Formal Methods in Software Development

Computational Tree Logic (CTL)
CTL, LTL, and CTL* model Checking

Ivano Salvo

Computer Science Department

Sapienza Università di Roma

Lesson **3**, October 8^{th}, 2019

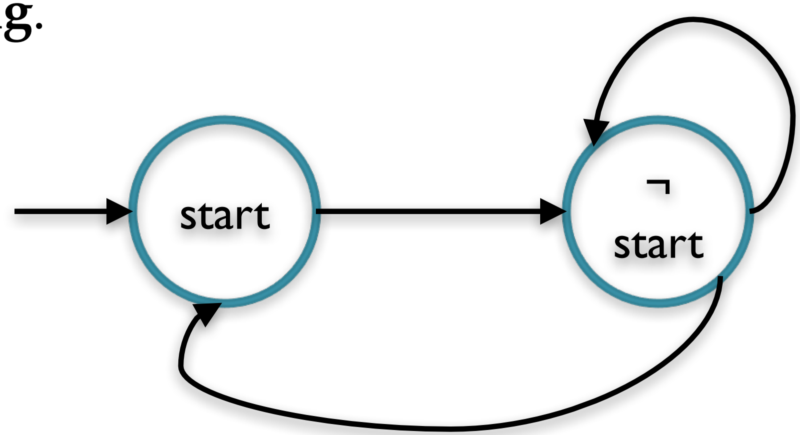
Lesson 2d:

Computation Tree Logic
CTL

Non Linear Time properties

Let us consider the property: "***For every computation, it is always possible to return to the initial state***"

A G F *start* **does not properly work**.
It is **too strong**.



This system intuitively satisfies our intended property, but not the linear property **A G F** start (**because of the path (¬start)$^{\omega}$**)

The solution is a **branching notion** of time, allowing nesting of path quantifiers **A** and **E**: in this case **A G E F** start.

CTL: syntax

State formulas are formulas that depend on a state of a transition system

- If $p \in AP$, then p is a state formula
- If f, g are state formulas, then so are $\neg f$, $f \wedge g$, $f \vee g$
- If f is a path formula, then $\mathbf{A}\, f$ and $\mathbf{E}\, f$ are state formulas

Path formulas are formulas that depend on a computation path

- If f, g are **state** formulas, then $\neg f$, $f \wedge g$, $f \vee g$, $\mathbf{X}\, f$, $\mathbf{F}\, f$, $\mathbf{G}\, f$, $f\, \mathbf{U}\, g$, and $f\, \mathbf{R}\, g$ are path formulas

Similar to CTL*, but **each temporal operator** (**X**, **F**, **G**, **U**, **R**) **must be preceded by a path quantifier** (**E** or **A**)

Examples: (il)legal CTL formulas

Let $AP = \{x = 1, x < 2, x \geq 3\}$ be the set of atomic propositions.

Legal CTL formulas are:

$\mathbf{EX}\,(x = 1)$, $\mathbf{AX}\,(x = 1)$, $x = 1 \vee x < 2$

Illegal CTL formulas are:

$\mathbf{E}\,(x = 1 \wedge \mathbf{AX}\; x \geq 3)$

because $\mathbf{AX}\; x \geq 3$ is not a path formula

$\mathbf{EX}\,(\text{true}\; \mathbf{U}\; x = 1)$

because **EX** nested with a path formula

By contrast, the following are legal CTL formulas:

$\mathbf{EX}\,(x = 1 \wedge \mathbf{AX}\; x \geq 3)$ $\mathbf{EX}\;\mathbf{A}\,(\text{true}\; \mathbf{U}\; x = 1)$

Common operators: $\mathbf{EF}\,\varphi \equiv$ "φ holds potentially"

$\mathbf{AF}\,\varphi \equiv$ "φ is inevitable"

$\mathbf{EG}\,\varphi \equiv$ "φ holds potentially always"

$\mathbf{AG}\,\varphi \equiv$ "invariantly φ"

Minimal Fragment of CTL

From a theoretical point of view, **3 operators only** are really needed: **EX**, **EG**, and **EU** (**via duality**):

$$\mathbf{AX}\, f \equiv \neg\, \mathbf{EX}\, \neg f$$

$$\mathbf{EF}\, f \equiv \neg\, \mathbf{E}\, (\text{true}\ \mathbf{U}\, f\,)$$

$$\mathbf{AG}\, f \equiv \neg\, \mathbf{EF}\, \neg f$$

$$\mathbf{AF}\, f \equiv \neg\, \mathbf{EG}\, \neg f$$

$$\mathbf{A}\, (f\, \mathbf{U}\, g\,) \equiv \neg\, \mathbf{E}\, (\neg g\, \mathbf{U}\, \neg f \wedge \neg g) \wedge \neg\, \mathbf{EG}\, \neg g$$

$$\mathbf{A}\, (\, f\, \mathbf{R}\, g\,) \equiv \neg\, \mathbf{E}\, (\neg f\, \mathbf{U}\, \neg g)$$

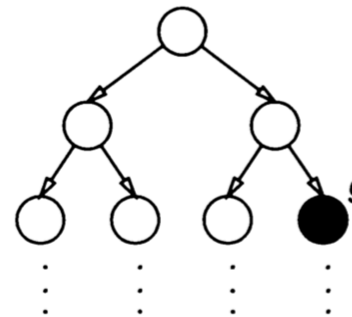
$$\mathbf{E}\, (\, f\, \mathbf{R}\, g\,) \equiv \neg\, \mathbf{A}\, (\neg f\, \mathbf{U}\, \neg g)$$

Attention! that propositional operators (∧, ∨ , ¬, etc.) **cannot be applied to path formula**, so it is not true that **EG** f ≡ **E** ¬ **F** ¬*f* simply because the latter **is not** a CTL formula.

