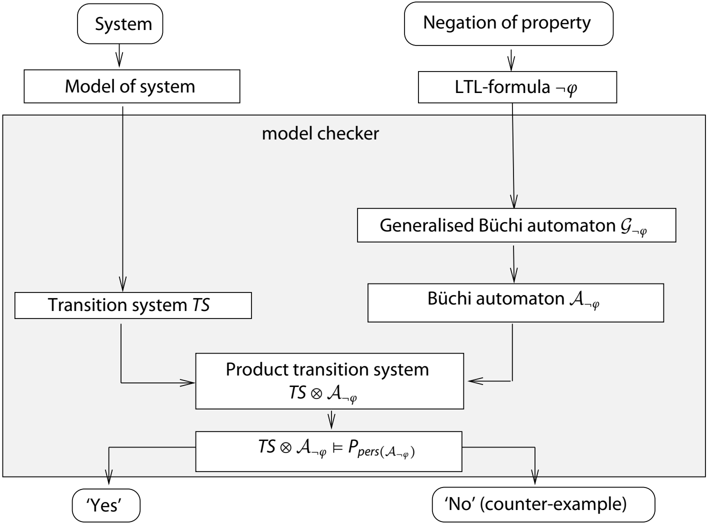
Verification

Lecture 9

Bernd Finkbeiner
Peter Faymonville
Michael Gerke

UNIVERSITÄT
DES
SAARLANDES

REVIEW: Overview of LTL model checking



REVIEW: The GNBA of LTL-formula φ

For LTL-formula φ, let $\mathcal{G}_\varphi = (Q, 2^{AP}, \delta, Q_0, \mathcal{F})$ where

- Q is the set of all elementary sets of formulas $B \subseteq closure(\varphi)$
  - $Q_0 = \{ B \in Q \mid \varphi \in B \}$
- $\mathcal{F} = \{ \{ B \in Q \mid \varphi_1 \mathsf{U} \varphi_2 \notin B \text{ or } \varphi_2 \in B \} \mid \varphi_1 \mathsf{U} \varphi_2 \in closure(\varphi) \}$
- The transition relation $\delta : Q \times 2^{AP} \to 2^Q$ is given by:
  - $\delta(B, B \cap AP)$ is the set of all elementary sets of formulas B' satisfying:
    - (i) For every $\bigcirc \psi \in closure(\varphi)$: $\bigcirc \psi \in B \;\Leftrightarrow\; \psi \in B'$, and
    - (ii) For every $\psi_1 \mathsf{U} \psi_2 \in closure(\varphi)$:

$$\psi_1 \mathsf{U} \psi_2 \in B \;\Leftrightarrow\; \big(\psi_2 \in B \vee (\psi_1 \in B \wedge \psi_1 \mathsf{U} \psi_2 \in B')\big)$$

REVIEW: Main result

[Vardi, Wolper & Sistla 1986]

For any LTL-formula φ (over AP) there exists a GNBA $\mathcal{G}_\varphi$ over 2^{AP} such that:

(a) $Words(\varphi) = \mathcal{L}_\omega(\mathcal{G}_\varphi)$

(b) $\mathcal{G}_\varphi$ can be constructed in time and space $\mathcal{O}\left(2^{|\varphi|}\right)$

(c) #accepting sets of $\mathcal{G}_\varphi$ is bounded above by $\mathcal{O}(|\varphi|)$

$\Rightarrow$ every LTL-formula expresses an ω-regular property!

REVIEW: NBA are more expressive than LTL

There is no LTL formula φ with $Words(\varphi) = P$ for the LT-property:

$$P \;=\; \left\{ A_0 A_1 A_2 \ldots \in \left(2^{\{a\}}\right)^\omega \mid a \in A_{2i} \text{ for } i \geq 0 \right\}$$

But there exists an NBA $\mathcal{A}$ with $\mathcal{L}_\omega(\mathcal{A}) = P$

$\Rightarrow$ there are ω-regular properties that cannot be expressed in LTL!

