Computation Tree Logic (CTL)

Antonio González Burgueño

University of Oslo, Norway

May 26, 2017

Outline

Outline

LTL vs CTL

LTL

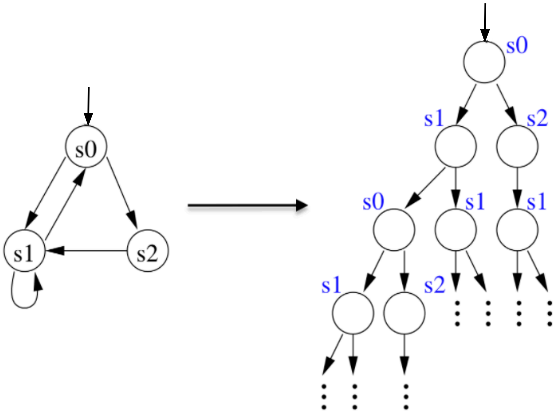
- Describes properties of individual executions.
- Semantics defined as a set of executions.
- LTL formulas ψ are evaluated on paths (path formulas).

CTL

- Describes properties of a computation tree.
  - Formulas can reason about many executions at once.
- Semantics defined in terms of states.
- CTL formulas ϕ are evaluated on states (state formulas)

Model of Comuputation (I)

- Computation trees are derived from state transition graphs.
- The graph structure is unwound into an infinite tree rooted at the initial state.



Unwind a Graph Into a Tree.

Model of Comuputation (II)

Formally, a *Kripke structure* is a triple $M = \langle S, R, L \rangle$, where

- S is the set of states,
- $R \subseteq S \times S$ is the transition relation, and
- $L : S \rightarrow \mathcal{P}(AP)$ gives the set of atomic propositions true in each state.

We assume that R is total

- $\forall s \in S,\ \exists s' \in S : (s, s') \in R$

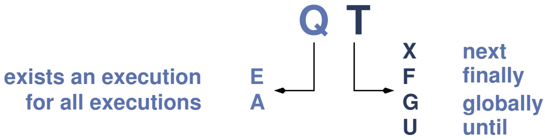
A path in M is an infinite sequence π of states:

- $\pi = s_0, s_1, ...$ such that for $i \geq 0$, $(s_i, s_{i+1}) \in R$

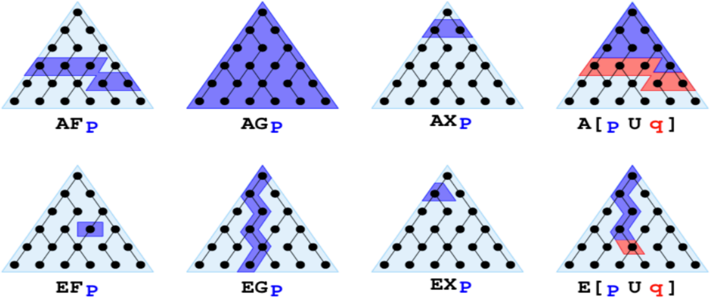
Outline

CTL Syntax.

- Combines temporal operators with quantification over runs.
- Operators have the following form:



Visualization of semantics



CTL Examples.

Let “P” mean “I like chocolate”.

- **AG**.P: “I will like chocolate from now on, no matter what happens”.
- **EF**.P: "It is possible I may like chocolate some day, at least for one day”.
- **AF.EG**.P: "It is always possible (AF) that I will suddenly start liking chocolate for the rest of time”.
- **EG.AF**.P: "Depending on what happens next, it is possible (E) that for the rest of time (G), there will always be some time in the future (AF) when I will like chocolate.

CTL semantics

The Backus-Naur form form CTL formula is the following:

$$\phi ::= \quad \top \mid \bot \mid p \mid \neg\,\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid AX\phi \mid EX\phi$$
$$AF\phi \mid EF\phi \mid AG\phi \mid EG\phi \mid A[\phi \mathbf{U} \phi] \mid E[\phi \mathbf{U} \phi]$$

Let ϕ be a CTL formula and $s \in S$. $M, s \models \phi$, where ϕ is true in all the initial states of the model, is defined as follows:

- $M, s \models \top$
- $M, s \not\models \bot$
- $M, s \models p$ iff $p \in L(s)$
- $M, s \models \neg\,\phi$ iff $M, s \not\models \phi$
- $M, s \models \phi \wedge \psi$ iff $M, s \models \phi$ and $M, s \models \psi$
- $M, s \models \phi \vee \psi$ iff $M, s \models \phi$ or $M, s \models \psi$

CTL semantics. Temporal Operators (I)

