02246 - Model Checking

$$M \models \Phi ?$$

Lecture 03 - Computation-tree Temporal Logic

Technical University of Denmark

System

refine/implement

Requirements

abstract model

$M \models \Phi$?

some logic here

Finite-state transition System

some smart algorithm here

yes :)

may be :|

no :(

Key points of this lecture

Temporal Logics as an extension of propositional logics to reason about transition systems.

Computation Tree Logic (CTL) as a logic to reason computation trees.

Grammars for CTL: standard, minimalistic and existential normal form.

Formal semantics of CTL: satisfaction relation for states and paths.

Derived operators and equivalences between CTL formulas.

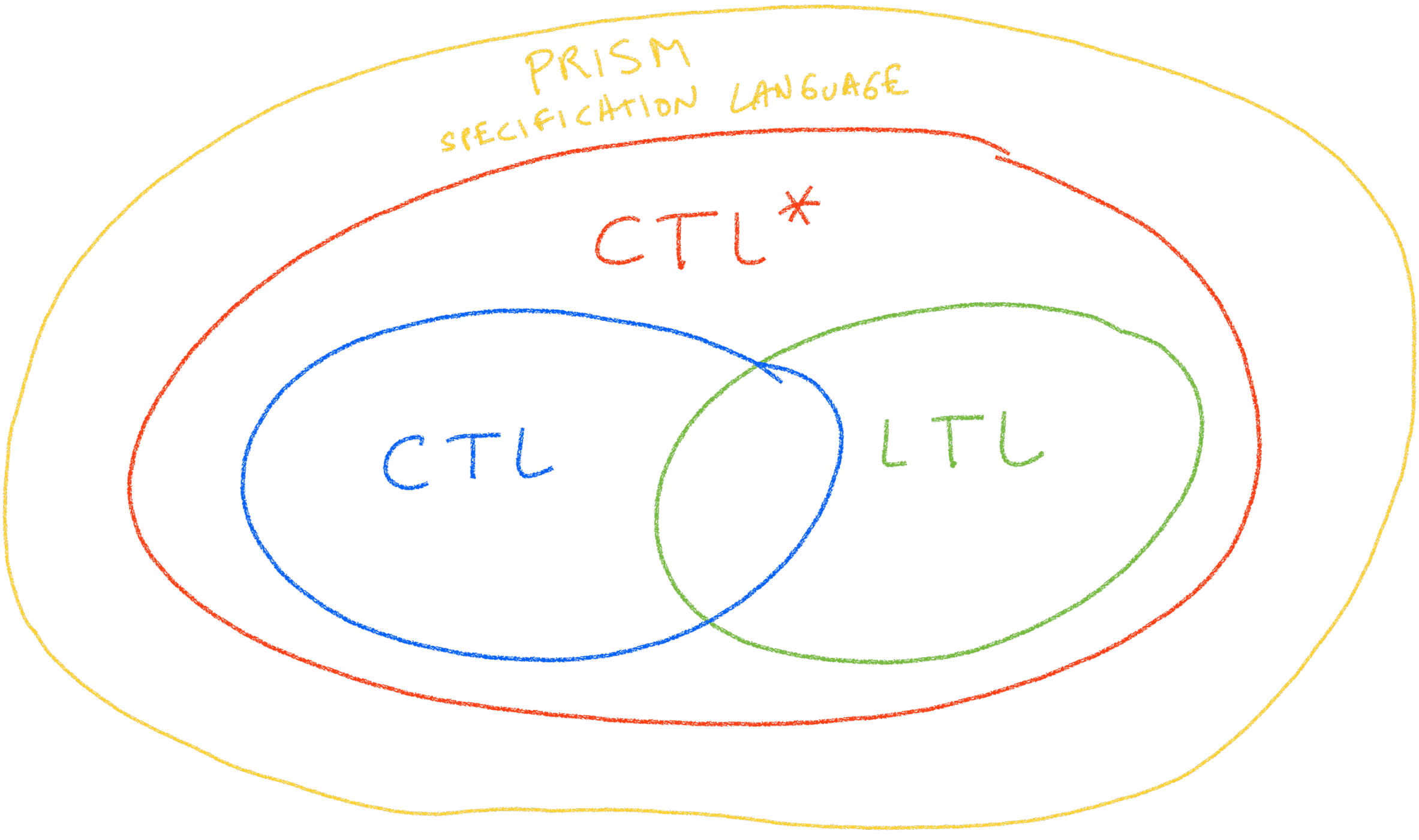
How to write CTL formulas in PRISM.

CTL* as a generalization of CTL and LTL.

Computation-tree Temporal Logic

- Reading Material
- LTL and CTL formulas, grammar and intuition
- Formal semantics of LTL and CTL
- Formula equivalences and alternative grammars
- LTL and CTL in PRISM
- Beyond CTL: CTL*
- Exercises & Homework

The CTL* family



Reading material

Sections 6 (intro), 6.1, 6.2, and 6.8 of Chapter 6 of “Principles of Model Checking”

CTL formulas, grammar and intuition

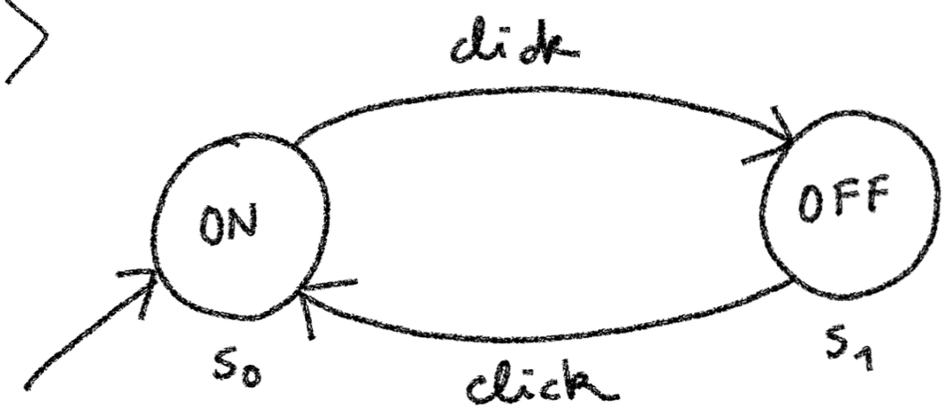
Transition Systems

Def. A transition system is a tuple

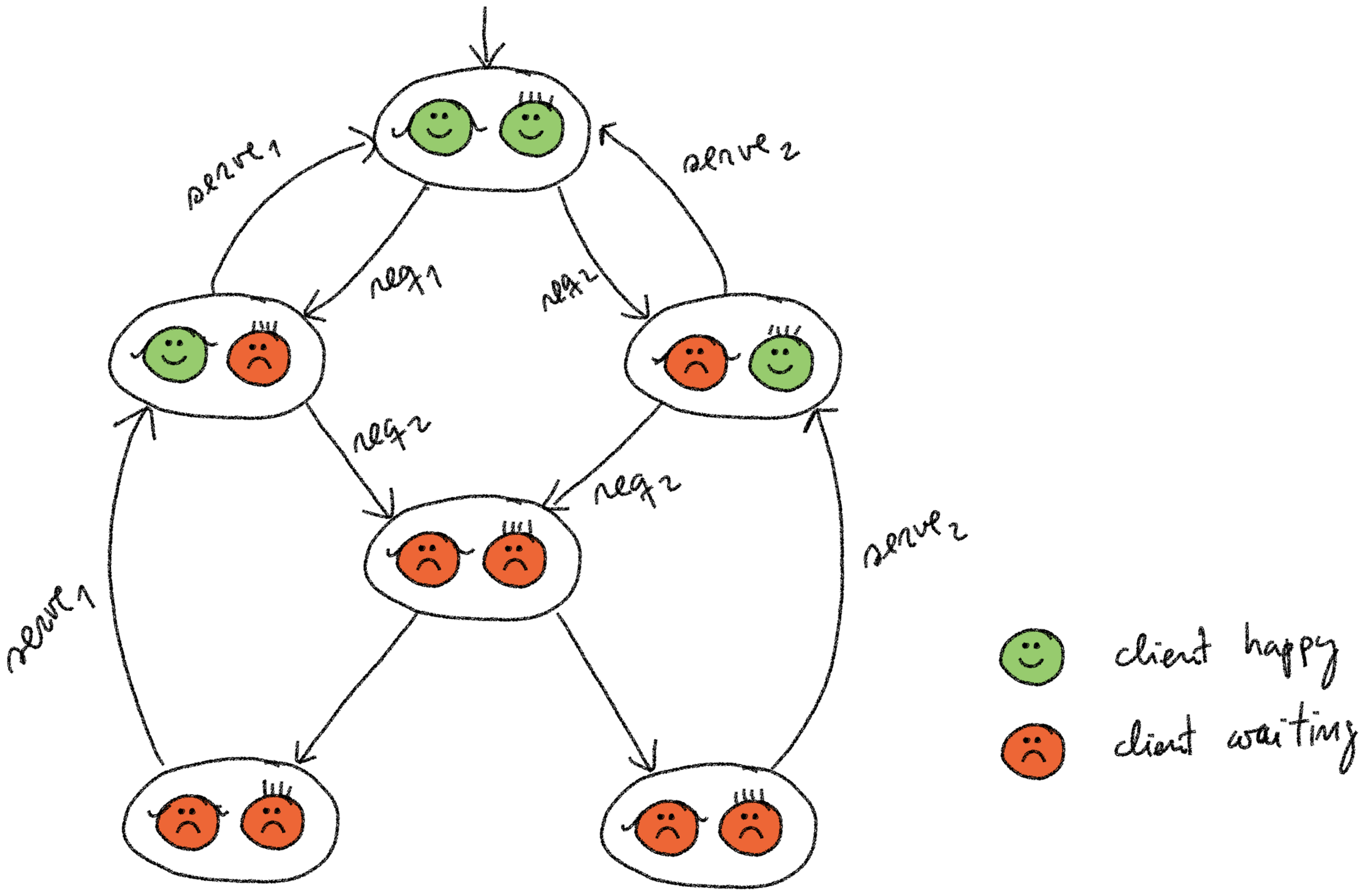
$$\langle S, A, \rightarrow, L, AP, I \rangle$$

such that

- S is a set of states
- A is a set of "Actions"
- $\rightarrow \subseteq S \times A \times S$ is a set of transitions
- $L : S \longrightarrow 2^{AP}$ is a labelling function
- AP is a finite state of "Atomic Propositions"
- $I \subseteq S$ is the set of initial states

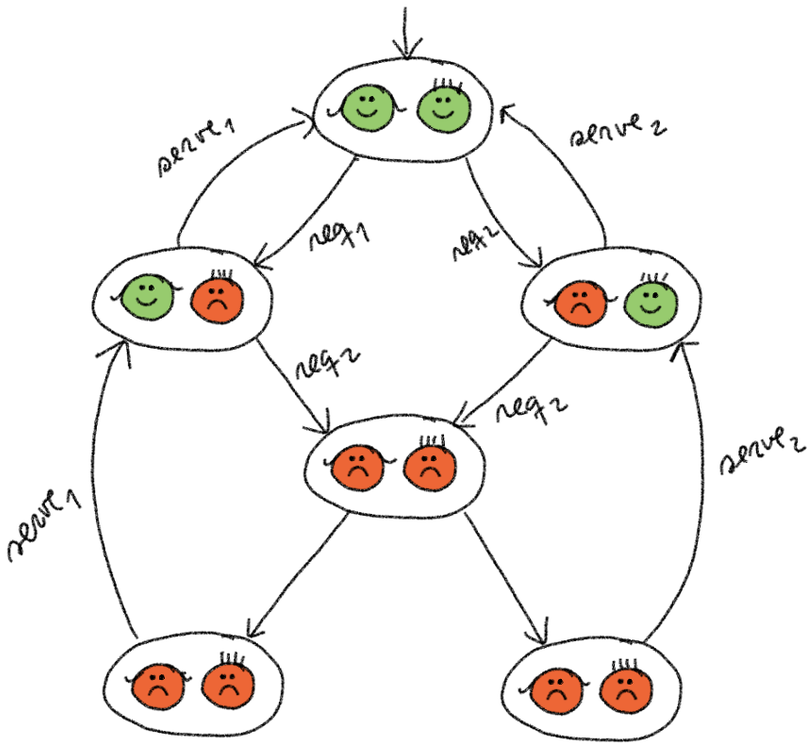


Example: 2 clients, 1 server

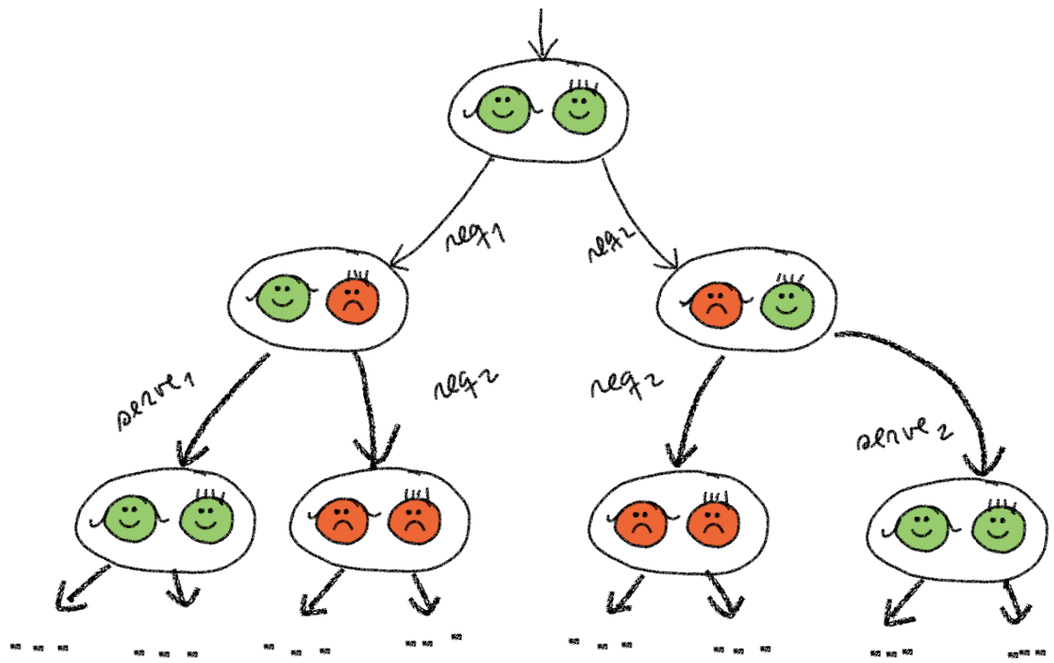


NOTE: model uploaded as `simple-scheduler.prism` on Learn. You can use PRISM to check the example we will see in class

Traces & Computation Trees

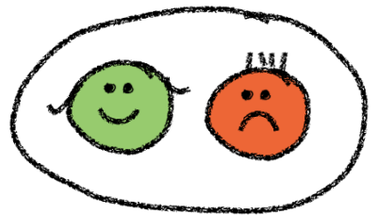


computation tree



Example of trace

Propositional Logic

We can use propositional logic to reason about a state 

$$true$$

$$false = \neg true$$

$$p$$

$$\neg \phi$$

$$\phi_1 \vee \phi_2$$

$$\phi_1 \wedge \phi_2$$

$$\phi_1 \rightarrow \phi_2$$

$$\ldots$$

the state satisfies p

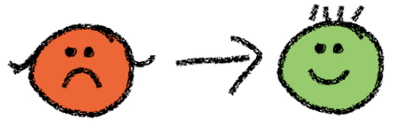
the state does not satisfy ϕ

the state satisfies at least one of ϕ_1, ϕ_2

the state satisfies both of ϕ_1, ϕ_2

if the state satisfies ϕ_1
then it also satisfies ϕ_2

Some examples



Derived operators and grammar for propositional logic

We don't need all operators, some can be derived

$$true$$

$$false = \neg true$$

$$p$$

$$\neg\phi$$

$$\phi_1 \vee \phi_2$$

$$\phi_1 \wedge \phi_2 \equiv \neg(\neg\phi_1 \vee \neg\phi_2)$$

$$\phi_1 \rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$$

We can then use a minimalistic grammar for formulas

$$\phi ::= true \mid p \mid \neg\phi \mid \phi_1 \vee \phi_2$$

Let’s have a look now at CTL….