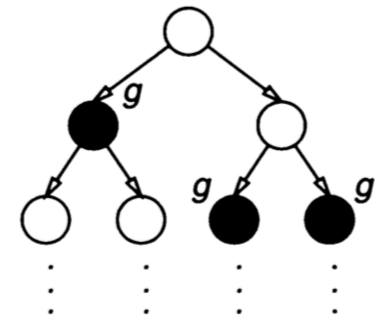
Non Linear Time (LT) examples

The semantics of CTL* formulas are relative to a computation Tree.

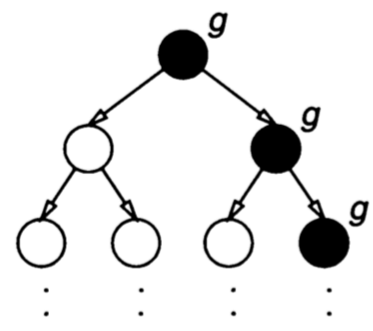
Here some example of computation trees and CTL* formulas valid in such computation trees.



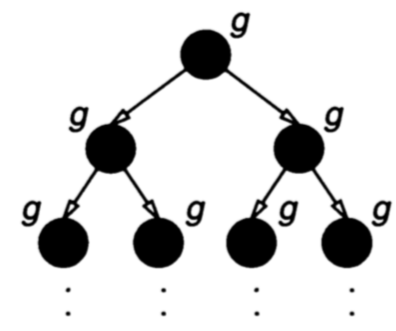
$M, s_0 \models \mathbf{EF}\, g$



$M, s_0 \models \mathbf{AF}\, g$



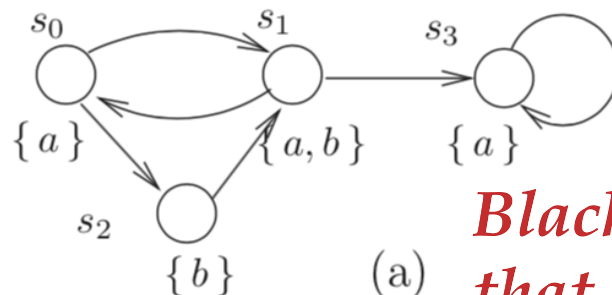
$M, s_0 \models \mathbf{EG}\, g$



$M, s_0 \models \mathbf{AG}\, g$

Other Examples

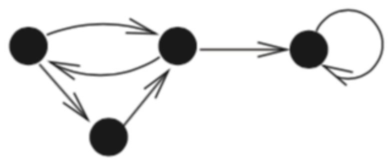
TS



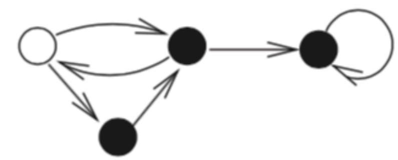
(a)

Black states are those that satisfy the formula

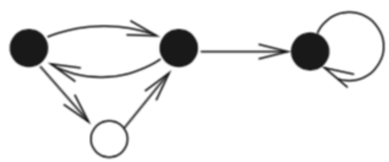
$\exists \bigcirc a$



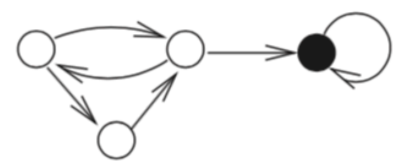
$\forall \bigcirc a$



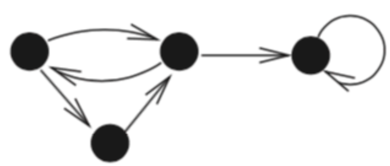
$\exists \Box a$



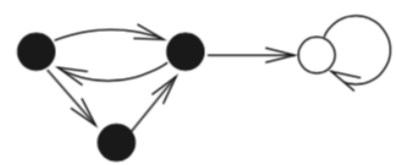
$\forall \Box a$



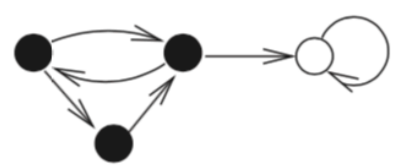
$\exists \Diamond (\exists \Box a)$



$\forall (a \, \mathsf{U} \, b)$



$\exists (a \, \mathsf{U} \, (\neg a \ \wedge \ \forall(\neg a \, \mathsf{U} \, b)))$



A remark on negation

Definition. A transition system $\mathcal{M}$ satisfies a CTL formula φ, notation $\mathcal{M} \vDash \varphi$ if and only if $\mathcal{M}, s \vDash \varphi$ for all $s \in S_0$, where S_0 is the set of initial states of $\mathcal{M}$.

Be careful! $\mathcal{M}, s \nvDash \varphi$ implies $\mathcal{M}, s \vDash \neg\varphi$, but **it is not true** that $\mathcal{M} \nvDash \varphi$ implies $\mathcal{M} \vDash \neg\varphi$ (**The same holds for LTL!**).

The problem is the universal quantification over **initial states**!

Example: Both a and $\neg a$ does not hold here:

Equivalent CTL formulas

Besides duality, there are a lot of interesting logical equivalence, useful, for example in Model Checking algorithms (in particular in Symbolic Model Checking).