- $M, s \models \mathrm{AX}\phi$ iff $\forall s'\ s.t\ sR_ts', M, s' \models \phi$
- $M, s \models \mathrm{EX}\phi$ iff $\exists s'\ s.t\ sR_ts'$ and $M, s' \models \phi$
- $M, s \models \mathrm{AG}\phi$ iff $\forall \pi = (s, s_2, s_3, s_4, ...)$ s.t. $s_i R_t s_{i+1}$ and for all i, it is the case that $M, s_i \models \phi$
- $M, s \models \mathrm{EG}\phi$ iff $\exists \pi = (s, s_2, s_3, s_4, ...)$ s.t. $s_i R_t s_{i+1}$ and for all i, it is the case that $M, s_i \models \phi$

CTL semantics. Temporal Operators (II)

- $M, s \models \mathsf{AF}\phi$ iff $\forall \pi = (s, s_2, s_3, s_4, ...)$ s.t. $s_i R_t s_{i+1}$, there is a state s_i s.t $M, s_i \models \phi$
- $M, s \models \mathsf{EF}\phi$ iff $\exists \pi = (s, s_2, s_3, s_4, ...)$ s.t. $s_i R_t s_{i+1}$, and there is a state s_i s.t $M, s_i \models \phi$
- $M, s \models A[\phi \mathbf{U} \psi]$ iff $\forall \pi = (s, s_2, s_3, s_4, ...)$ s.t. $s_i R_t s_{i+1}$, there is a state s_j s.t M, $s_i \models \phi$ and M, $s_j \models \psi$ for all $i < j$
- $M, s \models E[\phi \mathbf{U} \psi]$ iff $\exists \pi = (s, s_2, s_3, s_4, ...)$ s.t. $s_i R_t s_{i+1}$, there is a state s_j s.t M, $s_i \models \phi$ and M, $s_j \models \psi$ for all $i < j$

Basic Set of CTL Operators

There are eight basic CTL operators:

- **AX** and **EX**,
- **AG** and **EG**,
- **AF** and **EF**, and
- **AU** and **EU**.

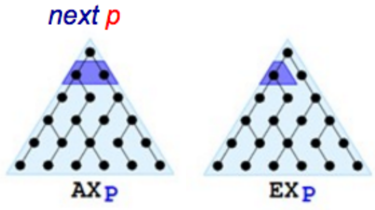
That can be expressed in terms of the operators **EX**, **EG** and **EU**

- $\textbf{AX}\, f = \neg\, \textbf{EX}(\neg f)$,
- $\textbf{AG}\, f = \neg\, \textbf{EF}(\neg f)$,
- $\textbf{AF}\, f = \neg\, \textbf{EG}(\neg f)$,
- $\textbf{EF}\, f = \textbf{E}\, [\, \textit{true}\ \mathbf{U}\, f\,]$
- $\textbf{A}\, [\, f\ \mathbf{U}\, g\,] = \neg\, \textbf{E}[\neg g\ \mathbf{U}\ \neg f \wedge \neg\, g\,] \wedge \neg\, \textbf{EG}\neg g$

Expressiveness of CTL and LTL(I)

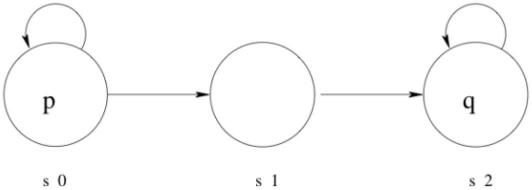
Any CTL formula ϕ using:

- $\mathbf{A}$ operator can be expressed in LTL; e.g. $\mathbf{AG}\phi_{CTL} \equiv \mathbf{G}\phi_{LTL}$ and $\mathbf{AX}\phi_{CTL} \equiv \mathbf{X}\phi_{LTL}$
- $\mathbf{E}$ operator cannot be expressed in LTL; e.g. $\mathbf{EX}\,p \not\equiv \mathbf{X}\,p$.



Expressiveness of CTL and LTL(II)

- $\mathbf{GF}\,p \Rightarrow \mathbf{GF}\,q$
  - $(\mathbf{GF}\,p \equiv \mathbf{AGAF}\,p)$ and $(\mathbf{GF}q \equiv \mathbf{AGAF}\,q)$
  - $(\mathbf{GF}\,p \Rightarrow \mathbf{GF}\,q) \not\equiv (\mathbf{AGAF}\,p \Rightarrow \mathbf{AGAF}\,q)$



- The CTL is trivially satisfied, because $\mathbf{AGAF}\,p$ is not satisfied.
- LTL is not satisfied, because the path cycling through s_0 forever satisfies $\mathbf{GF}\,p$ but not $\mathbf{GF}\,q$.
- The LTL formula is an implication about paths, but the two parts of the CTL formula determine subsets of states independently.

