Overview

Introduction

Modelling parallel systems

Linear Time Properties

Regular Properties

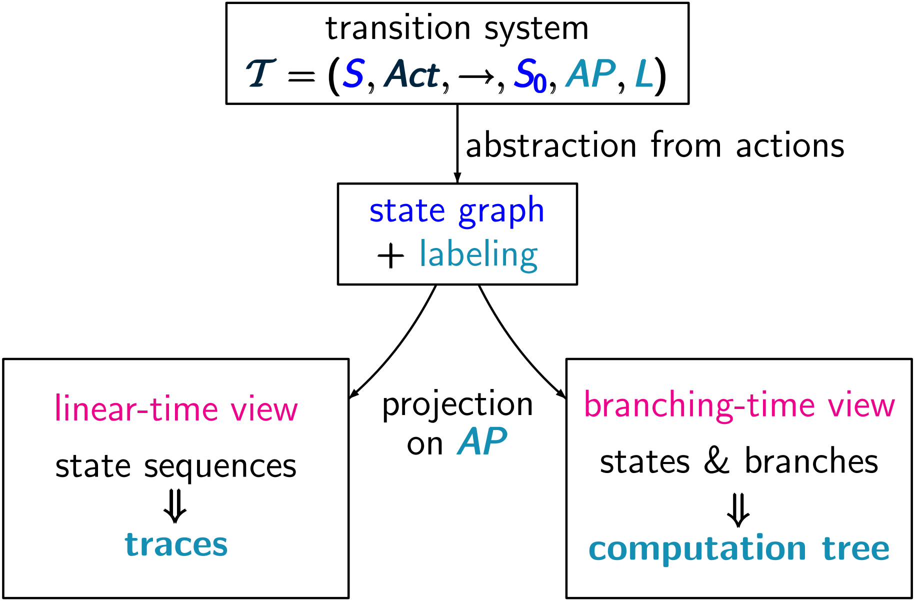
Linear Temporal Logic (LTL)

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

Linear vs branching time



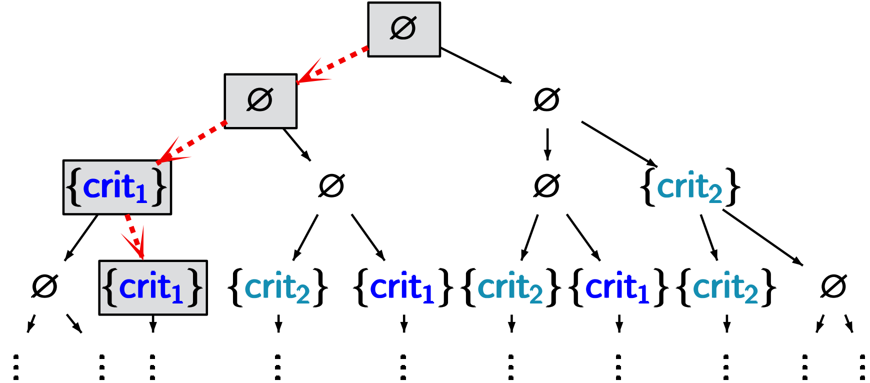
Computation tree

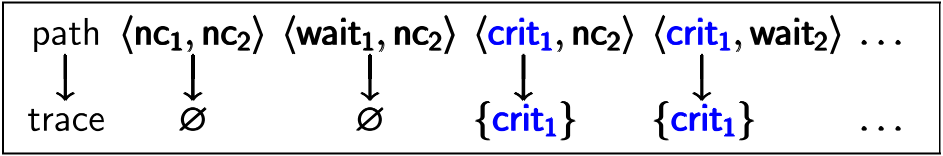
The computation tree of state s_0 in a transition system $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ arises by:

- unfolding $\mathcal{T}_{s_0} = (S, Act, \rightarrow, s_0, AP, L)$ into a tree
- abstraction from the actions
- projection of the states s to their labels $L(s) \subseteq AP$

Example: computation tree

mutual exclusion with semaphore and $AP = \{crit_1, crit_2\}$:





Linear vs. branching time

	linear time	branching time
behavior	path based traces	state based computation tree
temporal logic	**LTL** path formulas	**CTL** state formulas
model checking	PSPACE-complete $\mathcal{O}\big(size(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert)\big)$	PTIME $\mathcal{O}\big(size(\mathcal{T}) \cdot \lvert\Phi\rvert\big)$
impl. relation	trace inclusion trace equivalence PSPACE-complete	simulation bisimulation PTIME
fairness	can be encoded	requires special treatment

Computation Tree Logic (CTL)

CTL (state) formulas:

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

CTL path formulas:

$$\varphi \;::=\; \bigcirc\Phi \;\big|\; \Phi_1 \,\mathsf{U}\, \Phi_2$$

Computation Tree Logic (CTL)

CTL (state) formulas:

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

CTL path formulas:

$$\varphi ::= \bigcirc\Phi \mid \Phi_1 \, \mathsf{U} \, \Phi_2$$

eventually:

$\exists\Diamond\Phi \stackrel{\text{def}}{=} \exists(true\,\mathsf{U}\,\Phi)$

$\forall\Diamond\Phi \stackrel{\text{def}}{=} \forall(true\,\mathsf{U}\,\Phi)$

always:

$\exists\Box\Phi \stackrel{\text{def}}{=}$ **?**

note: $\exists\neg\Diamond\neg\Phi$ is <u>no</u> **CTL** formula

CTL (state) formulas:

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

CTL path formulas:

$$\varphi \quad ::= \quad \bigcirc\Phi \mid \Phi_1 \, \mathsf{U} \, \Phi_2$$

eventually:

$$\exists\Diamond\Phi \stackrel{\text{def}}{=} \exists(\textit{true}\,\mathsf{U}\,\Phi)$$

$$\forall\Diamond\Phi \stackrel{\text{def}}{=} \forall(\textit{true}\,\mathsf{U}\,\Phi)$$

always:

$$\exists\Box\Phi \stackrel{\text{def}}{=} \neg\forall\Diamond\neg\Phi$$

$$\forall\Box\Phi \stackrel{\text{def}}{=} \neg\exists\Diamond\neg\Phi$$

note: $\exists\neg\Diamond\neg\Phi$ is <u>no</u> **CTL** formula

Examples for CTL formulas

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:

$$\varphi \ ::= \ \bigcirc \Phi \mid \Phi_1 \, \mathsf{U} \, \Phi_2 \mid \Diamond \Phi \mid \Box \Phi$$

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

CTL (state) formulas:

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

CTL path formulas:

$$\varphi \quad ::= \quad \bigcirc\Phi \mid \Phi_1 \,\mathsf{U}\, \Phi_2 \mid \Diamond\Phi \mid \Box\Phi$$

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

"every request will be answered eventually"

$$\forall\Box(\mathit{request} \rightarrow \forall\Diamond \mathit{response})$$

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:

$$\varphi \;::=\; \bigcirc\Phi \mid \Phi_1 \mathsf{U} \Phi_2 \mid \Diamond\Phi \mid \Box\Phi$$

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

"every request will be answered eventually"

$\forall\Box(\mathit{request} \rightarrow \forall\Diamond\mathit{response})$

traffic lights $\forall\Box(\mathit{yellow} \rightarrow \forall\bigcirc\mathit{red})$

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:

$$\varphi \ ::= \ \bigcirc\Phi \mid \Phi_1 \,\mathsf{U}\, \Phi_2 \mid \Diamond\Phi \mid \Box\Phi$$

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

"every request will be answered eventually"