A CTL formula f is equivalent to g if and only if **for all transition systems** $\mathcal{M}$, $\mathcal{M} \models f$ **iff** $\mathcal{M} \models g$

Expansion Laws for CTL:

$$\mathbf{A}\,(f\,\mathbf{U}\,g) \equiv g \vee (f \wedge \mathbf{AX}\ \mathbf{A}(f\,\mathbf{U}\,g))$$

$$\mathbf{AG}\,f \equiv f \wedge \mathbf{AX}\ \mathbf{AG}\,f$$

$$\mathbf{AF}\,f \equiv f \vee \mathbf{AX}\ \mathbf{AF}\,f$$

$$\mathbf{E}\,(f\,\mathbf{U}\,g) \equiv g \vee (f \wedge \mathbf{EX}\ \mathbf{E}(f\,\mathbf{U}\,g))$$

$$\mathbf{EG}\,f \equiv f \wedge \mathbf{EX}\ \mathbf{EG}\,f$$

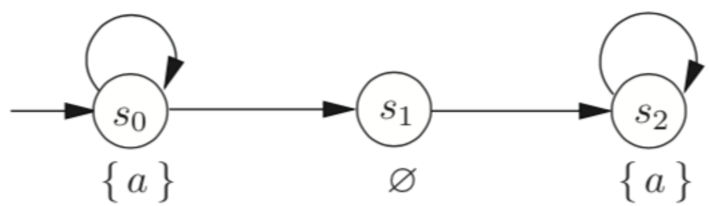
$$\mathbf{EF}\,f \equiv f \vee \mathbf{EX}\ \mathbf{EF}\,f$$

LTL versus CTL: eliminating A

Theorem. Let f be a CTL formula and let f^{LTL} be the LTL formula obtained by eliminating all path quantifiers in f. Then: $f \equiv f^{\mathrm{LTL}}$ or there does not exist any LTL formula equivalent to f

Lemma. [PERSISTENCE] The CTL formula $\mathbf{A}\,\mathbf{F}\,\mathbf{A}\,\mathbf{G}\,a$ and the LTL formula $\mathbf{F}\,\mathbf{G}\,a$ are not equivalent.

Proof: Just consider the following Kripke structure.



We have $s_0 \vDash_{\mathrm{LTL}} \mathbf{F}\,\mathbf{G}\,a$, since all path starting in s_0 will remain forever in s_0 or in s_2 (that satisfy $\mathbf{G}\,a$).

By contrast $s_0 \nvDash_{\mathrm{CTL}} \mathbf{A}\,\mathbf{F}\,\mathbf{A}\,\mathbf{G}\,a$, since $s_0^{\omega} \nvDash_{\mathrm{CTL}} \mathbf{F}\,\mathbf{A}\,\mathbf{G}\,a$ because of the paths $s_0^{*}\, s_1\, s_2^{\omega}$ which passes the $\neg a$-state s_1. ❑

LTL and CTL are not comparable

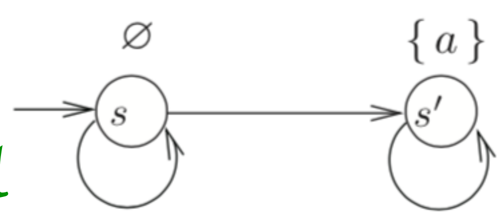
Theorem.

1. There exist LTL formulas for which no equivalent CTL formula exist. For instance: **F G** a or **F** ($a \wedge$ **X** a)
2. There exist CTL formulas for which no equivalent LTL formula exist. For instance: **AF AG** a or **AF** ($a \wedge$ **AX** a) or **AG EF** a

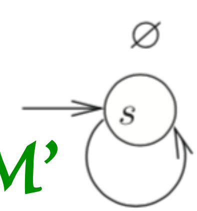
Proof (idea)**:** exhibit suitable transition systems $\mathcal{M}$ and $\mathcal{M}'$ such that $\mathcal{M} \models_{LTL} g$ and $\mathcal{M}' \not\models_{LTL} g$ but such that cannot be distinguished by any CTL formula, that is, for all CTL property g, $\mathcal{M} \models_{CTL} g$ if and only if f $\mathcal{M}' \models_{CTL} g$.

Example: Let us consider **AG EF** a. This is satisfied by $\mathcal{M}$ above, but not by $\mathcal{M}'$. On the other hand since traces($\mathcal{M}'$) ⊆ traces($\mathcal{M}$), $\mathcal{M}'$ satisfies all LTL formulas satisfied by $\mathcal{M}$. ❑

$\mathcal{M}$



$\mathcal{M}'$



Lesson 3a:

CTL Model Checking

CTL model checking: basics

As it is clear from CTL semantics **the truth** of a formula depends on the truth of its **subformula** (maybe in some other state as **EX** or **EF** shows).

CTL formulas are **state formula**: we can determine in each state which are formulas that are satisfied.

Putting together these two facts, and following a common pattern in graph algorithms (it is usually convenient **explore the whole graph** (visits), even when we need for example just a path between two nodes), we have:

Idea: Compute a set *label*(*s*) in such a way that for each sub-formula g of f, $g \in label(s)$ whenever $\mathcal{M}, s \models g$ **holds**.

Observation: the number of sub-formulas are **linear** in the **size** $| f |$ of a CTL formula f.

CTL model checking: basics

Start with the original labeling of states with atomic propositions, i.e. $label(s) = L(s)$.

$f \equiv \neg g \Rightarrow f \in label(s)$ if and only if $g \notin label(s)$

$f \equiv g \vee h \Rightarrow f \in label(s)$ if and only if $g \in label(s)$ or $h \in label(s)$

$f \equiv \mathbf{E}\,\mathbf{X}\, g \Rightarrow f \in label(s)$ if and only if $g \in label(s')$ for some s', $s \rightarrow s'$

The interesting cases are $f \equiv \mathbf{E}\,\mathbf{G}\, h$ and $f \equiv \mathbf{E}\,[\, g\, \mathbf{U}\, h\,]$

CTL model checking: EU f

When $f \equiv \mathbf{E}\,[\,g\ \mathbf{U}\ h\,]$ the idea is: **start** from the **set of states such that $h \in label(s)$** and then **proceed backwards** on states such that **$g \in label(s)$**. Label all these states with g.

