FORMAL METHODS

Lecture IV: Computation Tree Logic (CTL)

Alessandro Artale

Faculty of Computer Science – Free University of Bolzano

artale@inf.unibz.it  http://www.inf.unibz.it/~artale/

Some material (text, figures) displayed in these slides is courtesy of:
Computation Tree Logic: Intuitions.

CTL: Syntax and Semantics.

CTL in Computer Science.

CTL and Model Checking: Examples.

CTL Vs. LTL.

CTL*.
LTL implicitly quantifies *universally* over paths.

\[ \langle \mathcal{KM}, s \rangle \models \phi \quad \text{iff for every path } \pi \text{ starting at } s \langle \mathcal{KM}, \pi \rangle \models \phi \]

Properties that assert the *existence* of a path cannot be expressed. In particular, properties which *mix* existential and universal path quantifiers cannot be expressed.

The *Computation Tree Logic*, CTL, solves these problems!

- CTL explicitly introduces *path quantifiers*!
- CTL is the natural temporal logic interpreted over Branching Time Structures.
CTL at a glance

- CTL is evaluated over branching-time structures (Trees).
- CTL explicitly introduces path quantifiers:
  - All Paths: □
  - Exists a Path: ♦.

- Every temporal operator (□, ♦, ○, U) preceded by a path quantifier (□ or ♦).

- Universal modalities: □♦, □□, □○, □U
  The temporal formula is true in all the paths starting in the current state.

- Existential modalities: ♦♦, ♦□, ♦○, ♦U
  The temporal formula is true in some path starting in the current state.
Summary

- Computation Tree Logic: Intuitions.
- **CTL: Syntax and Semantics.**
- CTL in Computer Science.
- CTL and Model Checking: Examples.
- CTL Vs. LTL.
- CTL*. 
Countable set $\Sigma$ of *atomic propositions*: $p, q, \ldots$ the set $\text{ FORM}$ of formulas is:

$$\varphi, \psi \rightarrow p \mid \top \mid \bot \mid \neg \varphi \mid \varphi \land \psi \mid \varphi \lor \psi \mid$$

$$\blacksquare \varphi \mid \square \varphi \mid \Diamond \varphi \mid \lozenge (\varphi \cup \psi)$$

$$\Diamond \lozenge \varphi \mid \Diamond \square \varphi \mid \lozenge \Diamond \varphi \mid \lozenge (\varphi \cup \psi)$$
We interpret our CTL temporal formulas over Kripke Models linearized as trees (e.g. $\Box done$).

Universal modalities ($\Box P$, $\Box \Box P$, $\Box \Diamond P$, $\Box U$): the temporal formula is true in all the paths starting in the current state.

Existential modalities ($\Diamond P$, $\Diamond \Box P$, $\Diamond \Diamond P$, $\Diamond U$): the temporal formula is true in some path starting in the current state.
Let $\Sigma$ be a set of atomic propositions. We interpret our CTL temporal formulas over Kripke Models:

$KM = \langle S, I, R, \Sigma, L \rangle$

The semantics of a temporal formula is provided by the satisfaction relation:

$\models: (KM \times S \times \text{FORM}) \rightarrow \{\text{true, false}\}$
We start by defining when an atomic proposition is true at a state/time "s_i"

\[ \text{CTL Semantics: The Propositional Aspect} \]

\[ \mathcal{K}M, s_i \models p \iff p \in L(s_i) \quad (\text{for } p \in \Sigma) \]

The semantics for the classical operators is as expected:

\[ \mathcal{K}M, s_i \models \neg \varphi \iff \mathcal{K}M, s_i \not\models \varphi \]

\[ \mathcal{K}M, s_i \models \varphi \land \psi \iff \mathcal{K}M, s_i \models \varphi \text{ and } \mathcal{K}M, s_i \models \psi \]

\[ \mathcal{K}M, s_i \models \varphi \lor \psi \iff \mathcal{K}M, s_i \models \varphi \text{ or } \mathcal{K}M, s_i \models \psi \]

\[ \mathcal{K}M, s_i \models \varphi \Rightarrow \psi \iff \text{if } \mathcal{K}M, s_i \models \varphi \text{ then } \mathcal{K}M, s_i \models \psi \]

\[ \mathcal{K}M, s_i \models \top \]

\[ \mathcal{K}M, s_i \not\models \bot \]
Temporal operators have the following semantics where
\( \pi = (s_i, s_{i+1}, \ldots) \) is a generic path outgoing from state \( s_i \) in \( \mathcal{KM} \).

