Overview

Introduction

Modelling parallel systems

Linear Time Properties

Regular Properties

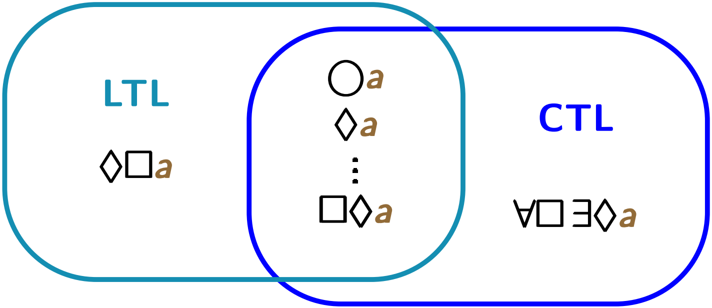
Linear Temporal Logic (LTL)

Computation Tree Logic

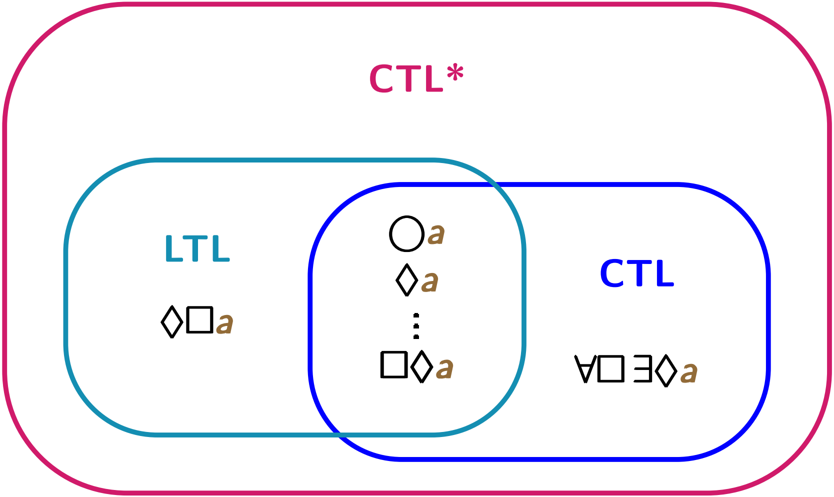
- syntax and semantics of CTL
- expressiveness of CTL and LTL
- CTL model checking
- fairness, counterexamples/witnesses
- CTL^+ and CTL^* ⟵

Equivalences and Abstraction

CTL and LTL



Combining LTL and CTL $\rightsquigarrow$ CTL*



Syntax of CTL*

Syntax of CTL*

state formulas:

$$\Phi \ ::= \ true \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

path formulas:

$$\varphi \ ::= \ \ldots$$

Syntax of CTL*

state formulas:

$$\Phi \ ::= \ \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

path formulas:

$$\varphi \ ::= \ \Phi \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \, \mathsf{U} \, \varphi_2$$

Syntax of CTL*

state formulas:

$$\Phi \; ::= \; \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

path formulas:

$$\varphi \; ::= \; \Phi \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \, \mathsf{U} \, \varphi_2$$

derived operators:

- $\vee$, $\longrightarrow$, etc.
- eventually, always

Syntax of CTL*

state formulas:

$$\Phi \ ::= \ \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

path formulas:

$$\varphi \ ::= \ \Phi \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \, \mathsf{U} \, \varphi_2$$

derived operators:

- $\vee$, $\rightarrow$, etc.
- eventually, always as in **LTL**:

$$\Diamond\varphi \ = \ \mathit{true} \, \mathsf{U} \, \varphi, \quad \Box\varphi \ = \ \neg\Diamond\neg\varphi$$

state formulas:

$$\Phi ::= \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

path formulas:

$$\varphi ::= \Phi \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2$$

derived operators:

- $\vee$, $\longrightarrow$, etc.
- eventually, always as in **LTL**:

$$\Diamond\varphi = \mathit{true}\,\mathsf{U}\,\varphi, \quad \Box\varphi = \neg\Diamond\neg\varphi$$

- universal quantification:

Syntax of CTL*

state formulas:

$$\Phi \ ::= \ \textit{true} \ \big| \ a \ \big| \ \Phi_1 \wedge \Phi_2 \ \big| \ \neg\Phi \ \big| \ \exists\varphi$$

path formulas:

$$\varphi \ ::= \ \Phi \ \big| \ \varphi_1 \wedge \varphi_2 \ \big| \ \neg\varphi \ \big| \ \bigcirc\varphi \ \big| \ \varphi_1 \, \mathsf{U} \, \varphi_2$$

derived operators:

- $\vee$, $\rightarrow$, etc.
- eventually, always as in **LTL**:

  $$\Diamond\varphi \ = \ \textit{true}\,\mathsf{U}\,\varphi, \quad \Box\varphi \ = \ \neg\Diamond\neg\varphi$$

- universal quantification: $\forall\varphi = \neg\exists\neg\varphi$

Semantics of CTL*

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a transition system without terminal states.

Semantics of CTL*

Let $\mathcal{T} = (S, Act, \to, S_0, AP, L)$ be a transition system without terminal states.

define by structural induction:

- a satisfaction relation $\models$ for states $s \in S$ and **CTL*** state formulas
- a satisfaction relation $\models$ for infinite path fragments π in $\mathcal{T}$ and **CTL*** path formulas

Semantics of CTL* state formulas

Semantics of CTL* state formulas

$s \models true$		
$s \models a$	iff	$a \in L(s)$
$s \models \neg\Phi$	iff	$s \not\models \Phi$
$s \models \Phi_1 \wedge \Phi_2$	iff	$s \models \Phi_1$ and $s \models \Phi_2$
$s \models \exists\varphi$	iff	there exists a path $\pi \in Paths(s)$ such that $\pi \models \varphi$

Semantics of CTL* state formulas

$s \models true$

$s \models a$ iff $a \in L(s)$

$s \models \neg\Phi$ iff $s \not\models \Phi$

$s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1$ and $s \models \Phi_2$

$s \models \exists\varphi$ iff there exists a path $\pi \in Paths(s)$ such that $\pi \models \varphi$

↑ satisfaction relation $\models$ for **CTL*** path formulas

Semantics of CTL* path formulas

let $\pi = s_0\, s_1\, s_2 \ldots$ be an infinite path fragment in $\mathcal{T}$

let $\pi = s_0\, s_1\, s_2 \ldots$ be an infinite path fragment in $\mathcal{T}$

$\pi \models \Phi$	iff	$\ldots$
$\pi \models \neg\varphi$	iff	$\pi \not\models \varphi$
$\pi \models \varphi_1 \wedge \varphi_2$	iff	$\pi \models \varphi_1$ and $\pi \models \varphi_2$
$\pi \models \bigcirc\varphi$	iff	$\mathit{suffix}(\pi, 1) \models \varphi$
$\pi \models \varphi_1 \, \mathsf{U} \, \varphi_2$	iff	there exists $j \geq 0$ such that $\mathit{suffix}(\pi, j) \models \varphi_2$ $\mathit{suffix}(\pi, i) \models \varphi_1$ for $0 \leq i < j$