$\forall\Box(request \rightarrow \forall\Diamond response)$

traffic lights $\forall\Box(yellow \rightarrow \forall\bigcirc red)$

reset possiblity $\forall\Box\exists\Diamond start$

Examples for CTL formulas

CTL (state) formulas:

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

CTL path formulas:

$$\varphi \quad ::= \quad \bigcirc\Phi \mid \Phi_1 \, \mathsf{U} \, \Phi_2 \mid \Diamond\Phi \mid \Box\Phi$$

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

"every request will be answered eventually"

$$\forall\Box(\, request \rightarrow \forall\Diamond response\,)$$

traffic lights $\forall\Box(\, yellow \rightarrow \forall\bigcirc red\,)$

reset possiblity $\forall\Box\exists\Diamond start$

unconditional process fairness $\forall\Box\forall\Diamond crit_1 \wedge \forall\Box\forall\Diamond crit_2$

Alternative (standard) Syntax for CTL

E	**(∃)**	**there exists a path**
A	**(∀)**	**for all paths**
X	**( )**	**in the next state**
U		**until (strong) (W,V,R...)**
F	**(<>)**	**finally (eventually)**
G	**([])**	**generally (always)**

Examples

[mutual exclusion]

- ∀[ ](~crit1 ∨ ~crit2) AG(~crit1 ∨ ~crit2)

[every request will be answered eventually]

- ∀[](request → ∀<> response) AG(request → AE response)

[traffic light]

- ∀[](yellow → ∀○red) AG(yellow → AX red)

[reset possiblity]

- ∀[]∃<> start AG EF start

Semantics of CTL

define a satisfaction relation $\models$ for CTL formulas over $\boldsymbol{AP}$ and a given TS $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ without terminal states

- interpretation of state formulas over the states
- interpretation of path formulas over the paths (infinite path fragments)

Recall: semantics of LTL

for infinite path fragment $\pi = s_0\, s_1\, s_2\, \ldots$:

$\pi \models \mathit{true}$		
$\pi \models a$	iff	$s_0 \models a$,i.e., $a \in L(s_0)$
$\pi \models \varphi_1 \wedge \varphi_2$	iff	$\pi \models \varphi_1$ and $\pi \models \varphi_2$
$\pi \models \neg\varphi$	iff	$\pi \not\models \varphi$
$\pi \models \bigcirc\varphi$	iff	$\mathit{suffix}(\pi, 1) = s_1\, s_2\, s_3\, \ldots \models \varphi$
$\pi \models \varphi_1 \,\mathsf{U}\, \varphi_2$	iff	there exists $j \geq 0$ such that

$\mathit{suffix}(\pi, j) = s_j\, s_{j+1}\, s_{j+2}\, \ldots \models \varphi_2$ and

$\mathit{suffix}(\pi, k) = s_k\, s_{k+1}\, s_{k+2}\, \ldots \models \varphi_1$ for $0 \leq k < j$

Satisfaction relation for path formulas

Let $\pi = s_0\, s_1\, s_2\, \ldots$ be an infinite path fragment.

$\pi \models \bigcirc \Phi$	iff	$s_1 \models \Phi$
$\pi \models \Phi_1 \, \mathsf{U} \, \Phi_2$	iff	there exists $j \geq 0$ such that
		$s_j \models \Phi_2$
		$s_k \models \Phi_1$ for $0 \leq k < j$

semantics of derived operators:

$\pi \models \Diamond \Phi$	iff	there exists $j \geq 0$ with $s_j \models \Phi$
$\pi \models \Box \Phi$	iff	for all $j \geq 0$ we have: $s_j \models \Phi$

Satisfaction relation for state formulas

$s \models \mathit{true}$

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

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

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

$s \models \exists\varphi$ iff there is a path $\pi \in \mathit{Paths}(s)$ s.t. $\pi \models \varphi$

$s \models \forall\varphi$ iff for each path $\pi \in \mathit{Paths}(s)$: $\pi \models \varphi$

satisfaction set for state formula Φ:

$$\mathit{Sat}(\Phi) \stackrel{\text{def}}{=} \{s \in S : s \models \Phi\}$$

Interpretation of CTL formulas over a TS

satisfaction of state formulas over a TS $\mathcal{T}$:

$$\mathcal{T} \models \Phi \quad \text{iff} \quad S_0 \subseteq Sat(\Phi)$$
$$\quad\quad \text{iff} \quad s_0 \models \Phi \text{ for all initial states } s_0 \text{ of } \mathcal{T}$$

where S_0 is the set of initial states

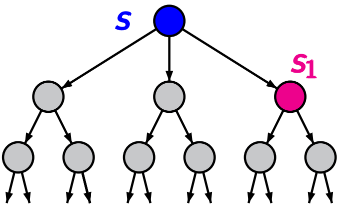
recall: $Sat(\Phi) = \{ s \in S : s \models \Phi \}$

Semantics of the next operator

$s \models \exists \bigcirc \Phi$ iff there exists $\pi = s\, s_1\, s_2 \ldots \in \mathit{Paths}(s)$ s.t. $\pi \models \bigcirc \Phi$, i.e., $s_1 \models \Phi$

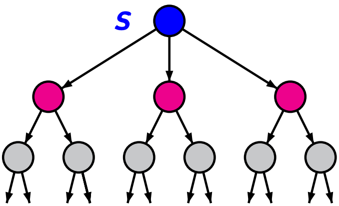
$s \models \forall \bigcirc \Phi$ iff for all $\pi = s\, s_1\, s_2 \ldots \in \mathit{Paths}(s)$: $\pi \models \bigcirc \Phi$, i.e., $s_1 \models \Phi$

$\exists \bigcirc \Phi$



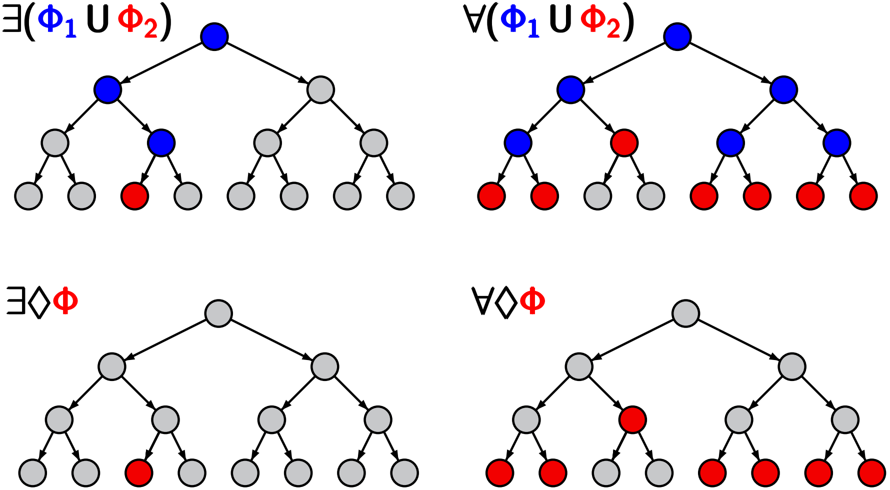
$\mathit{Post}(s) \cap \mathit{Sat}(\Phi) \neq \varnothing$

$\forall \bigcirc \Phi$



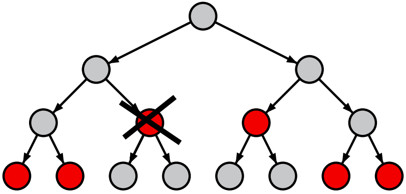
$\mathit{Post}(s) \subseteq \mathit{Sat}(\Phi)$