def $checkEU(g, h)$:
  $T = \{\, s \mid h \in labels(s) \,\}$
  forall $s \in T$ **do** $label(s) = label(s) \cup \{\, \mathbf{E}\,[g\ \mathbf{U}\ h] \,\}$
  while $\mathrm{T} \neq \varnothing$ **do**
    choose $s \in T$
    $T = T \setminus \{\, s \,\}$
    forall $t \in \mathrm{prec}(s)$ **do**
      if $\mathbf{E}[g\ \mathbf{U}\ h] \notin label(t)$ **and** $g \in label(t)$
        then
          $label(s) = label(s) \cup \{\, \mathbf{E}\,[g\ \mathbf{U}\ h] \,\}$
          $T = T \cup \{\, t \,\}$

$\mathrm{prec}(s) = \{t \mid R(t, s)\}$

It is essentially a backward visit of a graph. The complexity is $O(|S| + |R|)$

CTL model checking: EG f (1)

When g ≡ **E G** h, we must find **infinite paths labeled by** h.

In a finite directed graph, such paths must enter a **strongly connected component** where **all states** are **labeled by** h.

Roughly speaking:

1. Compute the set of states $S' = \{ s \in S \mid h \in label(s)\}$.
2. Decompose (S', R') in strongly connected components.
3. Add all states s such that $h \in label(s)$ and from which one of such strongly connected components is reachable.

CTL model checking: EG f (2)

Lemma. Let $S' = \{ s' \in S \mid \mathcal{M}, s' \vDash h \}$. Then $\mathcal{M}, s \vDash$ **E G** h if and only if the following conditions are satisfied:

1. $s \in S'$

2. There exists a path from s to a strongly connected component $C \subseteq S'$.

Proof: (**If**) Let π be an infinite path starting at s satisfying **G** h. Clearly, $s \vDash h$. Since π is an infinite path, it has the shape $\pi_0\pi_1$ and in π_1 each state occurs infinitely often. Both states in π_0 and π_1 belongs S'. Since each state appears infinitely often in π_1, there is a path between any pairs of states in π_1, therefore states in π_1 belong to some C that is a SCC in (S', R').

(**Only If**) Let π_0 be a finite path from s to $t \in C$ in S'. Then we can find a finite path π_1 from t to t. The path $\pi_0\, \pi_1^{\omega}$ satisfies **G** h. ❑

CTL model checking: EG f / 3

def *checkEG*(g):
    $S' = \{ s \mid g \in labels(s) \}$
    $SCC = \{ C \mid C$ *is a* nontrivial *SCC of* $S' \}$
    $T = \cup_{C \in SCC} \{ s \mid s \in C \}$
    forall $s \in T$ **do** $label(s)=label(s) \cup \{$ **EG** $g \}$
    while $T \neq \varnothing$ **do**
        choose $s \in T$
        $T = T \setminus \{ s \}$
        forall $t \in$ prec(s), $t \in S'$ **do**
            if EG $g \notin label(t)$
                then
                    $label(s)=label(s) \cup \{$ **EG** $g \}$
                    $T = T \cup \{ t \}$

Strongly connected components of the subgraph ⟨S′, R′⟩ of states where h holds

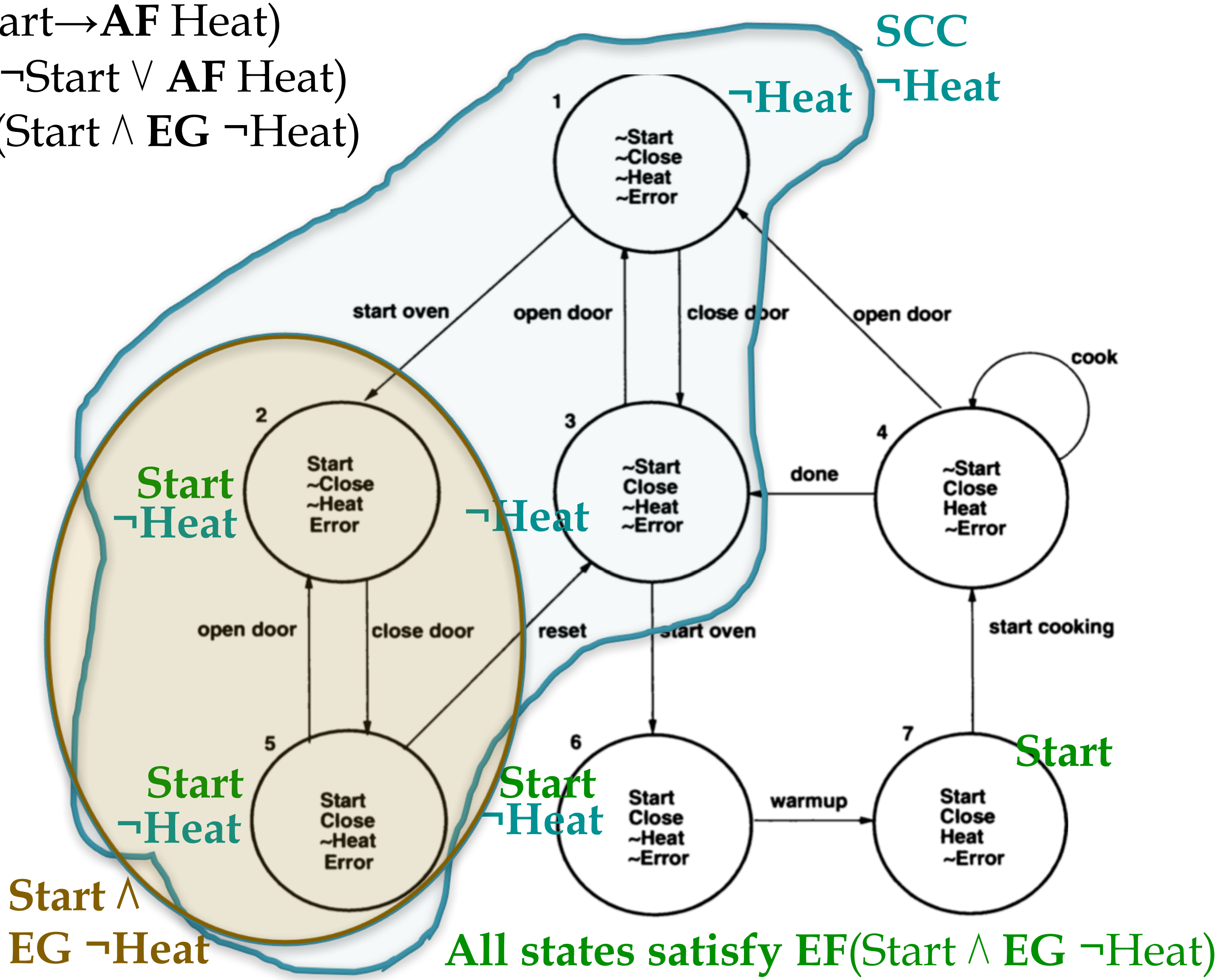
Backward reachability from strong connected components.