Adding "temporal" operators

Propositional logic is not enough if we want to talk about executions.

Whether some or all executions have some property

$\exists\psi$ Some execution satisfies ψ

$\forall\psi$ All executions satisfy ψ

and properties of executions

next $\bigcirc\phi$ the next state of the execution satisfies ϕ

eventually or finally $\Diamond\phi$ some state of the execution satisfies ϕ

always $\Box\phi$ all states of the execution satisfies ϕ

until $\phi_1 \mathsf{U} \phi_2$ ϕ_2 holds in some state of the execution and until then all states satisfy ϕ_1

…

CTL grammar

The grammar CTL extends the one of propositional logic with path quantifiers

$$\phi ::= true \mid p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \exists\psi \mid \forall\psi$$

and path formulas

$$\psi ::= \bigcirc\phi \mid \Diamond\phi \mid \Box\phi \mid \phi_1 \mathsf{U} \phi_2$$

Alternative / equivalent notation

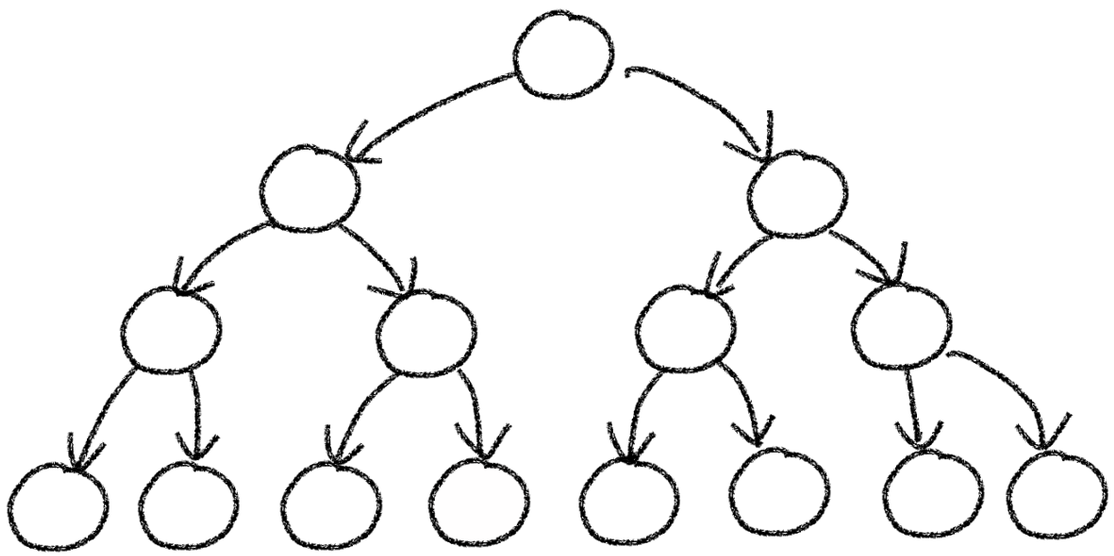
math/LaTeX style		ascii style
$\exists$	E	
$\forall$	A	
$\bigcirc$	X	
$\Diamond$	F	
$\Box$	G	

Intuition of “eventually” and “always” operators

On computation trees!

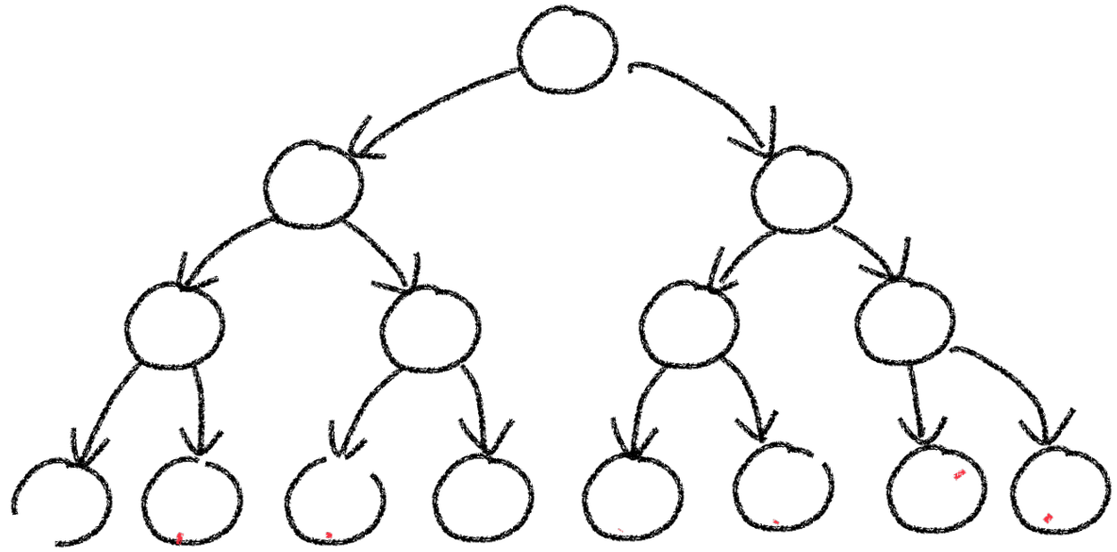
$\exists \Diamond \phi$

"ϕ holds potentially"



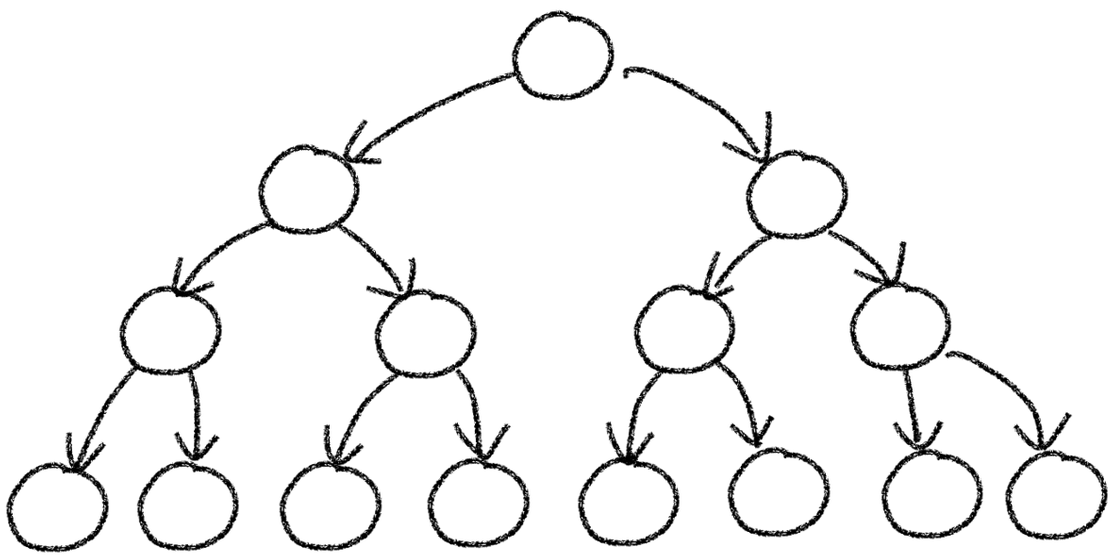
$\forall \Diamond \phi$

"ϕ is inevitable"



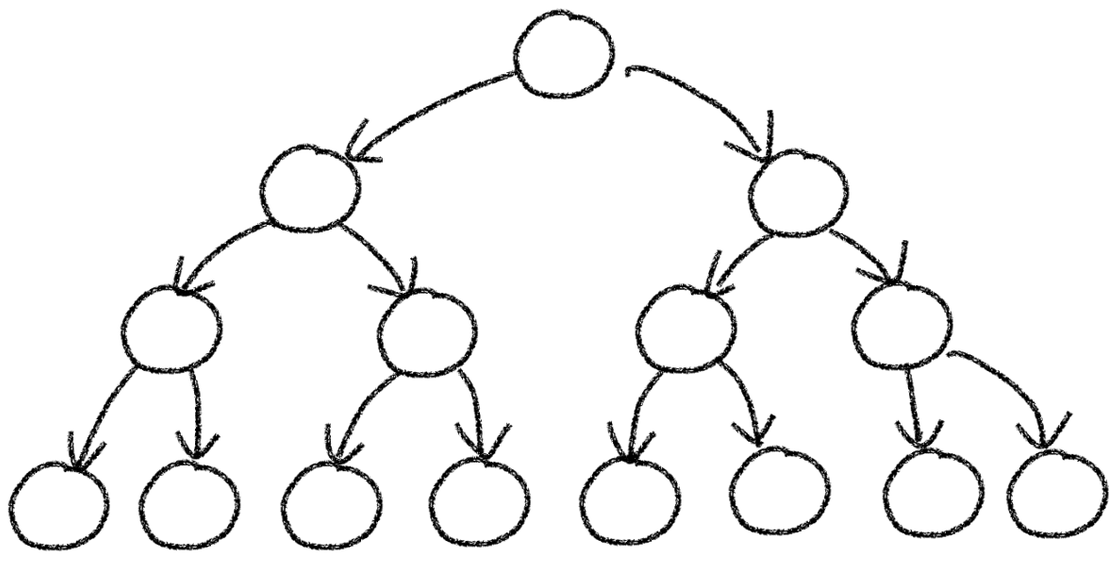
$\exists \Box \phi$

"potentially always ϕ"



$\forall \Box \phi$

"Globally always ϕ"

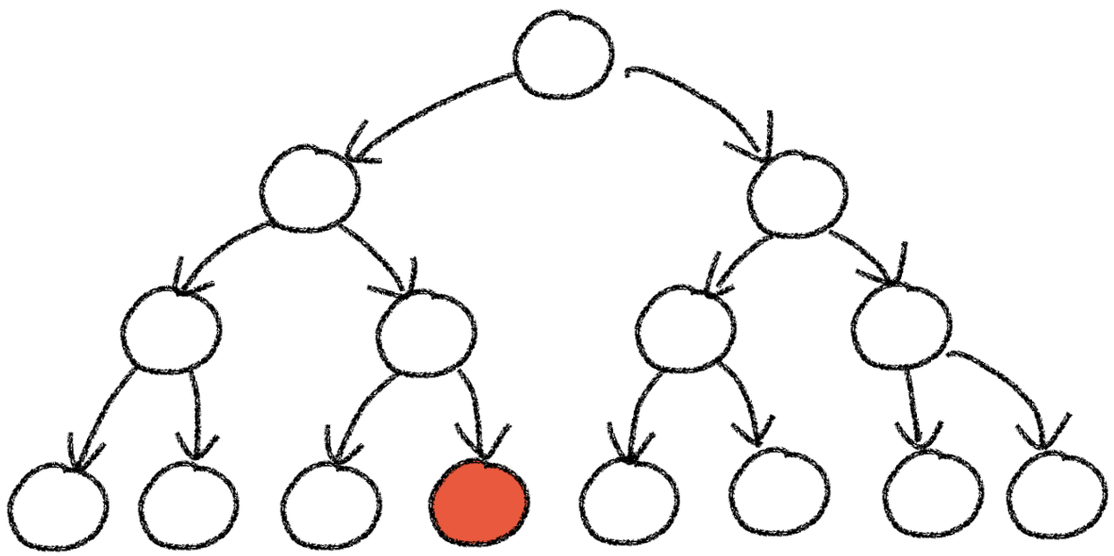


Intuition of “eventually” and “always” operators

On computation trees!

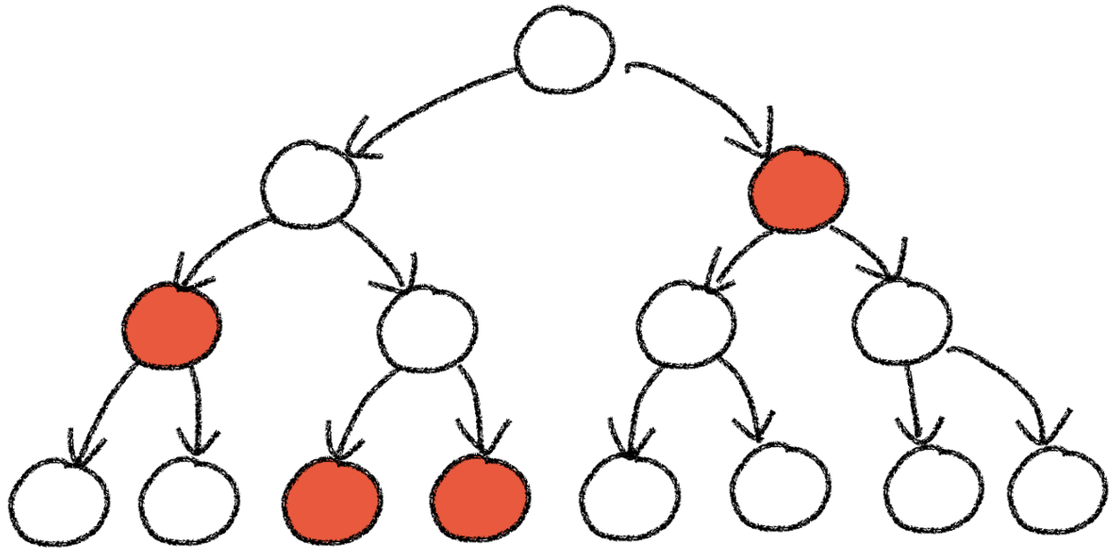
$\exists \Diamond \phi$

"ϕ holds potentially"



