https://didawiki.di.unipi.it/doku.php/magistraleinformatica/mpp/start

MPP 2025/26 (0077A, 9CFU)

Models for Programming Paradigms

Roberto Bruni Filippo Bonchi
http://www.di.unipi.it/~bruni/

22a - Temporal logic

Testing

how do you guarantee that your code is correct?

testing can show the presence of bugs

not their absence

coverage of all cases: difficult to achieve

especially in concurrent systems!
(because of nondeterminism)

Formal logics

what does it mean to be correct? to satisfy some properties

how are these properties expressed? in some syntax

formal logics serve to express properties about programs

safety: something bad will never happen

liveness: something good will happen

model checking are certain properties satisfied (by a model of the program)?

Modal logics

notion of time (discrete, infinite)

properties of states (atomic proposition)

modal operators at the next step

at any next step

(like HM logic)

fix point operators recursively defined formulas

minimal / maximal fixpoint

(meaning of a formula:
the set of states where it holds)

Temporal logics

notion of time (discrete, infinite)

properties of states (atomic proposition)

linear operators at the next instant

always

never

eventually

path quantifiers (nondeterministic systems)

for all possible futures

in a possible future

LTL
Linear temporal logic

Linear Temporal Logic

models

$$0 \to 1 \to 2 \to \cdots \to n \to \cdots$$

syntax

$$
\begin{array}{lcll}
\psi & ::= & \mathbf{tt} \mid \mathbf{ff} \mid \neg\psi \mid \psi_0 \wedge \psi_1 \mid \psi_0 \vee \psi_1 & \\
 & \mid & p & \text{atomic proposition} \quad p \in P \\
 & \mid & \mathsf{O}\psi & \text{NEXT: } \psi \text{ holds at the next instant of time} \\
 & \mid & \mathsf{F}\psi & \text{FINALLY: } \psi \text{ holds sometimes in the future} \\
 & \mid & \mathsf{G}\psi & \text{GLOBALLY: } \psi \text{ holds always in the future} \\
 & \mid & \psi_0 \mathsf{U} \psi_1 & \text{UNTIL: } \psi_0 \text{ holds until } \psi_1 \text{ is true}
\end{array}
$$

$\mathsf{O}\psi$ sometimes written $\mathsf{X}\psi$ or $\mathsf{N}\psi$

Linear Structure

$$S : P \to \wp(\mathbb{N})$$

set of atomic propositions

$S(p)$ is the set of time instants in which p holds

$$S(p) = \{n \mid p \text{ holds at } n\}$$

Shift $S^k : P \to \wp(\mathbb{N})$

$$S^k(p) = \{n - k \mid n \geq k \wedge n \in S(p)\}$$

$$S^k(p) = \{m \mid m + k \in S(p)\}$$

Example

$$S : P \to \wp(\mathbb{N})$$

$$\begin{tikzcd}[row sep=tiny] \bullet \arrow[r, bend left] & \bullet \arrow[r, bend left] & \bullet \arrow[r, bend left] & \cdots \arrow[r, bend left] & \bullet \arrow[r, bend left] & \cdots \\ \text{0} & \text{1} & \text{2} & & \text{n} & \\ \{p, q\} & \{p\} & \{q\} & & \{q, r\} & \end{tikzcd}$$

$$S(p) = \{0, 1, ...\} \qquad S(r) = \{n, ...\} \qquad S(q) = \{0, 2, n, ...\}$$

$$S^2 : P \to \wp(\mathbb{N}) \qquad S^2(q) = \{0, n-2, ...\}$$

$$\begin{tikzcd}[row sep=tiny] \bullet \arrow[r, bend left] & \bullet \arrow[r, bend left] & \bullet \arrow[r, bend left] & \cdots \arrow[r, bend left] & \bullet \arrow[r, bend left] & \cdots \\ & & \text{0} & & \text{n-2} & \\ & & \{q\} & & \{q, r\} & \end{tikzcd}$$

