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:

M. Benerecetti, A. Cimatti, M. Fisher, F. Giunchiglia, M. Pistore, M. Roveri, R.Sebastiani.

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.

  $\langle \mathcal{KM}, s\rangle \models \phi$ iff for every path π 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: $\boxed{\mathrm{P}}$

  Exists a Path: $\langle \mathrm{P} \rangle$.

- Every temporal operator ($\Box, \Diamond, \bigcirc, \mathcal{U}$) preceded by a path quantifier ($\boxed{\mathrm{P}}$ or $\langle \mathrm{P} \rangle$).
- **Universal modalities:** $\boxed{\mathrm{P}}\Diamond, \boxed{\mathrm{P}}\Box, \boxed{\mathrm{P}}\bigcirc, \boxed{\mathrm{P}}\,\mathcal{U}$
  The temporal formula is true in **all** the paths starting in the current state.
- **Existential modalities:** $\langle \mathrm{P} \rangle\Diamond, \langle \mathrm{P} \rangle\Box, \langle \mathrm{P} \rangle\bigcirc, \langle \mathrm{P} \rangle\,\mathcal{U}$
  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*.

CTL: Syntax

Countable set Σ of *atomic propositions*: $p, q, \ldots$ the set FORM of formulas is:

$$\varphi, \psi \quad \to \quad p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid$$

$$\boxed{\mathrm{P}} \bigcirc \varphi \mid \boxed{\mathrm{P}} \Box \varphi \mid \boxed{\mathrm{P}} \Diamond \varphi \mid \boxed{\mathrm{P}} (\varphi\, \mathcal{U}\, \psi)$$

$$\Diamond_{\mathrm{P}} \bigcirc \varphi \mid \Diamond_{\mathrm{P}} \Box \varphi \mid \Diamond_{\mathrm{P}} \Diamond \varphi \mid \Diamond_{\mathrm{P}} (\varphi\, \mathcal{U}\, \psi)$$

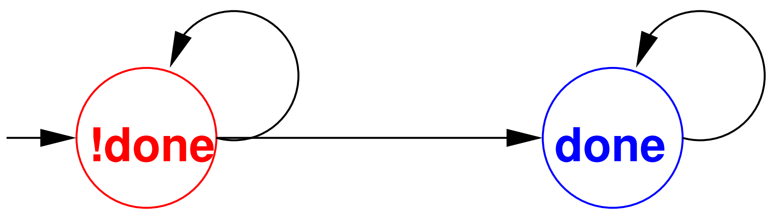
CTL Alternative Notation

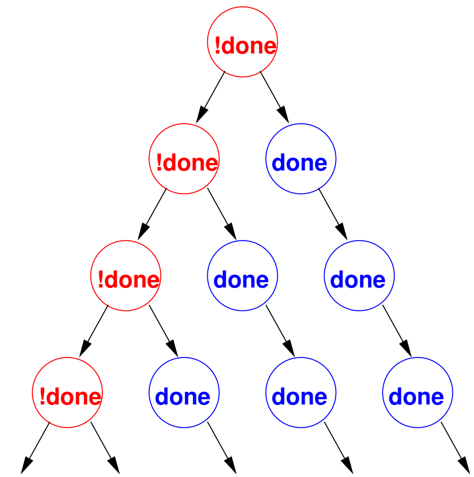
Alternative notations are used for temporal operators.

$\Diamond P$	$\rightsquigarrow$	E	there **E**xists a path
$\boxed{P}$	$\rightsquigarrow$	A	in **A**ll paths
$\Diamond$	$\rightsquigarrow$	F	sometime in the **F**uture
$\Box$	$\rightsquigarrow$	G	**G**lobally in the future
$\bigcirc$	$\rightsquigarrow$	X	ne**X**time

CTL: Semantics

- We interpret our CTL temporal formulas over Kripke Models linearized as trees.





- Universal modalities ($\forall\Diamond$, $\forall\Box$, $\forall\bigcirc$, $\forall\,\mathcal{U}$): the temporal formula is true in **all** the paths starting in the current state.
- Existential modalities ($\exists\Diamond$, $\exists\Box$, $\exists\bigcirc$, $\exists\,\mathcal{U}$): the temporal formula is true in **some** path starting in the current state.

CTL: Semantics (Cont.)

Let Σ 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:

$$\models : (\mathcal{KM} \times S \times \textsc{Form}) \rightarrow \{\mathbf{true}, \mathbf{false}\}$$

CTL Semantics: The Propositional Aspect

We start by defining when an atomic proposition is true at a state/time "s_i"

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

The semantics for the classical operators is as expected:

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

$$\mathcal{KM}, s_i \models \varphi \wedge \psi \quad \textbf{iff} \quad \mathcal{KM}, s_i \models \varphi \text{ and } \mathcal{KM}, s_i \models \psi$$

$$\mathcal{KM}, s_i \models \varphi \vee \psi \quad \textbf{iff} \quad \mathcal{KM}, s_i \models \varphi \text{ or } \mathcal{KM}, s_i \models \psi$$

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

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

$$\mathcal{KM}, s_i \not\models \bot$$

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 $s_i \mathrm{in} \mathcal{KM}$.

$\mathcal{KM}, s_i \models \boxed{\mathrm{P}} \bigcirc \varphi$	iff	$\forall \pi = (s_i, s_{i+1}, \ldots)$	$\mathcal{KM}, s_{i+1} \models \varphi$
$\mathcal{KM}, s_i \models \Diamond\!\!\!\!\mathrm{P}\ \bigcirc \varphi$	iff	$\exists \pi = (s_i, s_{i+1}, \ldots)$	$\mathcal{KM}, s_{i+1} \models \varphi$
$\mathcal{KM}, s_i \models \boxed{\mathrm{P}} \Box \varphi$	iff	$\forall \pi = (s_i, s_{i+1}, \ldots)$	$\forall j \geq i. \mathcal{KM}, s_j \models \varphi$
$\mathcal{KM}, s_i \models \Diamond\!\!\!\!\mathrm{P}\ \Box \varphi$	iff	$\exists \pi = (s_i, s_{i+1}, \ldots)$	$\forall j \geq i. \mathcal{KM}, s_j \models \varphi$
$\mathcal{KM}, s_i \models \boxed{\mathrm{P}} \Diamond \varphi$	iff	$\forall \pi = (s_i, s_{i+1}, \ldots)$	$\exists j \geq i. \mathcal{KM}, s_j \models \varphi$
$\mathcal{KM}, s_i \models \Diamond\!\!\!\!\mathrm{P}\ \Diamond \varphi$	iff	$\exists \pi = (s_i, s_{i+1}, \ldots)$	$\exists j \geq i. \mathcal{KM}, s_j \models \varphi$
$\mathcal{KM}, s_i \models \boxed{\mathrm{P}} (\varphi\, \mathcal{U}\, \psi)$	iff	$\forall \pi = (s_i, s_{i+1}, \ldots)$	$\exists j \geq i. \mathcal{KM}, s_j \models \psi$ and
			$\forall i \leq k < j\ : M, s_k \models \varphi$
$\mathcal{KM}, s_i \models \Diamond\!\!\!\!\mathrm{P}\ (\varphi\, \mathcal{U}\, \psi)$	iff	$\exists \pi = (s_i, s_{i+1}, \ldots)$	$\exists j \geq i. \mathcal{KM}, s_j \models \psi$ and
			$\forall i \leq k < j\ : \mathcal{KM}, s_k \models \varphi$