Semantics of until and eventually

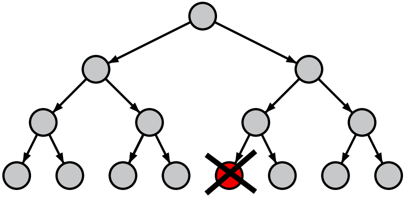


Semantics of eventually and always

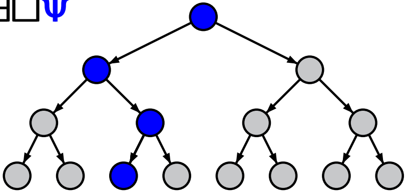
$\neg\forall\Diamond\Phi$



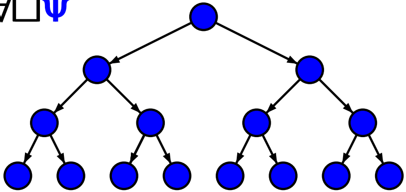
$\neg\exists\Diamond\Phi$



$\exists\Box\Psi$



$\forall\Box\Psi$



Specifying "infinitely often" in CTL

If s is a state in a TS and $a \in AP$ then:

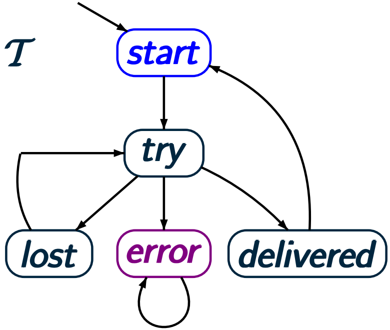
$$s \models_{\text{CTL}} \forall\Box\forall\Diamond a$$

iff for all paths $\pi = s_0\, s_1\, s_2 \ldots \in Paths(s)$:

$$\overset{\infty}{\exists}\, i \geq 0. \quad \text{s.t.} \quad s_i \models a$$

iff $s \models_{\text{LTL}} \Box\Diamond a$

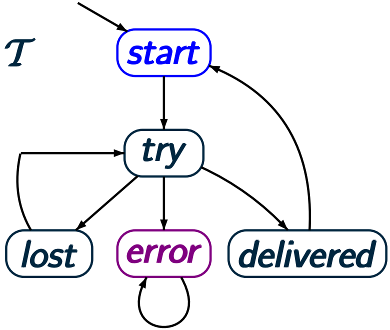
Example: CTL semantics



$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$?

$\Phi_1 = \exists\Diamond\forall\Box\neg start$

Example: CTL semantics

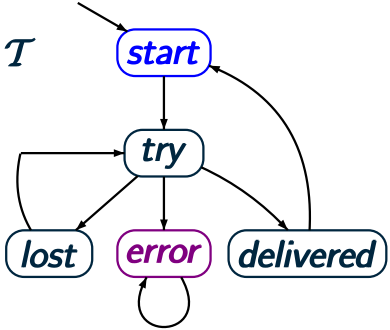


$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$ **?**

$$\Phi_1 = \exists\Diamond \boxed{\forall\Box\neg start}$$

$$Sat(\forall\Box\neg start) = \{error\}$$

Example: CTL semantics



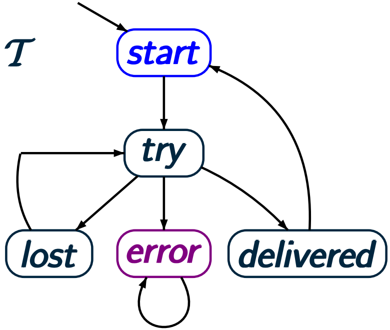
$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$?

$\Phi_1 = \exists\Diamond \boxed{\forall\Box\neg start} \rightsquigarrow \exists\Diamond \boxed{error}$

$Sat(\forall\Box\neg start) = \{error\}$

$Sat(\exists\Diamond\forall\Box\neg start) =$?

Example: CTL semantics

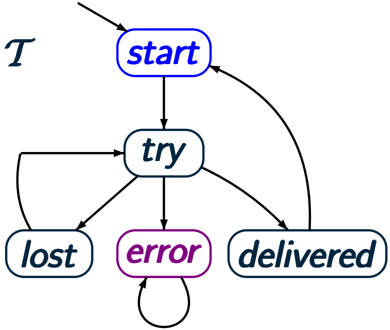


$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$ ✓

$\Phi_1 = \exists\Diamond \boxed{\forall\Box\neg start} \rightsquigarrow \exists\Diamond \boxed{error}$

$Sat(\forall\Box\neg start) = \{error\}$

$Sat(\exists\Diamond\forall\Box\neg start) = Sat(\exists\Diamond error) =$ "all states"

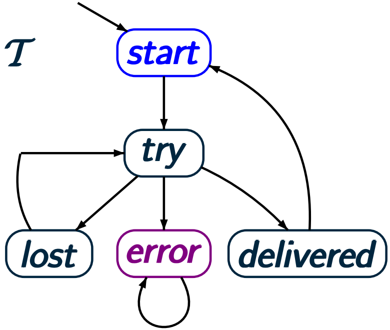


$\Phi_2 = \forall\bigcirc\exists\bigcirc\forall\Box\neg start$

$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$

$\mathcal{T} \models \forall\bigcirc\exists\bigcirc\forall\Box\neg start$ **?**

Example: CTL semantics



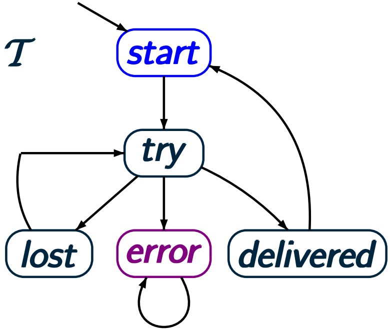
$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$

$\mathcal{T} \models \forall\bigcirc\exists\bigcirc\forall\Box\neg start$ **?**

$\Phi_2 = \forall\bigcirc\exists\bigcirc \boxed{\forall\Box\neg start}$

$$Sat(\forall\Box\neg start) = \{error\}$$

Example: CTL semantics



$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$

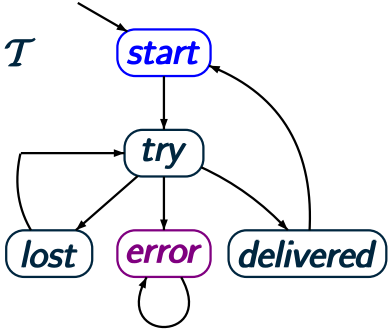
$\mathcal{T} \models \forall\bigcirc\exists\bigcirc\forall\Box\neg start$ **?**

$\Phi_2 = \forall\bigcirc\ \boxed{\exists\bigcirc\forall\Box\neg start} \quad \rightsquigarrow \quad \forall\bigcirc\ \boxed{\exists\bigcirc error}$

$$Sat(\forall\Box\neg start) = \{error\}$$

$$Sat(\exists\bigcirc\forall\Box\neg start) = \{error, try\}$$

Example: CTL semantics

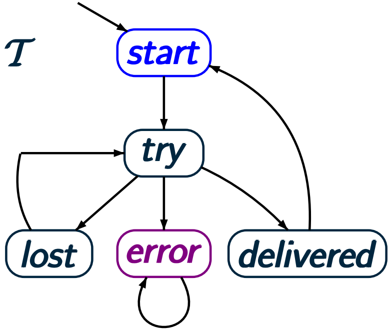


$\mathcal{T} \models \exists\Diamond\forall\Box\neg start$

$\mathcal{T} \models \forall\bigcirc\exists\bigcirc\forall\Box\neg start$ **?**

$\Phi_2 = \forall\bigcirc\exists\bigcirc\forall\Box\neg start \quad \rightsquigarrow \quad \boxed{\forall\bigcirc(error \vee try)}$

$$
\begin{aligned}
Sat(\forall\Box\neg start) &= \{error\} \\
Sat(\exists\bigcirc\forall\Box\neg start) &= \{error, try\} \\
Sat(\forall\bigcirc\exists\bigcirc\forall\Box\neg start) &= \textbf{?}
\end{aligned}
$$