Semantics of CTL* path formulas

let $\pi = s_0\, s_1\, s_2 \ldots$ be an infinite path fragment in $\mathcal{T}$

$\pi \models \Phi$	iff	$\ldots$
$\pi \models \neg\varphi$	iff	$\pi \not\models \varphi$
$\pi \models \varphi_1 \wedge \varphi_2$	iff	$\pi \models \varphi_1$ and $\pi \models \varphi_2$
$\pi \models \bigcirc\varphi$	iff	$suffix(\pi, 1) \models \varphi$
$\pi \models \varphi_1 \mathbf{U} \varphi_2$	iff	there exists $j \geq 0$ such that $suffix(\pi, j) \models \varphi_2$ $suffix(\pi, i) \models \varphi_1$ for $0 \leq i < j$

$$suffix(\pi, k) = s_k\, s_{k+1}\, s_{k+2} \ldots$$

Semantics of CTL* path formulas

let $\pi = s_0\, s_1\, s_2 \ldots$ be an infinite path fragment in $\mathcal{T}$

$\pi \models \Phi$	iff	$s_0 \models \Phi$
$\pi \models \neg\varphi$	iff	$\pi \not\models \varphi$
$\pi \models \varphi_1 \wedge \varphi_2$	iff	$\pi \models \varphi_1$ and $\pi \models \varphi_2$
$\pi \models \bigcirc\varphi$	iff	$suffix(\pi, 1) \models \varphi$
$\pi \models \varphi_1 \mathbf{U} \varphi_2$	iff	there exists $j \geq 0$ such that $suffix(\pi, j) \models \varphi_2$ $suffix(\pi, i) \models \varphi_1$ for $0 \leq i < j$

$$suffix(\pi, k) = s_k\, s_{k+1}\, s_{k+2} \ldots$$

Semantics of CTL* path formulas

let $\pi = s_0\, s_1\, s_2 \ldots$ be an infinite path fragment in $\mathcal{T}$

$\pi \models \Phi$	iff	$s_0 \models \Phi$ ⟵ satisfaction relation for **CTL*** state formulas
$\pi \models \neg\varphi$	iff	$\pi \not\models \varphi$
$\pi \models \varphi_1 \wedge \varphi_2$	iff	$\pi \models \varphi_1$ and $\pi \models \varphi_2$
$\pi \models \bigcirc\varphi$	iff	$suffix(\pi, 1) \models \varphi$
$\pi \models \varphi_1 \, \mathsf{U} \, \varphi_2$	iff	there exists $j \geq 0$ such that $suffix(\pi, j) \models \varphi_2$, $suffix(\pi, i) \models \varphi_1$ for $0 \leq i < j$

$$suffix(\pi, k) = s_k\, s_{k+1}\, s_{k+2} \ldots$$

Examples of CTL*-formulas

Examples of CTL*-formulas

mutual exclusion:

safety $\forall\Box(\neg \mathit{crit}_1 \vee \neg \mathit{crit}_2)$

Examples of CTL*-formulas

mutual exclusion:

safety	$\forall\Box(\neg crit_1 \vee \neg crit_2)$
liveness	$\forall\Box\Diamond crit_1 \;\wedge\; \forall\Box\Diamond crit_2$

Examples of CTL*-formulas

mutual exclusion:

safety $\forall\Box(\neg crit_1 \vee \neg crit_2)$

liveness $\forall\Box\Diamond crit_1 \ \wedge \ \forall\Box\Diamond crit_2$

progress property, e.g., $\forall\Box(request \rightarrow \Diamond response)$

Examples of CTL*-formulas

mutual exclusion:

safety $\forall\Box(\neg crit_1 \vee \neg crit_2)$

liveness $\forall\Box\Diamond crit_1 \;\wedge\; \forall\Box\Diamond crit_2$

progress property, e.g., $\forall\Box(request \rightarrow \Diamond response)$

persistence property, e.g., $\forall\Diamond\Box a$

Examples of CTL*-formulas

mutual exclusion:

safety $\forall\Box(\neg crit_1 \vee \neg crit_2)$

liveness $\forall\Box\Diamond crit_1 \;\wedge\; \forall\Box\Diamond crit_2$

progress property, e.g., $\forall\Box(request \rightarrow \Diamond response)$

persistence property, e.g., $\forall\Diamond\Box a$

CTL* formulas with existential quantification, e.g., Hamilton path problem (for fixed initial state)

$$\exists\Big(\bigwedge_{v\in V}\big(\Diamond v \wedge \Box(v \rightarrow \bigcirc\Box\neg v)\big)\Big)$$

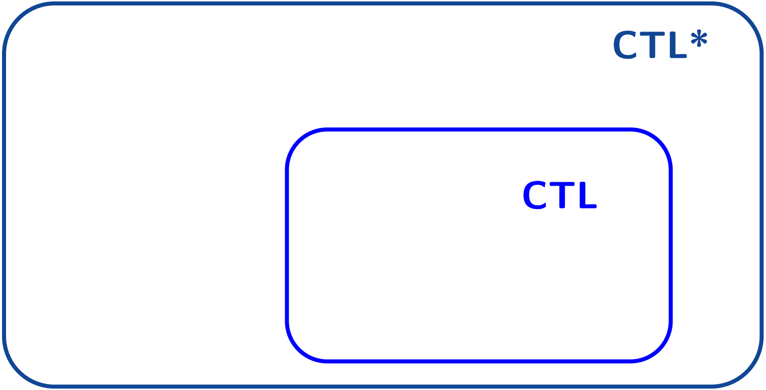
Expressiveness of CTL, LTL and CTL*

Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***

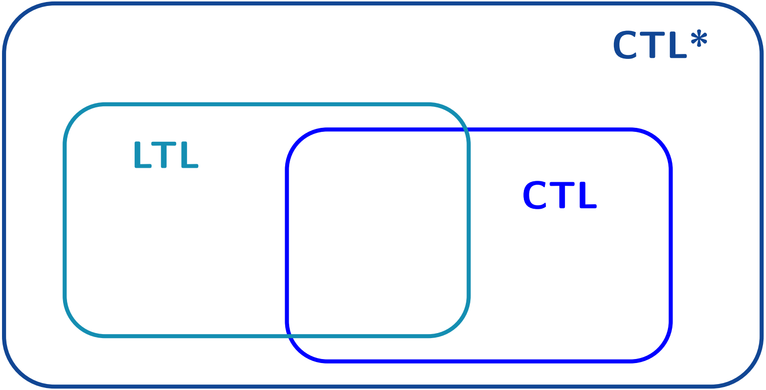
Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***



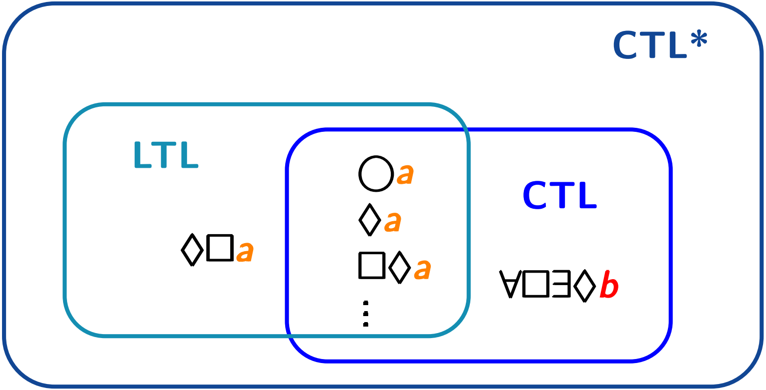
Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***
- **LTL** is a sublogic of **CTL***



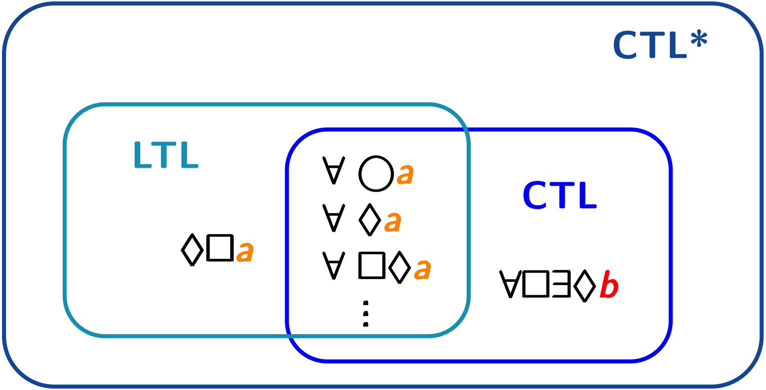
Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***
- **LTL** is a sublogic of **CTL***