CTL Semantics: Intuitions

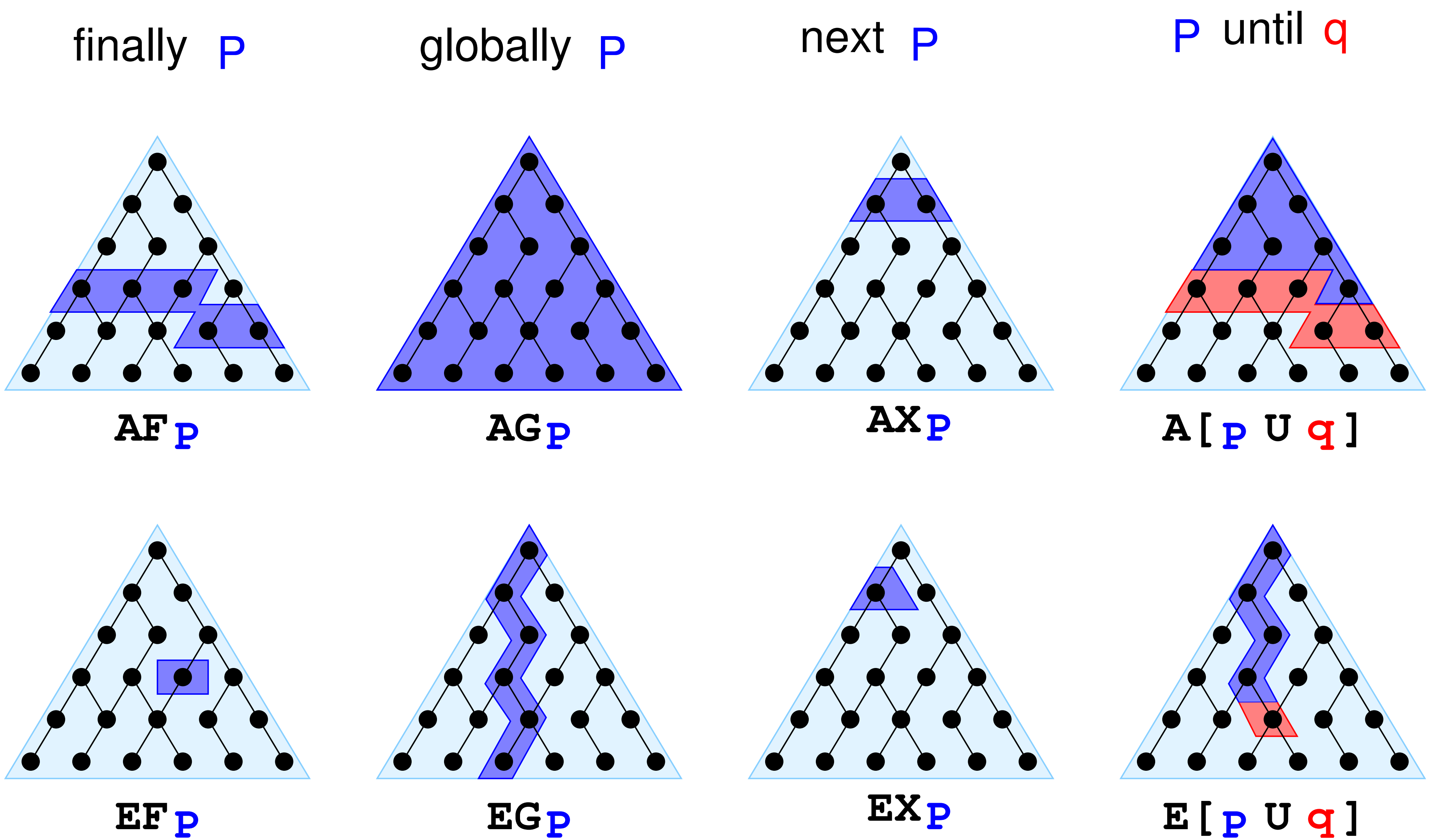
CTL is given by the standard boolean logic enhanced with temporal operators.

- "Necessarily Next". $\boxed{\text{P}}\bigcirc\varphi$ is true in s_t iff φ is true in every successor state s_{t+1}
- "Possibly Next". $\Diamond_{\text{P}}\bigcirc\varphi$ is true in s_t iff φ is true in one successor state s_{t+1}
- "Necessarily in the future" (or "Inevitably"). $\boxed{\text{P}}\Diamond\varphi$ is true in s_t iff φ is inevitably true in **some** $s_{t'}$ with $t' \geq t$
- "Possibly in the future" (or "Possibly"). $\Diamond_{\text{P}}\Diamond\varphi$ is true in s_t iff φ may be true in **some** $s_{t'}$ with $t' \geq t$

CTL Semantics: Intuitions (Cont.)

- "Globally" (or "always"). $\boxed{P}\,\Box\varphi$ is true in s_t iff φ is true in **all** $s_{t'}$ with $t' \geq t$
- "Possibly henceforth". $\Diamond\!\!\!\!\;P\,\Box\varphi$ is true in s_t iff φ is possibly true henceforth
- "Necessarily Until". $\boxed{P}\,(\varphi\,\mathcal{U}\,\psi)$ is true in s_t iff necessarily φ holds until ψ holds.
- "Possibly Until". $\Diamond\!\!\!\!\;P\,(\varphi\,\mathcal{U}\,\psi)$ is true in s_t iff possibly φ holds until ψ holds.

CTL Semantics: Intuitions (Cont.)



A Complete Set of CTL Operators

All CTL operators can be expressed via: $\Diamond_P \bigcirc$, $\Diamond_P \Box$, $\Diamond_P \mathcal{U}$

- $\boxed{P} \bigcirc \varphi \equiv \neg \Diamond_P \bigcirc \neg \varphi$
- $\boxed{P} \Diamond \varphi \equiv \neg \Diamond_P \Box \neg \varphi$
- $\Diamond_P \Diamond \varphi \equiv \Diamond_P (\top \, \mathcal{U} \, \varphi)$
- $\boxed{P} \Box \varphi \equiv \neg \Diamond_P \Diamond \neg \varphi \equiv \neg \Diamond_P (\top \, \mathcal{U} \, \neg \varphi)$
- $\boxed{P} (\varphi \, \mathcal{U} \, \psi) \equiv \neg \Diamond_P \Box \neg \psi \wedge \neg \Diamond_P (\neg \psi \, \mathcal{U} \, (\neg \varphi \wedge \neg \psi))$

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:

- $\Box \neg (reactor\_temp > 1000)$
- $\Box \neg (one\_way \wedge \bigcirc other\_way)$
- $\Box \neg ((x = 0) \wedge \bigcirc \bigcirc \bigcirc (y = z/x))$

and so on.....

Usually: $\Box \neg \ldots.$

Liveness Properties

Liveness:

"something good will happen"

Typical examples:

$\boxed{\text{P}}\ \Diamond rich$

$\boxed{\text{P}}\ \Diamond(x > 5)$

$\boxed{\text{P}}\ \Box(start \Rightarrow \boxed{\text{P}}\ \Diamond terminate)$

and so on.....