\[
\begin{align*}
\mathcal{KM}, s_i & \models \square \Diamond \phi \quad \text{iff} \quad \forall \pi = (s_i, s_{i+1}, \ldots) \quad \mathcal{KM}, s_{i+1} \models \phi \\
\mathcal{KM}, s_i & \models \Diamond \square \phi \quad \text{iff} \quad \exists \pi = (s_i, s_{i+1}, \ldots) \quad \mathcal{KM}, s_{i+1} \models \phi \\
\mathcal{KM}, s_i & \models \square \square \phi \quad \text{iff} \quad \forall \pi = (s_i, s_{i+1}, \ldots) \quad \exists j \geq i. \mathcal{KM}, s_j \models \phi \\
\mathcal{KM}, s_i & \models \Diamond \square \phi \quad \text{iff} \quad \exists \pi = (s_i, s_{i+1}, \ldots) \quad \exists j \geq i. \mathcal{KM}, s_j \models \phi \\
\mathcal{KM}, s_i & \models \square (\phi \cup \psi) \quad \text{iff} \quad \forall \pi = (s_i, s_{i+1}, \ldots) \quad \exists j \geq i. \mathcal{KM}, s_j \models \psi \quad \text{and} \quad \forall i \leq k < j : M, s_k \models \phi \\
\mathcal{KM}, s_i & \models \Diamond (\phi \cup \psi) \quad \text{iff} \quad \exists \pi = (s_i, s_{i+1}, \ldots) \quad \exists j \geq i. \mathcal{KM}, s_j \models \psi \quad \text{and} \quad \forall i \leq k < j : \mathcal{KM}, s_k \models \phi
\end{align*}
\]
CTL Semantics: Intuitions

CTL is given by the standard boolean logic enhanced with temporal operators.

- **“Necessarily Next”**. $\square \bigcirc \varphi$ is true in $s_t$ iff $\varphi$ is true in every successor state $s_{t+1}$

- **“Possibly Next”**. $\Diamond \bigcirc \varphi$ is true in $s_t$ iff $\varphi$ is true in one successor state $s_{t+1}$

- **“Necessarily in the future”** (or “Inevitably”). $\square \Diamond \varphi$ is true in $s_t$ iff $\varphi$ is inevitably true in some $s_{t'}$ with $t' \geq t$

- **“Possibly in the future”** (or “Possibly”). $\Diamond \Diamond \varphi$ is true in $s_t$ iff $\varphi$ may be true in some $s_{t'}$ with $t' \geq t$
“Globally” (or “always”). $\Box \varphi$ is true in $s_t$ iff $\varphi$ is true in all $s_{t'}$ with $t' \geq t$

“Possibly henceforth”. $\Diamond \Box \varphi$ is true in $s_t$ iff $\varphi$ is possibly true henceforth

“Necessarily Until”. $\Box (\varphi U \psi)$ is true in $s_t$ iff necessarily $\varphi$ holds until $\psi$ holds.

“Possibly Until”. $\Diamond (\varphi U \psi)$ is true in $s_t$ iff possibly $\varphi$ holds until $\psi$ holds.
Alternative notations are used for temporal operators.

