6. Computational Tree Logic

Computer-Aided Verification

Dave Parker

University of Birmingham

2016/17

Overview

- Linear temporal logic (LTL)
  - equivalence, expressiveness, negation
- Computation tree logic (CTL)
  - syntax, semantics
  - examples
  - CTL vs. LTL
- See [BK08] sections 5.1.4 and 6–6.3

Equivalence

- LTL formulae ψ_1 and ψ_2 are equivalent, written $\psi_1 \equiv \psi_2$ if:
  - they are satisfied by exactly the same traces
  - $\sigma \models \psi_1 \Leftrightarrow \sigma \models \psi_2$ (for any trace σ)
  - i.e. $\text{Words}(\psi_1) = \text{Words}(\psi_2)$

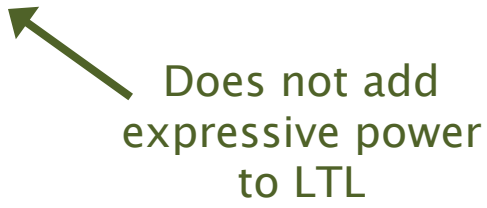
With respect to some set AP of propositions

- Or, equivalently:
  - if they are satisfied by exactly the same models
  - $M \models \psi_1 \Leftrightarrow M \models \psi_2$ (for any LTS M)

- This gives us a notion of expressiveness of LTL
  - "expressiveness" = "expressivity" = '"expressive power"
  - i.e. which models can LTL distinguish between?

LTL equivalences

- Equivalences
  - shorthand for common formulae, e.g.: $\Diamond \psi \equiv \text{true } U \psi$
  - simplifications, e.g.: $\neg\neg p \equiv p$
  - syntax vs. semantics



- Equivalences for: propositional logic + temporal operators

- Temporal operator equivalences:
  - $\Box\psi \equiv \neg\Diamond\neg\psi$ (duality)
  - $\Box\Box\psi \equiv \Box\psi$ (idempotency)
  - $\Diamond\psi \equiv \psi \vee \bigcirc\Diamond\psi$ (expansion law)
  - $\Box(\psi_1 \wedge \psi_2) \equiv \Box\psi_1 \wedge \Box\psi_2$ (distributive law)

Example

- Prove (or disprove):

$$\Diamond\psi \equiv \psi \vee \bigcirc\Diamond\psi \quad ? \qquad \text{Yes}$$

- We can directly consider the semantics for LTL:
- For any trace $\sigma \in (2^{AP})^{\omega}$ …

$$\begin{aligned}
\sigma \vDash \Diamond\psi &\Leftrightarrow \exists k{\geq}0 \text{ s.t. } \sigma[k\ldots] \vDash \psi \\
&\Leftrightarrow \sigma[0\ldots] \vDash \psi \text{ or } \exists k{\geq}1 \text{ s.t. } \sigma[k\ldots] \vDash \psi \\
&\Leftrightarrow \sigma \vDash \psi \text{ or } \exists k{\geq}1 \text{ s.t. } \sigma[k\ldots] \vDash \psi \\
&\Leftrightarrow \sigma \vDash \psi \text{ or } \exists j{\geq}0\ \sigma[1\ldots][j\ldots] \vDash \psi \\
&\Leftrightarrow \sigma \vDash \psi \text{ or } \sigma[1\ldots] \vDash \Diamond\psi \\
&\Leftrightarrow \sigma \vDash \psi \text{ or } \sigma \vDash \bigcirc\Diamond\psi \\
&\Leftrightarrow \sigma \vDash \psi \vee \bigcirc\Diamond\psi
\end{aligned}$$

Example

- Prove (or disprove):

$$\neg(\Box a \rightarrow \Diamond b) \equiv \Box a \wedge \Box \neg b \quad ? \quad \text{Yes}$$

- We can prove this using some basic equivalences, such as:
  - $\Psi_1 \rightarrow \Psi_2 \equiv \neg\Psi_1 \vee \Psi_2$
  - $\Box\Psi \equiv \neg\Diamond\neg\Psi$

Example

- Prove (or disprove):

$$\Box\Diamond a \wedge \Box\Diamond b \equiv \Box\Diamond(a \wedge b) \quad ? \quad \text{No}$$

LTL & Negation

- Are these statements equivalent? (for trace σ and LTL formula ψ)
  - $\sigma \models \neg\psi$
  - $\sigma \not\models \psi$
- Yes
  - in fact, this is just the semantics of LTL