Usually: $\boxed{\text{P}}\ \Diamond \ldots$

Fairness Properties

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

Fairness:

> "something is successful/allocated infinitely often"

Typical example:

$$\boxed{\text{P}}\ \Box(\boxed{\text{P}}\,\Diamond \mathit{enabled})$$

Usually: $\boxed{\text{P}}\ \Box\ \boxed{\text{P}}\,\Diamond \ldots$

Summary

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

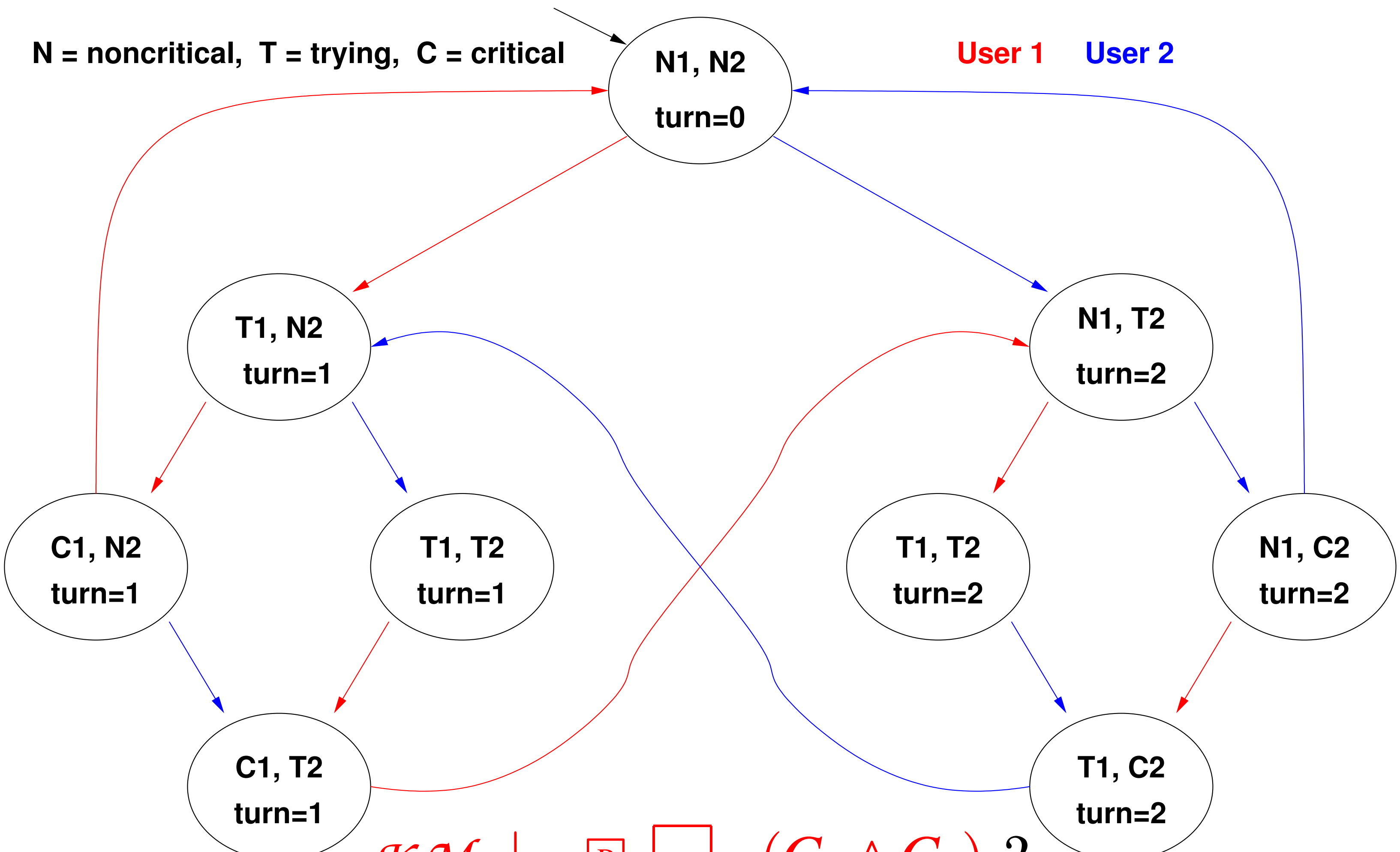
The CTL Model Checking Problem

The CTL Model Checking Problem is formulated as:

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

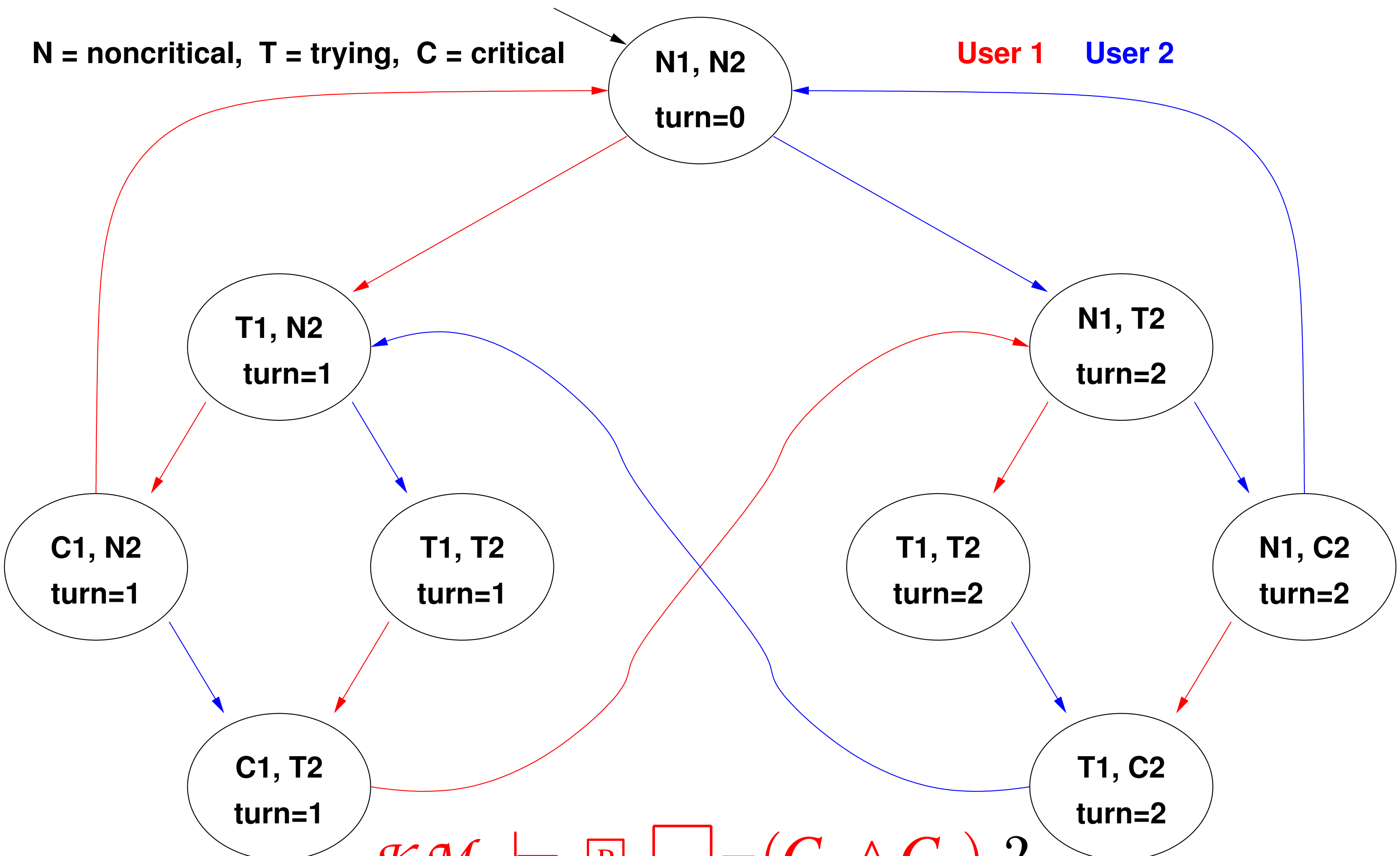
Check if $\mathcal{KM}, s_0 \models \phi$, for **every initial state**, s_0, of the Kripke structure $\mathcal{KM}$.