Theorem. Given a Kripke structure $\mathcal{M}=(S, R, L)$ and a CTL formula f, determining if $\mathcal{M} \models_{\text{CTL}} g$ can be decided in time

$$\mathcal{O}((| S | + | R |) \cdot \ |f|$$

Example: microwave oven

AG(Start→**AF** Heat)
≡ **AG**(¬Start ∨ **AF** Heat)
≡ ¬**EF**(Start ∧ **EG** ¬Heat)



Lesson 3b:

LTL Model Checking

LTL Model Checking (1)

Several algorithms. Today, we see a tableaux construction.

It suffices (thanks to duality) to check properties of the form $\mathbf{E}\, f$ ($\mathbf{A}\, f \equiv \neg \mathbf{E} \neg f$). Moreover (again thanks to duality), we consider only operators **X** and **U**.

Definition [Closure of f] CL(f) is the **smallest set** containing f and satisfying:

- $\neg g \in CL(f)$ iff $g \in CL(f)$
- If $g_1 \vee g_2 \in CL(f)$ then $g_1, g_2 \in CL(f)$
- If $\mathbf{X}\, g \in CL(f)$ then $g \in CL(f)$
- If $\neg \mathbf{X}\, g \in CL(f)$ then $\mathbf{X} \neg g \in CL(f)$
- If $g_1\, \mathbf{U}\, g_2 \in CL(f)$ then $g_1, g_2, \mathbf{X}\, (g_1\, \mathbf{U}\, g_2) \in CL(f)$

LTL Model Checking (2)

Definition An **atom** is a pair (s, K), where s is a state and K is a maximal **set of formulas in** $\mathrm{CL}(f)$ **consistent with** $L(s)$, that is (we identify : g with $\neg\neg g$) :

- for each atomic proposition p, $p \in K$ iff $p \in L(s)$
- for each $g \in \mathrm{CL}(f)$ then $g \in K$ iff $\neg g \notin K$
- for each $g_1 \vee g_2 \in \mathrm{CL}(f)$, $g_1 \vee g_2 \in K$ iff $g_1 \in K$ or $g_2 \in K$
- for each $\neg \mathbf{X}\, g \in \mathrm{CL}(f)$, $\neg \mathbf{X}\, g \in K$ iff $\mathbf{X} \neg g \in K$
- for each $g_1\, \mathbf{U}\, g_2 \in \mathrm{CL}(f)$, $g_1\, \mathbf{U}\, g_2 \in K$ iff $g_2 \in K$ or $g_1 \in K$ and $\mathbf{X}\,(g_1\, \mathbf{U}\, g_2) \in K$

Definition Given a LTL model checking problem $\mathcal{M}, s \models \mathbf{E} f$, the **atom graph** $G^{\mathcal{M}, f}$ is built with **atoms** as the set of **vertices**.

There is an **edge** from (s, K) to (s', K') iff **(s, s') is transition in $\mathcal{M}$**, and for each formula $\mathbf{X}\, g \in \mathrm{CL}(f)$, $\mathbf{X}\, g \in K$ iff $g \in K'$.

LTL Model Checking (3)

Definition An **eventuality sequence** is an infinite path π in $G^{\mathcal{M}, f}$ such that if g_1 **U** $g_2 \in K$ for some atom (s, K) then there exists an atom (s', K') reachable from (s, K) along π, such that $g_2 \in K'$.

Theorem $\mathcal{M}, s \models \mathbf{E} f \Leftrightarrow$ there exists an eventuality sequence in $G^{\mathcal{M}, f}$ starting at atom (s, K) such that $f \in K$.

Proof (sketch):

(**If**) Assume $(s_0, K_0)\ (s_1, K_1)\ (s_2, K_2)\ldots$ is an eventuality sequence with $(s, K)=(s_0, K_0)$. By def, $\pi = s_0\, s_1\, s_2\ldots$ is a path in $\mathcal{M}$.

To make induction hypothesis work, we prove that for every $g \in CL(f)$ and for all $i \geq 0$, $\pi^i \models g$ iff $g \in K_i$. The proof proceeds by induction on sub-formulas of f.

If $g = \neg h$, $\pi^i \models g \Leftrightarrow \pi^i \not\models h \Leftrightarrow h \notin K_i$ (**IND**) $\Leftrightarrow g \in K_i$ (by **maximality**)

If $g = \mathbf{X}\, h$ then $\pi^i \models g \Leftrightarrow \pi^{i+1} \models h$ **(IND)** $h \in K_{i+1}$. Since $(s_i, K_i) \rightarrow (s_{i+1}, K_{i+1})$ then $h \in K_{i+1} \Leftrightarrow \mathbf{X}\, h \in K_i$.