$\mathcal{T} \not\models \exists\Diamond\forall\Box\neg start$

$\mathcal{T} \models \forall\bigcirc\exists\bigcirc\forall\Box\neg start$ ✓

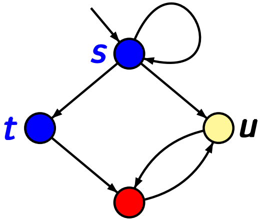
$\Phi_2 = \forall\bigcirc\exists\bigcirc\forall\Box\neg start \quad \rightsquigarrow \quad \forall\bigcirc(error \vee try)$

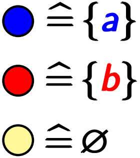
$$\begin{aligned} Sat(\forall\Box\neg start) &= \{error\} \\ Sat(\exists\bigcirc\forall\Box\neg start) &= \{error, try\} \\ Sat(\forall\bigcirc\exists\bigcirc\forall\Box\neg start) &= \{error, lost, start\} \end{aligned}$$

Example: CTL semantics

$\mathcal{T} \models \exists\Box\exists(a \,\mathsf{U}\, b)$?

Example: CTL semantics

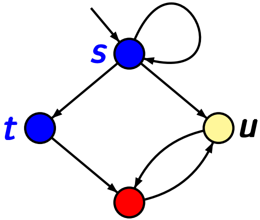


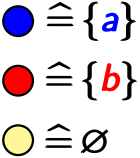


$\mathcal{T} \models \exists\Box\exists(a \,\mathsf{U}\, b)$ √ as $s\,s\,s\,\ldots \models \Box\exists(a \,\mathsf{U}\, b)$

$\mathcal{T} \models \exists((\exists\bigcirc a) \,\mathsf{U}\, b)$?

Example: CTL semantics



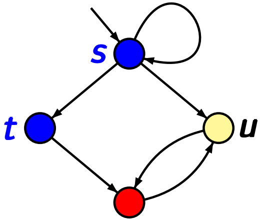


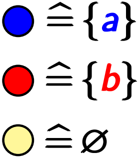
$\mathcal{T} \models \exists\Box\exists(a \,\mathsf{U}\, b)$ $\quad\surd\quad$ as $s\,s\,s\,\ldots \models \Box\exists(a \,\mathsf{U}\, b)$

$\mathcal{T} \not\models \exists((\exists\bigcirc a) \,\mathsf{U}\, b)$ $\quad$ as $t \not\models \exists\bigcirc a$, $u \not\models \exists\bigcirc a$

$\mathcal{T} \models \exists(a \,\mathsf{U}\, \forall(\neg a \,\mathsf{U}\, b))$ $\quad$?

Example: CTL semantics

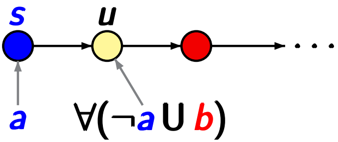




$\mathcal{T} \models \exists\Box\exists(a \,\mathsf{U}\, b)$ ✓ as $s\,s\,s \ldots \models \Box\exists(a \,\mathsf{U}\, b)$

$\mathcal{T} \not\models \exists((\exists\bigcirc a) \,\mathsf{U}\, b)$ as $t \not\models \exists\bigcirc a$, $u \not\models \exists\bigcirc a$

$\mathcal{T} \models \exists(a \,\mathsf{U}\, \forall(\neg a \,\mathsf{U}\, b))$ ✓



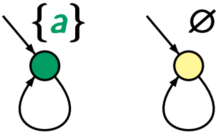
$\models a \,\mathsf{U}\, \forall(\neg a \,\mathsf{U}\, b)$

Let $\mathcal{T}$ be a transition system and Φ a CTL formula.
Is the following statement correct **?**

$$\text{if } \mathcal{T} \not\models \neg\Phi \text{ then } \mathcal{T} \models \Phi$$

answer: no

transition system $\mathcal{T}$ with 2 initial states:



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

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

Equivalence of CTL formulas

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

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

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

$$Sat(\Phi_1) = Sat(\Phi_2)$$

Examples:

$$\neg\neg\Phi \equiv \Phi$$

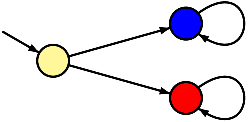
$$\neg(\Phi \wedge \Psi) \equiv \neg\Phi \vee \neg\Psi$$

$$\vdots$$

$$\neg\forall\bigcirc\Phi \equiv \exists\bigcirc\neg\Phi$$

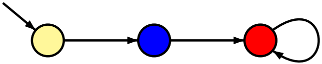
$$\exists\lozenge(a \wedge b) \;\equiv\; \exists\lozenge a \wedge \exists\lozenge b$$

wrong, e.g,



$$\forall\lozenge(a \wedge b) \;\equiv\; \forall\lozenge a \wedge \forall\lozenge b$$

wrong, e.g.,



but:

$$\forall\square(\Phi_1 \wedge \Phi_2) \;\equiv\; \forall\square\Phi_1 \;\wedge\; \forall\square\Phi_2$$
$$\exists\lozenge(\Phi_1 \vee \Phi_2) \;\equiv\; \exists\lozenge\Phi_1 \;\vee\; \exists\lozenge\Phi_2$$

Expansion laws

$$\exists(\Phi \,\mathsf{U}\, \Psi) \equiv \Psi \lor (\Phi \land \exists\bigcirc\exists(\Phi \,\mathsf{U}\, \Psi))$$

$$\forall(\Phi \,\mathsf{U}\, \Psi) \equiv \Psi \lor (\Phi \land \forall\bigcirc\forall(\Phi \,\mathsf{U}\, \Psi))$$

$$\exists\Diamond\Psi \equiv \Psi \lor \exists\bigcirc\exists\Diamond\Psi$$

$$\forall\Diamond\Psi \equiv \Psi \lor \forall\bigcirc\forall\Diamond\Psi$$

$$\exists(\Phi \,\mathsf{W}\, \Psi) \equiv \Psi \lor (\Phi \land \exists\bigcirc\exists(\Phi \,\mathsf{W}\, \Psi))$$

$$\forall(\Phi \,\mathsf{W}\, \Psi) \equiv \Psi \lor (\Phi \land \forall\bigcirc\forall(\Phi \,\mathsf{W}\, \Psi))$$

$$\exists\Box\Phi \equiv \Phi \land \exists\bigcirc\exists\Box\Phi$$

$$\forall\Box\Phi \equiv \Phi \land \forall\bigcirc\forall\Box\Phi$$

Duality laws

duality of $\Box$ and $\Diamond$:

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

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

self-duality of $\bigcirc$:

$$\forall\bigcirc\Phi \equiv \neg\exists\bigcirc\neg\Phi$$

$$\exists\bigcirc\Phi \equiv \neg\forall\bigcirc\neg\Phi$$

duality of **U** and **W**, e.g.:

$$\begin{aligned}\forall(\Phi \,\mathsf{U}\, \Psi) &\equiv \neg\exists((\Phi \wedge \neg\Psi) \,\mathsf{W}\, (\neg\Phi \wedge \neg\Psi)) \\ &\equiv \neg\exists((\neg\Psi) \,\mathsf{W}\, (\neg\Phi \wedge \neg\Psi)) \\ &\equiv \neg\exists((\neg\Psi) \,\mathsf{U}\, (\neg\Phi \wedge \neg\Psi)) \;\wedge\; \neg\exists\Box\neg\Psi\end{aligned}$$