Example 1: Mutual Exclusion (Safety)



$$\mathcal{KM} \models \boxed{\mathrm{P}}\ \Box \neg (C_1 \wedge C_2)\ ?$$

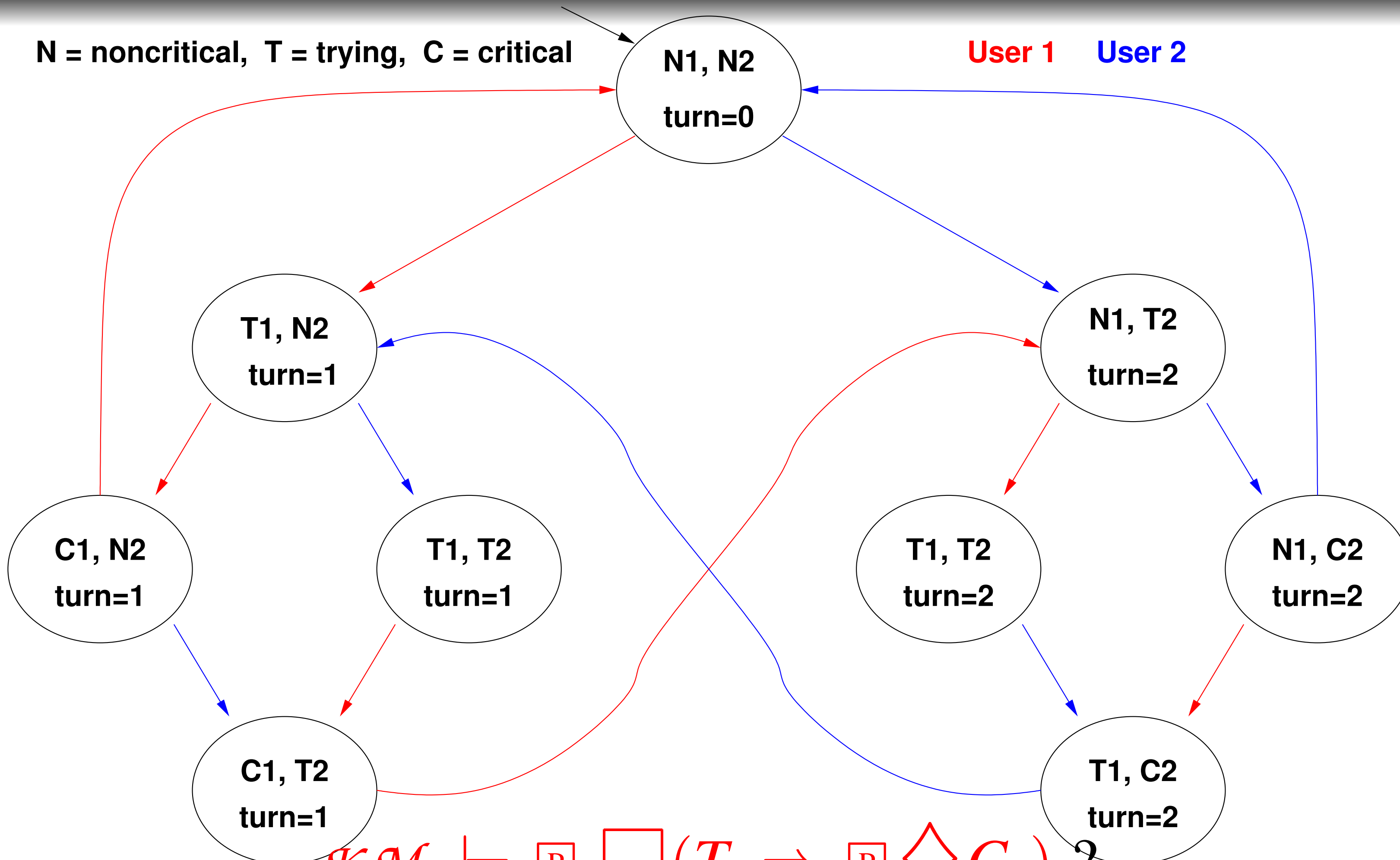
Example 1: Mutual Exclusion (Safety)



$\mathcal{KM} \models \boxed{\mathrm{P}}\ \Box\neg(C_1 \wedge C_2)$?

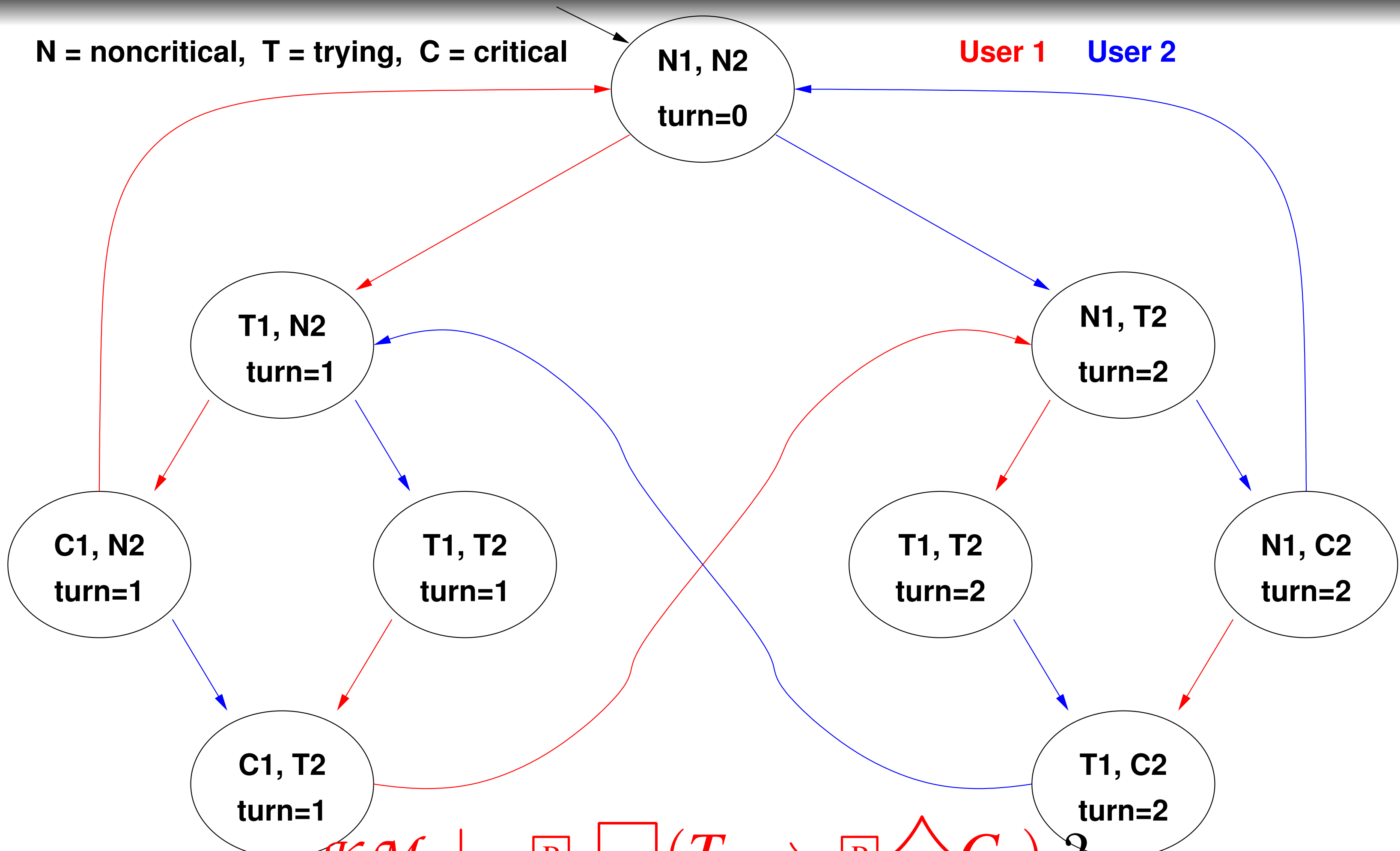
YES: There is no reachable state in which $(C_1 \wedge C_2)$ holds!
(Same as the $\Box\neg(C_1 \wedge C_2)$ in LTL.)