LTL Model Checking (4)

Proof (cntd.): If $g = h_1$ **U** h_2 we have $h_2 \in K_j$ for some $j \geq i$ and h_1, **X** $g \in K_k$ for $i \leq k < j$. This implies $h_1 \in K_k$ and $h_2 \in K_j$ and hence $\pi^i \vDash g$.

Conversely, if $\pi^i \vDash g$, there exists $j \geq i$ such that $\pi^j \vDash h_2$ and $\pi^k \vDash h_1$ K_k for $i \leq k < j$. (**HH**) $h_2 \in K_j$ and $h_1 \in K_k$. By absurd, $g \notin K_i$ and $h_1 \in K_i$ implies that **X** $g \notin K_i$ (def. of atom) and hence (def. of atom) **X** $\neg g \in K_i$ (def of transition relation) $\neg g \in K_{i+1}$ and $g \notin K_{i+1}$ and so on until $g \notin K_j$ against the fact that $h_2 \in K_j$.

(**only if**) Assuming that $\mathcal{M}, s \vDash$ **E** f there exists a path $\pi = s_0 s_1 s_2 \ldots$ in $\mathcal{M}$ such that $\pi \vDash f$. Define $K_i = \{ g \in \mathrm{CL}(f) \mid \pi^i \vDash g \}$.
One can show that:

1. (s_i, K_i) is an atom;
2. $(s_i, K_i) \rightarrow (s_{i+1}, K_{i+1})$ is a transition in G.
3. The sequence $(s_0, K_0)\ (s_1, K_1)\ (s_2, K_2) \ldots$ is an eventuality sequence. ❑

LTL Model Checking (5)

Definition: A non trivial strongly connected component C in $G^{\mathcal{M}, f}$ is **self-fullfilling** if for every atom (s, K) and for every h_1 **U** $h_2 \in K$ there exists an atom (s', K') in C such that $h_2 \in K'$.

Theorem. There exists an eventuality sequence in $G^{\mathcal{M}, f}$ starting at an atom (s, K) iff there exists a self-fulfilling strongly connected component in $G^{\mathcal{M}, f}$ reachable from (s, K) .

Proof (sketch): (**If**) Take an eventuality sequence and consider the set of atoms C' that appear infinitely often in it. $C' \subseteq C$, C strongly connected component. Take (s, K) in C and g = h_1 **U** $h_2 \in K$. There must be a path from (s, K) to C'. If h_2 appear in the path, ok. Otherwise, g is in every atom on the path and in an atom of C'. Since C' comes from an eventuality sequence, h_2 is in some atom of $C' \subseteq C$, thus C is self-fullfilling.

(**Only if**) Take a path from (s, K) to C. Clearly in C any subformula of the form h_1 **U** h_2 is followed by an atom containing h_2 . The only problem is along the path, but we can reason as in the (**If**) part. ❑

--

LTL Model Checking: Algorithm

Theorem $\mathcal{M}, s \models \mathbf{E} f$ if and only if there exists an atom (s, K) such that $f \in K$ and a path from (s, K) to a self-fullfilling SCC.

The size of the graph G is $(|S|+|R|) \cdot 2^{|f|}$.

Using Tarjan algorithm for SCC, this is also the complexity of this algorithm for LTL model checking.

Bad News: It is exponential in the size of the formula f.

Good News: Usually the transition system is huge, but the formula is small.

Is there any polynomial algorithm for LTL model checking?

Probably, no (unless P=NP).

--

LTL Model Checking: Complexity

The LTL model checking problem is **PSPACE-complete**.
Here we **prove** just that LTL model checking is ***NP*-hard**.

We reduce the **Hamiltonian path problem** for a graph $G=(V, E)$ to the LTL model checking problem $\mathcal{M}, s \vDash \mathbf{E}\, f$ where:

- $\mathcal{M}$ is the Kripke structure (S, R, L) where:
  - * S is $V \cup \{ s, t \}$
  - * R is $E \cup \{ (s, v) \mid v \in V \} \cup \{(v, t) \mid v \in V \}$
  - * $L(v_i)=\{ p_i \}$ and $L(s) = L(t) = \varnothing$.
- s is the initial state in $\mathcal{M}$, and
- f is the formula: **There exists a path that contains all nodes**

$$\mathbf{E}\,(\mathbf{F}\, p_1 \wedge \ldots \wedge \mathbf{F}\, p_n \wedge$$
$$\wedge\, \mathbf{G}\,(p_1 \rightarrow \mathbf{X}\,\mathbf{G} \neg p_1) \wedge \ldots \wedge \mathbf{G}\,(p_n \rightarrow \mathbf{X}\,\mathbf{G} \neg p_n)$$

Each node occurs just once

$\mathcal{M}, s \vDash \mathbf{E}\, f$ holds if and only if there exists an Hamiltonian path in G (observe that f has size polynomial in $|G|$).

Detour: Hamiltonian path in CTL

Taken a graph G=(V, E), we define a Kripke structure $\mathcal{M}$ =(S, R, L) where:

- $S=V \cup \{b\}$ **b is needed to make R total**
- $R=E \cup \{v \to b \mid v \in E\}$
- $L(v)=\{v\}$

We define $f = \bigvee_{(i_1,\ldots,i_n) \text{ permutation of } (1,\ldots,n)} g(v_{i_1}, \ldots, v_{i_n})$ and g inductively as follows:

$$g(v_i) = v_i$$
$$g(v_{i_1}, \ldots, v_{i_n}) = v_{i_1} \wedge \mathbf{E}\,\mathbf{X}\, g(v_{i_2}, \ldots, v_{i_n}) \text{ if } n>1$$

It is easy to see that **$g(v_{i_1}, \ldots, v_{i_n})$ holds** if and only if **$v_{i_1}, \ldots, v_{i_n}$ is a Hamiltonian path** in G.