REVIEW: Complexity for LTL to NBA

For any LTL-formula φ (over *AP*) there exists an NBA $\mathcal{A}_\varphi$
with $\mathit{Words}(\varphi) = \mathcal{L}_\omega(\mathcal{A}_\varphi)$ and
which can be constructed in time and space in $2^{\mathcal{O}(|\varphi|)}$

The LTL model-checking problem is co-NP-hard

The Hamiltonian path problem is polynomially reducible to the complement of the LTL model-checking problem

In fact, the LTL model-checking problem is PSPACE-complete

[Sistla & Clarke 1985]

Reduction to Hamiltonian Path Problem

- Hamiltonian Path for a graph (V, E) passes every vertex exactly once.
- State graph: $(V \cup \{b\}, E \cup (V \cup \{b\}) \times \{b\},$ $L(v) = \{v\}$ for $v \in V, L(b) = \emptyset)$
- LTL property "no path is Hamiltonian":

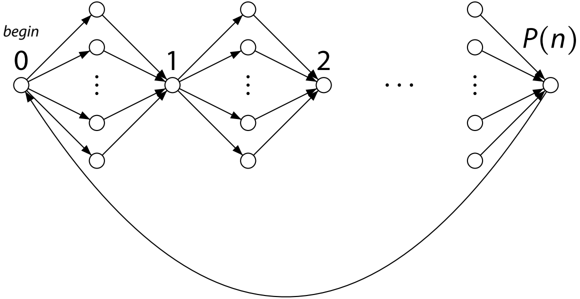
$$\neg \bigwedge_{v \in V} (\Diamond v \quad \wedge \quad \Box (v \rightarrow \bigcirc \Box \neg v))$$

PSPACE-hardness

- Let M be a polynomial space-bounded Turing machine that accepts words of a language K (i.e., K is a PSPACE-language)
- We construct for each word w a state graph S and an LTL formula φ such that $S \models \varphi$ iff $w \in K$.

Single-tape Turing machine $(Q, q_0, F, \Sigma, \delta)$
$\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{L, R, N\}$
L: left, R: right, N: no move

Space-bounded: there is a polynomial $P(n)$ such that the computation on input word of length n visits at most $P(n)$ tape cells.



$$S = \{0, 1, \ldots, P(n)\} \cup \{(q, A, i) \mid q \in Q \cup \{*\}, A \in \Sigma, 0 < i \leq P(n)\}$$

Idea: $q \in Q$ identifies current state of Turing machine and current position of cursor; $*$ everywhere else.

- Configuration (Tape content $A_1, \ldots, A_{P(n)}$, current state q, cursor position i)
  is encoded as path fragment
  $0(*, A_1, 1)1(*, A_2, 2)2 \ldots i-1(q, A_i, i)i(*, A_{i+1}, i+1) \ldots P(n)$
- Computation is encoded as a sequence of such fragments.
- Legal configurations:
  $\varphi_{conf} = \Box\,(begin \rightarrow \varphi^1_{conf} \wedge \varphi^2_{conf})$
  $\varphi^1_{conf} = \bigvee_{1 \leq i \leq P(n)} \bigcirc^{2i-1} \Phi_Q$ where $\Phi_Q = \bigvee_{(q,A,i) \in S, q \in Q} (q, A, i)$
  $\varphi^2_{conf} = \bigwedge_{1 \leq i \leq P(n)} (\bigcirc^{2i-1} \Phi_Q \rightarrow \bigwedge_{1 \leq j \leq P(n), j \neq i} \bigcirc^{2j-1} \neg \Phi_Q)$

Transition function

for $\delta(q, A) = (p, B, L)$:

$$\varphi_{q,A} = \Box \bigwedge_{1 \le i \le P(n)} \left(\bigcirc^{2i-1} (q, A, i) \rightarrow \psi(q, A, i, p, B, L) \right)$$

where

$$\psi(q, A, i, p, B, L) = \underbrace{\bigwedge_{1 \le j \le P(n), i \ne j, C \in \Sigma} \left(\bigcirc^{2j-1} C \leftrightarrow \bigcirc^{2j-1+2P(n)+1} C \right)}_{\text{content of all cells} \ne i \text{ unchanged}}$$

$$\wedge \quad \underbrace{\bigcirc^{2i-1+2P(n)+1} B}_{\text{overwrite } A \text{ by } B \text{ in cell } i}$$

$$\wedge \quad \underbrace{\bigcirc^{2i-1+2P(n)+1-2} p}_{\text{move to state } p \text{ and cursor to cell } i-1}$$

$$\varphi_\delta = \bigwedge_{q,A} \varphi_{q,A}$$

$\left[C \text{ short for } \bigvee_{r,j} (r, C, j),\ p \text{ short for } \bigvee_{D,j} (p, D, j) \right]$

- Starting configuration
  $\varphi^{w}_{start} = begin \wedge \bigcirc q_0 \wedge \bigwedge_{1 \leq i \leq n} \bigcirc^{2i-1} A_i \wedge \bigwedge_{n < i \leq P(n)} \bigcirc^{2i-1} blank$
- Accepting configuration
  $\varphi_{accept} = \Diamond \bigvee_{q \in F} q$
- Full encoding
  $\varphi_w = \varphi_{conf} \wedge \varphi^{w}_{start} \wedge \varphi_\delta \wedge \varphi_{accept}$
  $\Rightarrow$ Model check $\neg\varphi_w$.