Example 2: Liveness



$$\mathcal{KM} \models \boxed{\text{P}}\,\Box(T_1 \Rightarrow \boxed{\text{P}}\,\Diamond C_1)\ ?$$

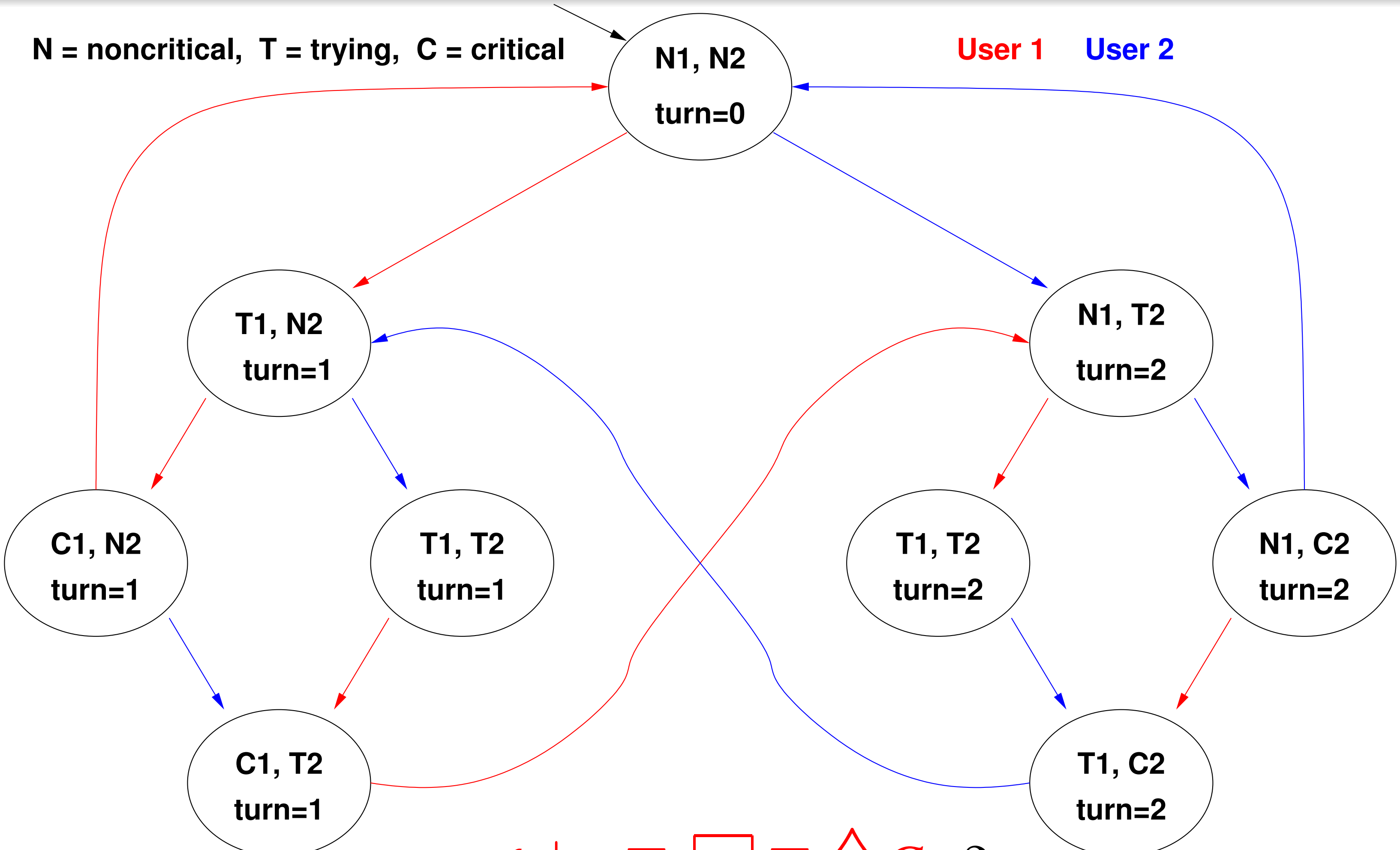
Example 2: Liveness



$\mathcal{KM} \models \boxed{\text{P}}\,\Box(T_1 \Rightarrow \boxed{\text{P}}\Diamond C_1)$?

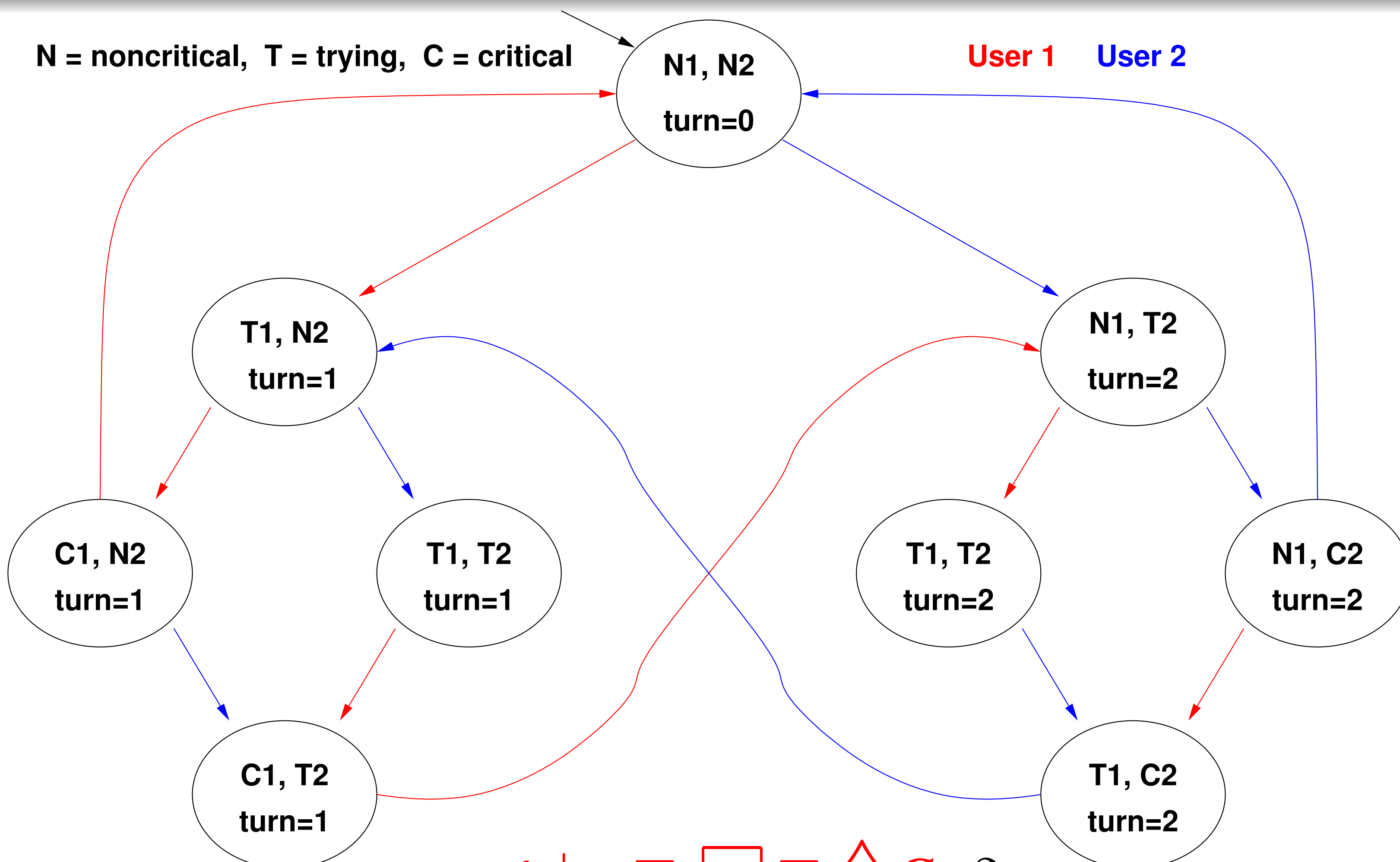
YES: every path starting from each state where T_1 holds passes through a state where C_1 holds.
(Same as $\Box(T_1 \Rightarrow \Diamond C_1)$ in LTL)

