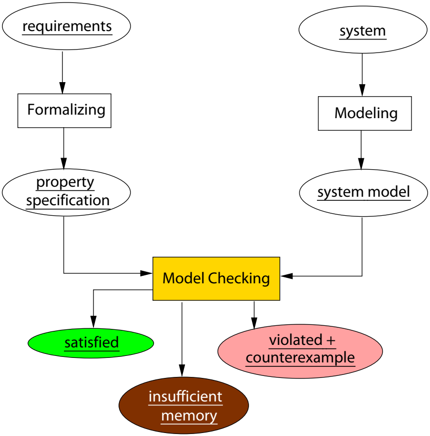
Verification

Lecture 2

Bernd Finkbeiner

UNIVERSITÄT
DES
SAARLANDES

Review: Model checking



Review: Transition systems

A transition system TS is a tuple $(S, Act, \rightarrow, I, AP, L)$ where

- S is a set of states
- Act is a set of actions
- $\longrightarrow \subseteq S \times Act \times S$ is a transition relation
- $I \subseteq S$ is a set of initial states
- AP is a set of atomic propositions
- $L : S \rightarrow 2^{AP}$ is a labeling function

S and Act are either finite or countably infinite

Notation: $s \xrightarrow{\alpha} s'$ instead of $(s, \alpha, s') \in \longrightarrow$

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
  - E φ — there exists a path fulfilling φ
  - A φ — all paths fulfill φ
- Statements over paths
  - X Φ — the next state fulfills Φ
  - Φ U Ψ — Φ holds until a Ψ-state is reached

⇒ note that X and U alternate with A and E

  - AX X Φ and A EX $\Phi \notin$ CTL, but AX AX Φ and AX EX $\Phi \in$ CTL

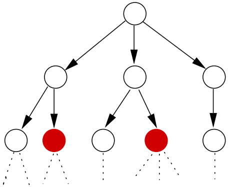
Alternative syntax: E ≈ ∃, A ≈ ∀, X ≈ ◯, G ≈ □, F ≈ ◇.

Derived operators

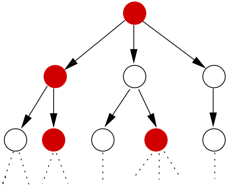
potentially Φ:	$\mathsf{EF}\,\Phi$	$=$	$\mathsf{E}\,(\mathsf{true}\,\mathsf{U}\,\Phi)$
inevitably Φ:	$\mathsf{AF}\,\Phi$	$=$	$\mathsf{A}\,(\mathsf{true}\,\mathsf{U}\,\Phi)$
potentially always Φ:	$\mathsf{EG}\,\Phi$	$:=$	$\neg\mathsf{AF}\,\neg\Phi$
invariantly Φ:	$\mathsf{AG}\,\Phi$	$=$	$\neg\mathsf{EF}\,\neg\Phi$
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

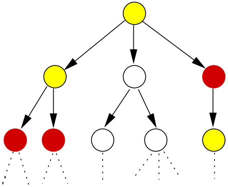
Visualization of semantics



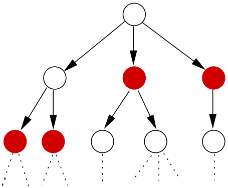
EF *red*



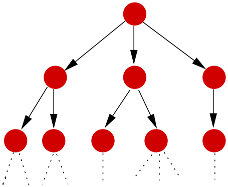
EG *red*



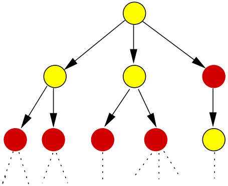
E (*yellow* U *red*)



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	$(s \models \Phi) \wedge (s \models \Psi)$
$s \models \mathrm{E}\,\varphi$	iff	$\pi \models \varphi$ for some path π that starts in s
$s \models \mathrm{A}\,\varphi$	iff	$\pi \models \varphi$ for all paths π that start in s

Semantics of CTL path-formulas

Defined by a relation $\vDash$ such that

$\pi \vDash \varphi$ if and only if path π satisfies φ

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

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

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 $I \subseteq Sat(\Phi)$

- Note: It is possible that both $TS \not\models \Phi$ and $TS \not\models \neg\Phi$
  - (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

$$\mathsf{A}(\Phi\,\mathsf{U}\,\Psi) \equiv \Psi \vee (\Phi \wedge \mathsf{AXA}(\Phi\,\mathsf{U}\,\Psi))$$

$$\mathsf{AF}\,\Phi \equiv \Phi \vee \mathsf{AXAF}\,\Phi$$

$$\mathsf{AG}\,\Phi \equiv \Phi \wedge \mathsf{AXAG}\,\Phi$$

$$\mathsf{E}(\Phi\,\mathsf{U}\,\Psi) \equiv \Psi \vee (\Phi \wedge \mathsf{EXE}(\Phi\,\mathsf{U}\,\Psi))$$

$$\mathsf{EF}\,\Phi \equiv \Phi \vee \mathsf{EXEF}\,\Phi$$

$$\mathsf{EG}\,\Phi \equiv \Phi \wedge \mathsf{EXEG}\,\Phi$$

Distributive laws

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

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

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:

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

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

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

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

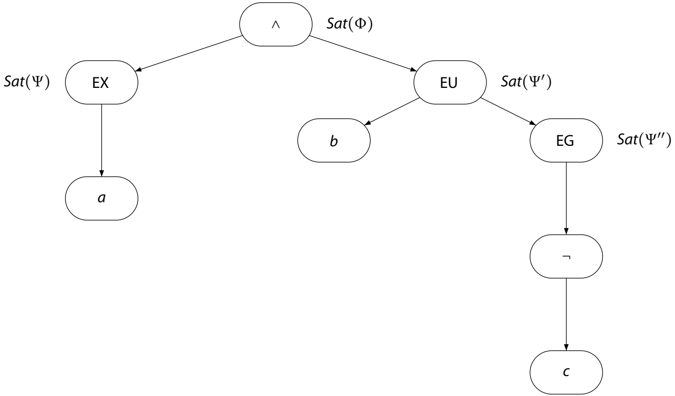
Model checking CTL

- 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 leaves in the tree
  - then go one level up in the tree and determine $Sat(\cdot)$ for these nodes

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

  - 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



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