- ♦ $\leadsto E$ there Exists a path
- $\exists$ $\leadsto A$ in All paths
- ♦ $\leadsto F$ sometime in the Future
- □ $\leadsto G$ Globally in the future
- ○ $\leadsto X$ neXtime
finally $P$

globally $P$

next $P$

$P$ until $q$

$ AF_P $

$ AG_P $

$ AX_P $

$ A[p \cup q] $

$ EF_P $

$ EG_P $

$ EX_P $

$ E[p \cup q] $
A Complete Set of CTL Operators

All CTL operators can be expressed via: $\lozenge \odot$, $\lozenge \square$, $\lozenge U$

- $P \odot \equiv \neg P \odot \neg \varphi$
- $P \lozenge \varphi \equiv \neg P \square \neg \varphi$
- $\lozenge \lozenge \varphi \equiv \lozenge (\top U \varphi)$
- $P \square \varphi \equiv \neg \lozenge \lozenge \neg \varphi \equiv \neg \lozenge (\top U \neg \varphi)$
- $P (\varphi U \psi) \equiv \neg P \square \neg \psi \land \neg P (\neg \psi U (\neg \varphi \land \neg \psi))$
Summary

- Computation Tree Logic: Intuitions.
- CTL: Syntax and Semantics.
- CTL in Computer Science.
- CTL and Model Checking: Examples.
- CTL Vs. LTL.
- CTL*.
Safety Properties

Safety:
“something bad will not happen”

Typical examples:

\[ P \  \Box \neg (reactor\_temp > 1000) \]
\[ P \  \Box \neg (one\_way \land P \  \Box other\_way) \]
\[ P \  \Box \neg ((x = 0) \land P \  \Box P \  \Box P \  \Box (y = z/x)) \]

and so on.....

Usually: \[ P \  \Box \neg .... \]
Liveness properties

Liveness:

“something good will happen”

Typical examples:

\( P \diamond rich \)

\( P \diamond (x > 5) \)

\( P \square (start \Rightarrow P \diamond terminate) \)

and so on.....

Usually: \( P \diamond \ldots \)
Fairness Properties

Often only really useful when scheduling processes, responding to messages, etc.

Fairness:

“something is successful/allocated infinitely often”

Typical example:

\[ P \square ( P \diamond \text{enabled}) \]

Usually: \[ P \square P \diamond \ldots \]
Summary

- Computation Tree Logic: Intuitions.
- CTL: Syntax and Semantics.
- CTL in Computer Science.
- CTL and Model Checking: Examples.
- CTL Vs. LTL.
- CTL*.
The CTL Model Checking Problem is formulated as:

$$\mathcal{K}\mathcal{M} \models \phi$$

Check if $\mathcal{K}\mathcal{M}, s_0 \models \phi$, for every initial state, $s_0$, of the Kripke structure $\mathcal{K}\mathcal{M}$.
Example 1: Mutual Exclusion (Safety)

N = noncritical,  T = trying,  C = critical

\[ \mathcal{K_M} \models \Box \neg (C_1 \land C_2) \]
Example 1: Mutual Exclusion (Safety)

\[ K_M \models P \Box \neg(C_1 \land C_2) ? \]

**YES:** There is no reachable state in which \((C_1 \land C_2)\) holds!
(Same as the \(\Box \neg(C_1 \land C_2)\) in LTL.)
Example 2: Liveness

N = noncritical, T = trying, C = critical

User 1

User 2

\[ \mathcal{KM} \models [\Box (T_1 \Rightarrow [\Box (C_1)]) ] \]
Example 2: Liveness

\[ KM \models \square (T_1 \Rightarrow \square \Diamond C_1) ? \]

**YES**: every path starting from each state where \( T_1 \) holds passes through a state where \( C_1 \) holds.

(Same as \( \square (T_1 \Rightarrow \Diamond C_1) \) in LTL)
Example 3: Fairness

N = noncritical, T = trying, C = critical

\[ \square P \quad \square \quad \square P \quad \Diamond C_1 \ ? \]
Example 3: Fairness

\[ \mathcal{K} \mathcal{M} \models \square \square \square \diamondsuit C_1 \]

**NO:** e.g., in the initial state, there is the blue cyclic path in which \( C_1 \) never holds! (Same as \( \square \diamondsuit C_1 \) in LTL)
Example 4: Non-Blocking

\[ \mathcal{K}_M \models \Box (N_1 \Rightarrow \Diamond \Diamond T_1) \]
Example 4: Non-Blocking

\[ \mathcal{K}_M \models P \Box (N_1 \Rightarrow \Diamond \Diamond T_1) ? \]