Example 3: Fairness



$\mathcal{KM} \models \boxed{P} \Box \boxed{P} \Diamond C_1$?

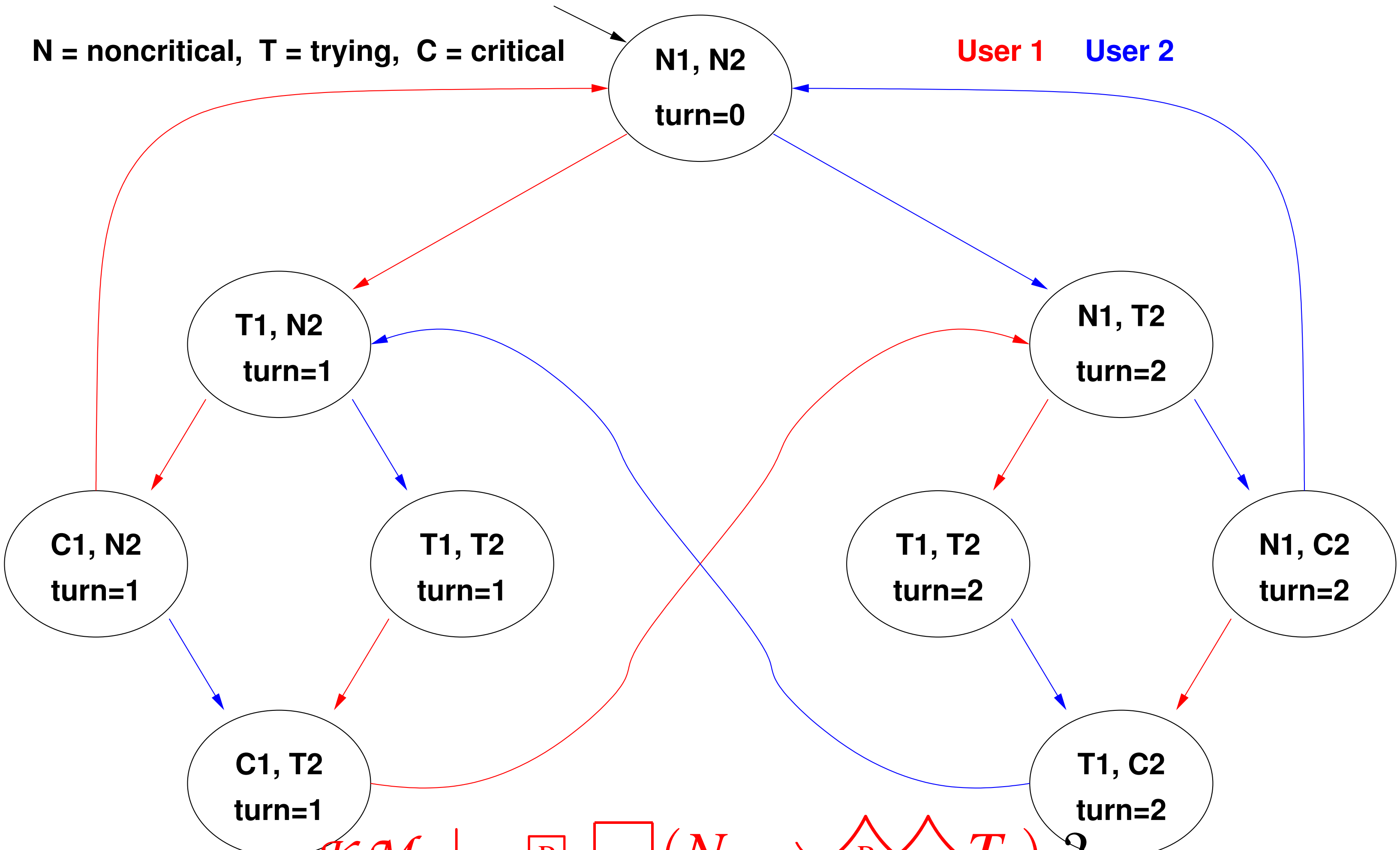
Example 3: Fairness



$$\mathcal{KM} \models \boxed{\mathrm{P}}\,\Box\,\boxed{\mathrm{P}}\Diamond C_1 \;?$$