PSPACE-completeness

Claim: The LTL model checking problem can be solved by a nondeterministic polynomial space-bounded algorithm

Idea: Guess, nondeterministically, an accepting run in $S \times G_\varphi$:

$u_0 u_1 \ldots u_{n-1} (v_0 v_1 \ldots v_{m-1})^\omega$

where $n, m \leq |S| \cdot 2^{|\varphi|}$

- Guess n, m nondeterministically by guessing $\lceil \log(|S| \cdot 2^{|\varphi|}) \rceil = O(\log(|S|) \cdot |\varphi|)$ bits.
- Guess the sequence $u_0 u_1 \ldots u_{n-1} u_n \ldots u_{n+m}$ where $n_i = (s_i, B_i)$ such that
  - s_i is a successor of s_{i-1} for $i \geq 1$
  - B_i is elementary
  - $B_i \cap AP = L(s_i)$
  - $B_i \in \delta(B_{i-1}, L(s_{i-1}))$ for $i \geq 1$.
- Check if $u_n = u_{n+m}$
- Check that whenever $\varphi_1 \,\mathsf{U}\, \varphi_2 \in B_i$ for some $i \in \{n, \ldots n+m-1\}$ then $\exists j \in \{n, \ldots, n+m-1\}$ with $\varphi_2 \in B_j$

Required space

$n + m$ can be exponential. However, we only need:

- pair of states u_{i-1}, u_i;
- flag which $\varphi_1 \mathrm{U} \varphi_2$ have appeared in loop;
- flag which φ_2 have appeared;
- u_n

$\Rightarrow$ polynomial space

LTL satisfiability and validity checking

- Satisfiability problem: $Words(\varphi) \neq \varnothing$ for LTL-formula φ?
  - does there exist a transition system for which φ holds?
- Solution: construct an NBA $\mathcal{A}_\varphi$ and check for emptiness
  - nested depth-first search for checking persistence properties
- Validity problem: is $\varphi \equiv$ true, i.e., $Words(\varphi) = \left(2^{AP}\right)^\omega$?
  - does φ hold for every transition system?
- Solution: as for satisfiability, as φ is valid iff $\neg\varphi$ is not satisfiable

runtime is exponential;
a more efficient algorithm most probably does not exist!

LTL satisfiability and validity checking

The satisfiability and validity problem for LTL are PSPACE-complete

Idea: Reduce model checking problem of φ to satisfiability problem by encoding transition system as LTL formula:

$$\psi = \psi_I \wedge \Box \psi_S \wedge \Box \psi_{AP}$$

- $\psi_I = \bigvee_{q \in I} q$
- $\psi_S = \bigwedge_{q \in S} q \rightarrow \bigcirc \bigvee_{q' \in Post(q)} q'$
- $\psi_{AP} = \bigwedge_{q \in S} q \rightarrow \bigwedge_{a \in L(q)} a \wedge \bigwedge_{a \notin L(q)} \neg a$

Check satisfiability of $\psi \wedge \neg\varphi$.

Summary of LTL model checking (1)

- LTL is a logic for formalizing path-based properties
- Expansion law allows for rewriting until into local conditions and next
- LTL-formula φ can be transformed algorithmically into NBA $\mathcal{A}_\varphi$
  - this may cause an exponential blow up
  - algorithm: first construct a GNBA for φ; then transform it into an equivalent NBA
- LTL-formulae describe ω-regular LT-properties
  - but do not have the same expressivity as ω-regular languages

Summary of LTL model checking (2)

- $TS \vDash \varphi$ can be solved by a nested depth-first search in $TS \otimes \mathcal{A}_{\neg\varphi}$
  - time complexity of the LTL model-checking algorithm is linear in *TS* and exponential in $|\varphi|$
- Fairness assumptions can be described by LTL-formulae

  the model-checking problem for LTL with fairness is reducible to the standard LTL model-checking problem
- The LTL-model checking problem is PSPACE-complete
- Satisfiability and validity of LTL amounts to NBA emptiness-check
- The satisfiability and validitiy problems for LTL are PSPACE-complete

Linear and branching temporal logic

- Linear temporal logic:

  "statements about (all) paths starting in a state"

  - $s \models \Box\,(x \leq 20)$ iff for all possible paths starting in s always $x \leq 20$
- Branching temporal logic:

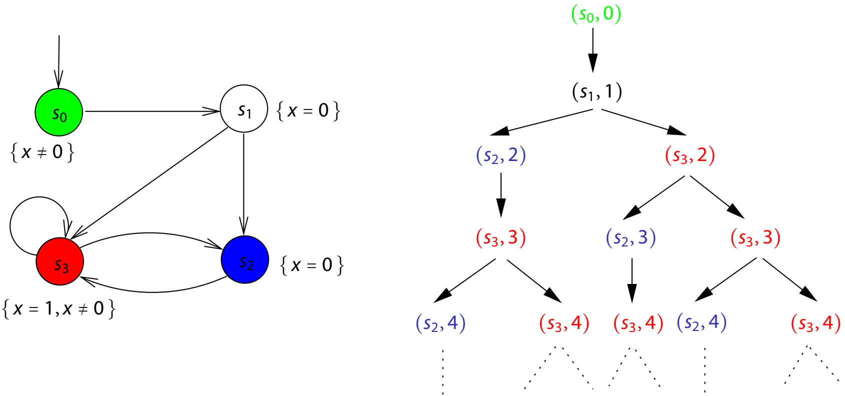
  "statements about all or some paths starting in a state"

  - $s \models \mathrm{AG}\,(x \leq 20)$ iff for **all** paths starting in s always $x \leq 20$
  - $s \models \mathrm{EG}\,(x \leq 20)$ iff for **some** path starting in s always $x \leq 20$
  - nesting of path quantifiers is allowed
- Checking $\mathrm{E}\,\varphi$ in LTL can be done using $\mathrm{A}\,\neg\varphi$
  - . . . but this does not work for nested formulas such as $\mathrm{AG}\,\mathrm{EF}\,a$

Linear versus branching temporal logic

- Semantics is based on a branching notion of time
  - an infinite tree of states obtained by unfolding transition system
  - one "time instant" may have several possible successor "time instants"
- Incomparable expressiveness
  - there are properties that can be expressed in LTL, but not in CTL
  - there are properties that can be expressed in CTL, but not in LTL
- Distinct model checking algorithms, and their time complexities
- Distinct treatment of fairness assumptions
- Distinct equivalences (pre-orders) on transition systems
  - that correspond to logical equivalence in LTL and branching temporal logics

Transition systems and trees



"behavior" in a state s	path-based: $trace(s)$	state-based: computation tree of s
temporal logic	LTL: path formulas φ $s \models \varphi$ iff $\forall \pi \in Paths(s).\, \pi \models \varphi$	CTL: state formulas existential path quantification $\exists\varphi$ universal path quantification: $\forall\varphi$
complexity of the model checking problems	PSPACE--complete $\mathcal{O}\left(\lvert TS\rvert \cdot 2^{\lvert\varphi\rvert}\right)$	PTIME $\mathcal{O}\left(\lvert TS\rvert \cdot \lvert\Phi\rvert\right)$
implementation-relation	trace inclusion and the like (proof is PSPACE-complete)	simulation and bisimulation (proof in polynomial time)
fairness	no special techniques	special techniques needed

Branching temporal logics

There are various branching temporal logics:

- Hennessy-Milner logic
- Computation Tree Logic (CTL)
- Extended Computation Tree Logic (CTL*)
  - combines LTL and CTL into a single framework
- Alternation-free modal μ-calculus
- Modal μ-calculus
- Propositional dynamic logic

Computation tree logic

modal logic over infinite trees [Clarke & Emerson 1981]

- Statements over states
  - $a \in AP$ — atomic proposition
  - $\neg\Phi$ and $\Phi \wedge \Psi$ — negation and conjunction
  - $\mathsf{E}\,\varphi$ — there exists a path fulfilling φ
  - $\mathsf{A}\,\varphi$ — all paths fulfill φ
- Statements over paths
  - $\mathsf{X}\,\Phi$ — the next state fulfills Φ
  - $\Phi\,\mathsf{U}\,\Psi$ — Φ holds until a Ψ-state is reached

$\Rightarrow$ note that X and U alternate with A and E

- $\mathsf{AX\,X}\,\Phi$ and $\mathsf{A\,EX}\,\Phi \notin$ CTL, but $\mathsf{AX\,AX}\,\Phi$ and $\mathsf{AX\,EX}\,\Phi \in$ CTL

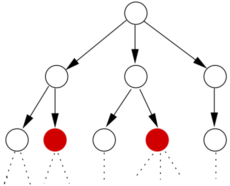
Alternative syntax: $\mathsf{E} \approx \exists$, $\mathsf{A} \approx \forall$, $\mathsf{X} \approx \bigcirc$, $\mathsf{G} \approx \Box$, $\mathsf{F} \approx \Diamond$.