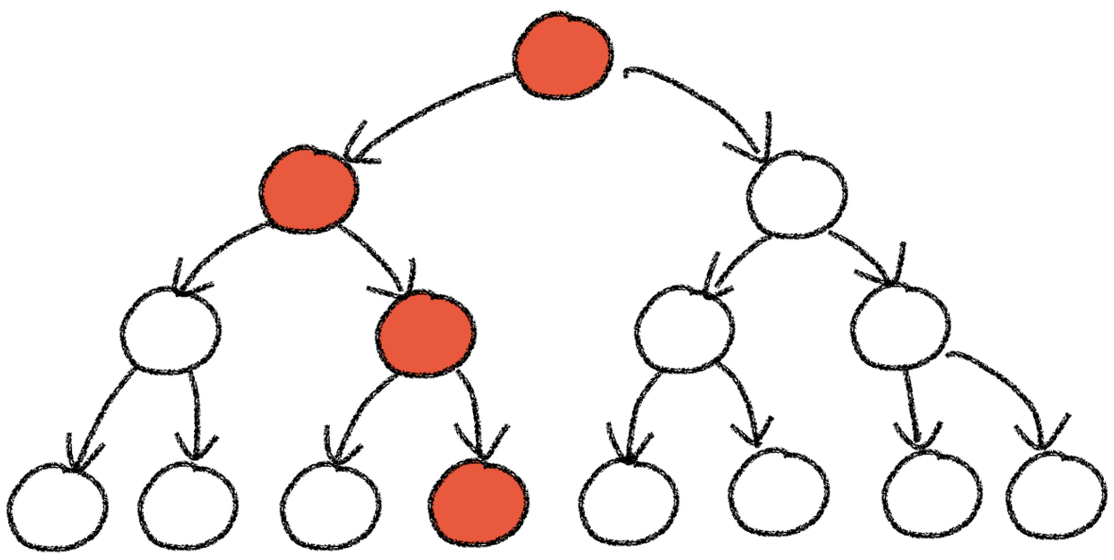
$\forall \Diamond \phi$

"ϕ is inevitable"



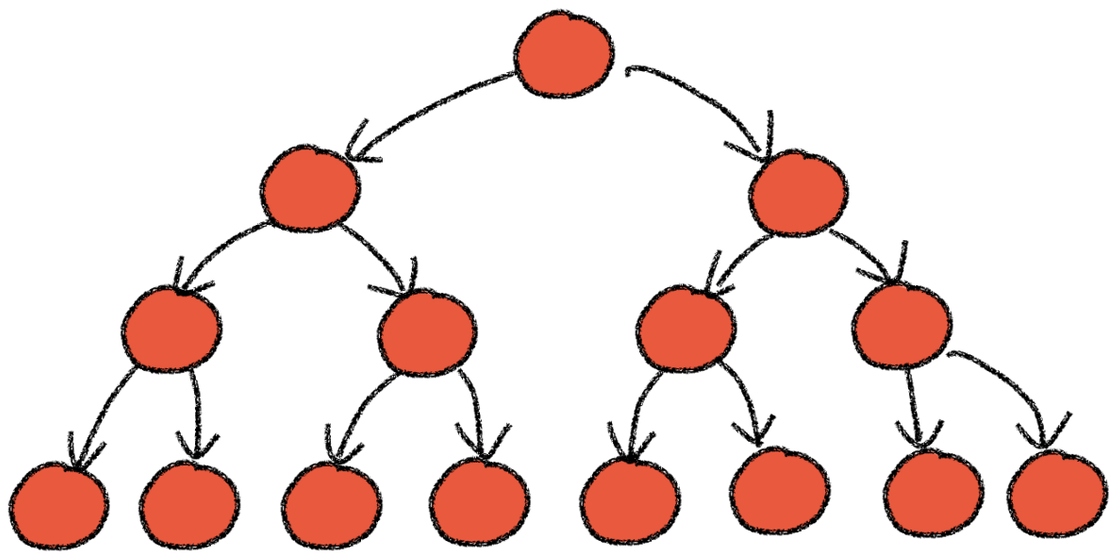
$\exists \Box \phi$

"potentially always ϕ"



$\forall \Box \phi$

"Globally always ϕ"

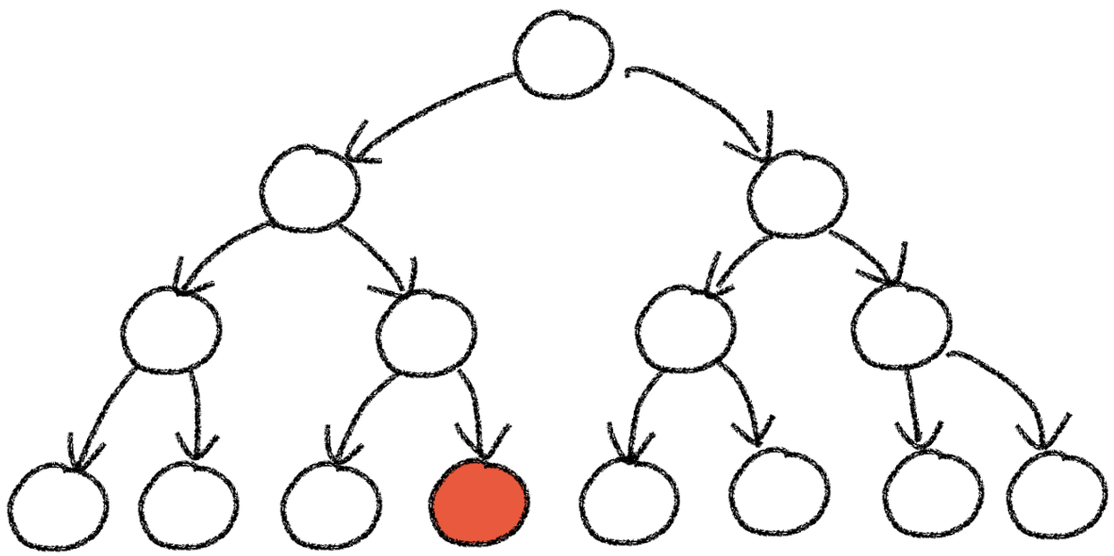


Intuition of the “until” operator

On computation trees!

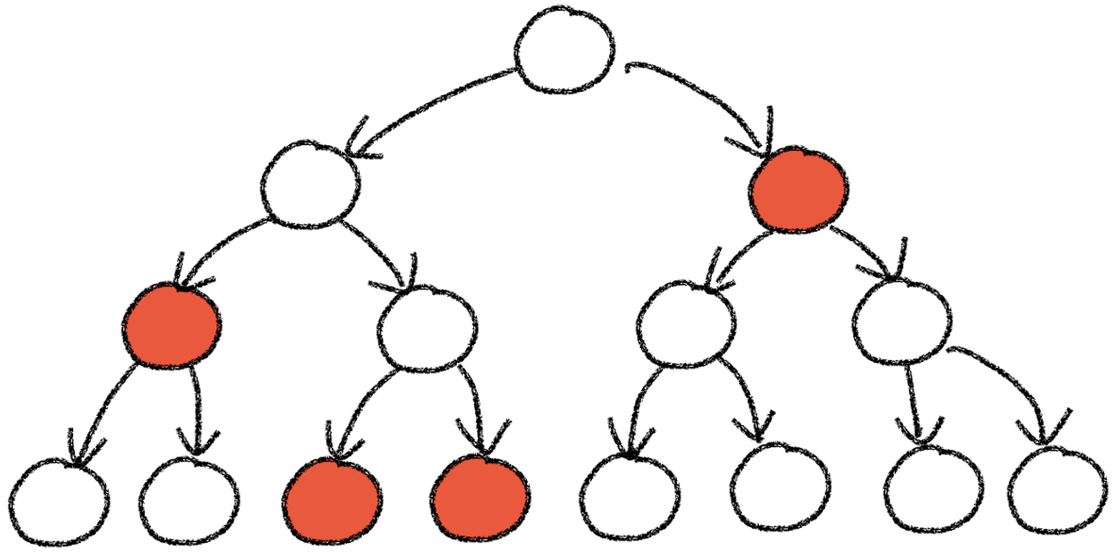
$\exists \Diamond \phi$

"ϕ holds potentially"



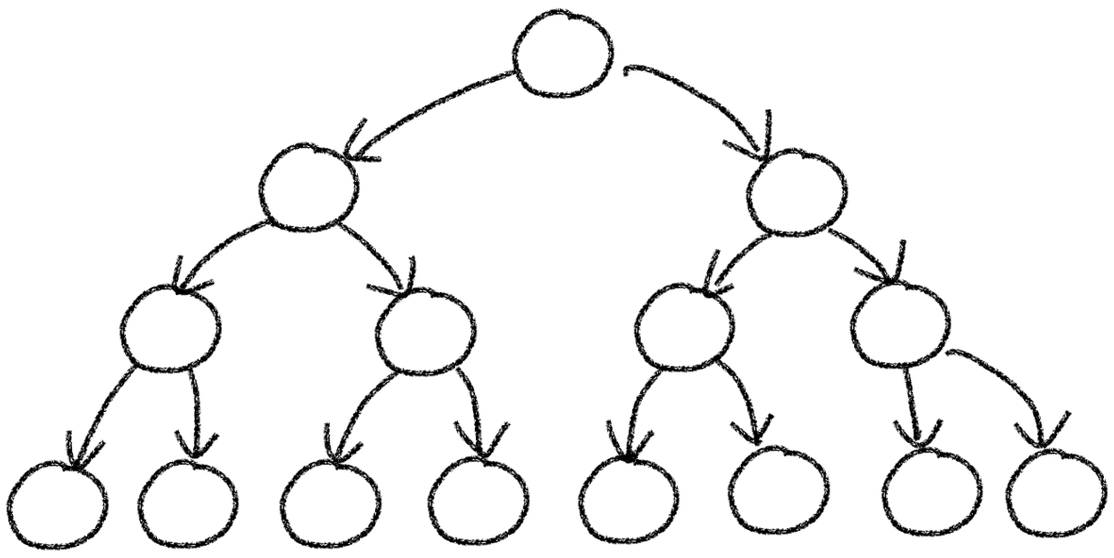
$\forall \Diamond \phi$

"ϕ is inevitable"



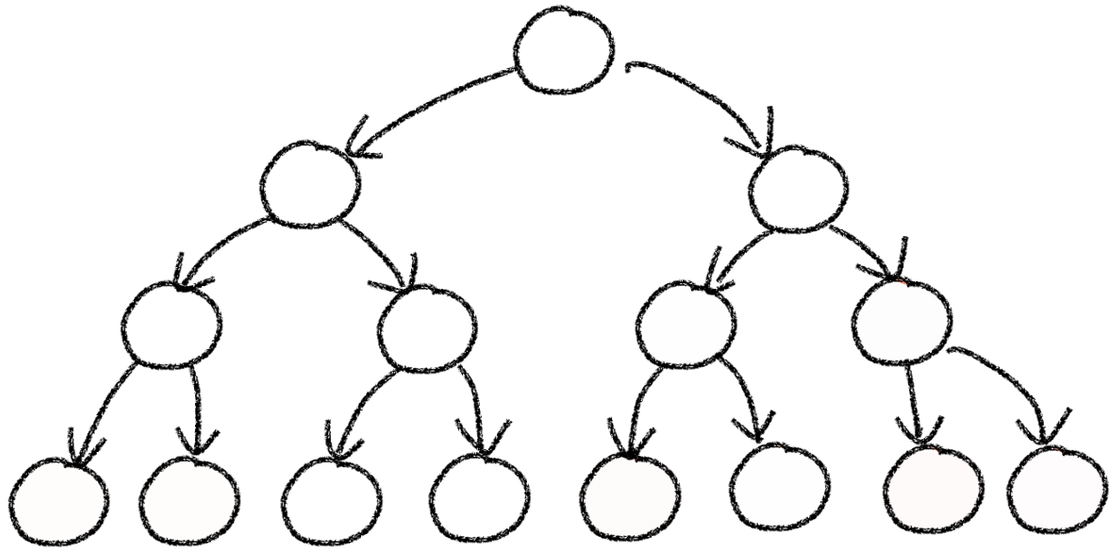
$\exists \ \phi_1 \cup \phi_2$

"there is a path s.t. $\phi_1 \cup \phi_2$ holds"



$\forall \ \phi_1 \cup \phi_2$

"in all paths $\phi_1 \cup \phi_2$ holds"

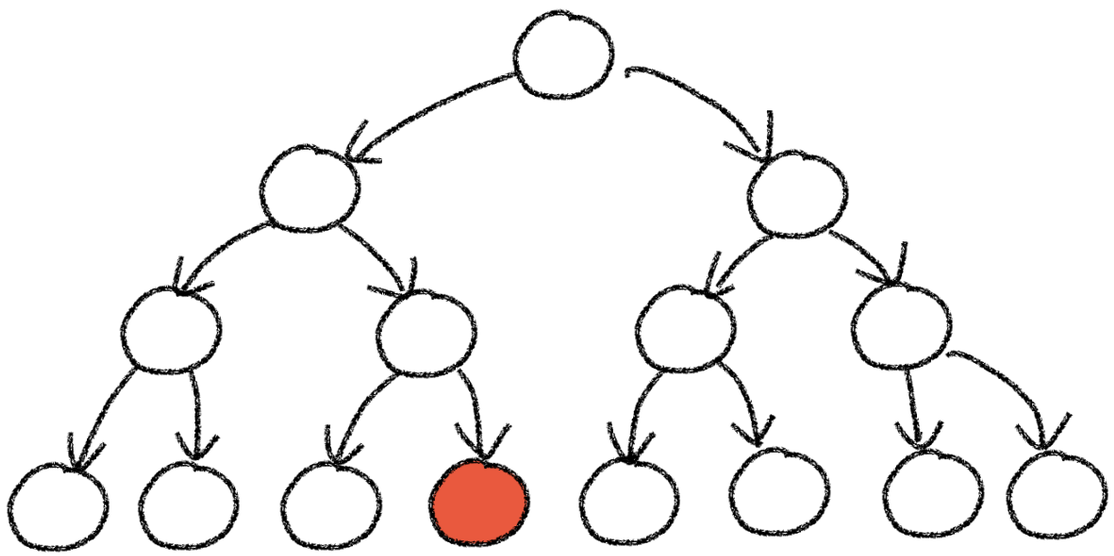


Intuition of the “until” operator

On computation trees!