LTL: satisfaction

LTL formula

$$S \models \psi$$

linear structure

LTL: satisfaction

current time: 0

$S \models \mathbf{tt}$

$S \models \neg\psi$ iff $S \not\models \psi$

$S \models \psi_0 \wedge \psi_1$ iff $S \models \psi_0$ and $S \models \psi_1$

$S \models \psi_0 \vee \psi_1$ iff $S \models \psi_0$ or $S \models \psi_1$

$S \models p$ iff $0 \in S(p)$

$S \models \mathsf{O}\psi$ iff $S^1 \models \psi$

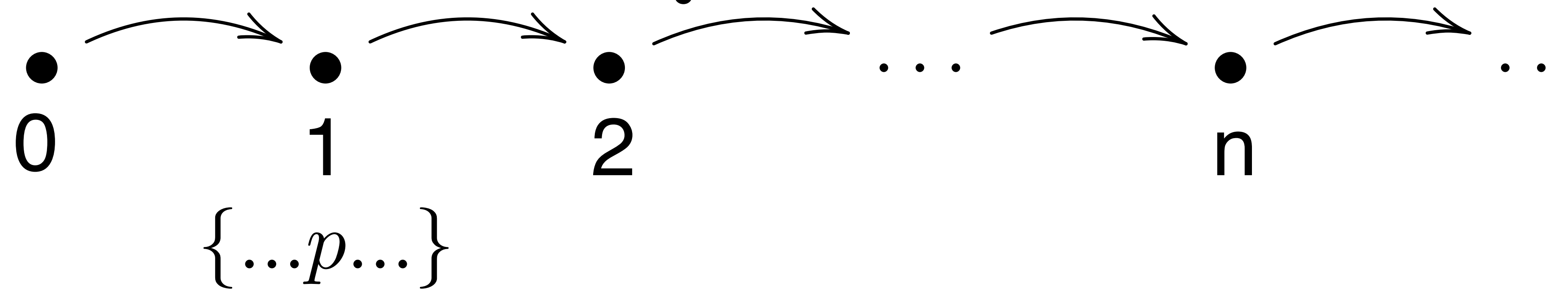
$S \models \mathsf{F}\psi$ iff $\exists k \in \mathbb{N}.\ S^k \models \psi$

$S \models \mathsf{G}\psi$ iff $\forall k \in \mathbb{N}.\ S^k \models \psi$

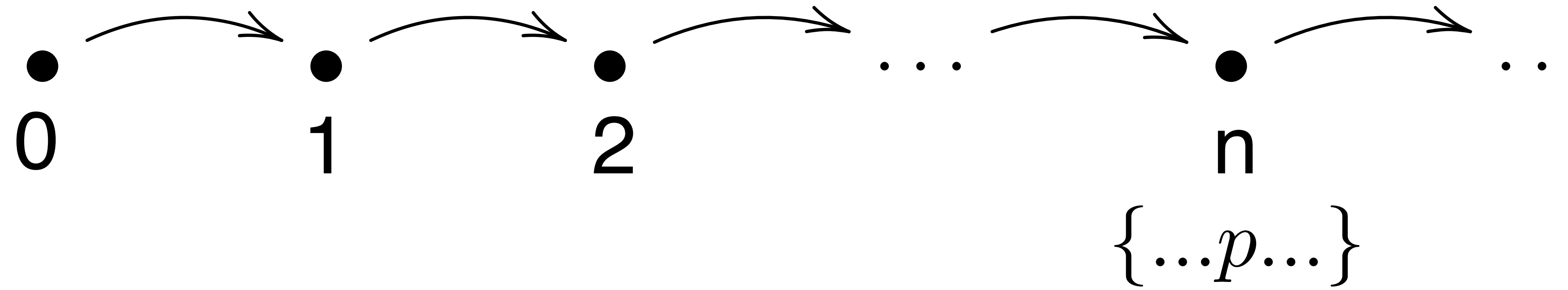
$S \models \psi_0 \mathsf{U} \psi_1$ iff $\exists k \in \mathbb{N}.\ S^k \models \psi_1$ and $\forall i < k.\ S^i \models \psi_0$