- Are these statements equivalent? (for LTS M and LTL formula ψ)
  - $M \models \neg\psi$
  - $M \not\models \psi$
- No:
  - $M \models \neg\psi$ means no trace satisfies ψ
  - $M \not\models \psi$ means it is not true that all traces satisfy ψ
    - i.e. there exists some trace that does not satisfy ψ

Existential properties

- Can we verify this, using LTL?
  - "there exists an execution that reaches program location l_2"
- Yes: $M \not\models \Box\neg l_2$

- Can we verify this, using LTL?
  - "there exists an execution that visits l_2 infinitely often, and never passes through program location l_4"
- Yes: $M \not\models \neg((\Box\Diamond l_2) \wedge (\Box\neg l_4))$

- Can we verify this, using LTL?
  - "for every execution, it is always possible to return to the initial state of the program"
- No…

CTL

- CTL – Computation Tree Logic
  - branching notion of time (compared to linear time for LTL)
  - infinite trees of states, not infinite sequences of states
- Two path quantifiers: $\forall$ (for all) and $\exists$ (there exists)
  - LTL implicitly uses $\forall$
- Example
  - $\exists \Diamond l_2$ – "does there exist an execution that reaches l_2?"
- CTL model checking
  - quite different to (and simpler than) LTL model checking

CTL syntax

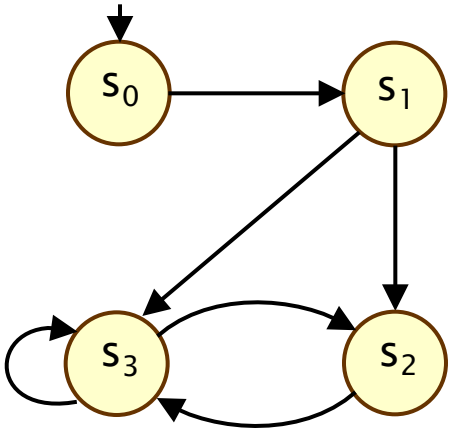
- Syntax split into state and path formulae
  - specify properties of states/paths, respectively
  - a CTL formula is a state formula ϕ
- State formulae:
  - ϕ ::= true | a | $\phi \wedge \phi$ | $\neg\phi$ | $\forall\ \psi$ | $\exists\ \psi$
  - where a $\in$ AP and ψ is a path formula
- Path formulae
  - ψ ::= $\bigcirc\ \phi$ | ϕ U ϕ | $\Diamond\ \phi$ | $\Box\ \phi$
  - where ϕ is a state formula
- Example:
  - $\forall\Box\ \exists\Diamond$ initial

CTL – Alternative styles

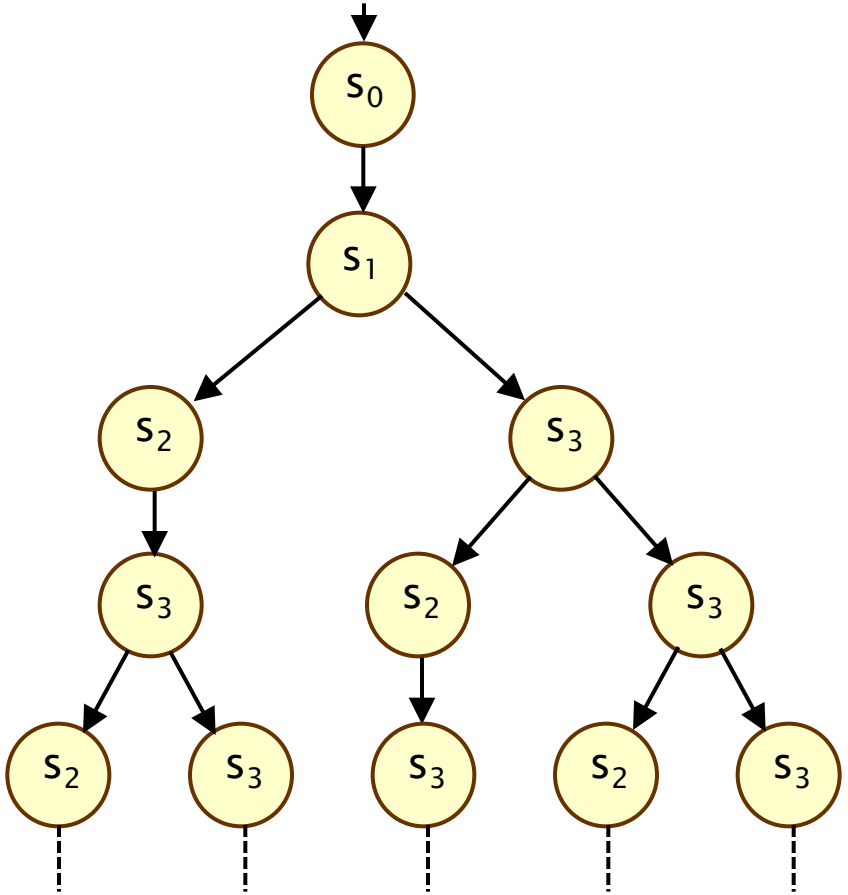
- Temporal operators:
  - ◯ a ≡ X a ("next")
  - ◇ a ≡ F a ("future", "finally")
  - □ a ≡ G a ("globally")
- Path quantifiers:
  - ∀ ψ ≡ A ψ
  - ∃ ψ ≡ E ψ
- Brackets: none/round/square
  - ∀◇ ψ
  - ∀ (ψ_1 U ψ_2)
  - ∀ [ψ_1 U ψ_2]

