FORMAL METHODS

LECTURE IV: COMPUTATION TREE LOGIC (CTL)

Alessandro Artale

Faculty of Computer Science – Free University of Bolzano
Room 2.03

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

Some material (text, figures) displayed in these slides is courtesy of:
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*.
Computation Tree logic Vs. LTL

- LTL implicitly quantifies *universally* over paths.

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

- 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: $\Box P$
  - Exists a Path: $\diamond P$.

- Every temporal operator ($\Box$, $\diamond$, $\bigcirc$, $U$) preceded by a path quantifier ($\Box$ or $\diamond$).

- **Universal modalities:** $\Box \diamond$, $\Box \Box$, $\Box \bigcirc$, $\Box U$
  The temporal formula is true in **all** the paths starting in the current state.

- **Existential modalities:** $\diamond \diamond$, $\diamond \Box$, $\diamond \bigcirc$, $\diamond U$
  The temporal formula is true in **some** path starting in the current state.
• 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:

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

$$\blacksquare p \mid \square \phi \mid \Diamond \phi \mid \blacksquare (\phi U \psi)$$

$$\Diamond \blacksquare \phi \mid \Diamond \square \phi \mid \Diamond \Diamond \phi \mid \Diamond (\phi U \psi)$$
Alternative notations are used for temporal operators.

◊ \( \leadsto \) \( E \) there Exists a path
\[ \Box \leadsto A \] in All paths
\[ \Diamond \leadsto F \] sometime in the Future
\[ \square \leadsto G \] Globally in the future
\[ \circ \leadsto X \] neXtime
We interpret our CTL temporal formulas over Kripke Models linearized as trees.

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

Existential modalities (\(\Diamond \Diamond\), \(\Diamond \Box\), \(\Diamond \bigcirc\), \(\Diamond \bigtriangledown\)): 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:

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

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

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

$$\mathcal{KM}, s_i \models p \iff p \in L(s_i) \quad \text{(for } p \in \Sigma)$$

The semantics for the classical operators is as expected:

$$\mathcal{KM}, s_i \models \neg\varphi \iff \mathcal{KM}, s_i \not\models \varphi$$

$$\mathcal{KM}, s_i \models \varphi \land \psi \iff \mathcal{KM}, s_i \models \varphi \text{ and } \mathcal{KM}, s_i \models \psi$$

$$\mathcal{KM}, s_i \models \varphi \lor \psi \iff \mathcal{KM}, s_i \models \varphi \text{ or } \mathcal{KM}, s_i \models \psi$$

$$\mathcal{KM}, s_i \models \varphi \Rightarrow \psi \iff \text{if } \mathcal{KM}, s_i \models \varphi \text{ then } \mathcal{KM}, s_i \models \psi$$

$$\mathcal{KM}, s_i \models \top$$

$$\mathcal{KM}, s_i \not\models \bot$$
CTL Semantics: The Temporal Aspect

Temporal operators have the following semantics where
\( \pi = (s_i, s_{i+1}, \ldots) \) is a generic path outgoing from state \( s_i \) in \( \mathcal{KM} \).

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

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

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

- “Possibly Next”. $\Diamond \circ \varphi$ 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 $t' \geq t$
CTL Semantics: Intuitions (Cont.)

▷ “Globally” (or “always”). $\Box\Box\phi$ is true in $s_t$ iff $\phi$ is true in all $s_{t'}$ with $t' \geq t$

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

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

▷ “Possibly Until”. $\Diamond (\phi U \psi)$ is true in $s_t$ iff possibly $\phi$ holds until $\psi$ holds.
finally $P$

globally $P$

next $P$

$P$ until $q$

$AF_P$

$AG_P$

$AX_P$

$A[ P \cup q ]$

$EF_P$

$EG_P$

$EX_P$

$E[ P \cup q ]$
All CTL operators can be expressed via: $\diamond \circ, \diamond \square, \diamond \cup$