Examples

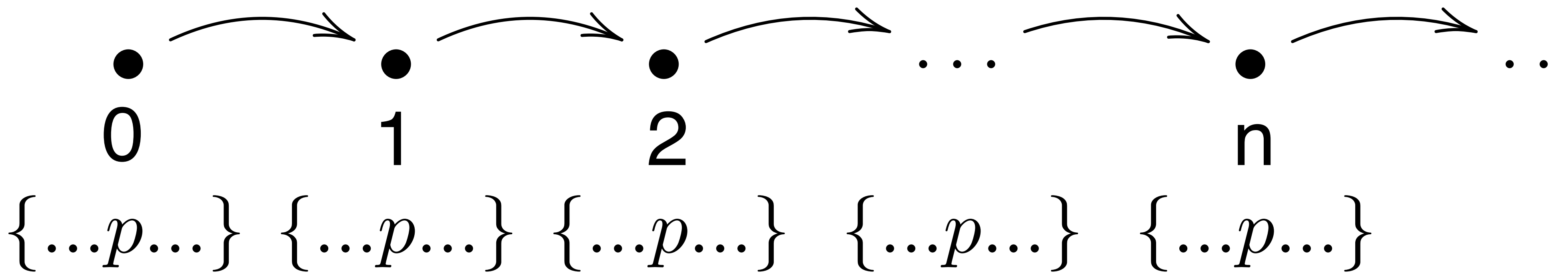
$S \models \mathsf{O}\ p$



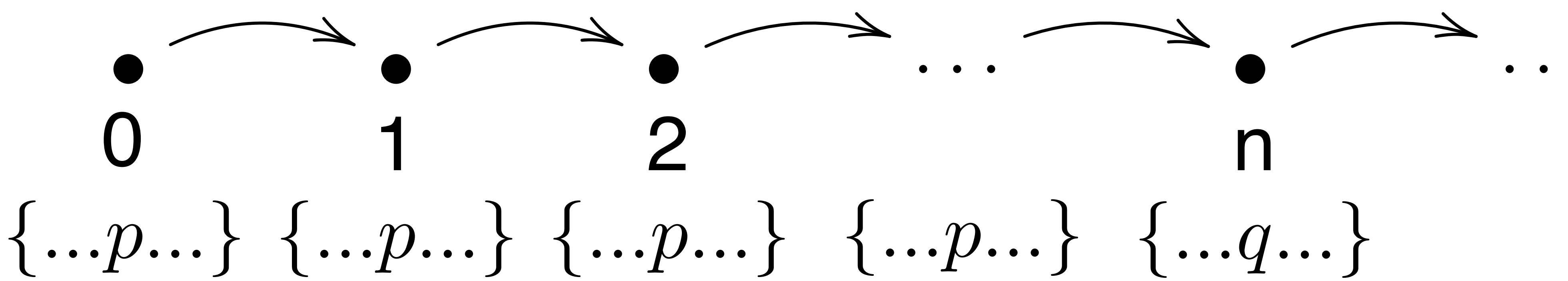
$S \models \mathsf{F}\ p$



$S \models \mathsf{G}\ p$



$S \models p\ \mathsf{U}\ q$



LTL: equivalent formulas

$$\psi_0 \equiv \psi_1 \quad \text{iff} \quad \forall S.\ S \models \psi_0 \Leftrightarrow S \models \psi_1$$

$$\mathsf{F}\ \psi \equiv \mathbf{tt}\ \mathsf{U}\ \psi$$

$$\begin{aligned} \mathsf{G}\ \psi &\equiv \neg(\mathsf{F}\ \neg\psi) \\ &\equiv \neg(\mathbf{tt}\ \mathsf{U}\ \neg\psi) \end{aligned}$$

$$\psi_0 \Rightarrow \psi_1 \triangleq \psi_1 \vee \neg\psi_0$$

Examples

$$\mathsf{G}\ \neg error$$

error will never arise

$$press \Rightarrow \mathsf{F}\ error$$

if you press now, an error will arise

$$\mathsf{G}\ \mathsf{F}\ enter$$

enter will happen infinitely often (fairness)

$$\mathsf{F}\ \mathsf{G}\ idle$$

the system will stay idle from some time in the future onward

$$\mathsf{G}\ (req \Rightarrow (req\ \mathsf{U}\ eval))$$

whenever a request is made, it holds until evaluated

LTL
automata-like models

LTL, again

models

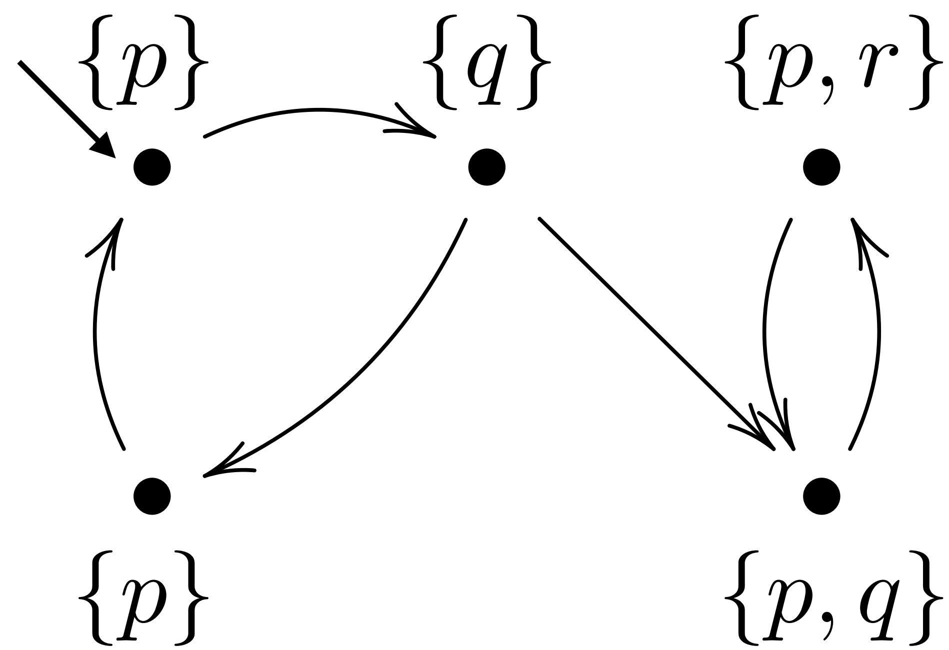
$$\underset{0}{\bullet} \longrightarrow \underset{1}{\bullet} \longrightarrow \underset{2}{\bullet} \longrightarrow \cdots \longrightarrow \underset{n}{\bullet} \longrightarrow \cdots$$

syntax

$$\begin{array}{lcll}
\psi & ::= & \mathbf{tt} \mid \mathbf{ff} \mid \neg\psi \mid \psi_0 \wedge \psi_1 \mid \psi_0 \vee \psi_1 & \\
 & \mid & p & \text{atomic proposition} \quad p \in P \\
 & \mid & \mathsf{O}\psi & \text{NEXT: } \psi \text{ holds at the next instant of time} \\
 & \mid & \mathsf{F}\psi & \text{FINALLY: } \psi \text{ holds sometimes in the future} \\
 & \mid & \mathsf{G}\psi & \text{GLOBALLY: } \psi \text{ holds always in the future} \\
 & \mid & \psi_0 \mathsf{U} \psi_1 & \text{UNTIL: } \psi_0 \text{ holds until } \psi_1 \text{ is true}
\end{array}$$