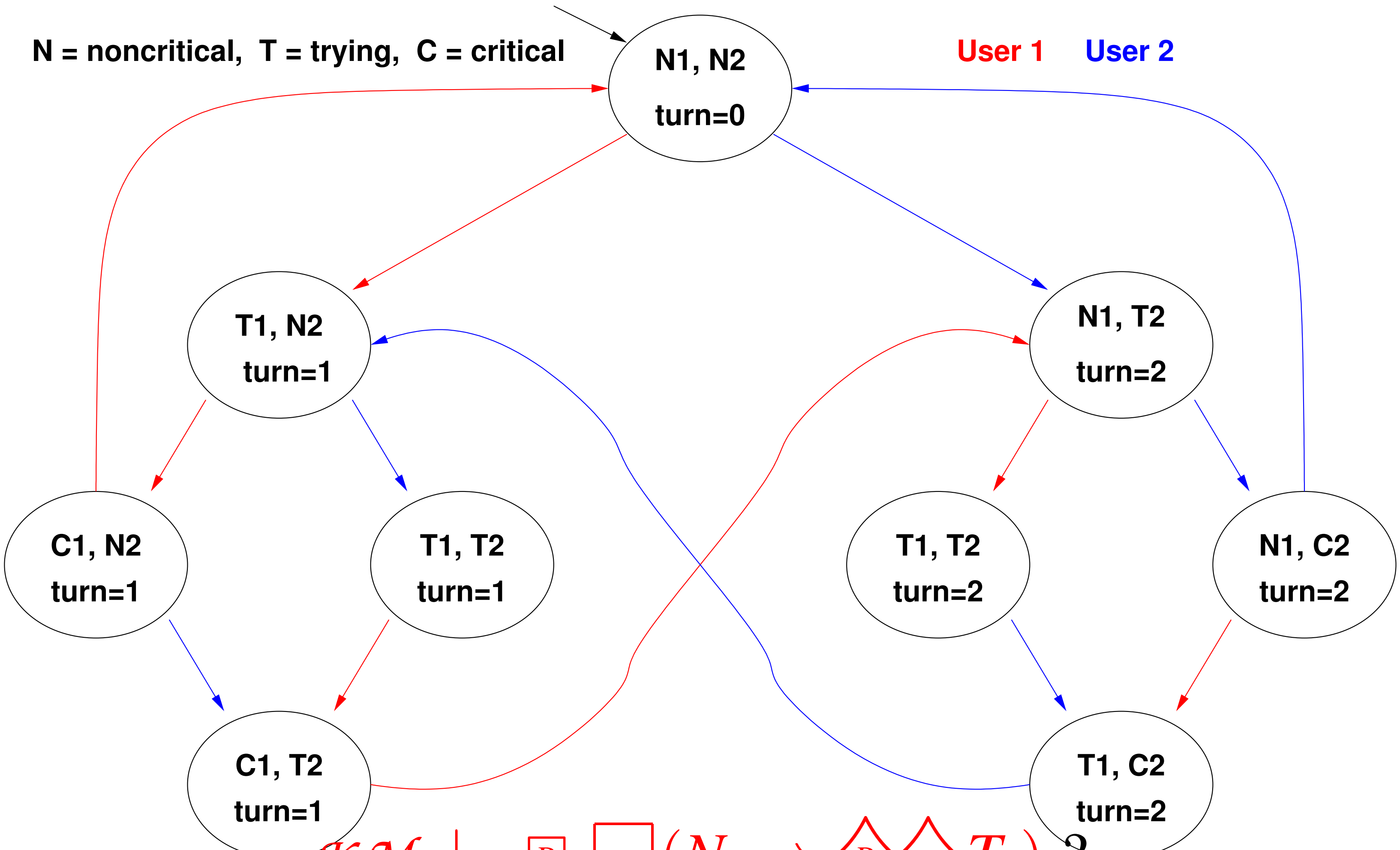
NO: e.g., in the initial state, there is the blue cyclic path in which C_1 never holds! (Same as $\Box\Diamond C_1$ in LTL)

Example 4: Non-Blocking



$$\mathcal{KM} \models \boxed{\mathrm{P}}\, \Box (N_1 \Rightarrow \Diamond_{\mathrm{P}} \Diamond T_1) \;?$$

Example 4: Non-Blocking



$$\mathcal{KM} \models \boxed{\text{P}}\, \Box (N_1 \Rightarrow \langle \text{P} \rangle \Diamond T_1) \ ?$$

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*.

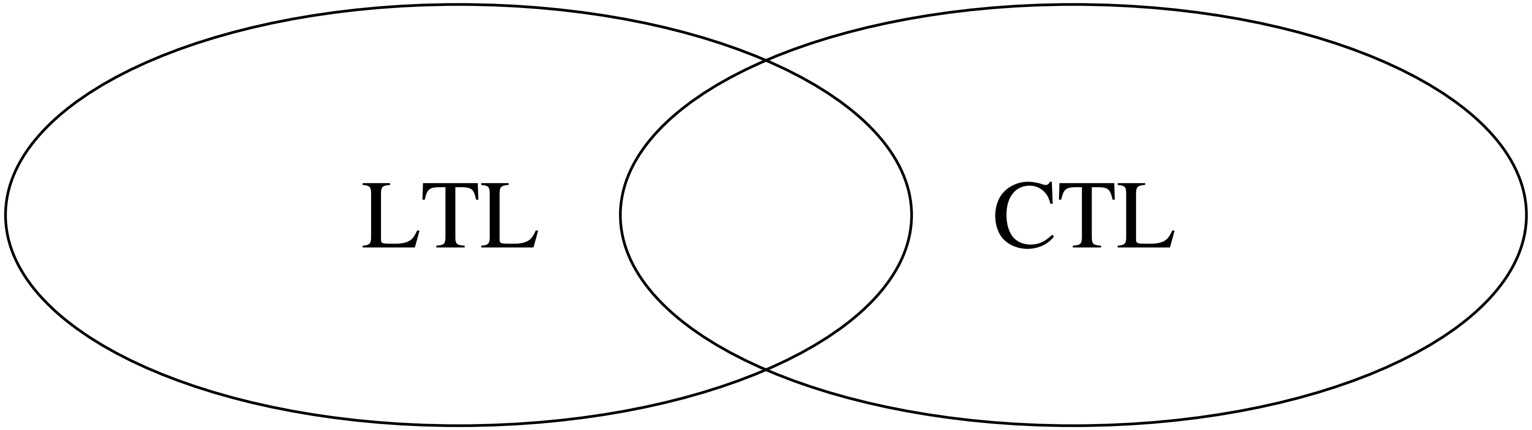
LTL Vs. CTL: Expressiveness

- Many CTL formulas cannot be expressed in LTL (e.g., those containing paths quantified existentially)
  E.g., $\boxed{\mathrm{P}}\ \Box(N_1 \Rightarrow \mathrm{P}\!\!\!\!\!\!\Diamond\ \Diamond T_1)$
- 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
  ($\Diamond p \Rightarrow \Diamond q$ Vs. $\boxed{\mathrm{P}}\ \Box(p \Rightarrow \boxed{\mathrm{P}}\ \Diamond q)$)
- Some formluas can be expressed both in LTL and in CTL (typically LTL formulas with operators of nesting depth 1)
  E.g., $\Box\neg(C_1 \wedge C_2)$, $\Diamond C_1$, $\Box(T_1 \Rightarrow \Diamond C_1)$, $\Box\Diamond C_1$

LTL Vs. CTL: Expressiveness (Cont.)

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*.

The Computation Tree Logic CTL*

- CTL* is a logic that combines the expressive power of LTL and CTL.
- Temporal operators can be applied without any constraints.
  - $\boxed{\text{P}}\,(\bigcirc\varphi \vee \bigcirc\bigcirc\varphi)$.
    Along all paths, φ is true in the next state or the next two steps.
  - $\Diamond_{\text{P}}\,(\Box\Diamond\varphi)$.
    There is a path along which φ is infinitely often true.

CTL*: Syntax

Countable set Σ of atomic propositions: $p, q, \ldots$ we distinguish between *States Formulas* (evaluated on states):

$$\begin{aligned} \varphi, \psi \quad \rightarrow \quad & p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \\ & \boxed{P}\, \alpha \mid \Diamond_{P}\, \alpha \end{aligned}$$

and *Path Formulas* (evaluated on paths):

$$\begin{aligned} \alpha, \beta \quad \rightarrow \quad & \varphi \mid \\ & \neg\alpha \mid \alpha \wedge \beta \mid \alpha \vee \beta \mid \\ & \bigcirc\alpha \mid \Box\alpha \mid \Diamond\alpha \mid (\alpha\, \mathcal{U}\, \beta) \end{aligned}$$

The set of CTL* formulas FORM is the set of state formulas.