- $\square \circ \varphi \equiv \neg \diamond \circ \neg \varphi$
- $\square \diamond \varphi \equiv \neg \diamond \square \neg \varphi$
- $\diamond \diamond \varphi \equiv \diamond (\top \cup \varphi)$
- $\square \uparrow \varphi \equiv \neg \diamond \diamond \neg \varphi \equiv \neg \diamond (\top \cup \neg \varphi)$
- $\square (\varphi \cup \psi) \equiv \neg \diamond \square \neg \psi \land \neg \diamond (\neg \psi \cup (\neg \varphi \land \neg \psi))$
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:

\[ P \Box \neg (\text{reactor\_temp} > 1000) \]

\[ P \Box \neg (\text{one\_way} \land P \bigcirc \text{other\_way}) \]

\[ P \Box \neg ((x = 0) \land P \bigcirc P \Box \bigcirc P \bigcirc (y = z/x)) \]

and so on.....

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

Liveness:

“something good will happen”

Typical examples:

$\square P \diamond rich$

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

$\square (\text{start} \Rightarrow \square P \diamond \text{terminate})$

and so on.....

Usually: $\square P \diamond \ldots$
Fairness Properties

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

Fairness:

“something is successful/allocated infinitely often”

Typical example:

\[\mathbb{P} \Box (\mathbb{P} \diamond enabled)\]

Usually: \[\mathbb{P} \Box \mathbb{P} \diamond \ldots\]
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:

$$\mathcal{KM} \models \phi$$

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

\[ KM \models \Box \neg (C_1 \land C_2) \]
Example 1: Mutual Exclusion (Safety)

\[ \mathcal{K}\mathcal{M} \models P \Box \neg (C_1 \land C_2) \? \]

**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

\( KM \models \square(p \rightarrow \square(T_1 \Rightarrow \square \Diamond C_1)) \)
Example 2: Liveness

N = noncritical, T = trying, C = critical

User 1

User 2

\[ \mathcal{K} = \Box (T_1 \Rightarrow \Diamond C_1) \]

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

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

\[ KM \models P \square P \diamondsuit C_1 \]
Example 3: Fairness

N = noncritical, T = trying, C = critical

N1, N2

User 1

User 2

T1, N2

turn=0

T1, T2

N1, T2

turn=1

C1, T2

T1, C2

turn=1

C1, N2

T1, C2

KM

|$\square\Box P \Box P \diamondsuit C_1$?

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

N = noncritical, T = trying, C = critical

N1, N2

T1, N2

C1, N2

T1, T2

C1, T2

T1, C2

N1, T2

T1, T2

N1, C2

N1, T2

T1, C2

\[ KM \models P \Box (N_1 \Rightarrow P \Diamond \Diamond T_1) ? \]
Example 4: Non-Blocking

\[ N = \text{noncritical}, \quad T = \text{trying}, \quad C = \text{critical} \]

\[ N_1, N_2, \quad \text{turn}=0 \]

\[ T_1, N_2, \quad \text{turn}=1 \]

\[ C_1, T_2, \quad \text{turn}=1 \]

\[ T_1, T_2, \quad \text{turn}=1 \]

\[ T_1, N_2, \quad \text{turn}=1 \]

\[ C_1, N_2, \quad \text{turn}=1 \]

\[ T_1, T_2, \quad \text{turn}=2 \]

\[ N_1, T_2, \quad \text{turn}=2 \]

\[ T_1, C_2, \quad \text{turn}=2 \]

\[ N_1, C_2, \quad \text{turn}=2 \]

\[ \mathcal{K} M \models [P] (N_1 \Rightarrow [P] [P] T_1) \]

**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., $\Box \Box (N_1 \Rightarrow \Diamond \Diamond T_1)$

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

▷ Some formluas can be expressed both in LTL and in CTL (typically LTL formulas with operators of nesting depth 1)
E.g., $\Box \neg (C_1 \land C_2), \Diamond C_1, \Box (T_1 \Rightarrow \Diamond C_1), \Box \Diamond C_1$
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.