$\mathsf{O}\psi$ sometimes written $\mathsf{X}\psi$ or $\mathsf{N}\psi$

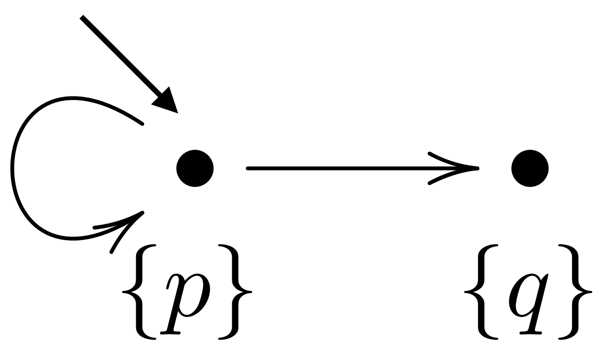
Automata-like models



the formula must be satisfied along all (infinite) traces

(if we enter a deadlock state, the last state is repeated forever)

Exercise



$\not\models$ F q ✗ $\{p\}\ \{p\}\ \{p\}\ \cdots$

$\not\models$ G p ✗ $\{p\}\ \{q\}\ \{q\}\ \cdots$

$\not\models$ p U q ✗ $\{p\}\ \{p\}\ \{p\}\ \cdots$

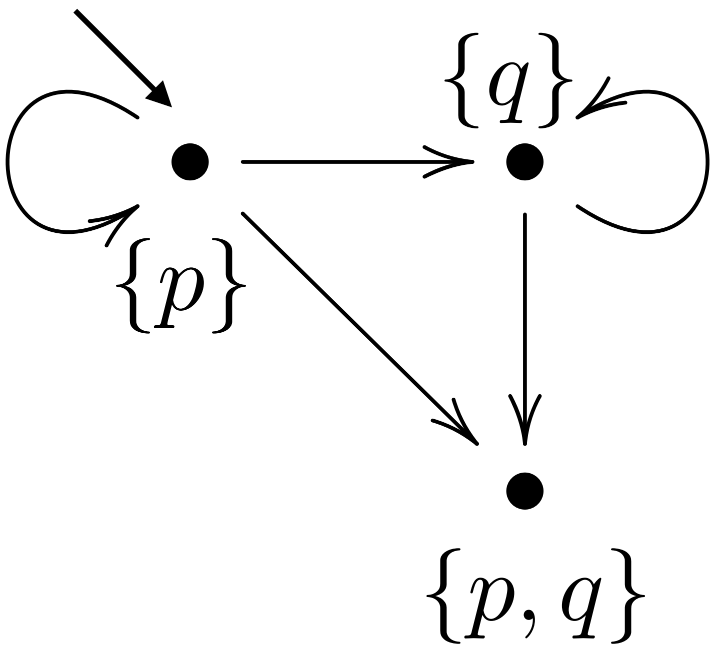
$\models$ q U p ✓

$\models$ G$(q \Rightarrow$ G $q)$ ✓

the formula must be satisfied along all (infinite) traces

(if we enter a deadlock state, the last state is repeated forever)

Exercise



$\not\models \ \mathsf{G}(q \ \mathsf{U}\ p)$ ✗ $\{p\} \cdots \{p\}\ \{q\}\ \{q\} \cdots$

$\models \ \mathsf{G}\ p \lor \mathsf{F}\ q$ ✓

$\not\models \ \mathsf{F}\ q \Rightarrow \neg\mathsf{G}\ p$ ✗ $\{p\}\ \{p, q\}\ \{p, q\} \cdots$

$\models \ G(q \Rightarrow \mathsf{O}\ q)$ ✓

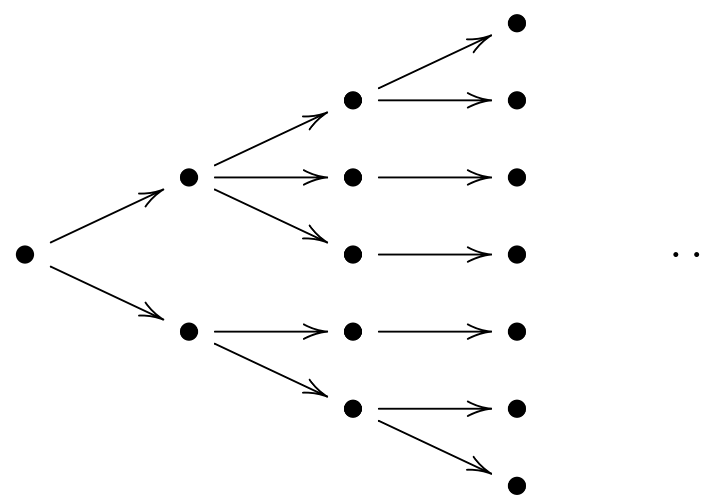
the formula must be satisfied along all (infinite) traces

(if we enter a deadlock state, the last state is repeated forever)

CTL*, CTL
Computational tree logic

Computational Tree Logic

models



syntax (CTL*)