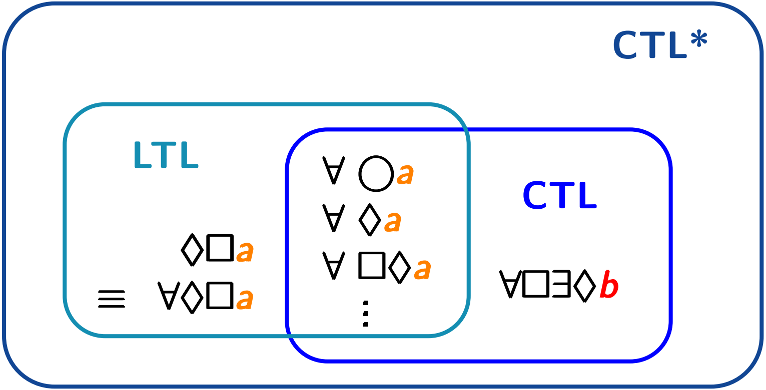
Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***
- **LTL** is a sublogic of **CTL***



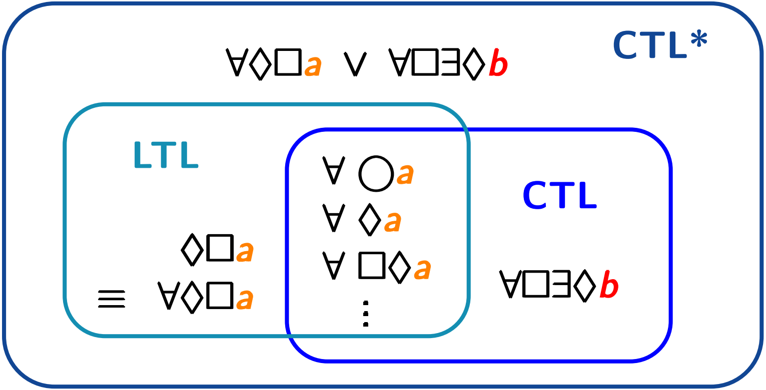
Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***
- **LTL** is a sublogic of **CTL***



Expressiveness of CTL, LTL and CTL*

- **CTL** is a sublogic of **CTL***
- **LTL** is a sublogic of **CTL***
- **CTL*** is more expressive than **LTL** and **CTL**



Equivalence of CTL*-formulas

$\Phi_1 \equiv \Phi_2$ iff for all transition systems $\mathcal{T}$:

$$\mathcal{T} \models \Phi_1 \iff \mathcal{T} \models \Phi_2$$

Equivalence of CTL*-formulas

$\Phi_1 \equiv \Phi_2$ iff for all transition systems $\mathcal{T}$:
$$\mathcal{T} \models \Phi_1 \iff \mathcal{T} \models \Phi_2$$

Examples:

$$\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$$

$$\forall\Box\Diamond a \equiv \forall\Box\,\forall\Diamond a$$

$$\vdots$$

$\Phi_1 \equiv \Phi_2$ iff for all transition systems $\mathcal{T}$:

$$\mathcal{T} \models \Phi_1 \iff \mathcal{T} \models \Phi_2$$

Examples:

$$\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$$

$$\forall\Box\Diamond a \equiv \forall\Box\,\forall\Diamond a$$

$$\vdots$$

$$\forall\,\forall\varphi \equiv \forall\varphi$$

$$\exists\,\exists\varphi \equiv \exists\varphi$$

Equivalence of CTL*-formulas

$\Phi_1 \equiv \Phi_2$ iff for all transition systems $\mathcal{T}$:

$$\mathcal{T} \models \Phi_1 \iff \mathcal{T} \models \Phi_2$$

Examples:

$$\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$$

$$\forall\Box\Diamond a \equiv \forall\Box\,\forall\Diamond a$$

$$\vdots$$

$$\forall\forall\varphi \equiv \forall\varphi$$

$$\exists\exists\varphi \equiv \exists\varphi$$

$$\forall\exists\varphi \equiv \,?$$

$\Phi_1 \equiv \Phi_2$ iff for all transition systems $\mathcal{T}$:

$$\mathcal{T} \models \Phi_1 \iff \mathcal{T} \models \Phi_2$$

Examples:

$$\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$$

$$\forall\Box\Diamond a \equiv \forall\Box\,\forall\Diamond a$$

$$\vdots$$

$$\forall\forall\varphi \equiv \forall\varphi$$

$$\exists\exists\varphi \equiv \exists\varphi$$

$$\forall\exists\varphi \equiv \exists\varphi$$

Correct or wrong?

$$\forall(\varphi_1 \vee \varphi_2) \equiv \forall\varphi_1 \vee \forall\varphi_2$$

Correct or wrong?

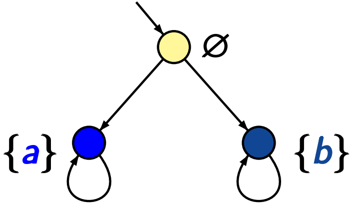
$$\forall(\varphi_1 \vee \varphi_2) \;\equiv\; \forall\varphi_1 \;\vee\; \forall\varphi_2$$

wrong, e.g., $\forall(\Diamond a \vee \Diamond b) \;\not\equiv\; \forall\Diamond a \vee \forall\Diamond b$

Correct or wrong?

$$\forall(\varphi_1 \vee \varphi_2) \equiv \forall\varphi_1 \vee \forall\varphi_2$$

wrong, e.g., $\forall(\Diamond a \vee \Diamond b) \not\equiv \forall\Diamond a \vee \forall\Diamond b$



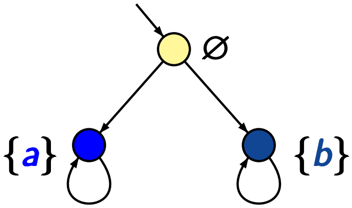
$\models \forall(\Diamond a \vee \Diamond b)$

$\not\models \forall\Diamond a$

$\not\models \forall\Diamond b$

$$\forall(\varphi_1 \vee \varphi_2) \;\equiv\; \forall\varphi_1 \;\vee\; \forall\varphi_2$$

wrong, e.g., $\forall(\Diamond a \vee \Diamond b) \;\not\equiv\; \forall\Diamond a \vee \forall\Diamond b$



$\models\ \forall(\Diamond a \ \vee\ \Diamond b)$

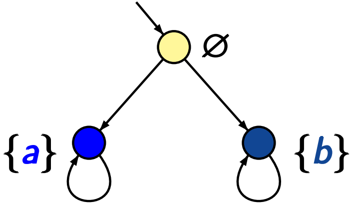
$\not\models\ \forall\Diamond a$

$\not\models\ \forall\Diamond b$

$$\exists(\varphi_1 \;\vee\; \varphi_2) \;\equiv\; \exists\varphi_1 \;\vee\; \exists\varphi_2$$

$$\forall(\varphi_1 \lor \varphi_2) \equiv \forall\varphi_1 \lor \forall\varphi_2$$

wrong, e.g., $\forall(\Diamond a \lor \Diamond b) \not\equiv \forall\Diamond a \lor \forall\Diamond b$



$\models \forall(\Diamond a \lor \Diamond b)$

$\not\models \forall\Diamond a$

$\not\models \forall\Diamond b$

$$\exists(\varphi_1 \lor \varphi_2) \equiv \exists\varphi_1 \lor \exists\varphi_2$$

correct

Correct or wrong?

$$\exists\Diamond\exists\Box a \;\equiv\; \exists\Diamond\Box a$$

Correct or wrong?

$$\exists\Diamond\exists\Box a \;\equiv\; \exists\Diamond\Box a$$

correct.