Derived operators

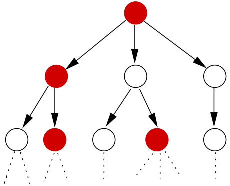
potentially Φ:	$\mathsf{E}\,\mathsf{F}\,\Phi$	$=$	$\mathsf{E}\,(\mathrm{true}\,\mathsf{U}\,\Phi)$
inevitably Φ:	$\mathsf{A}\,\mathsf{F}\,\Phi$	$=$	$\mathsf{A}\,(\mathrm{true}\,\mathsf{U}\,\Phi)$
potentially always Φ:	$\mathsf{E}\,\mathsf{G}\,\Phi$	$:=$	$\neg\mathsf{A}\,\mathsf{F}\,\neg\Phi$
invariantly Φ:	$\mathsf{A}\,\mathsf{G}\,\Phi$	$=$	$\neg\mathsf{E}\,\mathsf{F}\,\neg\Phi$
weak until:	$\mathsf{E}\,(\Phi\,\mathsf{W}\,\Psi)$	$=$	$\neg\mathsf{A}\big((\Phi \wedge \neg\Psi)\,\mathsf{U}\,(\neg\Phi \wedge \neg\Psi)\big)$
	$\mathsf{A}\,(\Phi\,\mathsf{W}\,\Psi)$	$=$	$\neg\mathsf{E}\big((\Phi \wedge \neg\Psi)\,\mathsf{U}\,(\neg\Phi \wedge \neg\Psi)\big)$

the boolean connectives are derived as usual

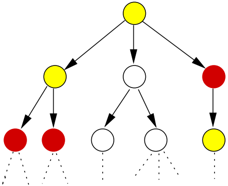
Visualization of semantics



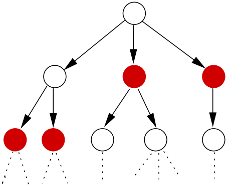
EF *red*



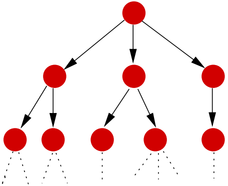
EG *red*



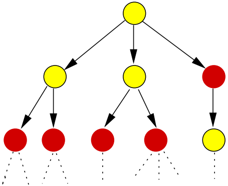
E (*yellow* U *red*)



AF *red*



AG *red*



A (*yellow* U *red*)

Semantics of CTL state-formulas

Defined by a relation $\models$ such that

$s \models \Phi$ *if and only if formula* Φ *holds in state* s

$s \models a$	iff	$a \in L(s)$
$s \models \neg\Phi$	iff	$\neg(s \models \Phi)$
$s \models \Phi \wedge \Psi$	iff	$(s \models \Phi) \wedge (s \models \Psi)$
$s \models \mathsf{E}\,\varphi$	iff	$\pi \models \varphi$ for <u>some</u> path π that starts in s
$s \models \mathsf{A}\,\varphi$	iff	$\pi \models \varphi$ for <u>all</u> paths π that start in s

Semantics of CTL path-formulas

Defined by a relation $\models$ such that

$\boxed{\pi \models \varphi \textit{ if and only if path } \pi \textit{ satisfies } \varphi}$

$\pi \models \mathrm{X}\,\Phi$ iff $\pi[1] \models \Phi$

$\pi \models \Phi\,\mathrm{U}\,\Psi$ iff $(\exists j \geq 0.\, \pi[j] \models \Psi \ \wedge\ (\forall\, 0 \leq k < j.\, \pi[k] \models \Phi))$

where $\pi[i]$ denotes the state s_i in the path π

Transition system semantics

- For CTL-state-formula Φ, the satisfaction set $Sat(\Phi)$ is defined by:

$$Sat(\Phi) \ = \ \{ \, s \in S \mid s \models \Phi \, \}$$

- TS satisfies CTL-formula Φ iff Φ holds in all its initial states:

$$TS \models \Phi \quad \text{if and only if} \quad \forall s_0 \in I. \, s_0 \models \Phi$$

  - this is equivalent to $I \subseteq Sat(\Phi)$
- Point of attention: $TS \not\models \Phi$ and $TS \not\models \neg\Phi$ is possible!
  - because of several initial states, e.g. $s_0 \models \mathrm{EG}\,\Phi$ and $s_0' \not\models \mathrm{EG}\,\Phi$