- $\Box (\bigcirc \varphi \lor \bigcirc \bigcirc \varphi)$. 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):

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

and *Path Formulas* (evaluated on paths):

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

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

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

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

$$\mathcal{K}M, s_0 \models \neg \varphi \quad \text{iff} \quad \mathcal{K}M, s_0 \not\models \varphi$$

$$\mathcal{K}M, s_0 \models \varphi \wedge \psi \quad \text{iff} \quad \mathcal{K}M, s_0 \models \varphi \text{ and } \mathcal{K}M, s_0 \models \psi$$

$$\mathcal{K}M, s_0 \models \varphi \lor \psi \quad \text{iff} \quad \mathcal{K}M, s_0 \models \varphi \text{ or } \mathcal{K}M, s_0 \models \psi$$

$$\mathcal{K}M, s_0 \models \Diamond \alpha \quad \text{iff} \quad \exists \pi = (s_0, s_1, \ldots) \text{ such that } \mathcal{K}M, \pi \models \alpha$$

$$\mathcal{K}M, s_0 \models \Box \alpha \quad \text{iff} \quad \forall \pi = (s_0, s_1, \ldots) \text{ then } \mathcal{K}M, \pi \models \alpha$$
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 \( (s_i, s_{i+1}, \ldots) \):

\[
\begin{align*}
\mathcal{KM}, \pi &\models \varphi \quad \text{iff} \quad \mathcal{KM}, s_0 \models \varphi \\
\mathcal{KM}, \pi &\models \neg \alpha \quad \text{iff} \quad \mathcal{KM}, \pi \not\models \alpha \\
\mathcal{KM}, \pi &\models \alpha \land \beta \quad \text{iff} \quad \mathcal{KM}, \pi \models \alpha \text{ and } \mathcal{KM}, \pi \models \beta \\
\mathcal{KM}, \pi &\models \alpha \lor \beta \quad \text{iff} \quad \mathcal{KM}, \pi \models \alpha \text{ or } \mathcal{KM}, \pi \models \beta \\
\mathcal{KM}, \pi &\models \Diamond \alpha \quad \text{iff} \quad \exists i \geq 0 \text{ such that } \mathcal{KM}, \pi^i \models \alpha \\
\mathcal{KM}, \pi &\models \Box \alpha \quad \text{iff} \quad \forall i \geq 0 \text{ then } \mathcal{KM}, \pi^i \models \alpha \\
\mathcal{KM}, \pi &\models \lozenge \alpha \quad \text{iff} \quad \mathcal{KM}, \pi^1 \models \alpha \\
\mathcal{KM}, \pi &\models \alpha \cup \beta \quad \text{iff} \quad \exists i \geq 0 \text{ such that } \mathcal{KM}, \pi^i \models \beta \text{ and } \\
& \quad \forall j.(0 \leq j \leq i) \text{ then } \mathcal{KM}, \pi^j \models \alpha 
\end{align*}
\]
CTL* subsumes both CTL and LTL

- \( \varphi \) in CTL \( \iff \varphi \) in CTL* (e.g., \( □ P (N_1 \Rightarrow \lozenge \lozenge T_1) \))
- \( \varphi \) in LTL \( \iff □ \varphi \) in CTL* (e.g., \( □ (□ \lozenge T_1 \Rightarrow □ \lozenge C_1) \))
- LTL \( \cup \) CTL \( ⊂ \) CTL* (e.g., \( P (□ \lozenge p \Rightarrow □ \lozenge q) \))
The following Table shows the Computational Complexity of checking *Satisfaction*.

<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, $\varphi$ – we give two complexity measures.

<table>
<thead>
<tr>
<th>Logic</th>
<th>Complexity w.r.t. $\varphi$</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*.