Correct or wrong?

$$\exists\Diamond\exists\Box a \equiv \exists\Diamond\Box a$$

correct. $\exists\Diamond\exists\Box a \equiv \neg\forall\Box\forall\Diamond\neg a$

Correct or wrong?

$$\exists\Diamond\exists\Box a \equiv \exists\Diamond\Box a$$

correct. $\exists\Diamond\exists\Box a \equiv \neg\forall\Box\forall\Diamond\neg a$

$\equiv \neg\forall\Box\Diamond\neg a$

Correct or wrong?

$$\exists\Diamond\exists\Box a \equiv \exists\Diamond\Box a$$

correct.

$$\begin{aligned}\exists\Diamond\exists\Box a &\equiv \neg\forall\Box\forall\Diamond\neg a\\ &\equiv \neg\forall\Box\Diamond\neg a\\ &\equiv \exists\Diamond\Box a\end{aligned}$$

Correct or wrong?

$$\exists\Diamond\exists\Box a \equiv \exists\Diamond\Box a$$

correct.

$$\begin{aligned}\exists\Diamond\exists\Box a &\equiv \neg\forall\Box\forall\Diamond\neg a\\ &\equiv \neg\forall\Box\Diamond\neg a\\ &\equiv \exists\Diamond\Box a\end{aligned}$$

$$\exists\bigcirc\exists\Diamond a \equiv \exists\bigcirc\Diamond a$$

Correct or wrong?

$$\exists\Diamond\exists\Box a \;\equiv\; \exists\Diamond\Box a$$

correct.

$$\begin{aligned} \exists\Diamond\exists\Box a &\equiv \neg\forall\Box\forall\Diamond\neg a \\ &\equiv \neg\forall\Box\Diamond\neg a \\ &\equiv \exists\Diamond\Box a \end{aligned}$$

$$\exists\bigcirc\exists\Diamond a \;\equiv\; \exists\bigcirc\Diamond a$$

correct.

Correct or wrong?

$$\exists\Diamond\exists\Box a \equiv \exists\Diamond\Box a$$

correct.

$$\begin{aligned}\exists\Diamond\exists\Box a &\equiv \neg\forall\Box\forall\Diamond\neg a\\ &\equiv \neg\forall\Box\Diamond\neg a\\ &\equiv \exists\Diamond\Box a\end{aligned}$$

$$\exists\bigcirc\exists\Diamond a \equiv \exists\bigcirc\Diamond a$$

correct. Both formulas assert that an a-state is reachable from the current state within one or more steps.

Combinations of □ and ◊ in CTL*

Combinations of $\Box$ and $\Diamond$ in CTL*

we already saw:

$$\forall\Box\,\forall\Diamond a \;\equiv\; \forall\Box\Diamond a$$

$$\exists\Diamond\,\exists\Box a \;\equiv\; \exists\Diamond\Box a$$

Combinations of $\Box$ and $\Diamond$ in CTL*

we already saw:

$$\forall\Box\,\forall\Diamond a \;\equiv\; \forall\Box\Diamond a$$

$$\exists\Diamond\,\exists\Box a \;\equiv\; \exists\Diamond\Box a$$

does $\exists\Box\,\exists\Diamond a \;\equiv\; \exists\Box\Diamond a$ hold **?**

Combinations of $\Box$ and $\Diamond$ in CTL*

we already saw:

$$\forall\Box\,\forall\Diamond a \;\equiv\; \forall\Box\Diamond a$$

$$\exists\Diamond\,\exists\Box a \;\equiv\; \exists\Diamond\Box a$$

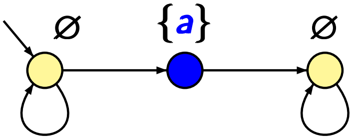
does $\exists\Box\,\exists\Diamond a \;\equiv\; \exists\Box\Diamond a$ hold **?**

answer: **no**

$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

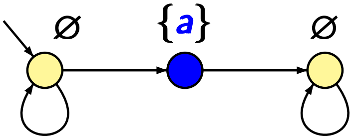
$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

$\mathcal{T}$:

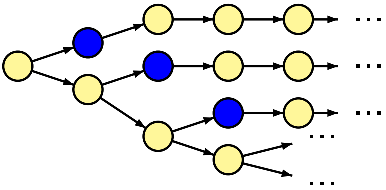


$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

$\mathcal{T}$:

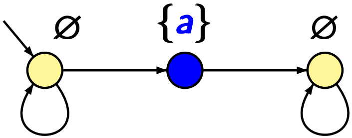


computation tree:



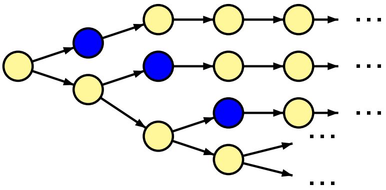
$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

$\mathcal{T}$:



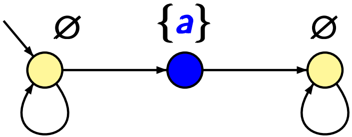
$\mathcal{T} \not\models \exists\Box\Diamond a$

computation tree:



$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

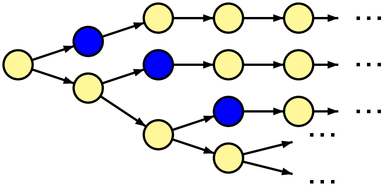
$\mathcal{T}$:



$\mathcal{T} \not\models \exists\Box\Diamond a$

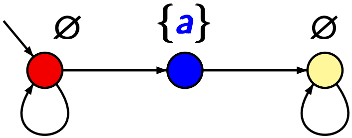
$\mathcal{T} \models \exists\Box\exists\Diamond a$

computation tree:



$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

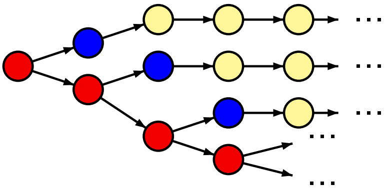
$\mathcal{T}$:



$\mathcal{T} \not\models \exists\Box\Diamond a$

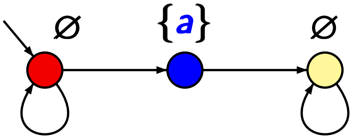
$\mathcal{T} \models \exists\Box\exists\Diamond a$ note: $Sat(\exists\Diamond a) = \{ \bullet, \bullet \}$

computation tree:



$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

$\mathcal{T}$:

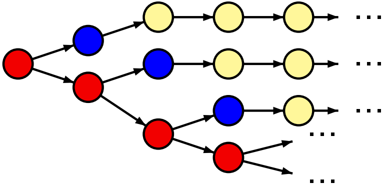


$\mathcal{T} \not\models \exists\Box\Diamond a$

$\mathcal{T} \models \exists\Box\exists\Diamond a$ note: $Sat(\exists\Diamond a) = \{$ ●, ● $\}$

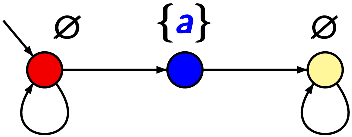
hence: ●●●... $\models \Box\exists\Diamond a$

computation tree:



$\exists\Box\exists\Diamond a$ and $\exists\Box\Diamond a$ are not equivalent

$\mathcal{T}$:



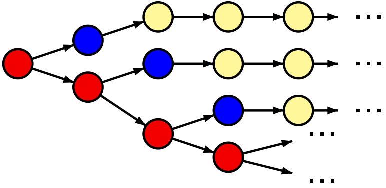
$$\exists\Box\exists\Diamond a \not\equiv \exists\Box\Diamond a$$

$\mathcal{T} \not\models \exists\Box\Diamond a$

$\mathcal{T} \models \exists\Box\exists\Diamond a$ note: $Sat(\exists\Diamond a) = \{$ ● (red), ● (blue) $\}$

hence: ●●●… (red) $\models \Box\exists\Diamond a$

computation tree:



Equivalence of CTL*-formulas

Equivalence of CTL*-formulas

$$\neg\exists\varphi \equiv \forall\neg\varphi$$

$$\neg\forall\varphi \equiv \exists\neg\varphi$$

Equivalence of CTL*-formulas

$\neg\exists\varphi \equiv \forall\neg\varphi$ e.g., $\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$

$\neg\forall\varphi \equiv \exists\neg\varphi$ e.g., $\neg\forall\Box\Diamond a \equiv \exists\Diamond\Box\neg a$