CTL* Semantics: State Formulas

We start by defining when an atomic proposition is true at a state "s_0"

$$\mathcal{KM}, s_0 \models p \quad \textbf{iff} \quad p \in L(s_0) \qquad (\text{for } p \in \Sigma)$$

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

$$\mathcal{KM}, s_0 \models \neg\varphi \quad \textbf{iff} \quad \mathcal{KM}, s_0 \not\models \varphi$$

$$\mathcal{KM}, s_0 \models \varphi \wedge \psi \quad \textbf{iff} \quad \mathcal{KM}, s_0 \models \varphi \text{ and } \mathcal{KM}, s_0 \models \psi$$

$$\mathcal{KM}, s_0 \models \varphi \vee \psi \quad \textbf{iff} \quad \mathcal{KM}, s_0 \models \varphi \text{ or } \mathcal{KM}, s_0 \models \psi$$

$$\mathcal{KM}, s_0 \models \Diamond\!\!\!\!\!\!\;\text{P}\ \alpha \quad \textbf{iff} \quad \exists \pi = (s_0, s_1, \ldots) \text{ such that } \mathcal{KM}, \pi \models \alpha$$

$$\mathcal{KM}, s_0 \models \boxed{\text{P}}\ \alpha \quad \textbf{iff} \quad \forall \pi = (s_0, s_1, \ldots) \text{ then } \mathcal{KM}, \pi \models \alpha$$

CTL* Semantics: Path Formulas

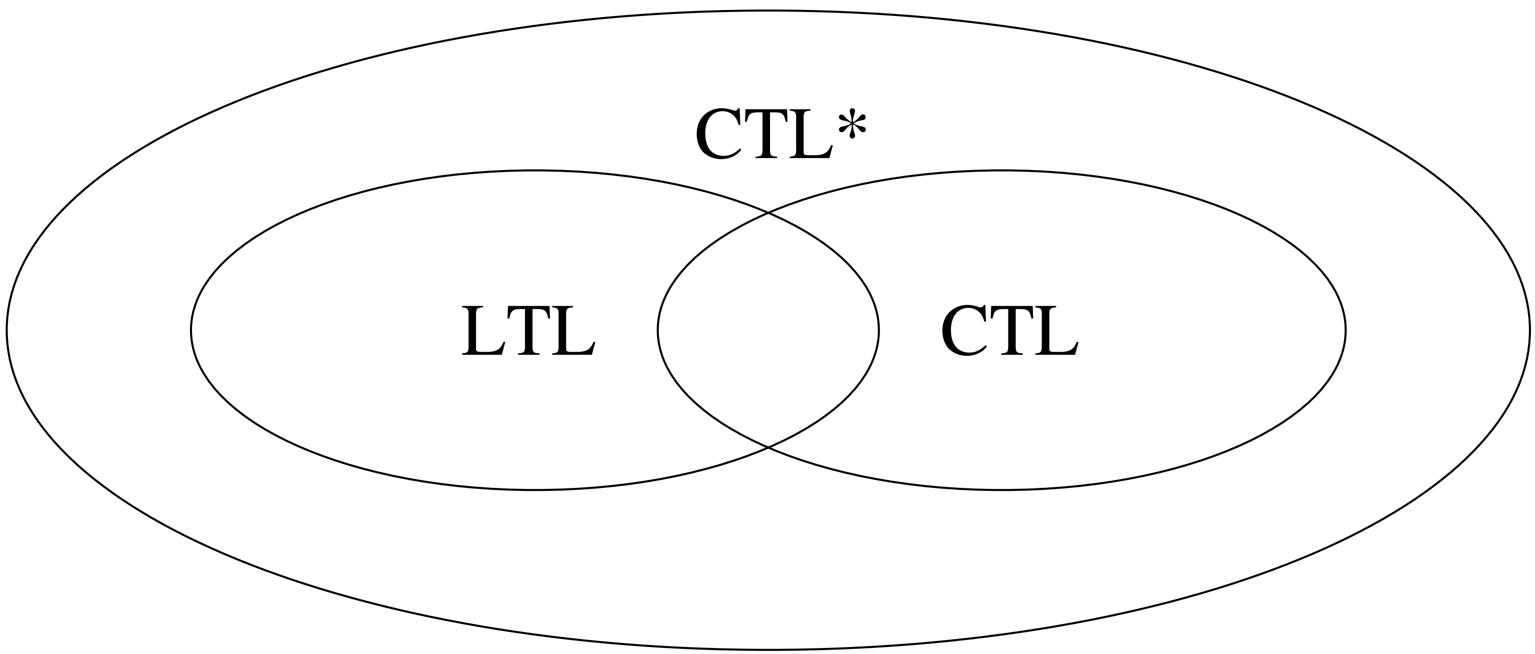
The semantics for *Path Formulas* is the following where $\pi = (s_0, s_1, \ldots)$ is a generic path outgoing from state s_0 and π^i denotes the suffix path $(s_i, s_{i+1}, \ldots)$:

$\mathcal{KM}, \pi \models \varphi$	**iff**	$\mathcal{KM}, s_0 \models \varphi$
$\mathcal{KM}, \pi \models \neg\alpha$	**iff**	$\mathcal{KM}, \pi \not\models \alpha$
$\mathcal{KM}, \pi \models \alpha \wedge \beta$	**iff**	$\mathcal{KM}, \pi \models \alpha$ and $\mathcal{KM}, \pi \models \beta$
$\mathcal{KM}, \pi \models \alpha \vee \beta$	**iff**	$\mathcal{KM}, \pi \models \alpha$ or $\mathcal{KM}, \pi \models \beta$
$\mathcal{KM}, \pi \models \Diamond\alpha$	**iff**	$\exists i \geq 0$ such that $\mathcal{KM}, \pi^i \models \alpha$
$\mathcal{KM}, \pi \models \Box\alpha$	**iff**	$\forall i \geq 0$ then $\mathcal{KM}, \pi^i \models \alpha$
$\mathcal{KM}, \pi \models \bigcirc\alpha$	**iff**	$\mathcal{KM}, \pi^1 \models \alpha$
$\mathcal{KM}, \pi \models \alpha\, \mathcal{U}\, \beta$	**iff**	$\exists i \geq 0$ such that $\mathcal{KM}, \pi^i \models \beta$ and $\forall j.(0 \leq j \leq i)$ then $\mathcal{KM}, \pi^j \models \alpha$

CTLs Vs LTL Vs CTL: Expressiveness

CTL* subsumes both CTL and LTL

- φ in CTL $\Longrightarrow$ φ in CTL* (e.g., $\boxed{\mathrm{P}}\,\Box(N_1 \Rightarrow \Diamond\!\!\!\!\!\!\;\mathrm{P}\;\Diamond T_1)$)
- φ in LTL $\Longrightarrow$ $\boxed{\mathrm{P}}\,\varphi$ in CTL* (e.g., $\boxed{\mathrm{P}}\,(\Box\Diamond T_1 \Rightarrow \Box\Diamond C_1)$)
- LTL $\cup$ CTL $\subset$ CTL* (e.g., $\Diamond\!\!\!\!\!\!\;\mathrm{P}\;(\Box\Diamond p \Rightarrow \Box\Diamond q)$)



CTL* Vs LTL Vs CTL: Complexity

The following Table shows the Computational Complexity of checking *Satisbiability*

Logic	Complexity
LTL	PSpace-Complete
CTL	ExpTime-Complete
CTL*	2ExpTime-Complete

CTL* Vs LTL Vs CTL: Complexity (Cont.)

The following Table shows the Computational Complexity of *Model Checking* (M.C.)

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

Logic	**Complexity w.r.t. $\vert\varphi\vert$**	**Complexity w.r.t. $\vert\mathcal{M}\vert$**
LTL	PSpace-Complete	P (linear)
CTL	P-Complete	P (linear)
CTL*	PSpace-Complete	P (linear)

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*.