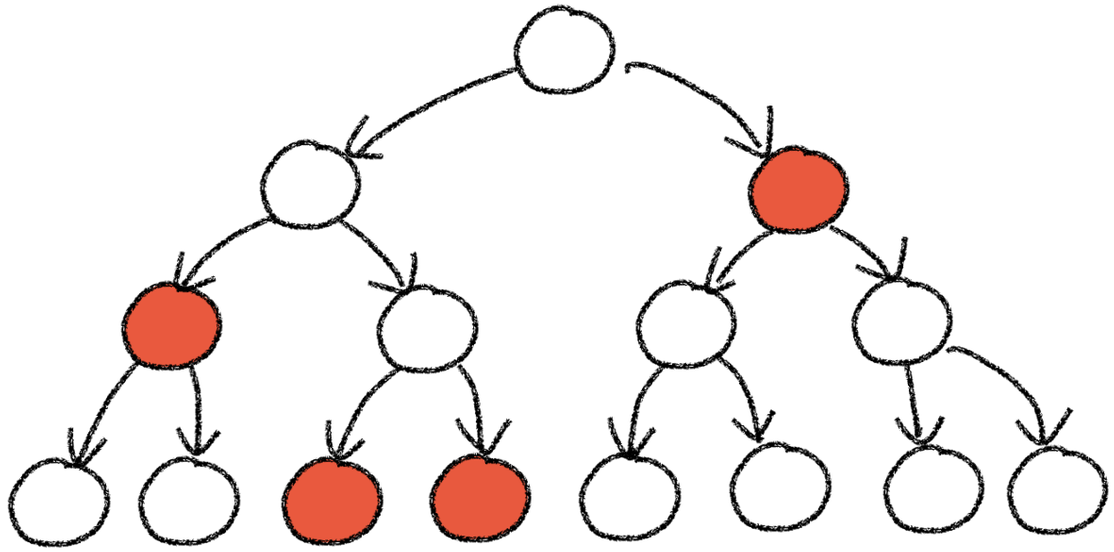
$\exists \Diamond \phi$

"ϕ holds potentially"



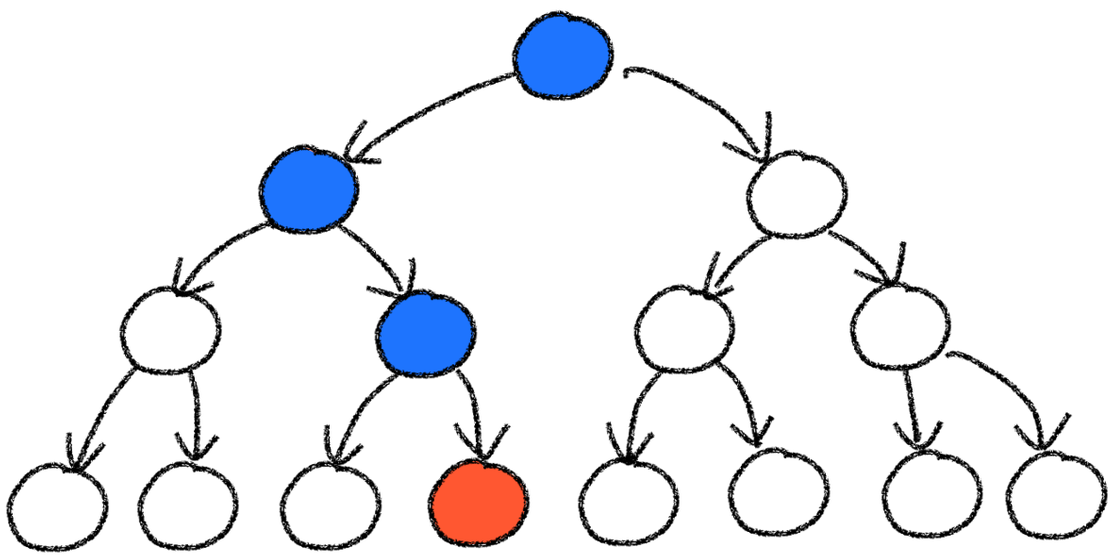
$\forall \Diamond \phi$

"ϕ is inevitable"



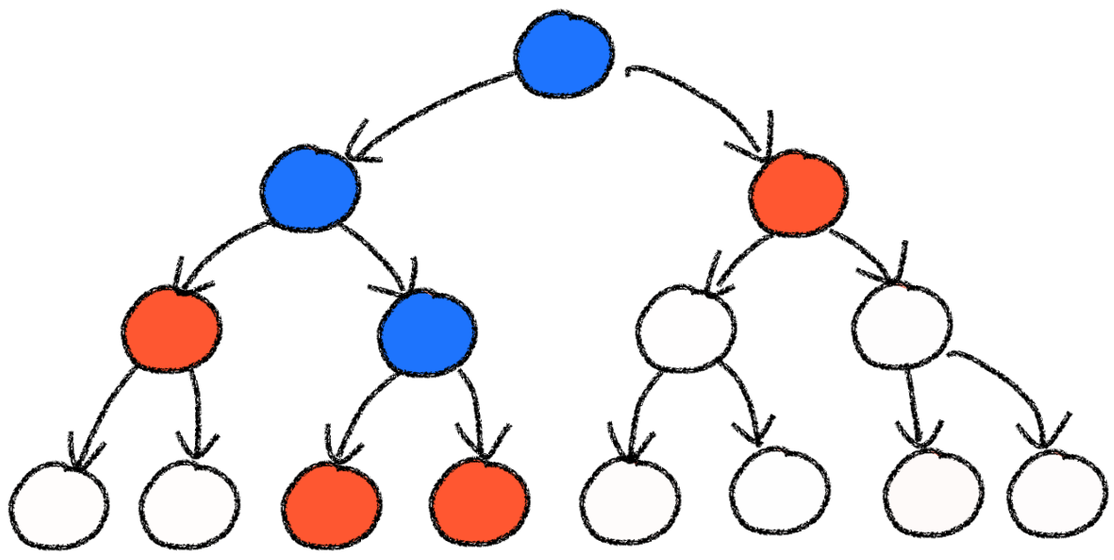
$\exists \ \phi_1 \cup \phi_2$

"there is a path s.t. $\phi_1 \cup \phi_2$ holds"



$\forall \ \phi_1 \cup \phi_2$

"in all paths $\phi_1 \cup \phi_2$ holds"



Witnesses and Counterexamples for CTL

A **witness** for a formula of the form

$$\exists \psi$$

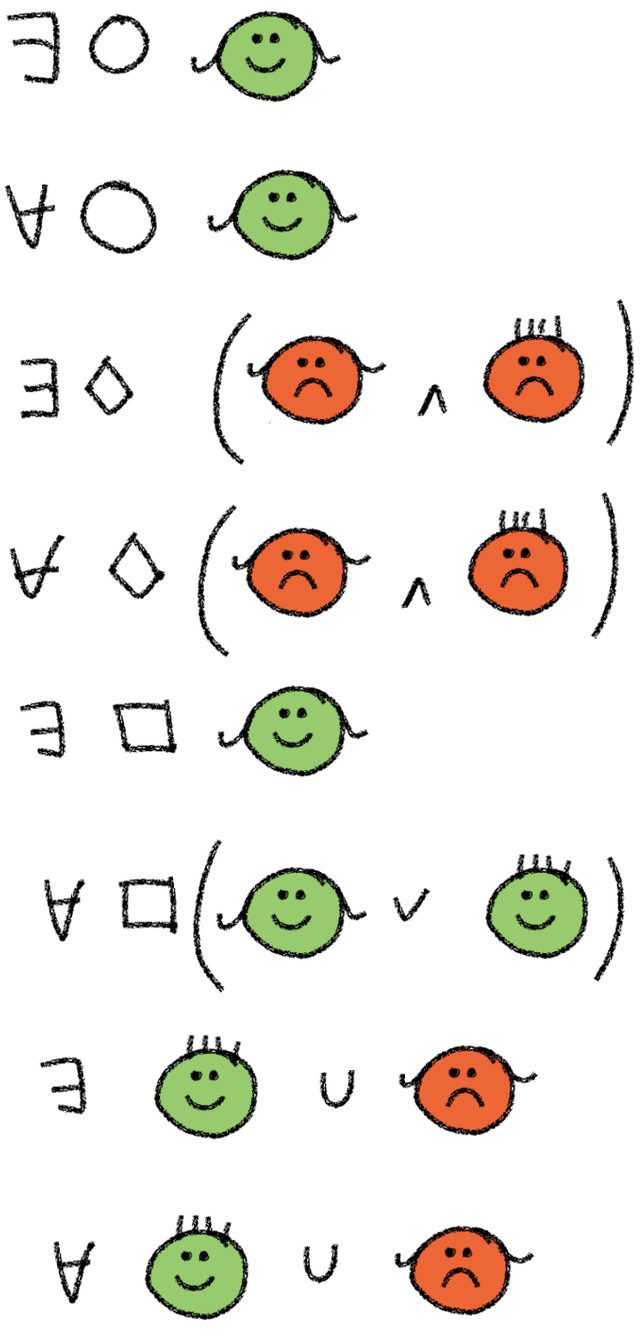
is just a path satisfying ψ

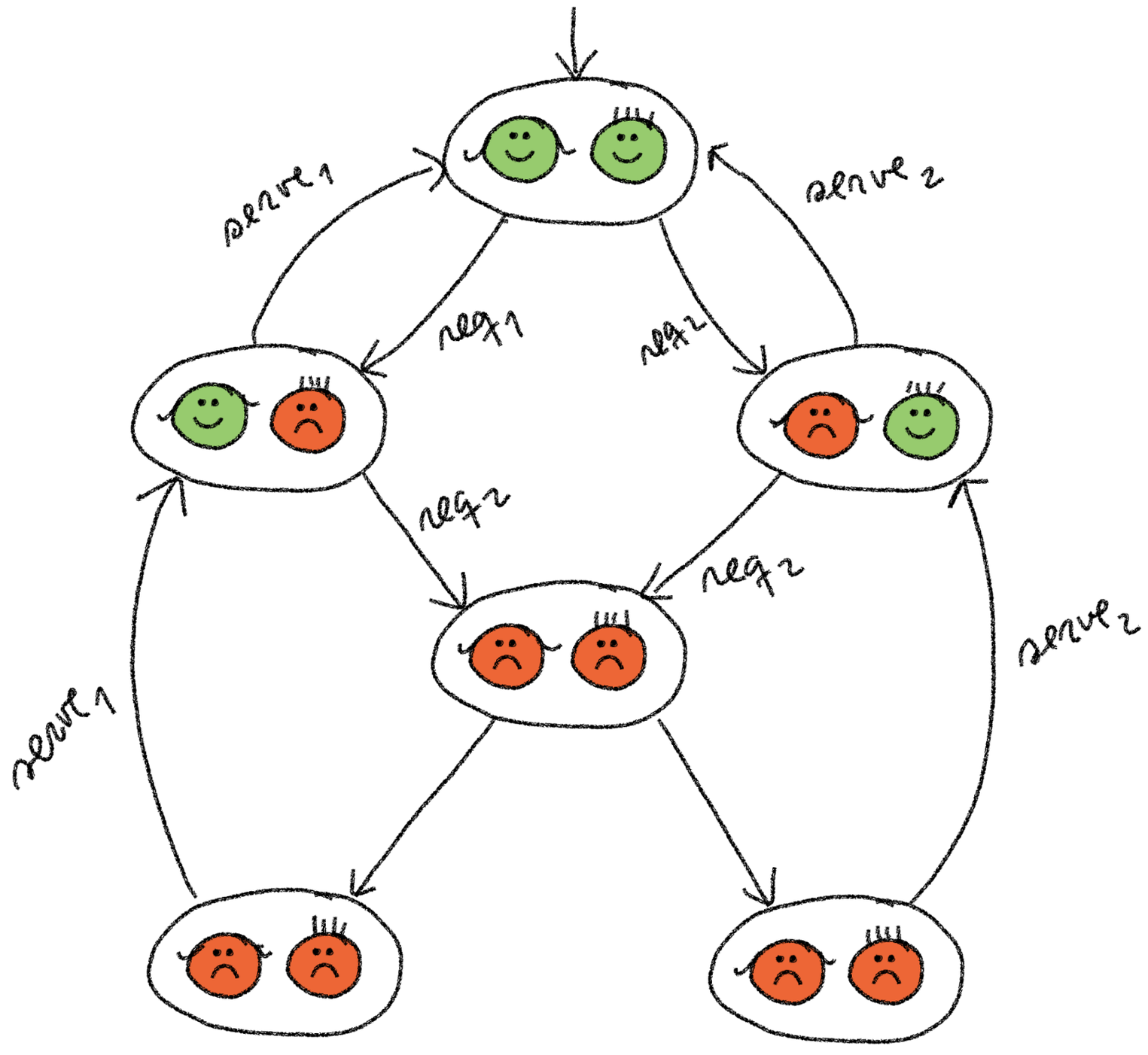
A **counterexample** is unfortunately the entire computation tree :(

We have the dual situation with formulas of the form

$$\forall \psi$$

Some examples

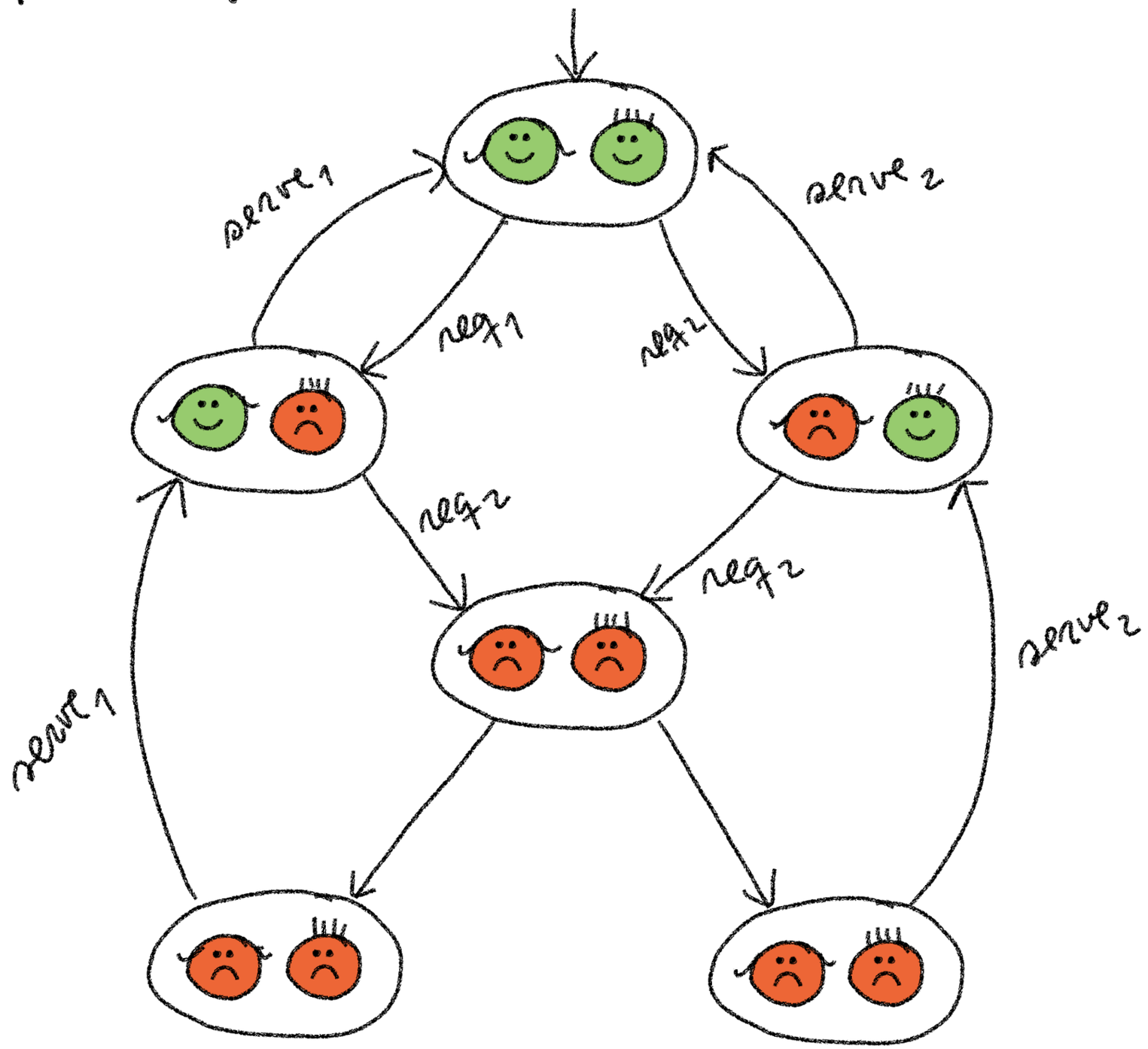




Some examples

Find formulas satisfied only by the initial state ...