Existential normalform for CTL

For each **CTL** formula Ψ there is an equivalent **CTL** formula Φ built by

- operators of propositional logic
- the modalities $\exists\bigcirc$, $\exists\mathsf{U}$ and $\exists\Box$.

$$\Phi \ ::= \ \mathit{true} \ \big| \ a \ \big| \ \Phi_1 \wedge \Phi_2 \ \big| \ \neg\Phi \ \big| \ \exists\bigcirc\Phi \ \big| \ \exists(\Phi_1 \, \mathsf{U} \, \Phi_2) \ \big| \ \exists\Box\Phi$$

transformation $\Psi \rightsquigarrow \Phi$ relies on:

$$\forall\bigcirc\Psi \rightsquigarrow \neg\exists\bigcirc\neg\Psi$$

$$\forall(\Psi_1 \, \mathsf{U} \, \Psi_2) \rightsquigarrow \neg\exists(\neg\Psi_2 \, \mathsf{U}(\neg\Psi_1 \wedge \neg\Psi_2)) \wedge \neg\exists\Box\neg\Psi_2$$

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

Equivalence of CTL and LTL formulas

Let Φ be a **CTL** formula and φ an **LTL** formula.

> $\Phi \equiv \varphi$ iff for all transition systems $\mathcal{T}$ and all states s in $\mathcal{T}$:
>
> $$s \models_{\mathrm{CTL}} \Phi \iff s \models_{\mathrm{LTL}} \varphi$$

e.g.,

CTL formula Φ	LTL formula φ
a	a
$\forall \bigcirc a$	$\bigcirc a$
$\forall(a \, \mathsf{U} \, b)$	$a \, \mathsf{U} \, b$

$a, b \in AP$

More examples

CTL formula Φ	LTL formula φ
a	a
$\forall \bigcirc a$	$\bigcirc a$
$\forall (a \, \mathsf{U} \, b)$	$a \, \mathsf{U} \, b$
$\forall \Box a$	$\Box a$
$\forall \Diamond a$	$\Diamond a$
$\forall (a \, \mathsf{W} \, b)$	$a \, \mathsf{W} \, b$
$\forall \Box \forall \Diamond a$	$\Box \Diamond a$

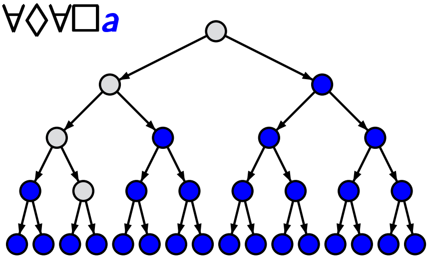
but: $\forall \Diamond \forall \Box a \not\equiv \Diamond \Box a$

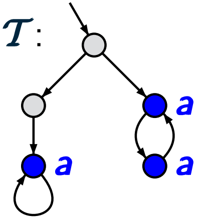
$\nwarrow$ infinitely often a $\nearrow$

The CTL formula $\forall\Diamond\forall\Box a$

$s \models \forall\Diamond\forall\Box a$ iff on each path π from s there is a state t with $t \models \forall\Box a$

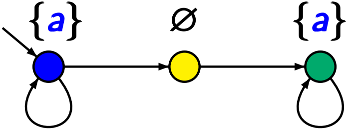
i.e., all states in the computation tree of t fulfill a





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

transition system $\mathcal{T}$

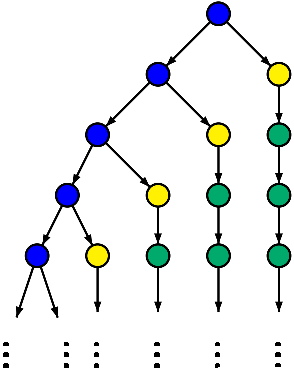


$\mathcal{T} \models_{\text{LTL}} \Diamond\Box a$

$\mathcal{T} \not\models_{\text{CTL}} \forall\Diamond\forall\Box a$

$Sat(\forall\Box a) = \{$●$\}$

computation tree



From CTL to LTL, if possible

For each **CTL** formula Φ the following holds:

- either there is **no** equivalent LTL formula
- or $\Phi \equiv \varphi$

  where φ is the **LTL** formula obtained from Φ by removing of all path quantifiers $\exists$ and $\forall$

without proof

$\Phi = \forall\Diamond\forall\Box a$

$\downarrow$

$\varphi = \Diamond\Box a \not\equiv \Phi$

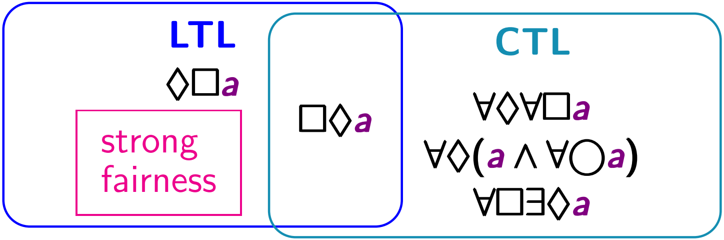
hence: there is no LTL formula equivalent to Φ

Expressiveness of LTL and CTL

The expressive powers of **LTL** and **CTL** are incomparable

- The **CTL** formulas $\forall\Diamond(a \wedge \forall\bigcirc a)$, $\forall\Diamond\forall\Box a$ and $\forall\Box\exists\Diamond a$ have no equivalent LTL formula

- The **LTL** formula $\Diamond\Box a$ has no equivalent CTL formula



LTL formula $\Diamond\Box a$

There is no **CTL** formula which is equivalent to the **LTL** formula $\Diamond\Box a$

Proof (sketch): provide sequences $(\mathcal{T}_n)_{n\geq 0}$, $(\mathcal{T}'_n)_{n\geq 0}$ of transition systems such that for all $n \geq 0$:

(1) $\mathcal{T}_n \not\models \Diamond\Box a$

(2) $\mathcal{T}'_n \models \Diamond\Box a$

(3) $\mathcal{T}_n$ and $\mathcal{T}'_n$ satisfy the same **CTL** formulas length $\leq n$

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 model checking

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

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

idea:

- compute $Sat(\Phi) = \{s \in S : s \models \Phi\}$
- check whether $S_0 \subseteq Sat(\Phi)$

CTL model checking

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

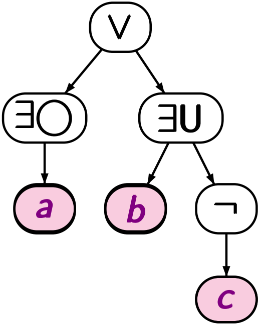
question: does $\mathcal{T} \models \Phi$ hold ?

inner subformulas first

```
FOR ALL subformulas Ψ of Φ DO
     compute Sat(Ψ)
     replace Ψ by a new atomic proposition a_Ψ
     FOR ALL s ∈ Sat(Ψ) DO add a_Ψ to L(s) OD
OD
IF S_0 ⊆ Sat(Φ) THEN output "yes"
                ELSE output "no"
FI
```