Equivalence of CTL*-formulas

$\neg\exists\varphi \equiv \forall\neg\varphi$ e.g., $\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$

$\neg\forall\varphi \equiv \exists\neg\varphi$ e.g., $\neg\forall\Box\Diamond a \equiv \exists\Diamond\Box\neg a$

$\forall(\varphi_1 \wedge \varphi_2) \equiv \forall\varphi_1 \wedge \forall\varphi_2$

$\exists(\varphi_1 \vee \varphi_2) \equiv \exists\varphi_1 \vee \exists\varphi_2$

Equivalence of CTL*-formulas

$\neg\exists\varphi \equiv \forall\neg\varphi$ e.g., $\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$

$\neg\forall\varphi \equiv \exists\neg\varphi$ e.g., $\neg\forall\Box\Diamond a \equiv \exists\Diamond\Box\neg a$

$$\forall(\varphi_1 \wedge \varphi_2) \equiv \forall\varphi_1 \wedge \forall\varphi_2$$

$$\exists(\varphi_1 \vee \varphi_2) \equiv \exists\varphi_1 \vee \exists\varphi_2$$

but: $\forall(\varphi_1 \vee \varphi_2) \not\equiv \forall\varphi_1 \vee \forall\varphi_2$

$\exists(\varphi_1 \wedge \varphi_2) \not\equiv \exists\varphi_1 \wedge \exists\varphi_2$

Equivalence of CTL*-formulas

$\neg\exists\varphi \equiv \forall\neg\varphi$ e.g., $\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$

$\neg\forall\varphi \equiv \exists\neg\varphi$ e.g., $\neg\forall\Box\Diamond a \equiv \exists\Diamond\Box\neg a$

$\forall(\varphi_1 \wedge \varphi_2) \equiv \forall\varphi_1 \wedge \forall\varphi_2$

$\exists(\varphi_1 \vee \varphi_2) \equiv \exists\varphi_1 \vee \exists\varphi_2$

but: $\forall(\varphi_1 \vee \varphi_2) \not\equiv \forall\varphi_1 \vee \forall\varphi_2$

$\exists(\varphi_1 \wedge \varphi_2) \not\equiv \exists\varphi_1 \wedge \exists\varphi_2$

$\forall\Box\Diamond\varphi \equiv \forall\Box\forall\Diamond\varphi$

$\exists\Diamond\Box\varphi \equiv \exists\Diamond\exists\Box\varphi$

Equivalence of CTL*-formulas

$\neg\exists\varphi \equiv \forall\neg\varphi$ e.g., $\neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$

$\neg\forall\varphi \equiv \exists\neg\varphi$ e.g., $\neg\forall\Box\Diamond a \equiv \exists\Diamond\Box\neg a$

$\forall(\varphi_1 \wedge \varphi_2) \equiv \forall\varphi_1 \wedge \forall\varphi_2$

$\exists(\varphi_1 \vee \varphi_2) \equiv \exists\varphi_1 \vee \exists\varphi_2$

but: $\forall(\varphi_1 \vee \varphi_2) \not\equiv \forall\varphi_1 \vee \forall\varphi_2$

$\exists(\varphi_1 \wedge \varphi_2) \not\equiv \exists\varphi_1 \wedge \exists\varphi_2$

$\forall\Box\Diamond\varphi \equiv \forall\Box\forall\Diamond\varphi$ but: $\forall\Diamond\Box\varphi \not\equiv \forall\Diamond\forall\Box\varphi$

$\exists\Diamond\Box\varphi \equiv \exists\Diamond\exists\Box\varphi$ $\exists\Box\Diamond\varphi \not\equiv \exists\Box\exists\Diamond\varphi$

CTL* model checking

CTL* model checking

given: finite TS $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$

CTL* formula Φ

question: does $\mathcal{T} \models \Phi$ hold **?**

CTL* model checking

given: finite TS $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$
CTL* formula Φ

question: does $\mathcal{T} \models \Phi$ hold ?

main procedure as for **CTL**:

```
FOR ALL subformulas Ψ of Φ DO
     compute Sat(Ψ) = {s ∈ S : s ⊨ Ψ}
OD
```

given: finite TS $\mathcal{T} = (S, Act, \to, S_0, AP, L)$

CTL* formula Φ

question: does $\mathcal{T} \models \Phi$ hold ?

main procedure as for **CTL**:

```
FOR ALL subformulas Ψ of Φ DO
     compute Sat(Ψ) = {s ∈ S : s ⊨ Ψ}
OD
IF S₀ ⊆ Sat(Φ)
     THEN   return "yes"
     ELSE   return "no"
FI
```

Recursive computation of satisfaction sets

Recursive computation of satisfaction sets

$$
\begin{aligned}
Sat(true) &= S \\
Sat(a) &= \{s \in S : a \in L(s)\} \\
Sat(\Phi_1 \wedge \Phi_2) &= Sat(\Phi_1) \cap Sat(\Phi_2) \\
Sat(\neg\Phi) &= S \setminus Sat(\Phi)
\end{aligned}
$$

Recursive computation of satisfaction sets

$$\left.\begin{array}{lcl} Sat(true) & = & S \\ Sat(a) & = & \{s \in S : a \in L(s)\} \\ Sat(\Phi_1 \wedge \Phi_2) & = & Sat(\Phi_1) \cap Sat(\Phi_2) \\ Sat(\neg\Phi) & = & S \setminus Sat(\Phi) \end{array}\right\} \text{ as for } \mathbf{CTL}$$

Recursive computation of satisfaction sets

$$\left.\begin{aligned} Sat(true) &= S \\ Sat(a) &= \{s \in S : a \in L(s)\} \\ Sat(\Phi_1 \wedge \Phi_2) &= Sat(\Phi_1) \cap Sat(\Phi_2) \\ Sat(\neg\Phi) &= S \setminus Sat(\Phi) \end{aligned}\right\}$$ as for **CTL**

$$Sat(\forall\varphi) = Sat_{LTL}(\varphi) \quad \}$$ using an **LTL** model checker

Recursive computation of satisfaction sets

$$\left.\begin{array}{lcl} Sat(true) & = & S \\ Sat(a) & = & \{s \in S : a \in L(s)\} \\ Sat(\Phi_1 \wedge \Phi_2) & = & Sat(\Phi_1) \cap Sat(\Phi_2) \\ Sat(\neg\Phi) & = & S \setminus Sat(\Phi) \end{array}\right\} \text{ as for } \mathbf{CTL}$$

$$\left.\begin{array}{lcl} Sat(\forall\varphi) & = & Sat_{LTL}(\varphi) \\ Sat(\exists\varphi) & = & S \setminus Sat_{LTL}(\neg\varphi) \end{array}\right\} \text{ using an } \mathbf{LTL} \text{ model checker}$$

Example: CTL* model checking

$$\Phi = \exists\Diamond\Box a \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \neg\exists(a \,\mathsf{U}\, b)\big)$$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\Big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\Big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\Big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\Big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\Box\big(\bigcirc b \wedge \Diamond a_2\big)$$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\text{LTL formula } \varphi}$$

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\Big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\Big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\Big(\bigcirc b \wedge \Diamond a_2\Big)}_{\textbf{LTL} \text{ formula } \varphi} = a_1 \wedge \exists\varphi$$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\textbf{LTL}\text{ formula } \varphi} = a_1 \wedge \exists\varphi$$

3. use an **LTL** model checker to compute $Sat(\exists\varphi)$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\text{LTL formula } \varphi} = a_1 \wedge \exists\varphi$$

3. use an **LTL** model checker to compute $Sat(\exists\varphi)$

more precisely: existential **LTL** model checker

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\textbf{LTL}\text{ formula } \varphi} = a_1 \wedge \exists\varphi$$

3. use an **LTL** model checker to compute $Sat(\exists\varphi)$

more precisely: existential **LTL** model checker

1. construct an **NBA** for φ
2. check via nested DFS whether $\mathcal{T} \otimes \mathcal{A} \models \exists\Box\Diamond F$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\Big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\Big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\textbf{LTL}\text{ formula } \varphi} = a_1 \wedge \exists\varphi$$