↓



Typical Patterns of Formulas

"inv is an invariant"

$$\forall \Box\, inv$$

"every request is followed by a response"

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

"infinitely often p holds"

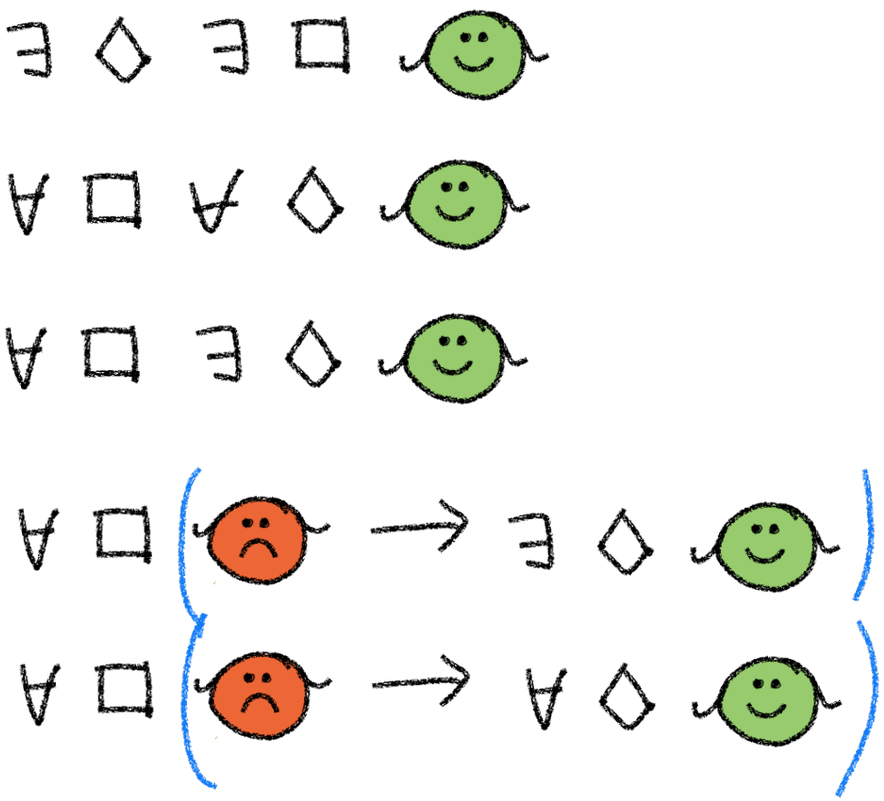
$$\forall \Box \forall \Diamond p$$

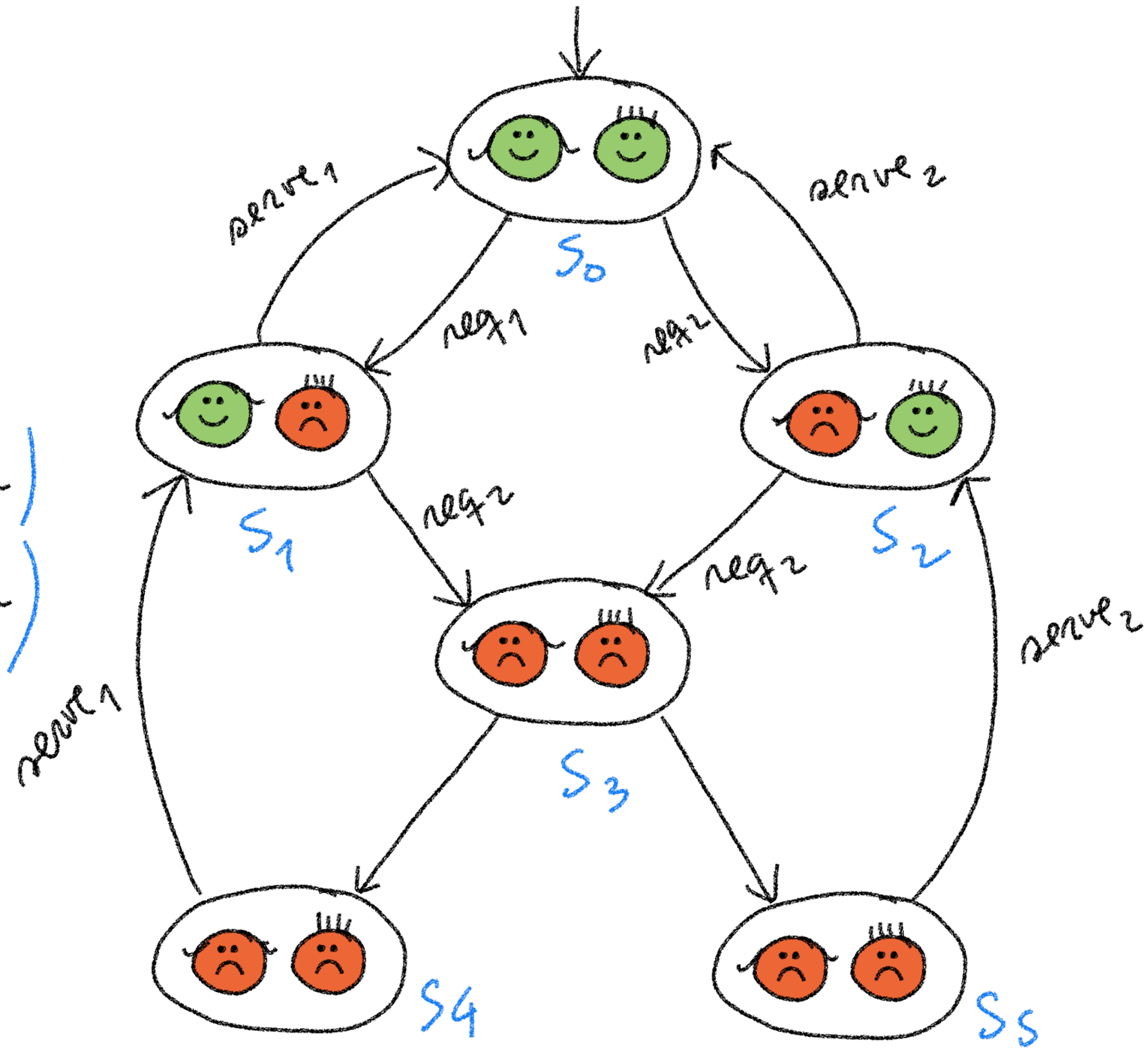
"q is persistent"

$$\forall \Diamond \exists \Box\, q$$

NOTE: A collection of property patterns is available at https://matthewbdwyer.github.io/psp/

More examples





NOTE: In PRISM, logical operators have precedence over temporal ones

Formal semantics of CTL

Some notation: paths, prefixes, states

For a state s we define the **set of all paths** starting from s, as the set of all **maximal executions** starting from s.

$$Paths(s) = \{s_0, s_1, s_2, \ldots \mid s_0 = s \text{ and } s_i \rightarrow s_{i+1}\}$$

For a path $\pi = s_0, s_1, s_2, \ldots$ we define the (i-1)-th state by

$$\pi[i] = s_i$$

We can also define the prefix starting at the (i-1)-th state

$$\pi[i\,..\,] = s_i, s_{i+1}, \ldots$$

CTL - formal semantics

The formal semantics of CTL needs 2 relations.

A relation between states and state formulas

$s \models_{CTL} true$ (holds always)

$s \models_{CTL} p$ iff $p \in L(s)$

$s \models_{CTL} \neg\phi$ iff $s \not\models_{CTL} \phi$

$s \models_{CTL} \phi_1 \wedge \phi_2$ iff $s \models_{CTL} \phi_1$ and $s \models_{CTL} \phi_2$

$s \models_{CTL} \exists\psi$ iff $\exists\pi \in Paths(s) \,.\, \pi \models_{CTL} \psi$

$s \models_{CTL} \forall\psi$ iff $\forall\pi \in Paths(s) \,.\, \pi \models_{CTL} \psi$

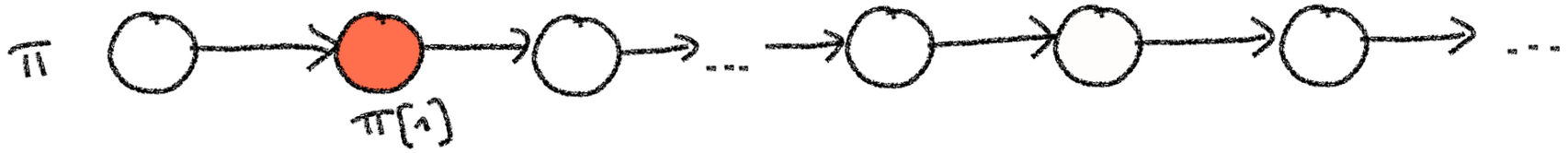
and a relation between paths and path formulas (next slide)

$\pi \models_{CTL} \psi$ iff …

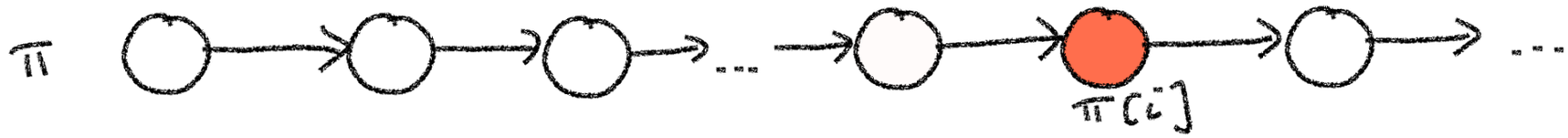
Semantics of CTL path formulas

And here is the formal semantics of path formulas

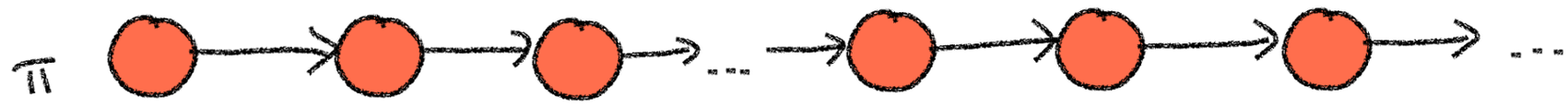
$\pi \models_{CTL} \bigcirc \phi$ iff $\pi[1] \models_{CTL} \phi$



$\pi \models_{CTL} \Diamond \phi$ iff $\exists i \in \mathbb{N} . \pi[i] \models_{CTL} \phi$



$\pi \models_{CTL} \Box \phi$ iff $\forall i \in \mathbb{N} . \pi[i] \models_{CTL} \phi$



$\pi \models_{CTL} \phi_1 \mathsf{U} \phi_2$ iff $\exists i \in \mathbb{N} . \pi[i] \models_{CTL} \phi_2 \wedge \forall 0 \leq j < i . \pi[j] \models_{CTL} \phi_1$

Can you spot the diff w.r.t. the semantics of LTL?

Semantics of CTL path formulas

Can you spot the diff w.r.t. the semantics of LTL?

$$\pi \vDash_{CTL} \phi_1 \mathsf{U} \phi_2 \quad \text{iff} \quad \exists i \in \mathbb{N} \,.\, \pi[i] \vDash_{CTL} \phi_2 \wedge \forall 0 \leq j < i \,.\, \pi[j] \vDash_{CTL} \phi_1$$

$$\pi \vDash_{LTL} \phi_1 \mathsf{U} \phi_2 \quad \text{iff} \quad \exists i \in \mathbb{N} \,.\, \pi[i\,..] \vDash_{LTL} \phi_2 \wedge \forall 0 \leq j < i \,.\, \pi[j\,..] \vDash_{LTL} \phi_1$$

In **CTL**, the path formula is valid if the **initial state** of the path satisfies ϕ_2

In **LTL**, the formula is valid if the **suffix starting in (i-1)-th state** satisfies ϕ_2

Satisfaction Sets

We define the satisfaction set of an LTL or CTL formula as the set of states that satisfy the formula

$$sat(\phi) = \{s \mid s \vDash \phi\}$$

Semantics over a TS

We say that a transition system satisfies a formula if all its initial states satisfy the formula:

$$T \models \phi \text{ iff } \forall s \in I \,.\, s \models \phi$$

or, equivalently

$$T \models \phi \text{ iff } I \subseteq sat(\phi)$$

Computation-tree Temporal Logic

- Reading Material
- LTL and CTL formulas, grammar and intuition
- Formal semantics of LTL and CTL
- Formula equivalences and alternative grammars
- LTL and CTL in PRISM
- Beyond CTL: CTL*
- Exercises & Homework

Equivalences

Two formulas are equivalent iff their validity (hold/not-holds) is **the same for all transitions systems**, i.e. for all transition systems T we have:

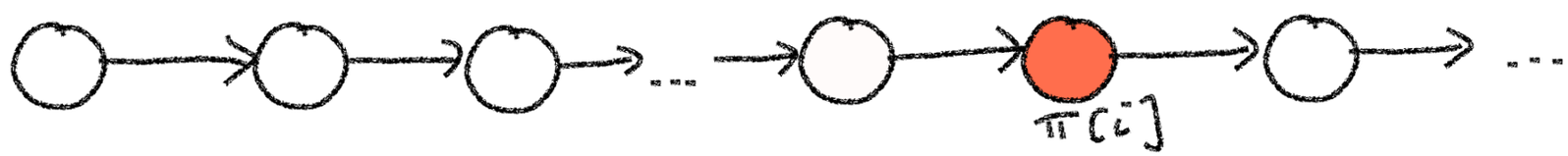
$$T \vDash \phi_1 \qquad \text{iff} \qquad T \vDash \phi_2$$

“Eventually” is a case of “Until”

A simple equivalence allows us to define “eventually” with “until”:

$$\Diamond\phi \;\equiv\; true \;\mathsf{U}\; \phi$$

$\pi \vDash \Diamond\phi_2$ iff $\exists i \in \mathbb{N} \,.\, \pi[i] \vDash \phi_2$



$\pi \vDash \phi_1 \mathsf{U} \phi_2$ iff $\exists i \in \mathbb{N} \,.\, \pi[i] \vDash \phi_2 \wedge \forall 0 \leq j < i \,.\, \pi[j] \vDash \phi_1$

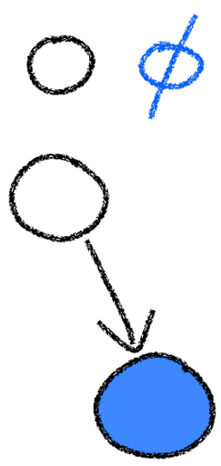
It holds for both LTL and CTL! (here illustrated just for CTL)

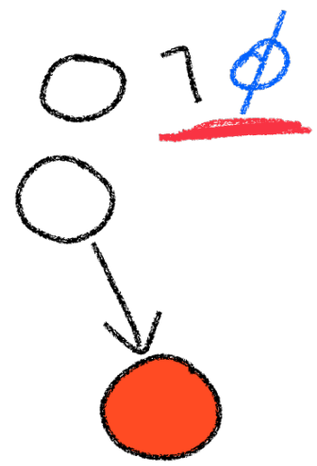
Dualities for “next”

