Computing $[\![\mathbf{AF}\phi]\!]$

$$\mathbf{AF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{AX}\,\mathbf{AF}\phi$$

Fixpoint equation:

$$F(X) = [\![\phi]\!] \cup \mathbf{inev}(X)$$

Least or greatest fixpoint?

Computing $[\![\mathbf{AF}\phi]\!]$

$$\mathbf{AF}\phi \quad \Leftrightarrow \quad \phi \vee \mathbf{AX}\,\mathbf{AF}\phi$$

Fixpoint equation:

$$F(X) = [\![\phi]\!] \cup \mathbf{inev}(X)$$

Least or greatest fixpoint?

$$[\![\mathbf{AF}\phi]\!] = \mathbf{lfp}X.[\![\phi]\!] \cup \mathbf{inev}(X)$$

Computing $[\![\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2)]\!]$

$$\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2) \quad \Leftrightarrow$$

Computing $[\![\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2)]\!]$

$$\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2) \quad \Leftrightarrow \quad \phi_2 \vee (\phi_1 \wedge \mathbf{AX}\, \mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2))$$

Fixpoint equation:

$$F(X) =$$

Computing $[\![\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2)]\!]$

$$\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2) \quad \Leftrightarrow \quad \phi_2 \vee (\phi_1 \wedge \mathbf{AX}\,\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2))$$

Fixpoint equation:

$$F(X) = [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{inev}(X))$$

Least or greatest fixpoint? (This case is a bit trickier.)

Computing $[\![\mathbf{A}(\phi_1 \, \mathbf{U} \, \phi_2)]\!]$

$$\mathbf{A}(\phi_1 \, \mathbf{U} \, \phi_2) \quad \Leftrightarrow \quad \phi_2 \vee (\phi_1 \wedge \mathbf{AX} \, \mathbf{A}(\phi_1 \, \mathbf{U} \, \phi_2))$$

Fixpoint equation:

$$F(X) = [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{inev}(X))$$

Least or greatest fixpoint? (This case is a bit trickier.)

$$[\![\mathbf{A}(\phi_1 \, \mathbf{U} \, \phi_2)]\!] = \mathbf{lfp} X. [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{inev}(X))$$

Computing $[\![\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2)]\!]$

$$\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2) \quad \Leftrightarrow \quad \phi_2 \vee (\phi_1 \wedge \mathbf{AX}\,\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2))$$

Fixpoint equation:

$$F(X) = [\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{inev}(X))$$

Least or greatest fixpoint? (This case is a bit trickier.)

$$[\![\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{inev}(X))$$

Why not take the greatest fixpoint here? **Homework.**

Computing $[\![\phi]\!]$ – final version!

1. For atomic proposition $p \in$ AP: $[\![p]\!] = \{s \in S \mid p \in L(s)\}$
2. $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$
3. $[\![\neg\phi_1]\!] = \overline{[\![\phi_1]\!]} = S - [\![\phi_1]\!]$
4. $[\![\mathbf{EX}\phi_1]\!] = \mathbf{pre}([\![\phi_1]\!])$
5. $[\![\mathbf{AX}\phi_1]\!] = \mathbf{inev}([\![\phi_1]\!])$
6. $[\![\mathbf{EF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{pre}(X)$
7. $[\![\mathbf{AF}\phi_1]\!] = \mathbf{lfp}X.[\![\phi_1]\!] \cup \mathbf{inev}(X)$
8. $[\![\mathbf{EG}\phi_1]\!] = \mathbf{gfp}X.[\![\phi_1]\!] \cap \mathbf{pre}(X)$
9. $[\![\mathbf{AG}\phi_1]\!] = \mathbf{gfp}X.[\![\phi_1]\!] \cap \mathbf{inev}(X)$
10. $[\![\mathbf{E}(\phi_1 \,\mathbf{U}\, \phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{pre}(X))$
11. $[\![\mathbf{A}(\phi_1 \,\mathbf{U}\, \phi_2)]\!] = \mathbf{lfp}X.[\![\phi_2]\!] \cup ([\![\phi_1]\!] \cap \mathbf{inev}(X))$

Symbolic CTL model-checking

The definitions of $[\![\phi]\!]$ directly suggest symbolic implementations.

It suffices to be able to compute $\mathbf{pre}$ and $\mathbf{inev}$ symbolically.

Recall:

$$\mathbf{pre}(X) = \{s \in S \mid \exists s' \in X : s \longrightarrow s'\}$$

$$\mathbf{inev}(X) = \overline{\mathbf{pre}(\overline{X})}$$

Symbolic CTL model-checking

The definitions of $[\![\phi]\!]$ directly suggest symbolic implementations.

It suffices to be able to compute **pre** and **inev** symbolically.

Recall:

$$\begin{aligned}
\mathbf{pre}(X) &= \{s \in S \mid \exists s' \in X : s \longrightarrow s'\} \\
\mathbf{inev}(X) &= \overline{\mathbf{pre}(\overline{X})} \\
&= \{s \in S \mid \forall s' \in S : s \longrightarrow s' \Rightarrow s' \in X\}
\end{aligned}$$

Symbolic CTL model-checking

The definitions of $[\![\phi]\!]$ directly suggest symbolic implementations.

It suffices to be able to compute $\mathbf{pre}$ and $\mathbf{inev}$ symbolically.

Recall:

$$\mathbf{pre}(X) = \{s \in S \mid \exists s' \in X : s \longrightarrow s'\}$$

$$\begin{aligned}\mathbf{inev}(X) &= \overline{\mathbf{pre}(\overline{X})} \\ &= \{s \in S \mid \forall s' \in S : s \longrightarrow s' \Rightarrow s' \in X\}\end{aligned}$$

How can we implement these operators symbolically?
Hint: recall $\mathbf{succ}$.

Symbolic CTL model-checking

Symbolic **post** :

$$\mathbf{succ}(\phi) \;=\; \big(\exists x : \phi(x) \wedge \mathit{Trans}(x, x')\big)[x' \leadsto x]$$

Symbolic **pre** :

$$\mathbf{pred}(\phi) \;=$$

Symbolic CTL model-checking

Symbolic **post** :

$$\mathbf{succ}(\phi) \quad = \quad \big(\exists x : \phi(x) \wedge \mathit{Trans}(x, x')\big)[x' \leadsto x]$$

Symbolic **pre** :

$$\mathbf{pred}(\phi) \quad = \quad \exists x' : \phi(x') \wedge \mathit{Trans}(x, x')$$

Symbolic **inev** :

$$\mathbf{syminev}(\phi) \quad =$$

Symbolic CTL model-checking

Symbolic $\mathbf{post}$:

$$\mathbf{succ}(\phi) = \big(\exists x : \phi(x) \wedge \mathit{Trans}(x, x')\big)[x' \leadsto x]$$

Symbolic $\mathbf{pre}$:

$$\mathbf{pred}(\phi) = \exists x' : \phi(x') \wedge \mathit{Trans}(x, x')$$

Symbolic $\mathbf{inev}$:

$$\mathbf{syminev}(\phi) = \forall x' : \mathit{Trans}(x, x') \rightarrow \phi(x')$$

Bibliography

Baier, C. and Katoen, J.-P. (2008).
Principles of Model Checking.
MIT Press.

Clarke, E., Grumberg, O., and Peled, D. (2000).
Model Checking.
MIT Press.

Davey, B. A. and Priestley, H. A. (2002).
Introduction to Lattices and Order.
Cambridge University Press, 2nd edition.

Huth, M. and Ryan, M. (2004).
Logic in Computer Science: Modelling and Reasoning about Systems.
Cambridge University Press.