3. compute $Sat(\exists\varphi)$ via NBA $\mathcal{A}$ for φ and nested DFS in $\mathcal{T} \otimes \mathcal{A}$

Example: CTL* model checking

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \, \mathsf{U} \, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\textbf{LTL formula } \varphi} = a_1 \wedge \exists\varphi$$

3. compute $Sat(\exists\varphi)$ via NBA $\mathcal{A}$ for φ and nested DFS in $\mathcal{T} \otimes \mathcal{A}$
4. return $Sat(\Phi) = Sat(a_1 \wedge \exists\varphi)$

$$\Phi = \underbrace{\exists\Diamond\Box a}_{\Phi_1} \wedge \exists\Box\big(\bigcirc b \wedge \Diamond \underbrace{\neg\exists(a \,\mathsf{U}\, b)}_{\Phi_2}\big)$$

1. calculate recursively the satisfaction sets $Sat(\Phi_i)$
2. replace Φ_i with the atomic proposition a_i, $i = 1, 2$

$$\Phi \rightsquigarrow a_1 \wedge \exists\underbrace{\Box\big(\bigcirc b \wedge \Diamond a_2\big)}_{\textbf{LTL}\text{ formula } \varphi} = a_1 \wedge \exists\varphi$$

3. compute $Sat(\exists\varphi)$ via NBA $\mathcal{A}$ for φ and nested DFS in $\mathcal{T} \otimes \mathcal{A}$
4. return $Sat(\Phi) = Sat(a_1 \wedge \exists\varphi) = Sat(\Phi_1) \cap Sat(\exists\varphi)$

Fairness in CTL*

Correct or wrong?

Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

Correct or wrong?

Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

CTL with fairness → $\models_{fair}$

CTL* semantic → $\models$

Correct or wrong?

Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

↑ **CTL*** path formula

Correct or wrong?

Let $fair = \bigwedge_{1 \le i \le k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

CTL* path formula

correct.

Correct or wrong?

Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$

correct.

$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \wedge \Box a)$

Let $fair = \bigwedge_{1 \le i \le k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

correct.

$$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \wedge \Box a)$$

wrong.

Correct or wrong?

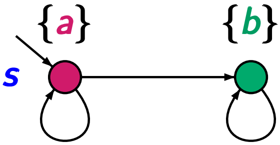
Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

correct.

$$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \wedge \Box a)$$

wrong.



$fair = \Box\Diamond\neg b$

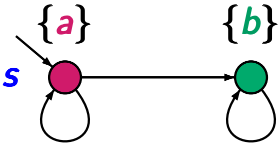
Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

correct.

$$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \wedge \Box a)$$

wrong.



$fair = \Box\Diamond\neg b$

$s \models_{fair} \forall\Box a$

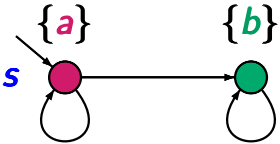
Let $fair = \bigwedge_{1 \le i \le k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

correct.

$$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \wedge \Box a)$$

wrong.



$fair = \Box\Diamond\neg b$

$s \models_{fair} \forall\Box a$

$s \not\models \forall(fair \wedge \Box a)$

Correct or wrong?

Let $fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond c_i$ be an unconditional **LTL** fairness assumption

$$s \models_{fair} \exists\Box a \quad \text{iff} \quad s \models \exists(fair \wedge \Box a)$$

correct.

$$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \wedge \Box a)$$

wrong. But we have:

$$s \models_{fair} \forall\Box a \quad \text{iff} \quad s \models \forall(fair \rightarrow \Box a)$$

FairCTL*

CTL* fairness assumptions are conjunctions of **CTL*** path formulas of the type

$\Box\Diamond\Phi$	unconditional fairness
$\Box\Diamond\Psi \to \Box\Diamond\Phi$	strong fairness
$\Diamond\Box\Psi \to \Box\Diamond\Phi$	weak fairness

FairCTL*

CTL* fairness assumptions are conjunctions of **CTL*** path formulas of the type

$\Box\Diamond\Phi$	unconditional fairness
$\Box\Diamond\Psi \rightarrow \Box\Diamond\Phi$	strong fairness
$\Diamond\Box\Psi \rightarrow \Box\Diamond\Phi$	weak fairness

where Φ and Ψ are **CTL*** state formulas

FairCTL*

CTL* fairness assumptions are conjunctions of **CTL*** path formulas of the type

$\Box\Diamond\Phi$	unconditional fairness
$\Box\Diamond\Psi \rightarrow \Box\Diamond\Phi$	strong fairness
$\Diamond\Box\Psi \rightarrow \Box\Diamond\Phi$	weak fairness

where Φ and Ψ are **CTL*** state formulas

obvious definition of the satisfaction relation $\models_{fair}$

Satisfaction relation $\models_{fair}$ for CTL*

$s \models_{fair} \exists\varphi$ iff there exists $\pi \in Paths(s)$ with $\pi \models fair$ and $\pi \models_{fair} \varphi$

$\models$ standard **CTL*** satisfaction relation

Satisfaction relation $\models_{fair}$ for CTL*

$s \models_{fair} \exists\varphi$ iff there exists $\pi \in Paths(s)$
with $\pi \models fair$ and $\pi \models_{fair} \varphi$

$s \models_{fair} \forall\varphi$ iff for all $\pi \in Paths(s)$:
if $\pi \models fair$ then $\pi \models_{fair} \varphi$

$\models$ standard **CTL*** satisfaction relation

Satisfaction relation $\models_{fair}$ for CTL*

$s \models_{fair} \exists\varphi$ iff there exists $\pi \in Paths(s)$ with $\pi \models fair$ and $\pi \models_{fair} \varphi$

iff $s \models \exists(fair \wedge \varphi)$

$s \models_{fair} \forall\varphi$ iff for all $\pi \in Paths(s)$: if $\pi \models fair$ then $\pi \models_{fair} \varphi$

$\models$ standard **CTL*** satisfaction relation

Satisfaction relation $\models_{fair}$ for CTL*

$s \models_{fair} \exists\varphi$ iff there exists $\pi \in Paths(s)$ with $\pi \models fair$ and $\pi \models_{fair} \varphi$

iff $s \models \exists(fair \wedge \varphi)$ ← if φ is quantifier-free

$s \models_{fair} \forall\varphi$ iff for all $\pi \in Paths(s)$: if $\pi \models fair$ then $\pi \models_{fair} \varphi$

$\models$ standard **CTL*** satisfaction relation

Satisfaction relation $\models_{fair}$ for CTL*

$s \models_{fair} \exists\varphi$ iff there exists $\pi \in Paths(s)$ with $\pi \models fair$ and $\pi \models_{fair} \varphi$

iff $s \models \exists(fair \wedge \varphi)$ ← if φ is quantifier-free

$s \models_{fair} \forall\varphi$ iff for all $\pi \in Paths(s)$: if $\pi \models fair$ then $\pi \models_{fair} \varphi$

iff $s \models \forall(fair \rightarrow \varphi)$ ← if φ is quantifier-free

$\models$ standard **CTL*** satisfaction relation

Complexity of CTL/LTL/CTL* model checking

Complexity of CTL/LTL model checking

	CTL	LTL	
		PSPACE-complete	
$\models$	$size(\mathcal{T}) \cdot \lvert\Phi\rvert$	$size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert)$	

Complexity of CTL/LTL model checking

	CTL	LTL	
	PTIME-complete	*PSPACE*-complete	
$\models$	$size(\mathcal{T}) \cdot \lvert\Phi\rvert$	$size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert)$	

Complexity of CTL/LTL model checking

	CTL	LTL
	PTIME-complete	*PSPACE*-complete
$\models$	$size(\mathcal{T}) \cdot \lvert\Phi\rvert$	$size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert)$
$\models_{fair}$	$size(\mathcal{T}) \cdot \lvert\Phi\rvert \cdot \lvert fair\rvert$	$size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert) \cdot \lvert fair\rvert$

Complexity of CTL/LTL/CTL* model checking

	CTL	LTL	CTL*
	PTIME-complete	*PSPACE*-complete	?
$\models$	$size(\mathcal{T}) \cdot \lvert\Phi\rvert$	$size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert)$	?
$\models_{fair}$	$size(\mathcal{T}) \cdot \lvert\Phi\rvert \cdot \lvert fair\rvert$	$size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert) \cdot \lvert fair\rvert$	?