CTL equivalence

CTL-formulas Φ and Ψ (over *AP*) are equivalent, denoted $\Phi \equiv \Psi$

if and only if $Sat(\Phi) = Sat(\Psi)$ for all transition systems *TS* over *AP*

$$\Phi \equiv \Psi \quad \text{iff} \quad (TS \models \Phi \quad \text{if and only if} \quad TS \models \Psi)$$

Duality laws

$$\begin{aligned}
\mathsf{A}\,\mathsf{X}\,\Phi &\equiv \neg\mathsf{E}\,\mathsf{X}\,\neg\Phi \\
\mathsf{E}\,\mathsf{X}\,\Phi &\equiv \neg\mathsf{A}\,\mathsf{X}\,\neg\Phi \\
\mathsf{A}\,\mathsf{F}\,\Phi &\equiv \neg\mathsf{E}\,\mathsf{G}\,\neg\Phi \\
\mathsf{E}\,\mathsf{F}\,\Phi &\equiv \neg\mathsf{A}\,\mathsf{G}\,\neg\Phi \\
\mathsf{A}\,(\Phi\,\mathsf{U}\,\Psi) &\equiv \neg\mathsf{E}\,((\Phi \wedge \neg\Psi)\,\mathsf{W}\,(\neg\Phi \wedge \neg\Psi))
\end{aligned}$$

Expansion laws

Recall in LTL: $\varphi \, \mathsf{U} \, \psi \equiv \psi \lor (\varphi \land \mathsf{X}(\varphi \, \mathsf{U} \, \psi))$

In CTL:

$$\mathsf{A}(\Phi \, \mathsf{U} \, \Psi) \equiv \Psi \lor (\Phi \land \mathsf{A}\mathsf{X}\mathsf{A}(\Phi \, \mathsf{U} \, \Psi))$$

$$\mathsf{A}\mathsf{F}\Phi \equiv \Phi \lor \mathsf{A}\mathsf{X}\mathsf{A}\mathsf{F}\Phi$$

$$\mathsf{A}\mathsf{G}\Phi \equiv \Phi \land \mathsf{A}\mathsf{X}\mathsf{A}\mathsf{G}\Phi$$

$$\mathsf{E}(\Phi \, \mathsf{U} \, \Psi) \equiv \Psi \lor (\Phi \land \mathsf{E}\mathsf{X}\mathsf{E}(\Phi \, \mathsf{U} \, \Psi))$$

$$\mathsf{E}\mathsf{F}\Phi \equiv \Phi \lor \mathsf{E}\mathsf{X}\mathsf{E}\mathsf{F}\Phi$$

$$\mathsf{E}\mathsf{G}\Phi \equiv \Phi \land \mathsf{E}\mathsf{X}\mathsf{E}\mathsf{G}\Phi$$

Distributive laws

Recall in LTL:

$$\mathsf{G}(\varphi \wedge \psi) \equiv \mathsf{G}\varphi \wedge \mathsf{G}\psi$$
$$\mathsf{F}(\varphi \vee \psi) \equiv \mathsf{F}\varphi \vee \mathsf{F}\psi$$

In CTL:

$$\mathsf{AG}(\Phi \wedge \Psi) \equiv \mathsf{AG}\Phi \wedge \mathsf{AG}\Psi$$

$$\mathsf{EF}(\Phi \vee \Psi) \equiv \mathsf{EF}\Phi \vee \mathsf{EF}\Psi$$

note that $\mathsf{EG}(\Phi \wedge \Psi) \not\equiv \mathsf{EG}\Phi \wedge \mathsf{EG}\Psi$ and $\mathsf{AF}(\Phi \vee \Psi) \not\equiv \mathsf{AF}\Phi \vee \mathsf{AF}\Psi$

Existential normal form (ENF)

The set of CTL formulas in existential normal form (ENF) is given by:

$$\Phi \;::=\; \text{true} \;\Big|\; a \;\Big|\; \Phi_1 \wedge \Phi_2 \;\Big|\; \neg\Phi \;\Big|\; \mathsf{EX}\,\Phi \;\Big|\; \mathsf{E}(\Phi_1 \,\mathsf{U}\, \Phi_2) \;\Big|\; \mathsf{EG}\,\Phi$$

For each CTL formula, there exists an equivalent CTL formula in ENF

$$\mathsf{AX}\,\Phi \equiv \neg\mathsf{EX}\,\neg\Phi$$

$$\mathsf{A}(\Phi\,\mathsf{U}\,\Psi) \equiv \neg\mathsf{E}(\neg\Psi\,\mathsf{U}\,(\neg\Phi \wedge \neg\Psi)) \wedge \neg\mathsf{EG}\,\neg\Psi$$