Therefore, $\mathcal{M} \models f$ if and only if G has a Hamiltonian path.

Obviously, **this reduction is not polynomial**!

Lesson 3c:

Summary of LTL Model Checking

LTL model checking: summary

The problem $\mathcal{M}, s \vDash \mathbf{A}\, f$ is transformed in the refutation of $\mathcal{M}, s \vDash \mathbf{E}\, \neg f$. To verify $\mathcal{M}, s \vDash \mathbf{E}\, f$ where $\mathcal{M} = (S, R, L)$:

1. Build the set of formulas: CL(f).

2. For each state $s \in S$, compute the set K of formulas in CL(f) consistent with $L(s)$ ⇨ atoms (s, K)

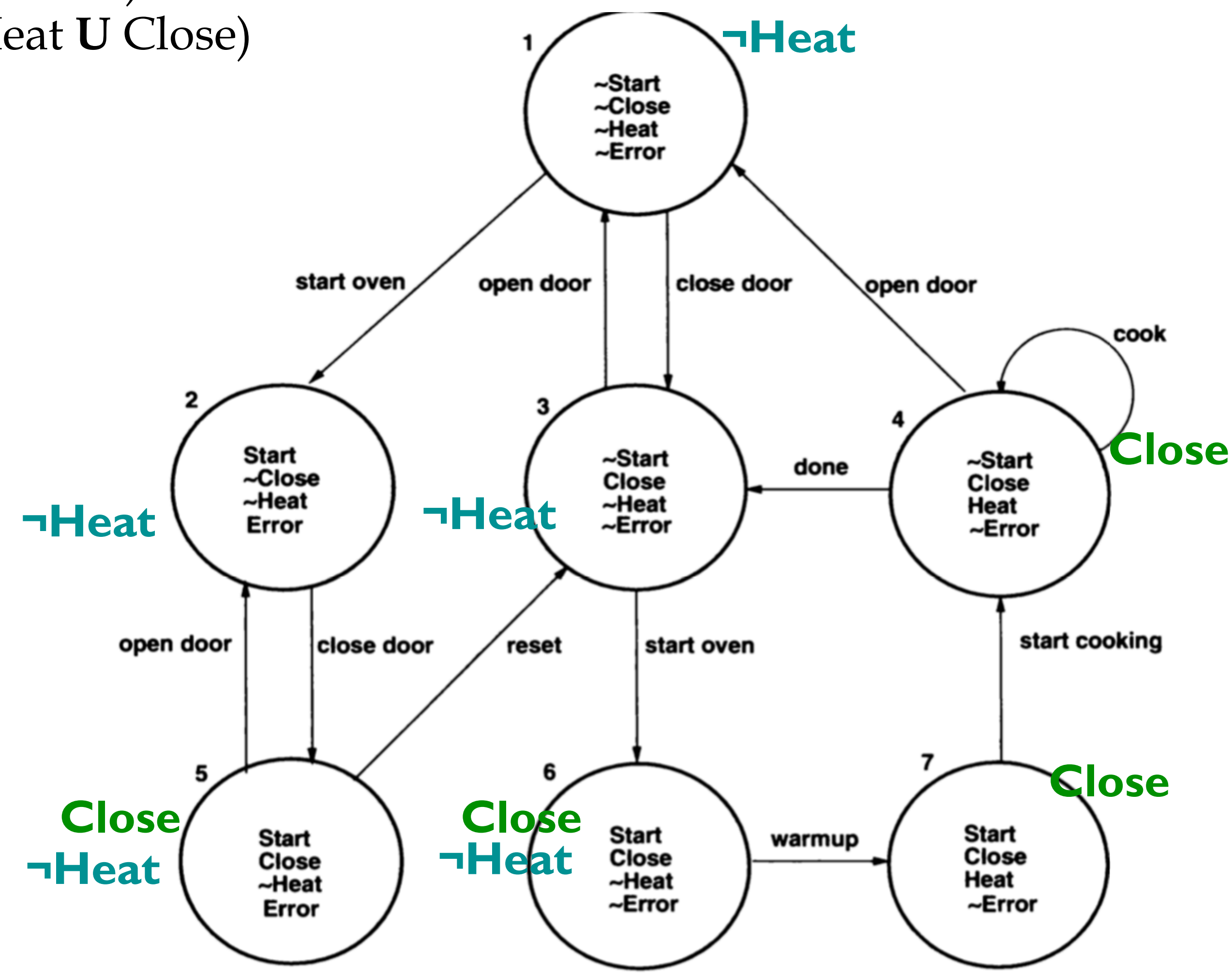
3. Build the graph $G^{\mathcal{M}, f}$ that contains an edge from (s, K) to (s', K') whenever (s, s') is in R, $\mathbf{X}\, g$ in K, and g in K'.

4. Find an eventuality sequence by finding strongly connected components of $G^{\mathcal{M}, f}$

[An **eventuality sequence** is an infinite path π in $G^{\mathcal{M}, f}$ such that if $g_1\, \mathbf{U}\, g_2 \in K$ for some atom (s, K) then there exists an atom (s', K') reachable from (s, K) along π, such that $g_2 \in K'$].

Example: microwave oven

A(¬Heat **U** Close)
≡ ¬**E** ¬ (¬Heat **U** Close)



LTL model checking: example 1

Taking $f \equiv (\neg\text{Heat}\ \mathbf{U}\ \text{Close})$

- Compute the closure of f, $CL(\neg f)$:

  $\{\neg f, f, \mathbf{X} f, \neg \mathbf{X} f, \mathbf{X} \neg f, \text{Heat}, \neg\text{Heat}, \text{Close}, \neg\text{Close}\}$

- Compute atoms:

  Not just subformulas!