Complexity of CTL/LTL/CTL* model checking

	CTL	**LTL** and **CTL***
	PTIME-complete	*PSPACE*-complete
$\models$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \lvert\Phi\rvert)$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert))$
$\models_{\mathit{fair}}$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \lvert\Phi\rvert \cdot \lvert\mathit{fair}\rvert)$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert) \cdot \lvert\mathit{fair}\rvert)$

	CTL	**LTL** and **CTL***
	PTIME-complete	*PSPACE*-complete
$\models$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \lvert\Phi\rvert)$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert))$
$\models_{\mathit{fair}}$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \lvert\Phi\rvert \cdot \lvert\mathit{fair}\rvert)$	$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert) \cdot \lvert\mathit{fair}\rvert)$

model complexity, i.e., for fixed formula:
$\mathcal{O}(\mathit{size}(\mathcal{T}))$

correct or wrong?

$$\exists(\Diamond a \ \wedge \ \Diamond b) \ \equiv \ \exists\Diamond(a \wedge \exists\Diamond b) \vee \exists\Diamond(b \wedge \exists\Diamond a)$$

$$\exists(\Diamond a \ \wedge \ \Diamond b) \ \equiv \ \exists\Diamond(a \wedge \exists\Diamond b) \vee \exists\Diamond(b \wedge \exists\Diamond a)$$

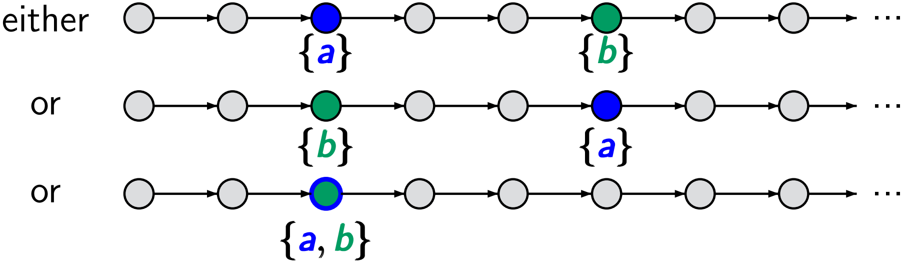
correct.

correct or wrong?

CTLST4.6-17

$$\exists(\Diamond a \wedge \Diamond b) \;\equiv\; \exists\Diamond(\, a \wedge \exists\Diamond b \,) \vee \exists\Diamond(\, b \wedge \exists\Diamond a \,)$$

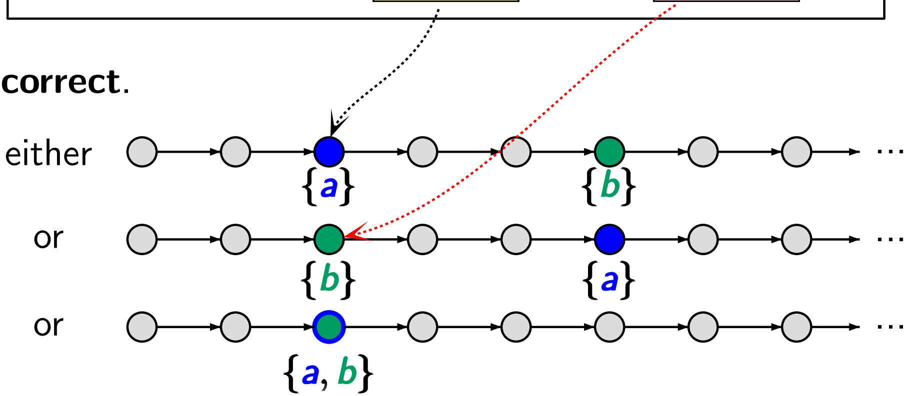
correct.



correct or wrong?

$$\exists(\Diamond a \wedge \Diamond b) \equiv \exists\Diamond(\, a \wedge \exists\Diamond b \,) \vee \exists\Diamond(\, b \wedge \exists\Diamond a \,)$$

correct.



The logic CTL$^+$

The logic CTL$^+$

- **CTL** with Boolean operators for path formulas

The logic CTL$^+$

- **CTL** with Boolean operators for path formulas
- sublogic of **CTL***

The logic CTL$^+$

- **CTL** with Boolean operators for path formulas
- sublogic of **CTL***

CTL$^+$ state formulas

$$\Phi ::= true \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

CTL$^+$ path formulas

$$\varphi ::= \ldots$$

The logic CTL^+

- **CTL** with Boolean operators for path formulas
- sublogic of **CTL***

CTL^+ state formulas

$$\Phi ::= \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi \mid \forall\varphi$$

CTL^+ path formulas

$$\varphi ::= \ldots$$

universal quantification can be derived: $\forall\varphi \overset{\text{def}}{=} \neg\exists\neg\varphi$

The logic CTL$^+$

- **CTL** with Boolean operators for path formulas
- sublogic of **CTL***

CTL$^+$ state formulas

$$\Phi ::= \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi \mid \forall\varphi$$

CTL$^+$ path formulas

$$\varphi ::= \bigcirc\Phi \mid \Phi_1 \,\mathsf{U}\, \Phi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi$$

The logic CTL$^+$

- **CTL** with Boolean operators for path formulas
- sublogic of **CTL***

CTL$^+$ state formulas

$$\Phi ::= \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi \mid \forall\varphi$$

CTL$^+$ path formulas

$$\varphi ::= \bigcirc\Phi \mid \Phi_1 \mathsf{U} \Phi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi$$

e.g., $\exists(\Diamond b \wedge \Box a)$

The logic CTL$^+$

- **CTL** with Boolean operators for path formulas
- sublogic of **CTL***

CTL$^+$ state formulas

$$\Phi ::= \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg \Phi \mid \exists \varphi \mid \forall \varphi$$

CTL$^+$ path formulas

$$\varphi ::= \bigcirc \Phi \mid \Phi_1 \, \mathsf{U} \, \Phi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg \varphi$$

e.g., $\exists(\Diamond b \wedge \Box a)$ and $\exists(\bigcirc b \rightarrow (a \, \mathsf{U} \, c))$
are **CTL$^+$** formulas

Expressiveness of CTL^+

Expressiveness of CTL$^+$

CTL$^+$ is as expressive as **CTL**, i.e.,

For each **CTL**$^+$-formula there exists an equivalent **CTL** formula.