Another example of simple equivalence is

$$\bigcirc\phi \equiv \neg\bigcirc\neg\phi$$

Easy to see graphically



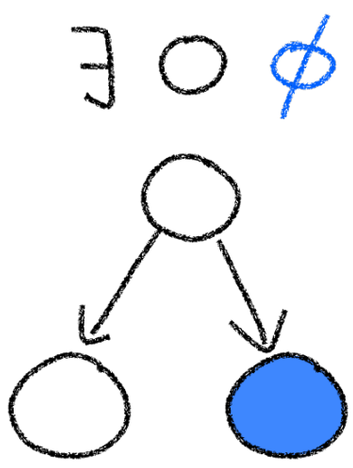


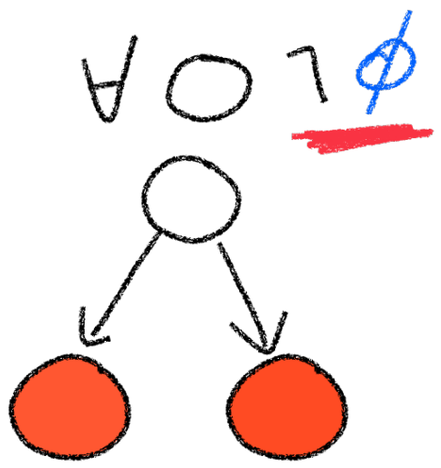
Dualities for “next”

Now in CTL:

$$\exists \bigcirc \phi \equiv \neg \forall \bigcirc \neg \phi \qquad \forall \bigcirc \phi \equiv \neg \exists \bigcirc \neg \phi$$

Easy to see graphically





Dualities for “eventually” and “always”

As in propositional logic, we could get rid of some operators:

$$\Box \phi \equiv \neg \Diamond \neg \phi$$

Dualities for “eventually” and “always”

In CTL we have

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

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

Why is this not fine in CTL?

$$\Box \phi \equiv \neg \Diamond \neg \phi$$

Dualities for “eventually” and “always”

In CTL we have

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

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

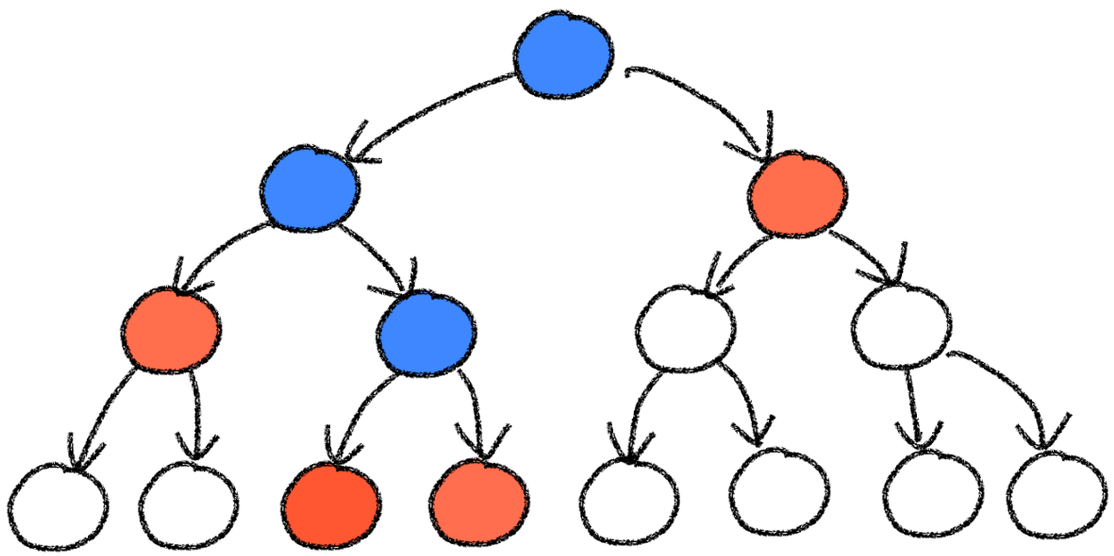
Why is this not fine in CTL? **Syntactically not allowed** (alternation with path quantifiers)

$$\Box \phi \equiv \neg \Diamond \neg \phi$$

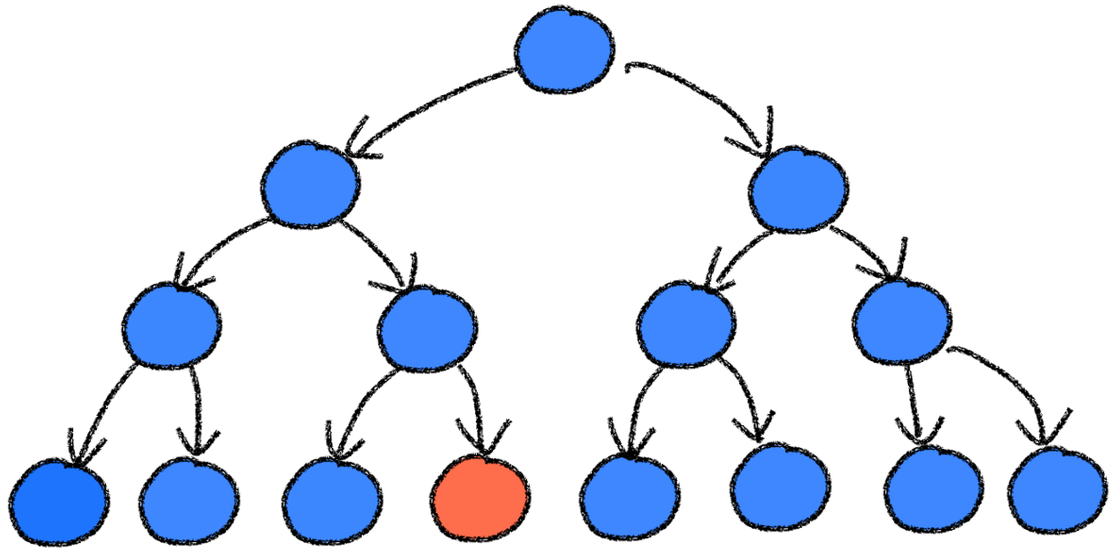
Intuition of dualities for “eventual” and “always”

$\forall \Diamond \neg \phi$

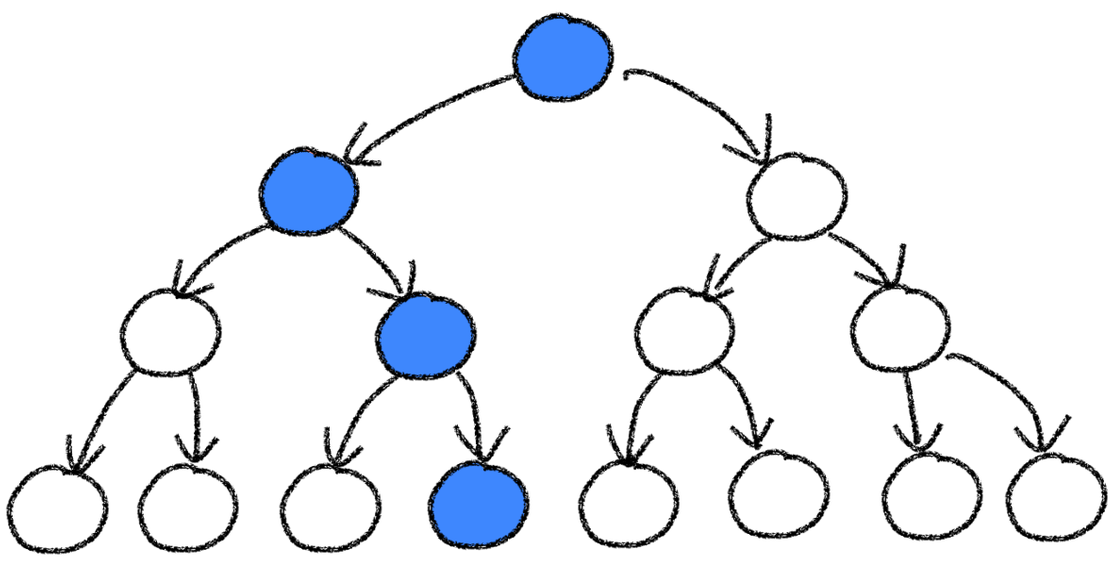
"ϕ holds potentially"



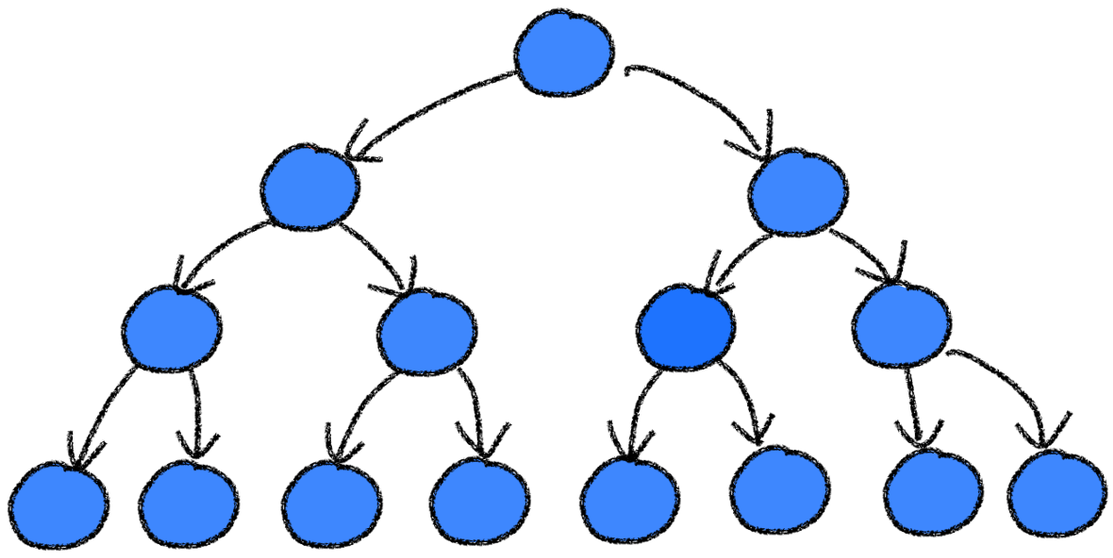
$\exists \Diamond \neg \phi$



$\exists \Box \phi$



$\forall \Box \phi$



More laws

See book…

duality law

$$\neg \bigcirc \varphi \equiv \bigcirc \neg \varphi$$
$$\neg \Diamond \varphi \equiv \Box \neg \varphi$$
$$\neg \Box \varphi \equiv \Diamond \neg \varphi$$

idempotency law

$$\Diamond \Diamond \varphi \equiv \Diamond \varphi$$
$$\Box \Box \varphi \equiv \Box \varphi$$
$$\varphi \,\mathsf{U}\, (\varphi \,\mathsf{U}\, \psi) \equiv \varphi \,\mathsf{U}\, \psi$$
$$(\varphi \,\mathsf{U}\, \psi) \,\mathsf{U}\, \psi \equiv \varphi \,\mathsf{U}\, \psi$$

absorption law

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

expansion law

$$\varphi \,\mathsf{U}\, \psi \equiv \psi \vee (\varphi \wedge \bigcirc (\varphi \,\mathsf{U}\, \psi))$$
$$\Diamond \psi \equiv \psi \vee \bigcirc \Diamond \psi$$
$$\Box \psi \equiv \psi \wedge \bigcirc \Box \psi$$

distributive law

$$\bigcirc (\varphi \,\mathsf{U}\, \psi) \equiv (\bigcirc \varphi) \,\mathsf{U}\, (\bigcirc \psi)$$
$$\Diamond (\varphi \vee \psi) \equiv \Diamond \varphi \vee \Diamond \psi$$
$$\Box (\varphi \wedge \psi) \equiv \Box \varphi \wedge \Box \psi$$

Figure 5.7: Some equivalence rules for LTL.

duality laws for path quantifiers

$$\forall \bigcirc \Phi \equiv \neg\, \mathsf{I} \bigcirc \neg \Phi \qquad \mathsf{I} \bigcirc \Phi \equiv \neg \forall \bigcirc \neg \Phi$$
$$\forall \Diamond \Phi \equiv \neg \exists \Box \neg \Phi \qquad \exists \Diamond \Phi \equiv \neg \forall \Box \neg \Phi$$
$$\begin{aligned} \forall (\Phi \,\mathsf{U}\, \Psi) &\equiv \neg \exists (\neg \Psi \,\mathsf{U}\, (\neg \Phi \wedge \neg \Psi)) \wedge \neg \exists \Box \neg \Psi \\ &\equiv \neg \mathsf{J} ((\Phi \wedge \neg \Psi) \,\mathsf{U}\, (\neg \Phi \wedge \neg \Psi)] \wedge \neg \mathsf{J} \Box (\Phi \wedge \neg \Psi) \\ &\equiv \neg \exists ((\Phi \wedge \neg \Psi) \,\mathsf{W}\, (\neg \Phi \wedge \neg \Psi)) \end{aligned}$$

expansion laws

$$\forall (\Phi \,\mathsf{U}\, \Psi) \equiv \Psi \vee (\Phi \wedge \forall \bigcirc \forall (\Phi \,\mathsf{U}\, \Psi))$$
$$\forall \Diamond \Phi \equiv \Phi \vee \forall \bigcirc \forall \Diamond \Phi$$
$$\forall \Box \Phi \equiv \Phi \wedge \forall \bigcirc \forall \Box \Phi$$
$$\mathsf{I}(\Phi \,\mathsf{U}\, \Psi) \equiv \Psi \vee (\Phi \wedge \mathsf{I} \bigcirc \mathsf{I}(\Phi \,\mathsf{U}\, \Psi))$$
$$\mathsf{I} \Diamond \Phi \equiv \Phi \vee \mathsf{I} \bigcirc \mathsf{I} \Diamond \Phi$$
$$\exists \Box \Phi \equiv \Phi \wedge \exists \bigcirc \exists \Box \Phi$$

distributive laws

$$\forall \Box (\Phi \wedge \Psi) \equiv \forall \Box \Phi \wedge \forall \Box \Psi$$
$$\mathsf{I} \Diamond (\Phi \vee \Psi) \equiv \mathsf{I} \Diamond \Phi \vee \mathsf{I} \Diamond \Psi$$

Figure 6.5: Some equivalence rules for CTL.

There exist complete axiomatisations.