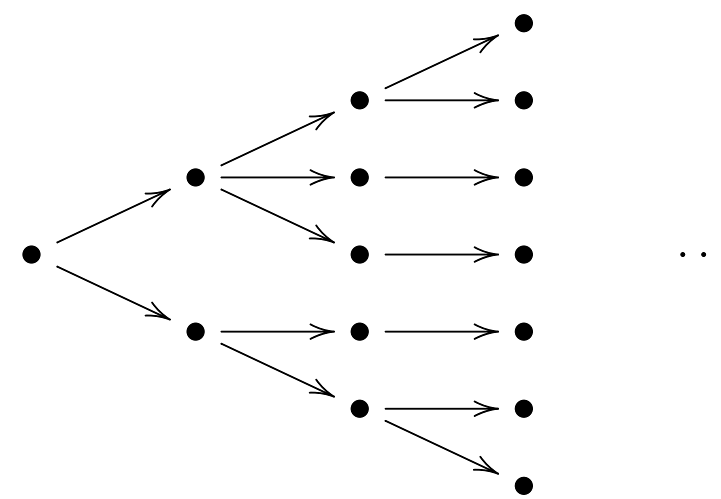
$$\psi \quad ::= \quad \mathbf{tt} \mid \mathbf{ff} \mid \neg\psi \mid \psi_0 \wedge \psi_1 \mid \psi_0 \vee \psi_1$$ classical ops

$$\mid \quad p \mid \mathsf{O}\psi \mid \mathsf{F}\psi \mid \mathsf{G}\psi \mid \psi_0 \mathsf{U} \psi_1$$ linear ops

$\mid \quad \mathsf{E}\psi$ POSSIBLY: there is a path that satisfies

$\mid \quad \mathsf{A}\psi$ ALWAYS: every path satisfies ψ

Infinite Tree



$T = (V, \to)$ directed graph

tree

$v_0 \in V$ root: a distinguished vertex (no incoming arc)