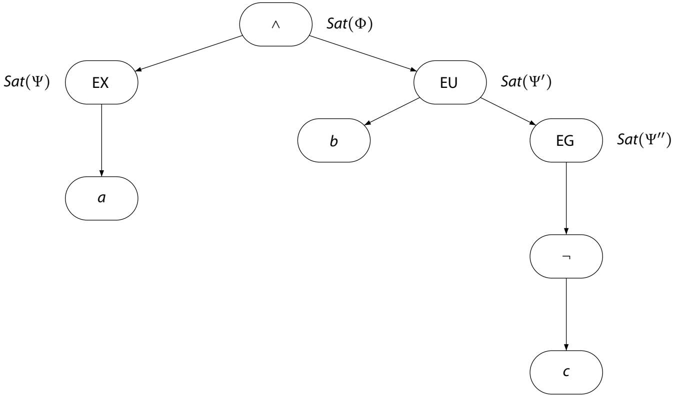
Model checking CTL

- How to check whether state graph *TS* satisfies CTL formula $\widehat{\Phi}$?
  - convert the formula $\widehat{\Phi}$ into the equivalent Φ in ENF
  - compute recursively the set $Sat(\Phi) = \{ q \in S \mid q \models \Phi \}$
  - $TS \models \Phi$ if and only if each initial state of *TS* belongs to $Sat(\Phi)$
- Recursive bottom-up computation of $Sat(\Phi)$:
  - consider the parse-tree of Φ
  - start to compute $Sat(a_i)$, for all leafs in the tree
  - then go one level up in the tree and determine $Sat(\cdot)$ for these nodes

$$\text{e.g.,: } Sat(\underbrace{\Psi_1 \wedge \Psi_2}_{\text{node at level } i}) = Sat(\underbrace{\Psi_1}_{\substack{\text{node at}\\ \text{level } i-1}}) \cap Sat(\underbrace{\Psi_2}_{\substack{\text{node at}\\ \text{level } i-1}})$$

  - then go one level up and determine $Sat(\cdot)$ of these nodes
  - and so on....... until the root is treated, i.e., $Sat(\Phi)$ is computed

Example



$$\Phi = \underbrace{\mathsf{EX}\, a}_{\Psi} \wedge \underbrace{\mathsf{E}\,(b \,\mathsf{U}\, \underbrace{\mathsf{EG}\, \neg c}_{\Psi''})}_{\Psi'} \quad .$$

Basic algorithm

Require: finite transition system *TS* with states *S* and initial states *I*, and CTL formula Φ (both over *AP*)
Ensure: $TS \models \Phi$

{compute the sets $Sat(\Phi) = \{ q \in S \mid q \models \Phi \}$}
for all $i \leq |\Phi|$ **do**
  for all $\Psi \in Sub(\Phi)$ with $|\Psi| = i$ **do**
    compute $Sat(\Psi)$ from $Sat(\Psi')$ {for maximal proper $\Psi' \in Sub(\Psi)$}
  end for
end for
return $I \subseteq Sat(\Phi)$

Characterization of *Sat* (1)

For all CTL formulas Φ, Ψ over *AP* it holds:

$$
\begin{aligned}
Sat(\text{true}) &= S \\
Sat(a) &= \{\, q \in S \mid a \in L(q) \,\}, \text{ for any } a \in AP \\
Sat(\Phi \wedge \Psi) &= Sat(\Phi) \cap Sat(\Psi) \\
Sat(\neg \Phi) &= S \setminus Sat(\Phi) \\
Sat(\text{EX}\,\Phi) &= \{\, q \in S \mid Post(q) \cap Sat(\Phi) \neq \varnothing \,\}
\end{aligned}
$$

for a given finite transition system with states *S*

Characterization of *Sat* (2)

- $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi))$ is the smallest subset T of S, such that:

  (1) $Sat(\Psi) \subseteq T$ and (2) $(q \in Sat(\Phi)$ and $Post(q) \cap T \neq \emptyset) \Rightarrow q \in T$

- $Sat(\mathrm{EG}\,\Phi)$ is the largest subset T of S, such that:

  (3) $T \subseteq Sat(\Phi)$ and (4) $q \in T$ implies $Post(q) \cap T \neq \emptyset$

Computing $Sat(\mathsf{E}\,(\Phi\,\mathsf{U}\,\Psi))$ (1)