Outline

CTL Model Checking

- Assumptions:
  - Finite number of processes, each having a finite number of finite-valued variables.
  - Finite length of CTL formula
- Problem: Determine whether ϕ is true in a finite structure M.
- Algorithm overview:
  1. Convert ϕ in terms of **AF**, **EU**, **EX**, $\wedge$, $\vee$, $\perp$.
  2. Label the states of M with the subformulas of ϕ that are satisfied there.
  3. If starting state s_0 is labeled with ϕ, then ϕ holds on M; i.e., $(s_0 \in \{s|M, s \models \phi\}) \Rightarrow (M \models \phi)$

Labeling Algorithm(I)

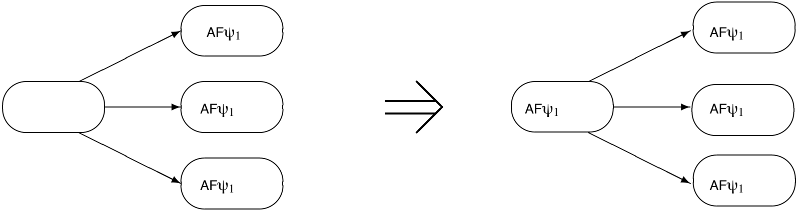
- Suppose ψ is a subformula of ϕ and states satisfying all the immediate subformulas of ψ have already been labeled.
- We want to determine which states to label with ψ.
- If ψ is:
  - $\perp$: Then no states are labeled with $\perp$
  - p: label s with p if $p \in L(s)$.
  - $\psi_1 \wedge \psi_2$: label s with $\psi_1 \wedge \psi_2$ if s is already labeled both with ψ_1 and with ψ_2.
  - $\neg\psi_1$: label s with $\neg\psi_1$ if s is not already labeled with ψ_1.
  - **EX** ψ_1: label any state with **EX** ψ_1 if one of its successors is labeled with ψ_1.

Labeling Algorithm(II)

AF ψ_1

- If any state s is labeled with ψ_1, label it with **AF** ψ_1.
- Repeat: label any state with **AF** ψ_1 if all successor states are labeled with **AF** ψ_1, until there is no change.

For example:

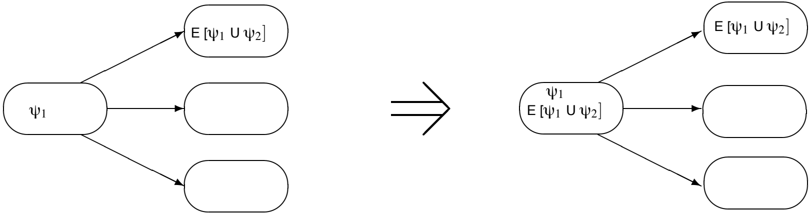


Labeling Algorithm(III)

$\mathbf{E}[\psi_1 \, \mathbf{U} \, \psi_2]$

- If any state s is labeled with ψ_2, label it with $\mathbf{E}[\psi_1 \, \mathbf{U} \, \psi_2]$.
- Repeat: label any state with $\mathbf{E}[\psi_1 \, \mathbf{U} \, \psi_2]$ if it is labeled with ψ_1 and at least one of its successors is labeled with $\mathbf{E}[\psi_1 \, \mathbf{U} \, \psi_2]$, until there is no change.

For example:



Output states labeled with f. Complexity: $O(|f| \times S \times (S + |R|))$ (linear in the size of the formula and quadratic in the size of the model).

Fairness (I)

- Often liveness properties (something good eventually happens) cannot be proven without certain assumptions, i.e., fairness.
- Fairness: something happens infinitely often or repeatedly.
  - Executions are fair if a system enters a state infinitely often, and
  - Takes every possible transition from that state.
- Example: Liveness condition at the Dining Philosophers Problem.
  - Any philosopher who tries to eat, eventually does.

Fairness (II)

Weak/strong fairness can be expressed in LTL

- Weak fairness: if an event is continuously enabled, it will occur infinitely often
  - LTL: **GF** ($\neg$ enabled $\vee$ occurs)
- Strong fairness: if a event is infinitely often enabled it will occur infinitely often
  - LTL: **GF** enabled $\Rightarrow$ GF occurs

Fairness (III)

- In LTL holds $M \models_{fair} \psi$ if and only if $M \models (\text{fair} \to \psi)$.
- Formulas of the form $\forall(fair \to \psi)$ and $\exists(fair \wedge \psi)$ needed.
- CTL problem:
  - Boolean combinations of path formulas are not allowed in CTL
  - Example: strong fairness constraints $\Box\Diamond b \to \Box\Diamond c \equiv \Diamond\Box\neg b \vee \Diamond\Box c$ cannot be expressed in CTL because persistence properties cannot be represented.
- Solution: change the semantics of CTL by ignoring unfair paths.