Computation trees

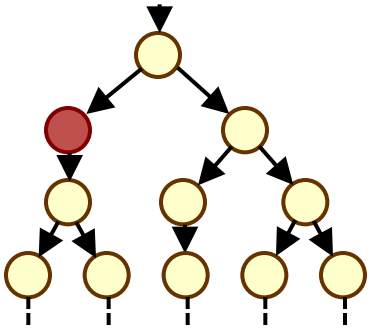
- LTS:



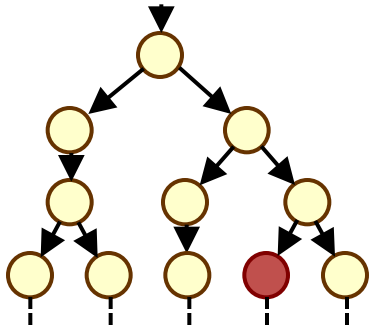
- (Prefix of) infinite computation tree



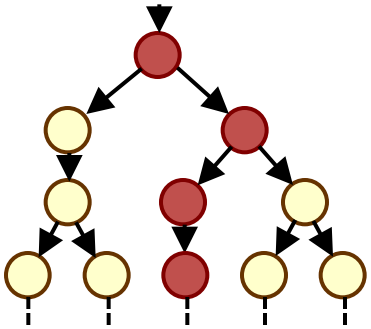
CTL – Intuitive semantics



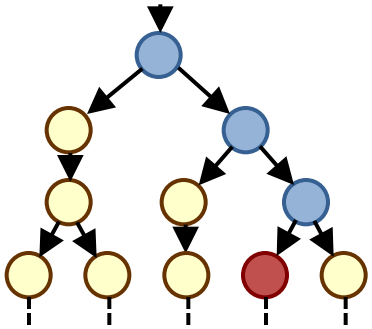
∃◯ red



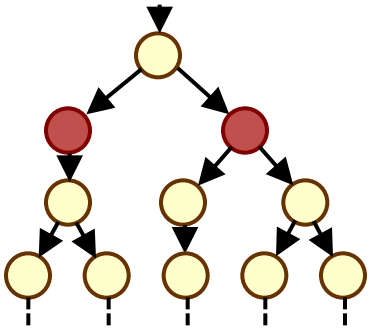
∃◇ red



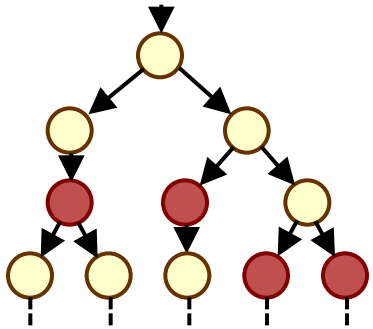
∃□ red



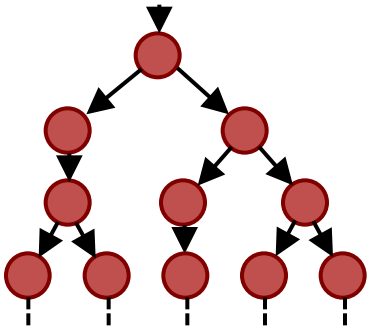
∃ [blue U red]