$\Phi = \quad \exists\bigcirc a \ \vee \exists(b \, \mathsf{U} \, \neg c)$

syntax tree for Φ



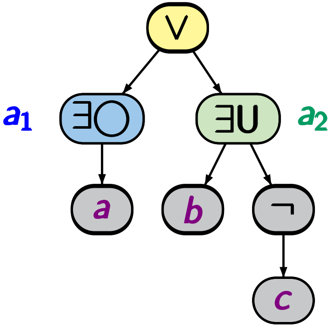
processed in bottom-up fashion

compute $Sat(a), Sat(b), Sat(c)$

Example: CTL model checking

$$\Phi = \underbrace{\exists \bigcirc a}_{\Phi_1} \vee \underbrace{\exists(b \, \mathsf{U} \, \neg c)}_{\Phi_2} \quad \rightsquigarrow \quad a_1 \vee a_2$$

syntax tree for Φ



processed in bottom-up fashion

compute $Sat(a), Sat(b), Sat(c)$

$Sat(\Phi_1) = \ldots = Sat(a_1)$

$Sat(\neg c) = S \setminus Sat(c)$

$Sat(\Phi_2) = \ldots = Sat(a_2)$

replace Φ_1 with a_1

replace Φ_2 with a_2

$Sat(\Phi) = Sat(a_1) \cup Sat(a_2)$

CTL model checking

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

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

method: regard in bottom-up manner all subformulas Ψ of Φ and compute their satisfaction sets

$$Sat(\Psi) = \{s \in S : s \models \Psi\}$$

here: explanations for the case that Φ is in existential normal form

analogous algorithms can be designed for standard CTL (and the derived operators)

Recall: Existential normal form for CTL

For each **CTL** formula there is an equivalent formula in **∃-normal form**, i.e., a **CTL** formula with the basis modalities $\exists\bigcirc$, $\exists\mathsf{U}$, $\exists\Box$.

CTL formulas in ∃-normal form:

$$\Psi \quad ::= \quad \mathit{true} \mid a \mid \neg\Psi \mid \Psi_1 \wedge \Psi_2 \mid$$
$$\exists\bigcirc\Psi \mid \exists(\Psi_1 \,\mathsf{U}\, \Psi_2) \mid \exists\Box\Psi$$

CTL formula ⇝ **CTL** formula in ∃-normal form

$$\forall\bigcirc\Phi \rightsquigarrow \neg\exists\bigcirc\neg\Phi$$

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

Recursive computation of the satisfaction sets

$$
\begin{aligned}
Sat(\mathit{true}) &= S \\
Sat(a) &= \{s \in S : a \in L(s)\} \\
Sat(\neg\Phi) &= S \setminus Sat(\Phi) \\
Sat(\Phi_1 \wedge \Phi_2) &= Sat(\Phi_1) \cap Sat(\Phi_2) \\
Sat(\exists\bigcirc\Phi) &= \{s \in S : Post(s) \cap Sat(\Phi) \neq \varnothing\} \\
Sat(\exists(\Phi_1 \,\mathsf{U}\, \Phi_2)) &= \ \ldots \\
Sat(\exists\Box\Phi) &= \ \ldots
\end{aligned}
$$

treatment of $\exists\mathsf{U}$ and $\exists\Box$:

via fixed point computation

Least fixed point characterization of ∃U

$$\exists(\Phi_1 \mathsf{U} \Phi_2) \equiv \Phi_2 \vee (\Phi_1 \wedge \exists\bigcirc\exists(\Phi_1 \mathsf{U} \Phi_2))$$

$$Sat(\exists(\Phi_1 \mathsf{U} \Phi_2)) = Sat(\Phi_2) \cup \{s \in Sat(\Phi_1) : Post(s) \cap Sat(\exists(\Phi_1 \mathsf{U} \Phi_2)) \neq \varnothing\}$$

satisfies the following conditions:

(1) $Sat(\Phi_2) \subseteq Sat(\exists(\Phi_1 \mathsf{U} \Phi_2))$

(2) If $s \in Sat(\Phi_1)$ and $Post(s) \cap Sat(\exists(\Phi_1 \mathsf{U} \Phi_2)) \neq \varnothing$ then $s \in Sat(\exists(\Phi_1 \mathsf{U} \Phi_2))$

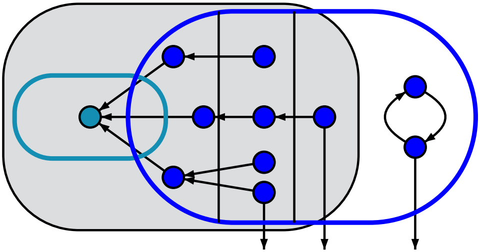
$Sat(\exists(\Phi_1 \mathsf{U} \Phi_2))$ is the smallest set s.t. (1) and (2) hold

CTL model checking: until operator

$$\exists(\Phi_1 \mathsf{U} \Phi_2) \equiv \Phi_2 \vee (\Phi_1 \wedge \exists\bigcirc\exists(\Phi_1 \mathsf{U} \Phi_2))$$

$Sat(\exists(\Phi_1 \mathsf{U} \Phi_2))$ = least set T of states s.t.

$$Sat(\Phi_2) \cup \{s \in Sat(\Phi_1) : Post(s) \cap T \neq \varnothing\} \subseteq T$$



$Sat(\exists(\Phi_1 \mathsf{U} \Phi_2))$

CTL model checking: always operator

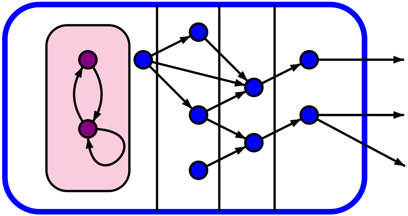
expansion law: $\exists\Box\Phi \equiv \Phi \wedge \exists\bigcirc\exists\Box\Phi$

$Sat(\exists\Box\Phi)$ = greatest set T of states with

$$T \subseteq \{ s \in Sat(\Phi) : Post(s) \cap T \neq \varnothing \}$$

$$T_0 := Sat(\Phi), \qquad T_{n+1} := \{ s \in T_n : Post(s) \cap T_n \neq \varnothing \}$$

$Sat(\Phi)$



$$Sat(\Phi_1 \wedge \Phi_2) = Sat(\Phi_1) \cap Sat(\Phi_2)$$

$$Sat(\neg\Phi) = S \setminus Sat(\Phi)$$

$$Sat(\exists\bigcirc\Phi) = \{s \in S : Post(s) \cap Sat(\Phi) = \varnothing\}$$

$Sat(\exists(\Phi_1 \mathsf{U} \Phi_2))$ = smallest set T of states s.t.

- $Sat(\Phi_2) \subseteq T$
- $s \in Sat(\Phi_1)$ and $Post(s) \cap T \neq \varnothing \implies s \in T$

$Sat(\exists\Box\Phi)$ = greatest set V of states s.t.

- $V \subseteq Sat(\Phi)$
- $s \in V \implies Post(s) \cap V \neq \varnothing$

CTL model checking: treatment of ∃U

compute $Sat(\exists(\Phi_1 \, \mathsf{U} \, \Phi_2))$ via an enumerative backward search

$T := Sat(\Phi_2)$ ⟵ collects all states $s \models \exists(\Phi_1 \, \mathsf{U} \, \Phi_2)$