Semantics of fair CTL

CTL fairness assumption *fair*, relation $\models_{fair}$ is defined by:

$$
\begin{array}{lll}
s \models_{fair} a & \textit{iff} & a \in Label(s) \\
s \models_{fair} \neg\phi & \textit{iff} & \neg(s \models_{fair} \phi) \\
s \models_{fair} \phi \vee \psi & \textit{iff} & (s \models_{fair} \phi) \vee (s \models_{fair} \psi) \\
s \models_{fair} \exists\varphi & \textit{iff} & \pi \models_{fair} \varphi \text{ for some fair path } \pi \text{ that starts in } s \\
s \models_{fair} \forall\varphi & \textit{iff} & \pi \models_{fair} \varphi \text{ for all fair paths } \pi \text{ that start in } s \\
\pi \models_{fair} \bigcirc\phi & \textit{iff} & \pi[1] \models_{fair} \phi \\
\pi \models_{fair} \phi \bigcup \psi & \textit{iff} & \exists j.\, j \geq 0, \pi[j] \models_{fair} \psi \wedge \forall k, 0 \leq k < j, \pi[k] \models_{fair} \phi
\end{array}
$$

where π is a fair path *iff* $\pi \models_{LTL} \textit{fair}$ for CTL fairness assumption *fair*.

CTL with fairness constraints

- Fair path: a path in the model along which each fairness condition holds infinitely often.
- Fair states: states reachable along fair paths
- Let $C = \psi_1, \psi_2, ..., \psi_n$ be a set of n fairness constraints.
  - Sets of states (constraint) that must occur infinitely often along a computation path to be considered.
  - Restrict the path quantifiers (**E** and **A**) to fair paths.
  - **EF**ψ holds at state s only if there exists a fair path from s along which ϕ holds.
  - **AG**ψ holds at s if ψ holds in all states reachable from s along fair paths.

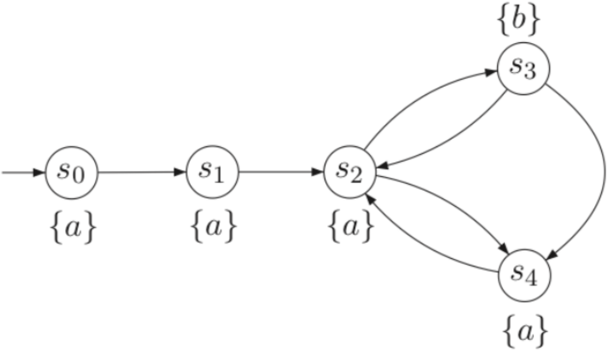
Algorithm for fairness in CTL

An algorithm for fairness in CTL is as follows:

1. Restrict the graph to states satisfying ϕ; of the resulting graph, we want to know from which states there is a fair path.
2. Find the maximal strongly connected components (SCC) of the restricted graph;
3. Remove a SCC if, for some ψ_i, it does not contain a state satisfying ψ_i. The resulting SCCs are the fair SCCs. Any state of the restricted graph that can reach one has a fair path from it.
4. Use backwards breadth-first searching to find the states on the restricted graph that can reach a fair SCC.

Fairness in CTL. Example

$M \not\models \forall a(a \to \forall\Diamond b)$.



- $C = \{(\Box\Diamond s_2 \to \Box\Diamond a), (\Box\Diamond s_2 \to \Box\Diamond b)\}$.
- Both loops should be visited fairly.
- $M \models_{fair} \forall a(a \to \forall\Diamond b)$.

References

- E. M. Clarke, E. A. Emerson and A. P. Sistla. Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. ACM Transactions on Programming Languages and Systems, April, 1986
- Z. Manna and A. Pnueli. Temporal Verification of Reactive Systems: Safety, 1995.
- Model Checking. Edmund M Clarke, jr., Orna Grumberg, and Doron Peled, MIT Press, 1999
- Michael Huth and Mark Ryan Logic in Computer Science: Modelling and Reasoning about Systems. The MIT Press, 1999.
- Logic in Computer Science: Modelling and Reasoning about Systems Michael Huth, Mark Ryan Cambridge University Press 2004. http://www.cs.bham.ac.uk/research/lics/
- Christel Baier and Joost-Pieter Katoen. Principles of Model Checking, MIT Press 2008.