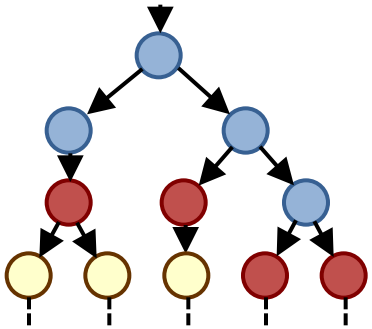
∀◯ red



∀◇ red



∀□ red



∀ [blue U red]

CTL examples

- $\forall\Box\ (\neg(crit_1 \wedge crit_2))$
  - mutual exclusion
- $\forall\Box\ \exists\Diamond$ initial
  - for every computation, it is always possible to return to the initial state
- $\forall\Box$ (request $\rightarrow\ \forall\Diamond$ response)
  - every request will eventually be granted
- $\forall\Box\ \forall\Diamond\ crit_1\ \wedge\ \forall\Box\ \forall\Diamond\ crit_2$
  - each process has access to the critical section infinitely often

CTL semantics

- Semantics of state formulae:
  - $s \models \phi$ denotes "s satisfies ϕ" or "ϕ is true in s"
- For a state s of an LTS $(S, Act, \rightarrow, I, AP, L)$:
  - $s \models true$ always
  - $s \models a \Leftrightarrow a \in L(s)$
  - $s \models \phi_1 \wedge \phi_2 \Leftrightarrow s \models \phi_1$ and $s \models \phi_2$
  - $s \models \neg\phi \Leftrightarrow s \not\models \phi$
  - $s \models \forall\, \psi \Leftrightarrow \pi \models \psi$ for all $\pi \in Path(s)$
  - $s \models \exists\, \psi \Leftrightarrow \pi \models \psi$ for some $\pi \in Path(s)$
- and for a path ω:
  - $\pi \models \bigcirc \phi \Leftrightarrow \pi[1] \models \phi$
  - $\pi \models \phi_1\, U\, \phi_2 \Leftrightarrow \exists k \geq 0$ s.t. $\pi[k] \models \phi_2$ and $\forall i < k\ \pi[i] \models \phi_1$

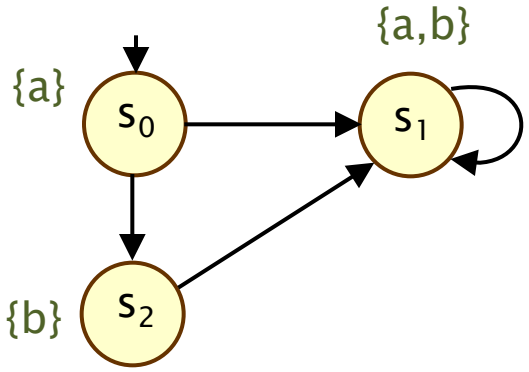
CTL equivalences

- Again various operators can be derived
  - propositional logic: $\vee$, $\rightarrow$, $\leftrightarrow$, $\oplus$
- Path quantifier duality:
  - $\forall\psi \equiv \neg\exists\neg\psi$
  - $\exists\psi \equiv \neg\forall\neg\psi$
- Temporal operators:
  - $\Diamond\, \phi \equiv$ true U ϕ
  - $\Box\, \phi \equiv$?
- For example:
  - $\forall\Box\, \phi \equiv \neg\exists\Diamond(\neg\phi)$

Examples

- $s_0 \models \forall\bigcirc b$?
- $s_0 \models \exists\bigcirc \neg b$?
- $s_0 \models \exists(a \; U \; a \wedge b)$?
- $s_0 \models \exists\bigcirc \forall\Box (a \wedge b)$?

M:



CTL vs LTL

- How do we compare the expressiveness of CTL and LTL?
  - evaluated over states and paths, respectively

- Satisfaction of a CTL formula ϕ by an LTS M:
  - $M \models \phi$ if $s \models \phi$ for all initial states s of M

- CTL formulae ϕ_1 and ϕ_2 are equivalent ($\phi_1 \equiv \phi_2$) if
  - $M \models \phi_1 \Leftrightarrow M \models \phi_2$ (for any LTS M)

- CTL formula ϕ and LTL formula ψ are equivalent ($\phi \equiv \psi$) if
  - $M \models \phi \Leftrightarrow M \models \psi$ (for any LTS M)

Questions

- Is CTL more expressive than LTL?

- What can we express in LTL that we cannot in CTL?