exactly one directed path from v_0 to any other vertex $v \in V$

infinite

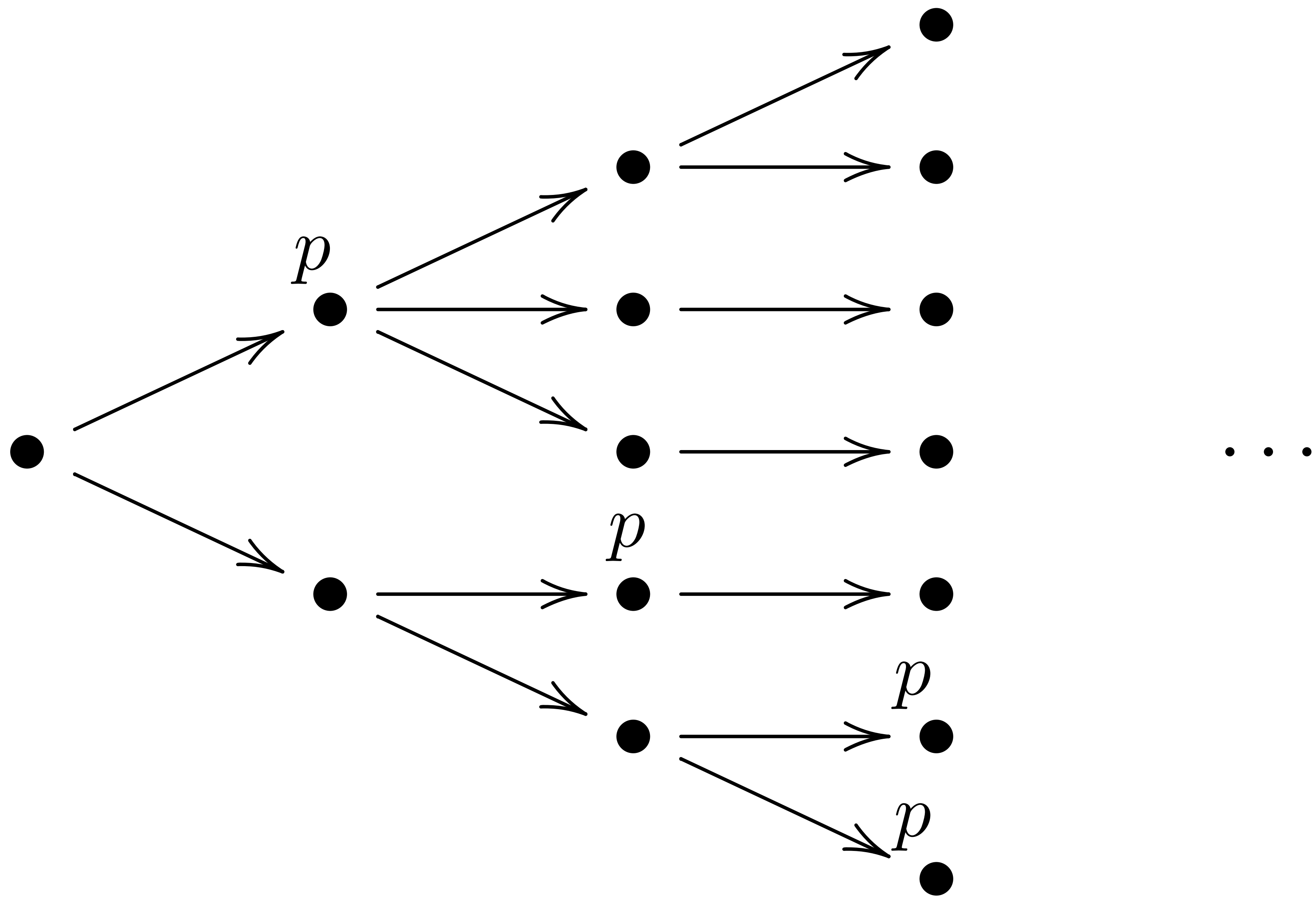
every node has a child

Branching Structure

$T = (V, \to)$ infinite tree

$S : P \to \wp(V)$

$$S(p) = \{x \in V \mid x \text{ satisfies } p\}$$



Infinite Path

$T = (V, \rightarrow)$ $S : P \rightarrow \wp(V)$ branching structure

infinite path $T = (V, \rightarrow)$ $\pi : \mathbb{N} \rightarrow V$ $(\pi = v_0 v_1 \cdots)$

such that $\forall k \in \mathbb{N}.\ v_k \rightarrow v_{k+1}$

path shifting $\pi = v_0 v_1 \cdots$ $\pi^k = v_k v_{k+1} \cdots$

$\pi : \mathbb{N} \rightarrow V$ $\pi^k : \mathbb{N} \rightarrow V$

$\pi^k(i) = \pi(k + i)$

CTL*: satisfaction

infinite path

CTL* formula

$$S, \pi \models \psi$$

branching structure

$S : P \to \wp(V)$
$\pi : \mathbb{N} \to V$

CTL*: satisfaction

$S, \pi \models \mathbf{tt}$

$S, \pi \models \neg\psi$ iff $S, \pi \not\models \psi$

$S, \pi \models \psi_0 \wedge \psi_1$ iff $S, \pi \models \psi_0$ and $S, \pi \models \psi_1$

$S, \pi \models \psi_0 \vee \psi_1$ iff $S, \pi \models \psi_0$ or $S, \pi \models \psi_1$

$S, \pi \models p$ iff $\pi(0) \in S(p)$

$S, \pi \models \mathsf{O}\psi$ iff $S, \pi^1 \models \psi$

$S, \pi \models \mathsf{F}\psi$ iff $\exists k \in \mathbb{N}.\ S, \pi^k \models \psi$

$S, \pi \models \mathsf{G}\psi$ iff $\forall k \in \mathbb{N}.\ S, \pi^k \models \psi$

$S, \pi \models \psi_0 \mathsf{U} \psi_1$ iff $\exists k \in \mathbb{N}.\ S, \pi^k \models \psi_1$ and $\forall i < k.\ S, \pi^i \models \psi_0$

state ops

$S, \pi \models \mathsf{E}\psi$ iff $\exists \pi'.\ \pi'(0) = \pi(0)$ and $S, \pi' \models \psi$

$S, \pi \models \mathsf{A}\psi$ iff $\forall \pi'.\ \pi'(0) = \pi(0)$ implies $S, \pi' \models \psi$

path ops

CTL*: equivalent formulas

$$\psi_0 \equiv \psi_1 \quad \text{iff} \quad \forall S.\ \forall \pi.\ \ S, \pi \models \psi_0 \Leftrightarrow S, \pi \models \psi_1$$