**YES:** from each state where \( N_1 \) holds there is a path leading to a state where \( T_1 \) holds. (No corresponding LTL formulas)
Summary

- Computation Tree Logic: Intuitions.
- CTL: Syntax and Semantics.
- CTL in Computer Science.
- CTL and Model Checking: Examples.
- CTL Vs. LTL.
- CTL*.
LTL Vs. CTL: Expressiveness

- Many CTL formulas cannot be expressed in LTL (e.g., those containing paths quantified existentially)
  E.g., $\mathbf{P} \mathbf{□}(N_1 \Rightarrow \mathbf{P} \mathbf{♢} T_1)$

- Many LTL formulas cannot be expressed in CTL
  E.g., $\mathbf{□} \mathbf{♢} T_1 \Rightarrow \mathbf{□} \mathbf{♢} C_1$ (Strong Fairness in LTL)
  i.e., formulas that select a range of paths with a property
  ($\mathbf{♢} p \Rightarrow \mathbf{♢} q$ Vs. $\mathbf{P} \mathbf{□}(p \Rightarrow \mathbf{P} \mathbf{♢} q)$)

- Some formulas can be expressed both in LTL and in CTL (typically LTL formulas with operators of nesting depth 1)
  E.g., $\mathbf{□}\neg(C_1 \land C_2)$, $\mathbf{♢} C_1$, $\mathbf{□}(T_1 \Rightarrow \mathbf{♢} C_1)$, $\mathbf{□} \mathbf{♢} C_1$
CTL and LTL have incomparable expressive power.

The choice between LTL and CTL depends on the application and the personal preferences.
Summary

- Computation Tree Logic: Intuitions.
- CTL: Syntax and Semantics.
- CTL in Computer Science.
- CTL and Model Checking: Examples.
- CTL Vs. LTL.
- CTL*.
CTL* is a logic that combines the expressive power of LTL and CTL.

Temporal operators can be applied without any constraints.

- $\square (\bigcirc \varphi \lor \bigcirc \bigcirc \varphi)$. Along all paths, $\varphi$ is true in the next state or the next two steps.

- $\diamond (\square \diamond \varphi)$. There is a path along which $\varphi$ is infinitely often true.
Countable set $\Sigma$ of atomic propositions: $p, q, \ldots$ we distinguish between *States Formulas* (evaluated on states):

$$\varphi, \psi \rightarrow p \mid \top \mid \bot \mid \neg \varphi \mid \varphi \land \psi \mid \varphi \lor \psi \mid \Box \alpha \mid \Diamond \alpha$$

and *Path Formulas* (evaluated on paths):

$$\alpha, \beta \rightarrow \varphi \mid \neg \alpha \mid \alpha \land \beta \mid \alpha \lor \beta \mid \Diamond \alpha \mid \Diamond \alpha \mid \Diamond \alpha \mid (\alpha \lor \beta)$$

The set of CTL* formulas $\text{FORM}$ is the set of state formulas.
We start by defining when an atomic proposition is true at a state “$s_0$”:

\[ \mathcal{K}M, s_0 \models p \iff p \in L(s_0) \quad (\text{for } p \in \Sigma) \]

The semantics for *State Formulas* is the following where $\pi = (s_0, s_1, \ldots)$ is a generic path outgoing from state $s_0$:

\[ \mathcal{K}M, s_0 \models \neg \varphi \iff \mathcal{K}M, s_0 \not\models \varphi \]
\[ \mathcal{K}M, s_0 \models \varphi \land \psi \iff \mathcal{K}M, s_0 \models \varphi \text{ and } \mathcal{K}M, s_0 \models \psi \]
\[ \mathcal{K}M, s_0 \models \varphi \lor \psi \iff \mathcal{K}M, s_0 \models \varphi \text{ or } \mathcal{K}M, s_0 \models \psi \]
\[ \mathcal{K}M, s_0 \models \Diamond \alpha \iff \exists \pi = (s_0, s_1, \ldots) \text{ such that } \mathcal{K}M, \pi \models \alpha \]
\[ \mathcal{K}M, s_0 \models \Box \alpha \iff \forall \pi = (s_0, s_1, \ldots) \text{ then } \mathcal{K}M, \pi \models \alpha \]
CTL* Semantics: Path Formulas