$E := Sat(\Phi_2)$ ⟵ set of states still to be expanded

WHILE $E \neq \varnothing$ DO
    select a state $s' \in E$ and remove s' from E
    FOR ALL $s \in Pre(s')$ DO
        IF $s \in Sat(\Phi_1) \setminus T$ THEN add s to T and E FI
    OD
OD

return T

complexity: $\mathcal{O}(size(\mathcal{T}))$

Computation of $Sat(\exists\Box\Phi)$

$T := Sat(\Phi)$ ← organizes the candidates for $s \models \exists\Box\Phi$

$E := S \setminus T$ ← set of states to be expanded

```
WHILE E ≠ ∅ DO
   pick a state s' ∈ E and remove s' from E
   FOR ALL s ∈ Pre(s') DO
    IF s ∈ T and Post(s) ∩ T = ∅ THEN
         remove s from T and add s to E
    FI
OD
return T
```

naïve implementation:
quadratic time complexity

Computation of $Sat(\exists\Box\Phi)$ using counters

$T := Sat(\Phi)$; $E := S \setminus T$

FOR ALL $s \in Sat(\Phi)$ DO $c[s] := |Post(s)|$ OD

loop invariant: $c[s] = |Post(s) \cap (T \cup E)|$ for $s \in T$

```
WHILE E ≠ ∅ DO
   pick a state s' ∈ E and remove s' from E
   FOR ALL s ∈ Pre(s') DO
     IF s ∈ T THEN
          c[s] := c[s] − 1
          IF c[s] = 0 THEN
                remove s from T and add s to E FI
   FI
OD
```

complexity: $\mathcal{O}(size(\mathcal{T}))$

Recursive computation of $Sat(\ldots)$

$$Sat(\Phi_1 \wedge \Phi_2) = Sat(\Phi_1) \cap Sat(\Phi_2)$$

$$Sat(\neg\Phi) = S \setminus Sat(\Phi)$$

$$Sat(\exists\bigcirc\Phi) = \{s \in S : Post(s) \cap Sat(\Phi) = \varnothing\}$$

$$Sat(\exists(\Phi_1 \,\mathsf{U}\, \Phi_2)) = \bigcup_{n \geq 0} T_n \text{ where}$$

$$T_0 = Sat(\Phi_2)$$

$$T_{n+1} = \{s \in Sat(\Phi_1) : Post(s) \cap T_n \neq \varnothing\}$$

$$Sat(\exists\Box\Phi) = \bigcap_{n \geq 0} V_n \text{ where}$$

$$V_0 = Sat(\Phi); \quad V_{n+1} = \{s \in V_n : Post(s) \cap V_n \neq \varnothing\}$$

Complexity of CTL/LTL model checking

CTL model checking: $\mathcal{O}\big(\mathit{size}(\mathcal{T}) \cdot |\Phi|\big)$

LTL model checking: $\mathcal{O}\big(\mathit{size}(\mathcal{T}) \cdot \exp(|\varphi|)\big)$

model complexity, i.e., for fixed specification:

CTL and **LTL**: $\mathcal{O}\big(\mathit{size}(\mathcal{T})\big)$

If $\Phi \equiv \varphi$ then "often" we have: $|\Phi| = \exp(|\varphi|)$

If $P \neq NP$ then there is a sequence $(\varphi_n)_{n \geq 0}$ of **LTL** formulas such that:

- $|\varphi_n| = \mathcal{O}(poly(n))$
- φ_n has an equivalent **CTL** formula, but no equivalent **CTL** formula of polynomial length

Proof. Let $\varphi_n = \neg\varphi'_n$. Then:

- $|\varphi_n| = |\varphi'_n| + 1 = \mathcal{O}(n^2)$
- φ_n has an equivalent **CTL** formula, e.g., $\neg\Phi'_n$

Suppose there is a **CTL** formula of polynomial length that is equivalent to φ_n. Then:

Hamilton path problem $\in P$ and $P = NP$

Overview

Complexity of CTL and LTL model checking

LTL model checking problem:

PSPACE-complete and solvable in time

$$\mathcal{O}(\textit{size}(\mathcal{T}) \cdot \exp(|\varphi|))$$

CTL model checking problem:

solvable in polynomial time (even PTIME-complete)

$$\mathcal{O}(\textit{size}(\mathcal{T}) \cdot |\Phi|)$$

Complexity of CTL and LTL model checking

LTL model checking problem:

PSPACE-complete and solvable in time

$$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(|\varphi|))$$

LTL with fairness: $\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(|\varphi|) \cdot |\mathit{fair}|)$

CTL model checking problem:

solvable in polynomial time (even PTIME-complete)

$$\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot |\Phi|)$$

CTL with fairness: $\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot |\Phi| \cdot |\mathit{fair}|)$

Recall: LTL fairness assumptions

are conjunctions of **LTL formulas** of the form

- unconditional fairness $\Box\Diamond\phi$
- strong fairness $\Box\Diamond\psi \rightarrow \Box\Diamond\phi$
- weak fairness $\Diamond\Box\psi \rightarrow \Box\Diamond\phi$

where ϕ, ψ are propositional formulas

Reduction of $\models_{fair}$ to $\models$

$\mathcal{T} \models_{fair} \varphi$ iff $\pi \models \varphi$ for all fair paths π in $\mathcal{T}$

iff for all paths π in $\mathcal{T}$:

$$\pi \models fair \rightarrow \varphi$$

CTL fairness assumptions

conjunctions of "formulas" of the type

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

where Ψ, Φ are CTL state formulas

note: CTL fairness assumptions

- are not CTL (state or path) formulas
- just a syntactic formalism to specify fairness assumptions

Satisfaction relation for CTL with fairness

$s \models_{fair} true$		
$s \models_{fair} a$	iff	$a \in L(s)$
$s \models_{fair} \neg\Phi$	iff	$s \not\models_{fair} \Phi$
$s \models_{fair} \Phi_1 \wedge \Phi_2$	iff	$s \models_{fair} \Phi_1$ and $s \models_{fair} \Phi_2$
$s \models_{fair} \exists\varphi$	iff	there exists $\pi \in Paths(s)$ with $\pi \models fair$ and $\pi \models_{fair} \varphi$
$s \models_{fair} \forall\varphi$	iff	for all $\pi \in Paths(s)$: $\pi \models fair$ implies $\pi \models_{fair} \varphi$

e.g., $s_0\, s_1\, s_2 \ldots \models \Box\Diamond\Phi$ iff $\overset{\infty}{\exists}\, i \geq 0$ s.t. $s_i \models \Phi$

Preprocessing of FairCTL model checking

given: finite transition system $\mathcal{T}$
CTL formula Φ in $\exists$-normal form
CTL fairness assumption *fair*, e.g.,

$$fair = \bigwedge_{1 \leq i \leq k} \Box\Diamond\Psi_{i,1} \rightarrow \Box\Diamond\Psi_{i,2}$$

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

preprocessing: apply a standard CTL model checker to evaluate the CTL state subformulas of *fair*

- compute $Sat(\Psi_{i,1})$ and $Sat(\Psi_{i,2})$
- replace $\Psi_{i,1}$ and $\Psi_{i,2}$ with fresh atomic propositions b_i and c_i, respectively

Idea of FairCTL model checking

given: finite transition system $\mathcal{T}$
CTL formula Φ in $\exists$-normal form
CTL fairness assumption ***fair***

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

1. ... preprocessing ...
2. Build the parse tree of Φ and process it in bottom-up-manner. Treatment of:
   - ***true***, ***a*** $\in$ ***AP***, $\wedge$, $\neg$: as for standard **CTL**
   - $\exists\bigcirc$, $\exists$**U**: via standard **CTL** model checking
   - $\exists\Box$: via analysis of **SCCs**

CTL fairness assumptions: formulas similar to **LTL**

e.g., $fair = \bigwedge_{1 \le i \le k} (\Box\Diamond\Psi_i \rightarrow \Box\Diamond\Phi_i)$