$$\mathsf{A}\ \psi \equiv \neg(\mathsf{E}\ \neg\psi)$$

$$\mathsf{A}\ \mathsf{A}\ \psi \equiv \mathsf{A}\ \psi$$

$$\mathsf{A}\ \mathsf{E}\ \psi \equiv \mathsf{E}\ \psi$$

LTL formulas as CTL* ones

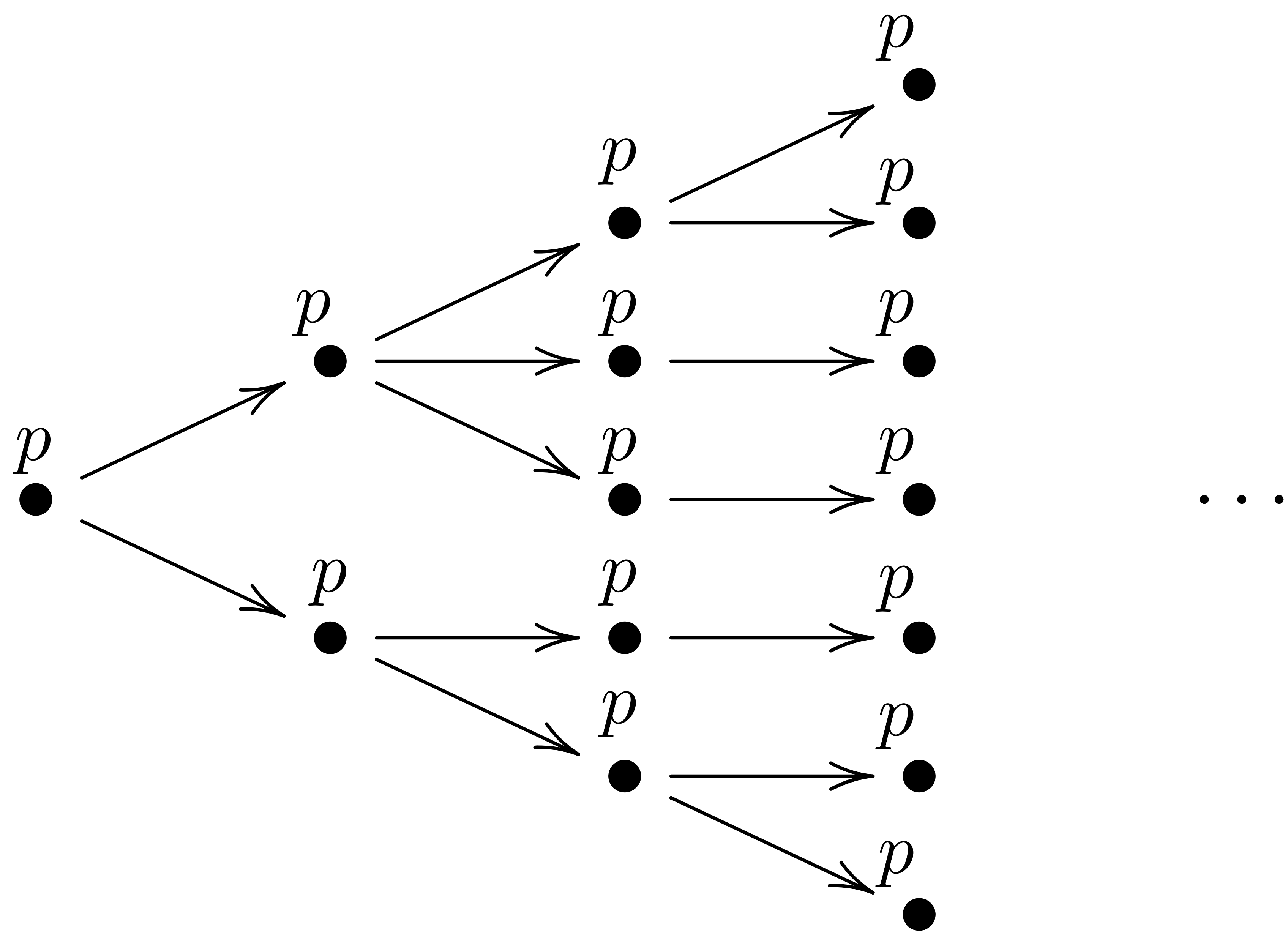
ψ $\qquad$ $\mathsf{A}\ \psi$

Examples

E O ψ	analogous to HML formula $\Diamond\psi$
A O ψ	analogous to HML formula $\Box\psi$
A G p	*p* holds at any reachable state
E F p	*p* holds at some reachable state
A F p	on every path there is a state where *p* holds
E $(p$ U $q)$	there is a path where *p* holds until *q*