The semantics for Path Formulas is the following where \( \pi = (s_0, s_1, \ldots) \) is a generic path outgoing from state \( s_0 \) and \( \pi^i \) denotes the suffix path \( (s_i, s_{i+1}, \ldots) \):

- \( \mathcal{K} \mathcal{M}, \pi \models \varphi \) iff \( \mathcal{K} \mathcal{M}, s_0 \models \varphi \)
- \( \mathcal{K} \mathcal{M}, \pi \models \neg \alpha \) iff \( \mathcal{K} \mathcal{M}, \pi \not\models \alpha \)
- \( \mathcal{K} \mathcal{M}, \pi \models \alpha \land \beta \) iff \( \mathcal{K} \mathcal{M}, \pi \models \alpha \) and \( \mathcal{K} \mathcal{M}, \pi \models \beta \)
- \( \mathcal{K} \mathcal{M}, \pi \models \alpha \lor \beta \) iff \( \mathcal{K} \mathcal{M}, \pi \models \alpha \) or \( \mathcal{K} \mathcal{M}, \pi \models \beta \)
- \( \mathcal{K} \mathcal{M}, \pi \models \Diamond \alpha \) iff \( \exists i \geq 0 \) such that \( \mathcal{K} \mathcal{M}, \pi^i \models \alpha \)
- \( \mathcal{K} \mathcal{M}, \pi \models \Box \alpha \) iff \( \forall i \geq 0 \) then \( \mathcal{K} \mathcal{M}, \pi^i \models \alpha \)
- \( \mathcal{K} \mathcal{M}, \pi \models \Diamond \alpha \) iff \( \mathcal{K} \mathcal{M}, \pi^1 \models \alpha \)
- \( \mathcal{K} \mathcal{M}, \pi \models \alpha \lor \beta \) iff \( \exists i \geq 0 \) such that \( \mathcal{K} \mathcal{M}, \pi^i \models \beta \) and \( \forall j.(0 \leq j \leq i) \) then \( \mathcal{K} \mathcal{M}, \pi^j \models \alpha \)
CTL* subsumes both CTL and LTL

- \( \varphi \) in CTL \( \Leftrightarrow \) \( \varphi \) in CTL* (e.g., \( \square P \square(N_1 \Rightarrow \Diamond \Diamond T_1) \))
- \( \varphi \) in LTL \( \Leftrightarrow \) \( \square \varphi \) in CTL* (e.g., \( \square (\square \Diamond T_1 \Rightarrow \square \Diamond C_1) \))
- LTL \( \cup \) CTL \( \subset \) CTL* (e.g., \( \Diamond (\square \Diamond p \Rightarrow \square \Diamond q) \))
The following Table shows the Computational Complexity of checking *Satisbiability*

<table>
<thead>
<tr>
<th>Logic</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL</td>
<td>PSpace-Complete</td>
</tr>
<tr>
<td>CTL</td>
<td>ExpTime-Complete</td>
</tr>
<tr>
<td>CTL*</td>
<td>2ExpTime-Complete</td>
</tr>
</tbody>
</table>
The following Table shows the Computational Complexity of Model Checking (M.C.)

- Since M.C. has 2 inputs – the model, $M$, and the formula, $\phi$ – we give two complexity measures.

<table>
<thead>
<tr>
<th>Logic</th>
<th>Complexity w.r.t. $\phi$</th>
<th>Complexity w.r.t. $M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL</td>
<td>PSpace-Complete</td>
<td>P (linear)</td>
</tr>
<tr>
<td>CTL</td>
<td>P-Complete</td>
<td>P (linear)</td>
</tr>
<tr>
<td>CTL*</td>
<td>PSpace-Complete</td>
<td>P (linear)</td>
</tr>
</tbody>
</table>
Summary of Lecture IV

- Computation Tree Logic: Intuitions.
- CTL: Syntax and Semantics.
- CTL in Computer Science.
- CTL and Model Checking: Examples.
- CTL Vs. LTL.
- CTL*.