- $Sat(\mathsf{E}\,(\Phi\,\mathsf{U}\,\Psi))$ is the smallest set $T \subseteq Q$ such that:

  $$(1)\; Sat(\Psi) \subseteq T \quad \text{and} \quad (2)\; (q \in Sat(\Phi) \text{ and } Post(q) \cap T \neq \varnothing) \;\Rightarrow\; q \in T$$

- This suggests to compute $Sat(\mathsf{E}\,(\Phi\,\mathsf{U}\,\Psi))$ iteratively:

  $$T_0 = Sat(\Psi) \quad \text{and} \quad T_{i+1} = T_i \cup \{\, q \in Sat(\Phi) \mid Post(q) \cap T_i \neq \varnothing \,\}$$

- T_i = states that can reach a Ψ-state in at most i steps via a Φ-path
- By induction on j it follows:

  $$T_0 \subseteq T_1 \subseteq \ldots \subseteq T_j \subseteq T_{j+1} \subseteq \ldots \subseteq Sat(\mathsf{E}\,(\Phi\,\mathsf{U}\,\Psi))$$

Computing $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi))$ (2)

- TS is finite, so for some $j \geq 0$ we have: $T_j = T_{j+1} = T_{j+2} = \ldots$
- Therefore: $T_j = T_j \cup \{\, q \in Sat(\Phi) \mid Post(q) \cap T_j \neq \varnothing \,\}$
- Hence: $\{\, q \in Sat(\Phi) \mid Post(q) \cap T_j \neq \varnothing \,\} \subseteq T_j$
  - hence, T_j satisfies (2), i.e.,
    $(q \in Sat(\Phi)$ and $Post(q) \cap T_j \neq \varnothing) \Rightarrow q \in T_j$
  - further, $Sat(\Psi) = T_0 \subseteq T_j$ so, T_j satisfies (1), i.e. $Sat(\Psi) \subseteq T_j$
- As $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi))$ is the smallest set satisfying (1) and (2):
  - $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)) \subseteq T_j$ and thus $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)) = T_j$
- Hence: $T_0 \subsetneqq T_1 \subsetneqq T_2 \subsetneqq \ldots \subsetneqq T_j = T_{j+1} = \ldots = Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi))$

Computing $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi))$ (3)

Require: finite transition system with states S CTL-formula $\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)$
Ensure: $Sat(\mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)) = \{\, q \in S \mid q \models \mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)\,\}$

$V := Sat(\Psi)$; {V administers states q with $q \models \mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)$}
$T := V$; {T contains the already visited states q with $q \models \mathrm{E}\,(\Phi\,\mathrm{U}\,\Psi)$}
while $V \neq \emptyset$ **do**
  let $q' \in V$;
  $V := V \setminus \{\, q' \,\}$;
  for all $q \in Pre(q')$ **do**
    if $q \in Sat(\Phi) \setminus T$ **then** $V := V \cup \{\, q \,\}$; $T := T \cup \{\, q \,\}$; **endif**
  end for
end while
return T

Computing $Sat(\mathrm{EG}\,\Phi)$

$V := S \setminus Sat(\Phi)$; {$V$ contains any not visited q' with $q' \not\models \mathrm{E\,G}\,\Phi$}

$T := Sat(\Phi)$; {T contains any q for which $q \models \mathrm{E\,G}\,\Phi$ has not yet been disproven}

for all $q \in Sat(\Phi)$ **do** $c[q] := |Post(q)|$; **od** {initialize array c}

while $V \neq \emptyset$ **do**
    {loop invariant: $c[q] = |Post(q) \cap (T \cup V)|$}
    let $q' \in V$; {$q' \not\models \Phi$}
    $V := V \setminus \{q'\}$; {q' has been considered}
    for all $q \in Pre(q')$ **do**
        if $q \in T$ **then**
            $c[q] := c[q] - 1$; {update counter $c[q]$ for predecessor q of q'}
            if $c[q] = 0$ **then**
                $T := T \setminus \{q\}$; $V := V \cup \{q\}$; {q does not have any successor in T}
            end if
        end if
    end for
end while
return T

Alternative algorithm for $Sat(\mathrm{EG}\,\Phi)$

1. Consider only state q if $q \models \Phi$, otherwise eliminate q
   - change states to $S' = Sat(\Phi)$,

   $\Rightarrow$ all removed states will not satisfy EG Φ, and thus can be safely removed
2. Determine all non-trivial strongly connected components in $TS[\Phi]$
   - non-trivial SCC = maximal, connected subgraph with at least one edge

   $\Rightarrow$ any state in such SCC satisfies EG Φ
3. $q \models \mathrm{EG}\,\Phi$ is equivalent to "some SCC is reachable from q"
   - this search can be done in a backward manner