We can exploit the equivalences to use

“lightweight” grammars

(i.e. with less operators)

CTL - minimal syntax

We can get rid of “always” and “eventually” in CTL

$$\phi ::= true \mid p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \exists\psi \mid \forall\psi$$

$$\psi ::= \bigcirc\phi \mid \Diamond\phi \mid \Box\phi \mid \phi_1 \mathsf{U} \phi_2$$

CTL - Existential Normal Form

Alternatively, we can get rid of the universal quantifier

$$\phi ::= true \mid p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \exists\psi$$

$$\psi ::= \bigcirc\phi \mid \Box\phi \mid \phi_1 \mathsf{U} \phi_2$$

The above grammar yields CTL formulas in so-called "**existential normal form**" (we shall consider it when we will see the model checking algorithms)

Expansion laws

In LTL

$$\Box\phi \equiv \phi \wedge \bigcirc \Box\phi$$

$$\Diamond\phi \equiv \phi \vee \bigcirc \Diamond\phi$$

$$\phi_1 \mathsf{U} \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge \bigcirc \phi_1 \mathsf{U} \phi_2)$$

In CTL

$$\exists \Box\phi \equiv \ldots$$

$$\exists \Diamond\phi \equiv \ldots$$

$$\exists \phi_1 \mathsf{U} \phi_2 \equiv \ldots$$

Expansion laws

In LTL

$$\Box\phi \equiv \phi \wedge \bigcirc \Box\phi$$

$$\Diamond\phi \equiv \phi \vee \bigcirc \Diamond\phi$$

$$\phi_1 \mathsf{U} \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge \bigcirc \phi_1 \mathsf{U} \phi_2)$$

In CTL

$$\exists\Box\phi \equiv \phi \wedge \exists\bigcirc\exists\Box\phi$$

$$\exists\Diamond\phi \equiv \phi \vee \exists\bigcirc\exists\Diamond\phi$$

$$\exists\phi_1 \mathsf{U} \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge \exists\bigcirc\exists\phi_1 \mathsf{U} \phi_2)$$

} basis for model checking algorithms (next lecture)

Distribution Laws

In LTL

$$\Diamond(\phi_1 \vee \phi_2) \equiv (\Diamond\phi_1) \vee (\Diamond\phi_2)$$

$$\Box(\phi_1 \wedge \phi_2) \equiv (\Box\phi_1) \wedge (\Box\phi_2)$$

In CTL

$$\exists\Diamond(\phi_1 \vee \phi_2) \equiv \ldots$$

$$\forall\Box(\phi_1 \wedge \phi_2) \equiv \ldots$$

Distribution Laws

In LTL

$$\Diamond(\phi_1 \vee \phi_2) \equiv (\Diamond\phi_1) \vee (\Diamond\phi_2)$$

$$\Box(\phi_1 \wedge \phi_2) \equiv (\Box\phi_1) \wedge (\Box\phi_2)$$

In CTL

$$\exists\Diamond(\phi_1 \vee \phi_2) \equiv (\exists\Diamond\phi_1) \vee (\exists\Diamond\phi_2)$$

$$\forall\Box(\phi_1 \wedge \phi_2) \equiv (\forall\Box\phi_1) \wedge (\forall\Box\phi_2)$$

More Distribution Laws?

Does this hold?

$$\exists \Diamond (\phi_1 \wedge \phi_2) \equiv (\exists \Diamond \phi_1) \wedge (\exists \Diamond \phi_2)$$

More Distribution Laws?

Does this hold?

$$\exists \Diamond (\phi_1 \wedge \phi_2) \equiv (\exists \Diamond \phi_1) \wedge (\exists \Diamond \phi_2)$$

Let’s see a proof

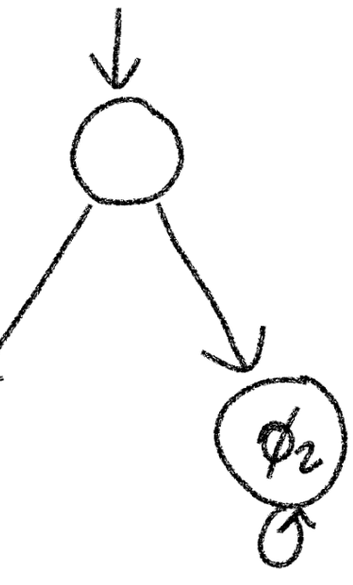
- Important for mandatory assignments
- Equivalence/implication holds —> demonstrate formally
- Equivalence/implication doesn’t hold —> show a counter-example

$$\exists\Diamond(\phi_1 \wedge \phi_2) \not\equiv (\exists\Diamond\phi_1) \wedge (\exists\Diamond\phi_2)$$

$$\not\Leftarrow$$

We need to find a TS $\mathcal{T}$ s.t.

$\mathcal{T} \models \exists\Diamond\phi_1 \wedge \exists\Diamond\phi_2$ $\Rightarrow$



<u>but</u>

$\mathcal{T} \not\models \exists\Diamond(\phi_1 \wedge \phi_2)$ $\not\Leftarrow$

$$\exists \Diamond(\phi_1 \wedge \phi_2) \overset{?}{\equiv} (\exists \Diamond \phi_1) \wedge (\exists \Diamond \phi_2)$$

$\Rightarrow$

Let $\mathcal{T}$ be any TS s.t. $\mathcal{T} \models \exists \Diamond (\phi_1 \wedge \phi_2)$.

By definition (semantics) we know that for all initial states $s_0 \in I$ there is a path $\pi = s_0 s_1 \dots$ and an $i \in \mathbb{N}$ s.t.

$\pi[i] \models \phi_1 \wedge \phi_2$

$\Downarrow$ $\wedge$-elimination

$\pi[i] \models \phi_1$

$\Downarrow$ semantics of $\Diamond$

$\pi \models \Diamond \phi_1$

$\Downarrow$ semantics of $\exists$

$s_0 \models \exists \Diamond \phi_1$

$\Downarrow$

$\pi[i] \models \phi_2$

$\Downarrow$

$\pi \models \Diamond \phi_2$

$\Downarrow$

$s_0 \models \exists \Diamond \phi_2$

$\Downarrow$ sem of $\wedge$

$s_0 \models \exists \Diamond \phi_1 \wedge \exists \Diamond \phi_2$

$\Downarrow$

$\mathcal{T} \models \exists \Diamond \phi_1 \wedge \exists \Diamond \phi_2$

CTL in PRISM

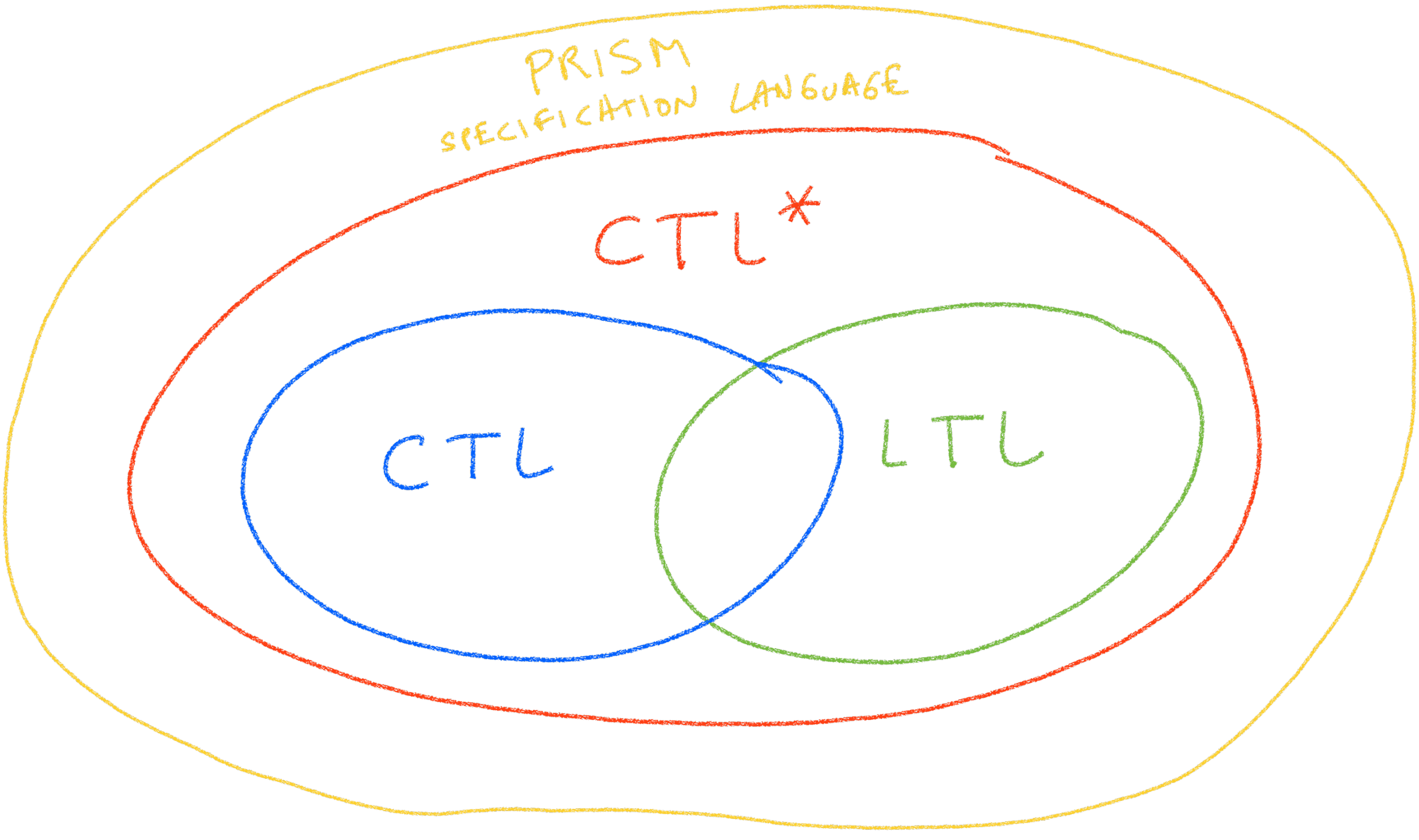
CTL in PRISM

PRISM uses a slightly different notation temporal logics:

CTL Formula (version 1)	CTL Formula (version 2)	PRISM Notation
$\neg\Phi$	$\neg\Phi$	$!\Phi$
$\Phi_1 \wedge \Phi_2$	$\Phi_1 \wedge \Phi_2$	Φ_1 & Φ_2
$\Phi_1 \Rightarrow \Phi_2$	$\Phi_1 \rightarrow \Phi_2$	Φ_1 => Φ_2
$\forall[\Phi_1\ U\ \Phi_2]$	$A[\Phi_1\ U\ \Phi_2]$	P>=1 [Φ_1 U Φ_2]
$\exists[\Phi_1\ U\ \Phi_2]$	$E[\Phi_1\ U\ \Phi_2]$	!P<=0 [Φ_1 U Φ_2]
$\forall \bigcirc\ \Phi$	$AX\ \Phi$	P>=1 [X Φ]
$\exists \bigcirc\ \Phi$	$EX\ \Phi$	!P<=0 [X Φ]
$\forall \Box\ \Phi$	$AG\ \Phi$	P>=1 [G Φ]
$\exists \Box\ \Phi$	$EG\ \Phi$	!P<=0 [G Φ]
$\forall \Diamond\ \Phi$	$AF\ \Phi$	P>=1 [F Φ]
$\exists \Diamond\ \Phi$	$EF\ \Phi$	!P<=0 [F Φ]

Beyond CTL: CTL*

The CTL* family



CTL* grammar

State formulas

$$\phi ::= true \mid p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \exists\psi \mid \forall\psi$$

Path formulas

$$\psi ::= \bigcirc\psi \mid \Diamond\psi \mid \Box\psi \mid \psi_1 \mathsf{U} \psi_2 \mid \neg\psi \mid \psi_1 \vee \psi_2 \mid \phi$$

What's the difference w.r.t. LTL and CTL?

CTL* grammar

The grammar of CTL* has the same state formulas as CTL…

$$\phi ::= true \mid p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \exists\psi \mid \forall\psi$$

… but (subtly) extended path formulas (akin to LTL)

$$\psi ::= \bigcirc\psi \mid \Diamond\psi \mid \Box\psi \mid \psi_1 \mathsf{U} \psi_2 \mid \neg\psi \mid \psi_1 \vee \psi_2 \mid \phi$$

Main differences:

(1) no need to interleave quantifiers with temporal operators

(2) All state formulas are also path formulas

LTL as fragment CTL*

LTL formulas can be seen as CTL* formulas of the form

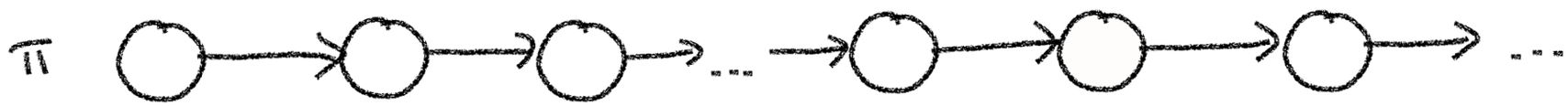
$$\forall \psi$$

where ψ is a path formula **without path quantifiers**

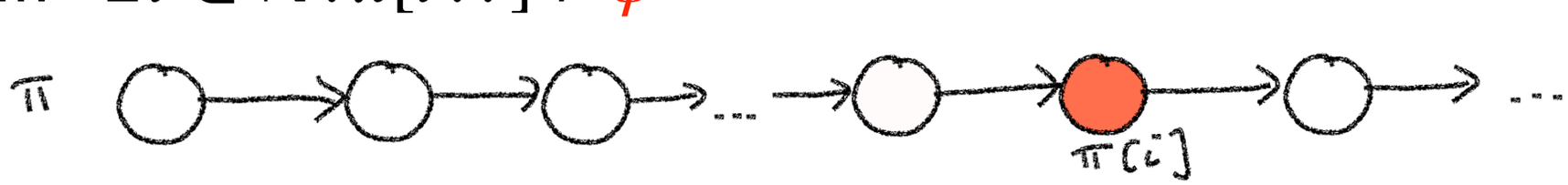
Semantics of path formulas

$\pi \models \phi$ iff $\pi[0] \models \phi$

$\pi \models \bigcirc \psi$ iff $\pi[1..] \models \psi$

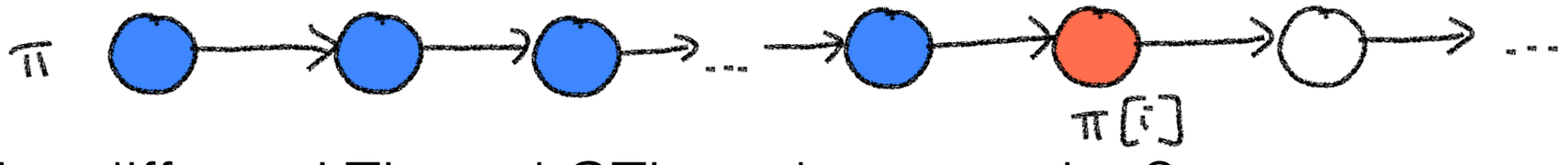


$\pi \models \Diamond \psi$ iff $\exists i \in \mathbb{N} . \pi[i\,..\,] \models \psi$



$\pi \models \Box \psi$ iff $\forall i \in \mathbb{N} . \pi[i\,..\,] \models \psi$