  - $\{\neg\text{Heat}, \neg\text{Close}\} \subseteq L(1), L(2)$

    $K_1' = \{\neg\text{Heat}, \neg\text{Close}, f, \mathbf{X} f\}$

    $K_1'' = \{\neg\text{Heat}, \neg\text{Close}, \neg f, \neg\mathbf{X} f, \mathbf{X} \neg f\}$

  - $\{\neg\text{Heat}, \text{Close}\} \subseteq L(3), L(5), L(6)$

    Close is not consistent with $\neg f$

    $K_2' = \{\neg\text{Heat}, \text{Close}, f, \mathbf{X} f\}$

    $K_2'' = \{\neg\text{Heat}, \text{Close}, f, \neg\mathbf{X} f, \mathbf{X} \neg f\}$

  - $\{\text{Heat}, \text{Close}\} \subseteq \text{L}(4), \text{L}(7)$

    $K_3' = \{\text{Heat}, \text{Close}, f, \mathbf{X} f\}$

    $K_3'' = \{\text{Heat}, \text{Close}, f, \neg\mathbf{X} f, \mathbf{X} \neg f\}$

Compute the graph G

Example of transitions:

$(1, K_1') \to (2, K_1')$ because $\mathbf{X} f \in K_1'$, $f \in K_1'$, and $(1,2) \in R$

$(1, K_1'') \to (2, K_1'')$ because $\mathbf{X} \neg f \in K_1'$, $\neg f \in K_1''$, and $(1,2) \in R$

There is no transition $(1, K_1') \to (2, K_1'')$ since $\mathbf{X} f \in K_1'$ but $f \notin K_1''$

Once the full graph is constructed, it is easy to see that there is no atom (s, K) from which there is a path into a self-fullfilling non trivial strong component of $G^{\mathcal{M}, f}$.

Therefore, no state s is such that $\mathcal{M}, s \models \mathbf{E} \neg f$ and hence all states satisfy $\mathcal{M}, s \models \mathbf{A}\, g$.

Lesson 3d:

CTL Model Checking*

Idea of CTL Model Checking*

Idea: use CTL and LTL model checking procedures on sub-formulas.

Substitute any maximal state sub-formulas with fresh atomic propositions. Like CTL algorithm, the CTL* algorithm works in stages.

Level 0: atomic propositions
Level *i*+1: all state sub-formulas *g* such that all state sub-formulas of *g* are of level *i* or less and *g* is not contained in any lower level.

Example: $\mathbf{AG}((\neg \text{Close} \wedge \text{Start}) \rightarrow \mathbf{A}\,(\mathbf{G}\, \neg\text{Heat} \vee \mathbf{F}\, \neg\text{Error}))$
Only **E** quantifier: $\neg\mathbf{EF}((\neg\text{Close} \wedge \text{Start} \wedge \mathbf{E}\,(\mathbf{F}\,\text{Heat} \wedge \mathbf{G}\,\text{Error}))$

Level 0: Close, Start, Heat, Error
Level 1: $\neg$Close, $\mathbf{E}\,(\mathbf{F}\,\text{Heat} \wedge \mathbf{G}\,\text{Error})$
Level 2: $\mathbf{EF}((\neg\text{Close} \wedge \text{Start} \wedge \mathbf{E}\,(\mathbf{F}\,\text{Heat} \wedge \mathbf{G}\,\text{Error}))$
Level 3: $\neg\mathbf{EF}((\neg\text{Close} \wedge \text{Start} \wedge \mathbf{E}\,(\mathbf{F}\,\text{Heat} \wedge \mathbf{G}\,\text{Error}))$

CTL Model Checking: algorithm*

Algorithm 27 CTL* model checking algorithm (basic idea)

Input: finite transition system TS with initial states I, and CTL* formula Φ
Output: $I \subseteq Sat(\Phi)$

for all $i \leqslant |\,\Phi\,|$ **do**
  for all $\Psi \in Sub(\Phi)$ with $|\,\Psi\,| = i$ **do**
    switch(Ψ):

true	:	$Sat(\Psi) := S$;
a	:	$Sat(\Psi) := \{\, s \in S \mid a \in L(s)\,\}$;
$a_1 \wedge a_2$	:	$Sat(\Psi) := Sat(a_1) \cap Sat(a_2)$;
$\neg a$	:	$Sat(\Psi) := S \setminus Sat(a)$;
$\exists \varphi$	:	determine $Sat_{LTL}(\neg\varphi)$ by means of an LTL model-checker;
	:	$Sat(\Psi) := S \setminus Sat_{LTL}(\neg\varphi)$

    end switch
    $AP := AP \cup \{\, a_\Psi \,\}$;   (* introduce fresh atomic proposition *)
    replace Ψ with a_Ψ
    forall $s \in Sat(\Psi)$ **do** $L(s) := L(s) \cup \{\, a_\Psi \,\}$; **od**
  od
od
return $I \subseteq Sat(\Phi)$

CTL: example and complexity*

Example: ¬**EF**((¬Close ∧ Start ∧ **E** (**F** Heat ∧ **G** Error))
Level 1: The level 1 formula ¬Close is added to $L(1)$ and $L(2)$
E (**F** Heat ∧ **G** Error) is pure LTL, but there is no state satisfying this formula.

Level 2: **E** (**F** Heat ∧ **G** Error) is replaced by a fresh atomic proposition a. LTL-model checking is then applied to the formula **EF**((¬Close ∧ Start ∧ a), that is unsatisfiable, so all states are labeled with ¬**EF**((¬Close ∧ Start ∧ **E** (**F** Heat ∧ **G** Error)).

Theorem: There exists a CTL* model checking algorithm with complexity $O(|\mathcal{M}|2^{|f|})$

Theorem: CTL* model checking is PSPACE-complete.

That's all Folks!

Thanks for your attention…

…Questions?