CTL$^+$ is as expressive as **CTL**, i.e.,

For each **CTL**$^+$-formula there exists an equivalent **CTL** formula.

proof relies on a series of equivalence rules, e.g.:

Expressiveness of CTL^+

$\textbf{CTL}^+$ is as expressive as **CTL**, i.e.,

> For each $\textbf{CTL}^+$-formula there exists an equivalent **CTL** formula.

proof relies on a series of equivalence rules, e.g.:

$\exists(\neg\bigcirc\Phi) \rightsquigarrow \exists\bigcirc\neg\Phi$

CTL$^+$ is as expressive as **CTL**, i.e.,

For each **CTL**$^+$-formula there exists an equivalent **CTL** formula.

proof relies on a series of equivalence rules, e.g.:

$$\exists(\neg\bigcirc\Phi) \rightsquigarrow \exists\bigcirc\neg\Phi$$

$$\exists(\neg(\Phi_1 \mathsf{U} \Phi_2)) \rightsquigarrow \exists\big((\Phi_1 \wedge \Phi_2) \mathsf{U} (\neg\Phi_1 \wedge \neg\Phi_2)\big) \vee \exists\Box\neg\Phi_2$$

CTL$^+$ is as expressive as **CTL**, i.e.,

> For each **CTL**$^+$-formula there exists an equivalent **CTL** formula.

proof relies on a series of equivalence rules, e.g.:

$$\exists(\neg\bigcirc\Phi) \rightsquigarrow \exists\bigcirc\neg\Phi$$

$$\exists(\neg(\Phi_1 \mathsf{U} \Phi_2)) \rightsquigarrow \exists((\Phi_1 \wedge \Phi_2)\mathsf{U}(\neg\Phi_1 \wedge \neg\Phi_2)) \vee \exists\Box\neg\Phi_2$$

$$\exists((\Psi_1 \mathsf{U} \Psi_2) \wedge (\Phi_1 \mathsf{U} \Phi_2)) \rightsquigarrow \ldots$$

$$\exists(\bigcirc\Psi \wedge (\Phi_1 \mathsf{U} \Phi_2)) \rightsquigarrow \ldots$$

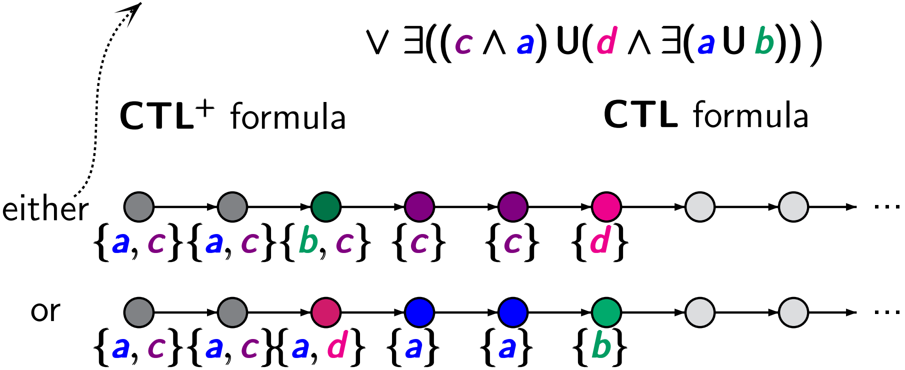
$$\exists((a \,\mathsf{U}\, b) \wedge (c \,\mathsf{U}\, d)) \equiv \exists\Big((a \wedge c) \,\mathsf{U}\, \big(b \wedge \exists(c \,\mathsf{U}\, d)\big)\Big) \vee \exists\Big((c \wedge a) \,\mathsf{U}\, \big(d \wedge \exists(a \,\mathsf{U}\, b)\big)\Big)$$

CTL$^+$ formula | **CTL** formula

$$\exists\big((a \,\mathsf{U}\, b) \wedge (c \,\mathsf{U}\, d)\big) \equiv \exists\big((a \wedge c)\,\mathsf{U}\,(b \wedge \exists(c \,\mathsf{U}\, d))\big) \vee \exists\big((c \wedge a)\,\mathsf{U}\,(d \wedge \exists(a \,\mathsf{U}\, b))\big)$$

CTL$^+$ formula

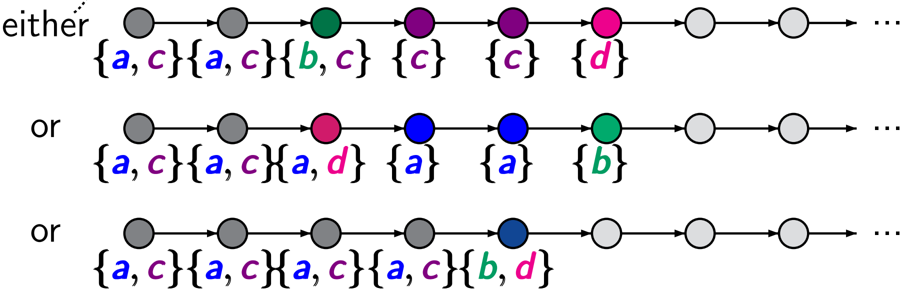
CTL formula



From CTL$^+$ to CTL

$$\exists\big((a \,\mathsf{U}\, b) \;\wedge\; (c \,\mathsf{U}\, d)\big) \;\equiv\; \exists\big((a \wedge c)\,\mathsf{U}\,(b \wedge \exists(c \,\mathsf{U}\, d))\big) \;\vee\; \exists\big((c \wedge a)\,\mathsf{U}\,(d \wedge \exists(a \,\mathsf{U}\, b))\big)$$

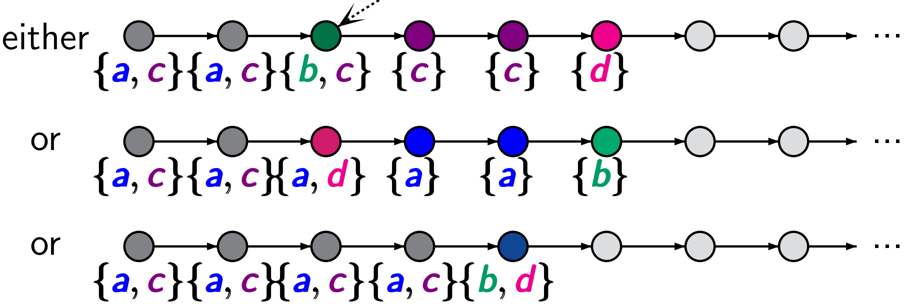
CTL$^+$ formula — **CTL** formula



From CTL$^+$ to CTL

$$\exists((a\,\mathsf{U}\,b) \;\wedge\; (c\,\mathsf{U}\,d)) \;\equiv\; \exists((a \wedge c)\,\mathsf{U}\,(b \wedge \exists(c\,\mathsf{U}\,d)))$$
$$\vee\; \exists((c \wedge a)\,\mathsf{U}\,(d \wedge \exists(a\,\mathsf{U}\,b)))$$

CTL$^+$ formula **CTL** formula



From CTL$^+$ to CTL

$$\exists(\bigcirc a \ \wedge\ (b \,\mathsf{U}\, c))$$

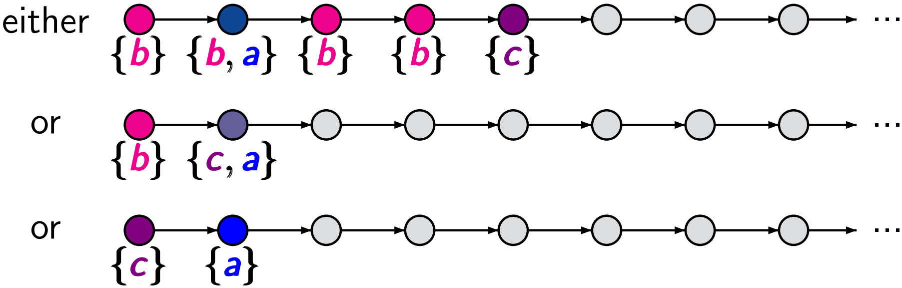
From CTL$^+$ to CTL

$$\exists(\bigcirc a \;\wedge\; (b \,\mathsf{U}\, c))$$

$$\equiv \quad (c \wedge \exists\bigcirc a) \vee \big(b \wedge \exists\bigcirc(a \wedge \exists(b \,\mathsf{U}\, c))\big)$$

From CTL$^+$ to CTL

$$\exists(\bigcirc a \ \wedge \ (b \, \mathsf{U} \, c))$$
$$\equiv \quad (c \wedge \exists\bigcirc a) \vee (b \wedge \exists\bigcirc(a \wedge \exists(b \, \mathsf{U} \, c)))$$



$$\exists(\bigcirc a \ \wedge\ (b \,\mathsf{U}\, c))$$

$$\equiv \quad (c \wedge \exists\bigcirc a) \vee \big(b \wedge \exists\bigcirc\big(\boxed{a \wedge \exists(b \,\mathsf{U}\, c)}\big)\big)$$