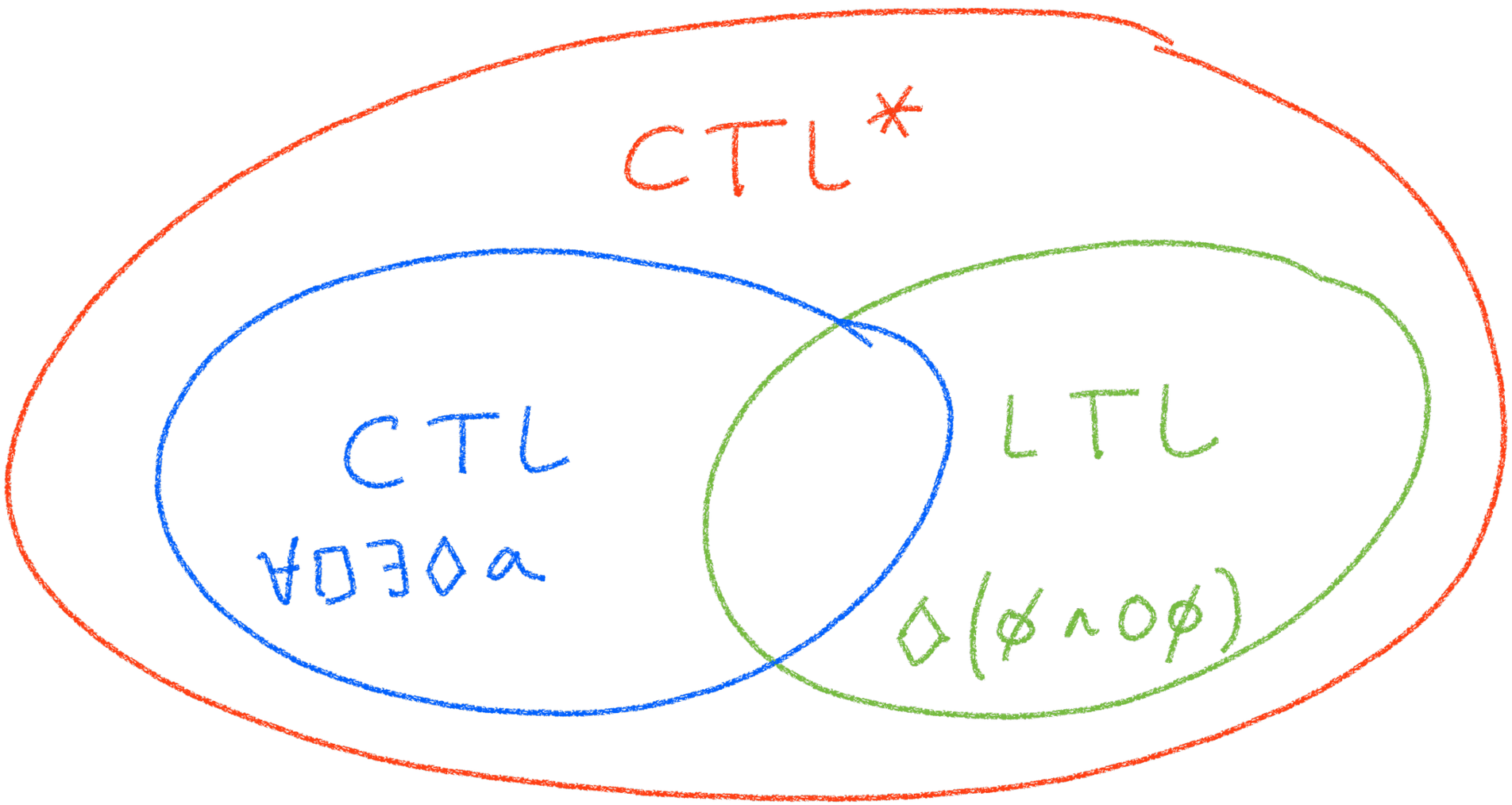
$\pi \models \psi_1 \mathsf{U} \psi_2$ iff $\exists i \in \mathbb{N} . \pi[i\,..\,] \models \psi_2 \wedge \forall 0 \leq j < i . \pi[j\,..\,] \models \psi_1$



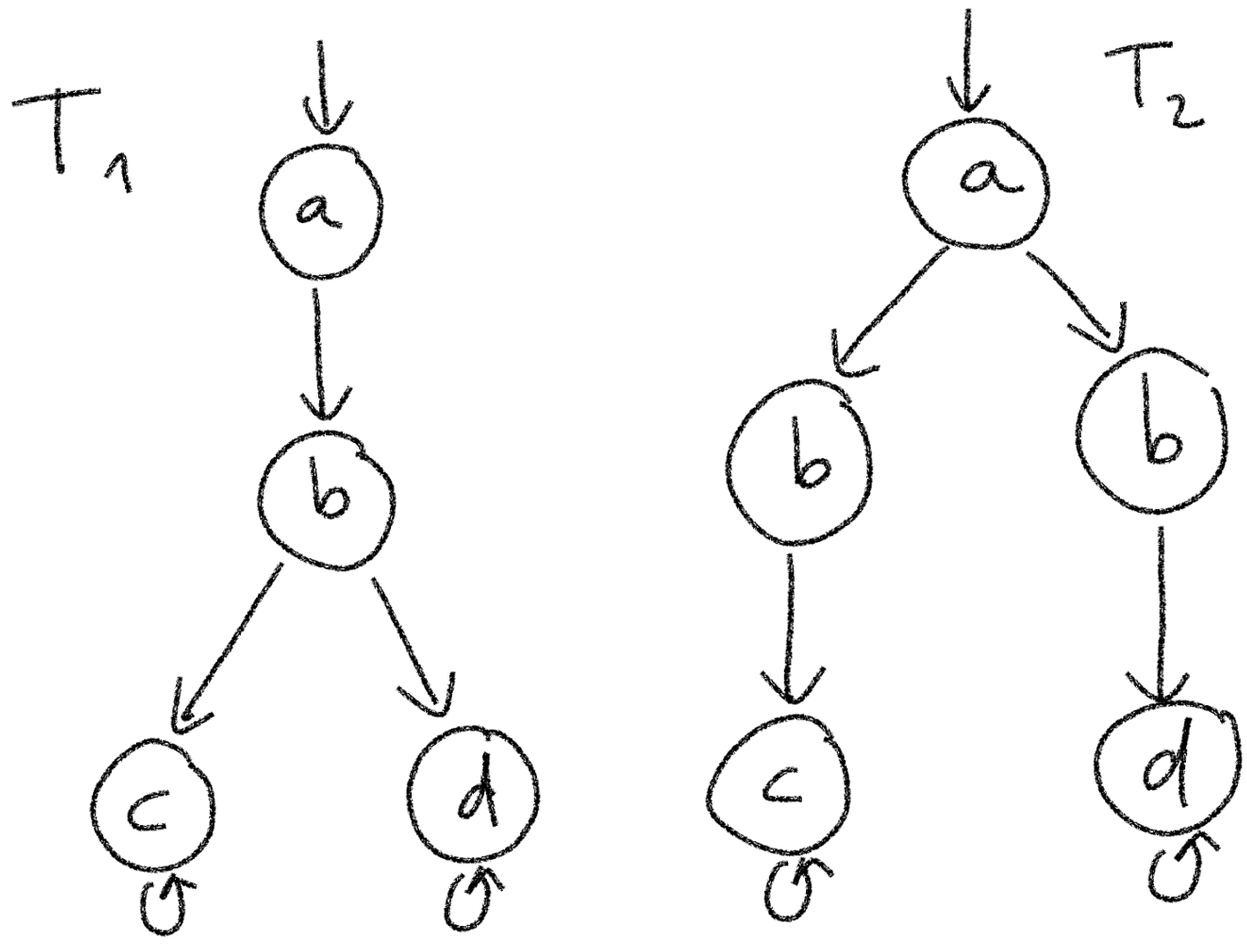
Can you spot the diff. wrt LTL and CTL path semantics?

The CTL* family

CTL is not the only logic for transition systems…



CTL can distinguish more than LTL

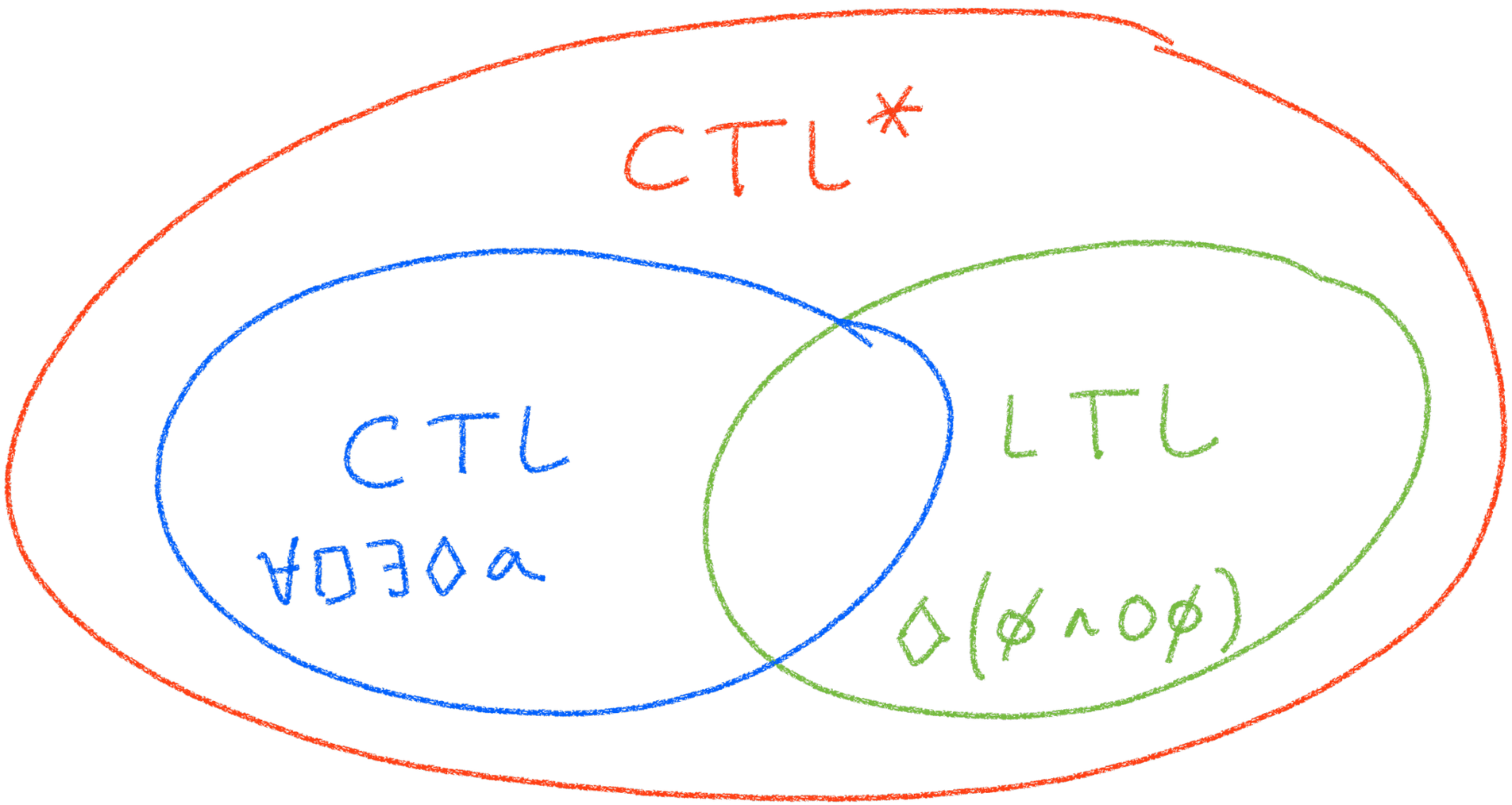


T_1 and T_2 have the same traces $\Rightarrow$ satisfy the same LTL formulas

but ... $T_1 \models \phi$, $T_2 \not\models \phi$ → Can you find such a CTL formula?

The CTL* family

CTL* is more expressive than both LTL and CTL*



As an example just take some combination of the above two formulas

Closing…

Key points of this lecture

Temporal Logics as an extension of propositional logics to reason about transition systems.

Computation Tree Logic (CTL) as a logic to reason computation trees.

Grammars for CTL: standard, minimalistic and existential normal form.

Formal semantics of CTL: satisfaction relation for states and paths.

Derived operators and equivalences between CTL formulas.

How to write CTL formulas in PRISM.

CTL* as a generalization of CTL and LTL.

APPENDIX: Exercises

Exercise 03.1

For the transition system below determine the set of states that satisfies the below CTL formulas

(1) $\forall \Diamond y$

(2) $\forall \Box y$

(3) $\forall \Box \forall \Diamond y$

(4) $\forall \Diamond g$

(5) $\forall \Diamond b$

(6) $\exists \Box y$

(7) $\exists \Box \neg g$

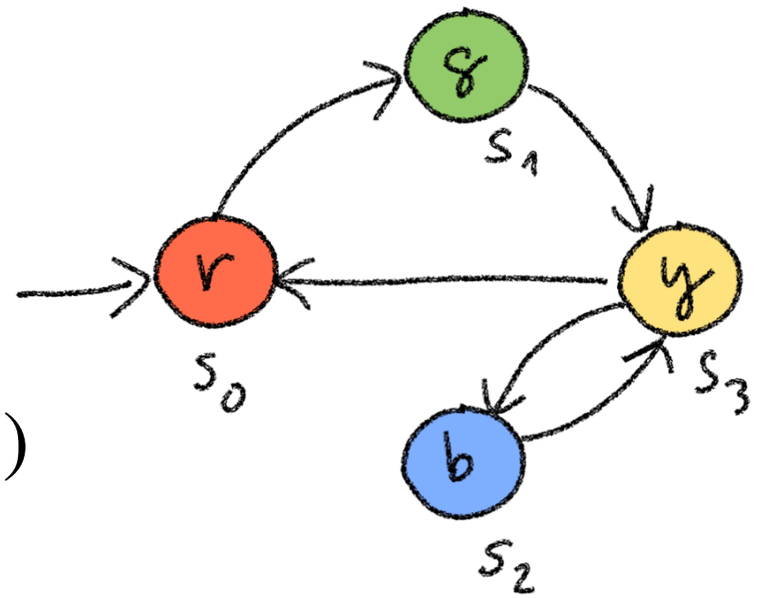
(8) $\forall (b \cup \neg b)$

(9) $\exists \bigcirc (g \vee r)$

(10) $\forall (g \cup \exists \Box (b \vee y))$

(11) $\forall (g \cup \exists (y \cup r))$

(12) $\forall (\neg b \cup b)$



NOTE: Reason using the semantics of CTL. If you are in doubt you can double-check with PRISM.

Exercise 03.2

Consider the following statements about CTL formulas:

(1) $\exists \Box \phi$ implies $\forall \Box \phi$

(2) $\forall \Box \phi$ implies $\exists \Box \phi$

(3) $(\forall \Diamond \phi_1) \vee (\forall \Diamond \phi_2)$ implies $\forall \Diamond (\phi_1 \vee \phi_2)$

(4) $\forall \Diamond (\phi_1 \vee \phi_2)$ implies $(\forall \Diamond \phi_1) \vee (\forall \Diamond \phi_2)$

For each statement determine if it holds or not.

If the statement does not hold, provide a counterexample (a transition system that satisfies the formula on the left but not the one on the right).

If the statement holds, provide an explanation (similar to the ones in the book or slides).

Exercise 03.3

Consider again exercise 03.1.

(a) Drop all quantifiers to obtain LTL formulas. You will get the formulas of exercise 02.1.
(b) For each of the formulas 1-10 compare the CTL and LTL formulas. Do they agree on the result (holds / does not hold) on the example transition system? Are they equivalent? Explain your answer.

Exercise 03.4

Show that the LTL formula below is not equivalent to any of the two CTL formulas. You just need one transition system for each CTL formula where they do not agree.

$$\phi_\wedge \equiv \forall \Box (\phi \wedge \bigcirc \phi)$$

$$\phi_2 \equiv \forall \Box (\phi \wedge \exists \bigcirc \phi)$$

$$\phi_3 \equiv \forall \Box (\phi \wedge \forall \bigcirc \phi)$$