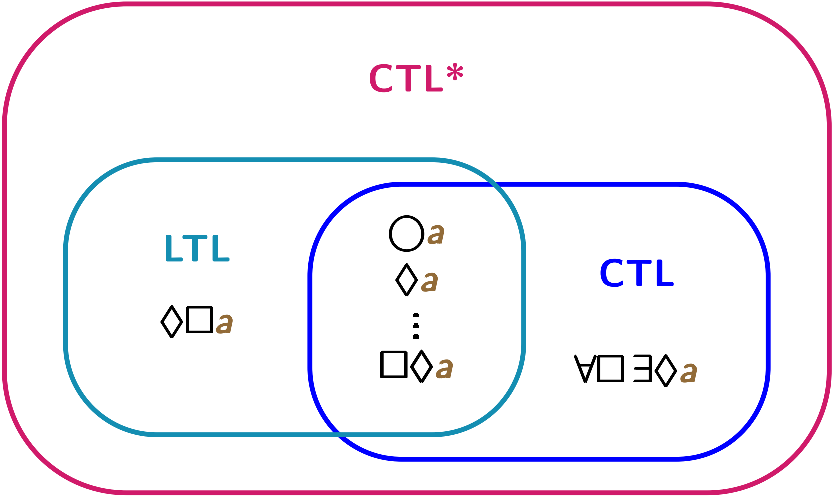
CTL satisfaction relation with fairness:

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

model checking for **CTL** with fairness:

- $\exists\bigcirc$, $\exists$**U**, $\forall\bigcirc$, $\forall\Box$ via **CTL** model checker
- analysis of **SCCs** for $\exists\Box$, $\forall$**U**
- complexity: $\mathcal{O}(size(\mathcal{T}) \cdot |\Phi| \cdot |fair|)$

Overview



CTL*
LTL
$\Diamond\Box a$
$\bigcirc a$
$\Diamond a$
⋮
$\Box\Diamond a$
CTL
$\forall\Box\,\exists\Diamond a$

Syntax of CTL*

state formulas:

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

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:

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

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

- 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.

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

$s \models true$		
$s \models a$	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 $\mathcal{T}$

$\pi \models \Phi$	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	$suffix(\pi, 1) \models \varphi$
$\pi \models \varphi_1 \mathbf{U} \varphi_2$	iff	there exists $j \geq 0$ such that $suffix(\pi, j) \models \varphi_2$, $suffix(\pi, i) \models \varphi_1$ for $0 \leq i < j$

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

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., $\forall\Box(request \rightarrow \Diamond response)$

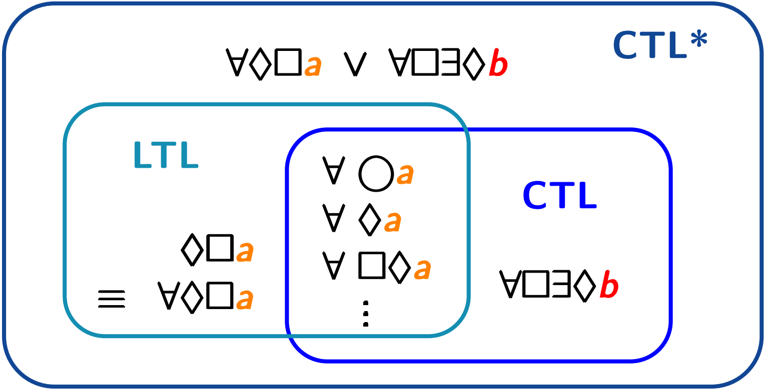
persistence property, e.g., $\forall\Diamond\Box a$

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*

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



$$\neg\exists\varphi \equiv \forall\neg\varphi \qquad \text{e.g., } \neg\exists\Box\Diamond a \equiv \forall\Diamond\Box\neg a$$

$$\neg\forall\varphi \equiv \exists\neg\varphi \qquad \text{e.g., } \neg\forall\Box\Diamond a \equiv \exists\Diamond\Box\neg a$$

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

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

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

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

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

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

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

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

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
IF S₀ ⊆ Sat(Φ)
    THEN  return "yes"
    ELSE  return "no"
FI
```

Recursive computation of satisfaction sets

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

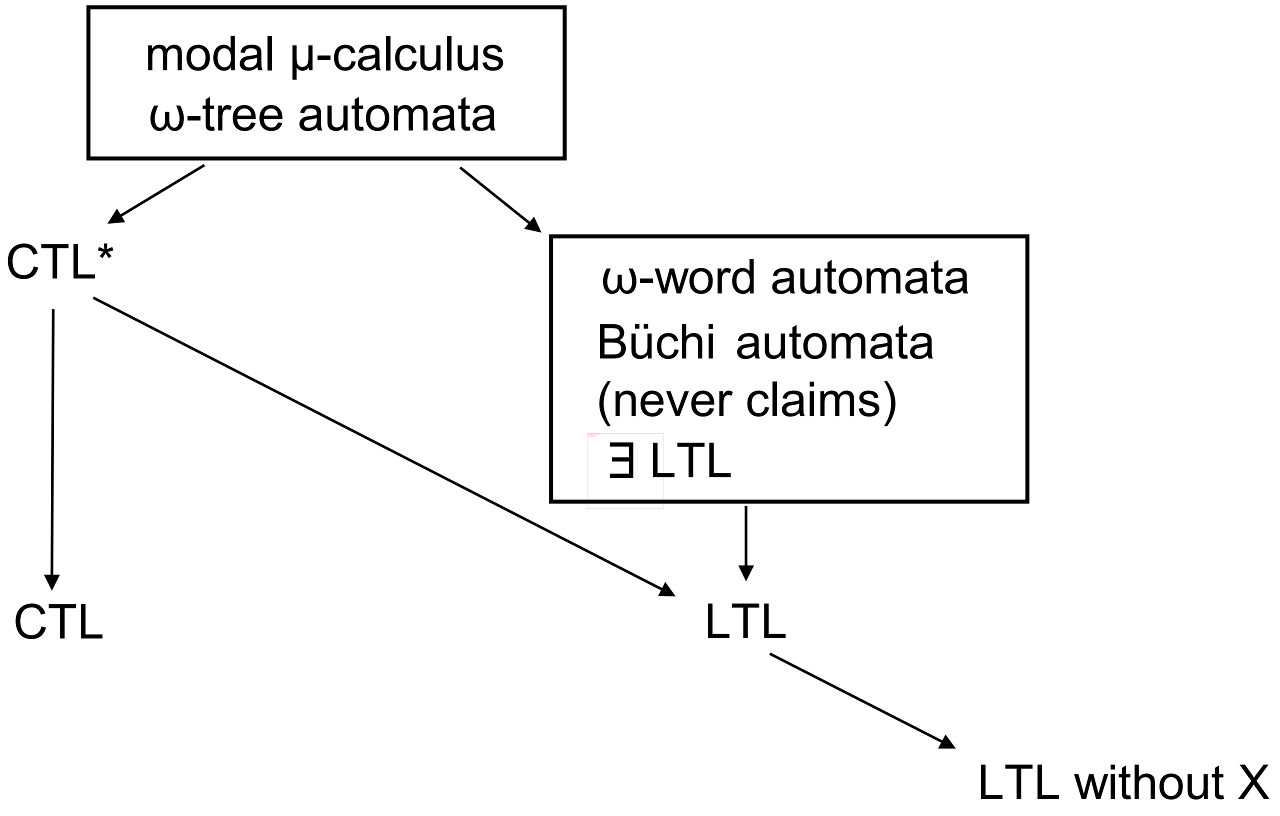
$$\left.\begin{array}{lcl} Sat(\forall\varphi) & = & Sat_{LTL}(\varphi) \\ Sat(\exists\varphi) & = & S \setminus Sat_{LTL}(\neg\varphi) \end{array}\right\} \text{ using an } \textbf{LTL} \text{ model checker}$$

Complexity of CTL/LTL/CTL* model checking

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

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

Relationships among logics



same box means ‘equally expressive’
single arrow means ‘more expressive than’
no arrow means ‘expressiveness is not comparable’