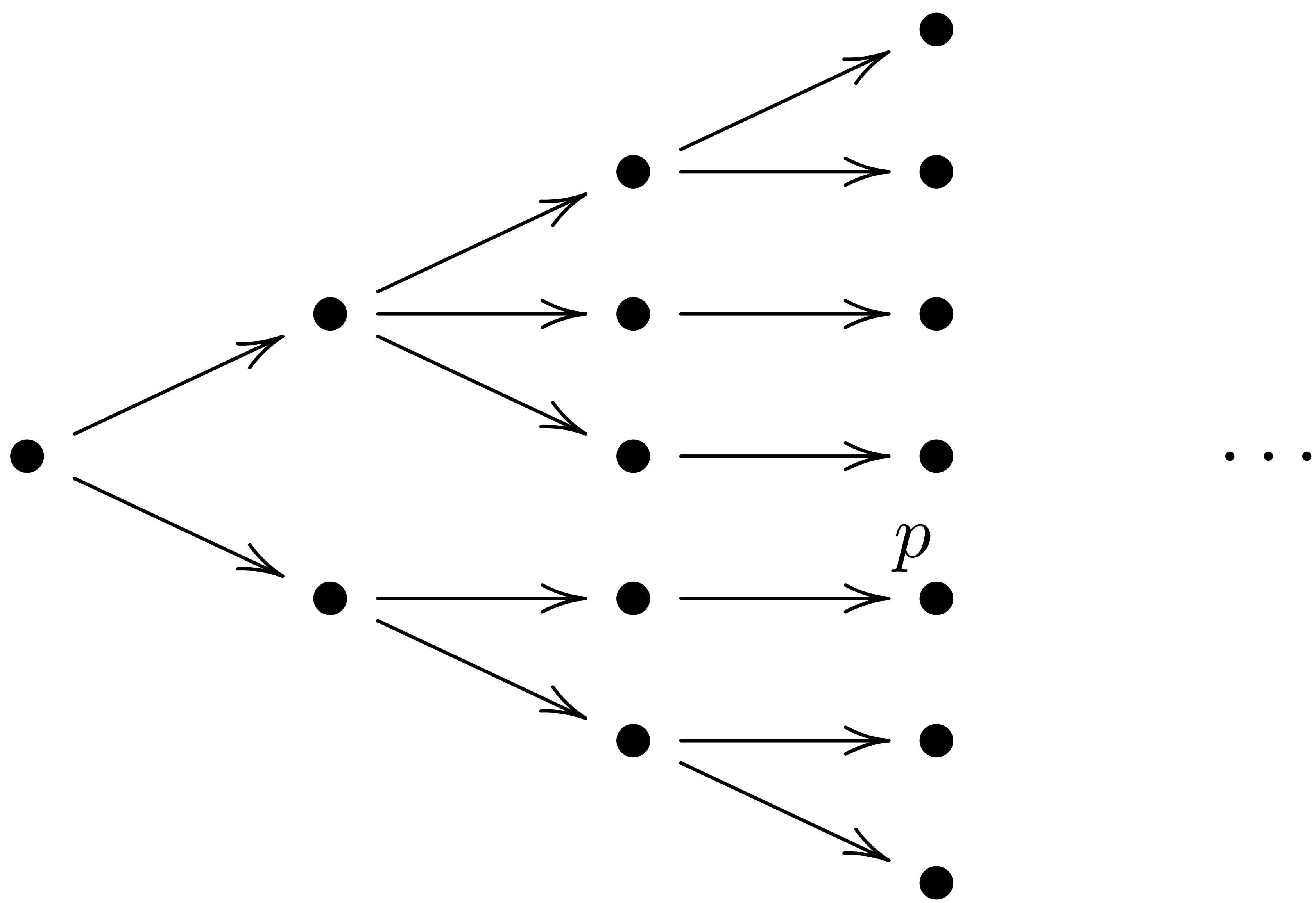
Example

A G p



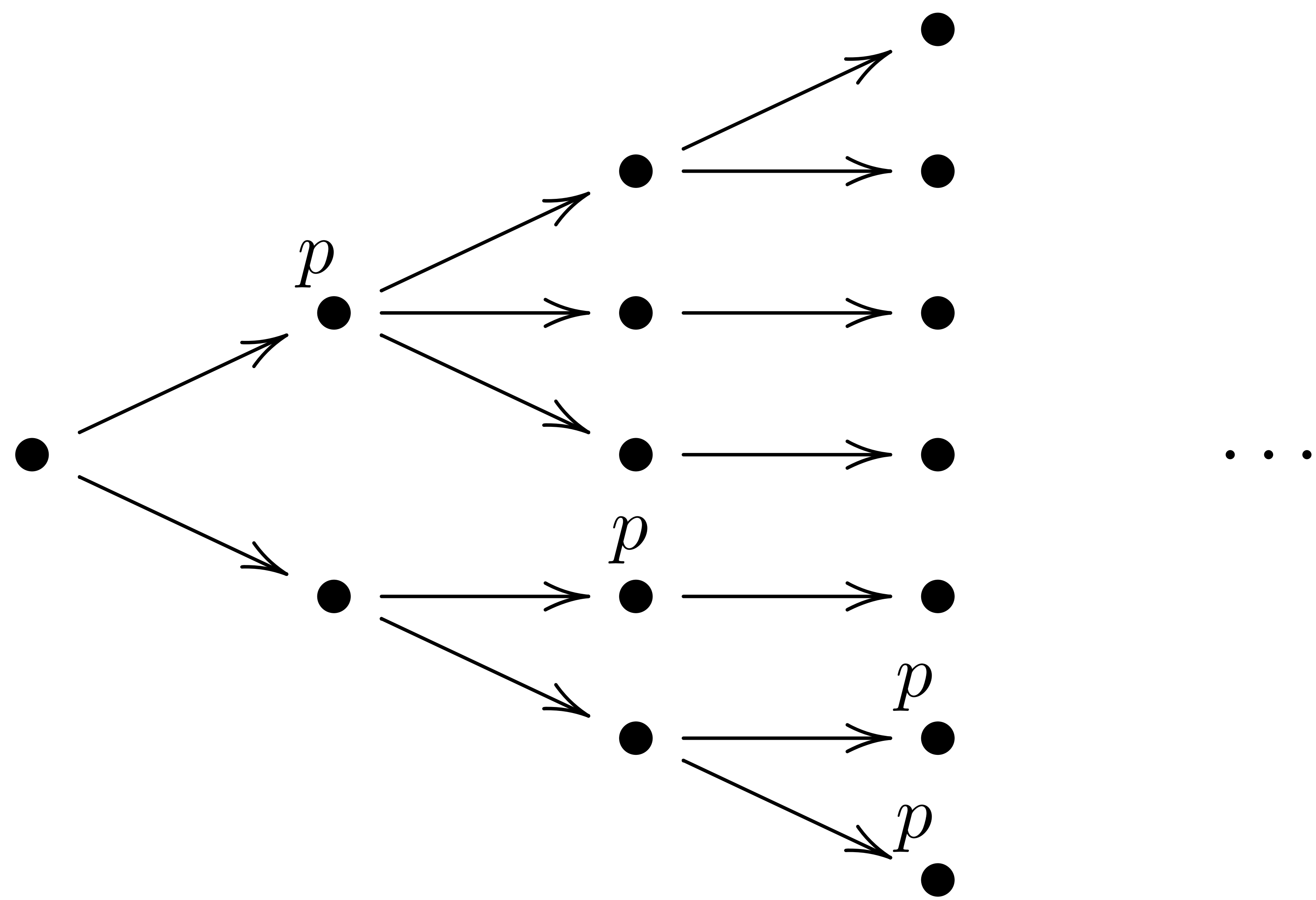
Example

E F p



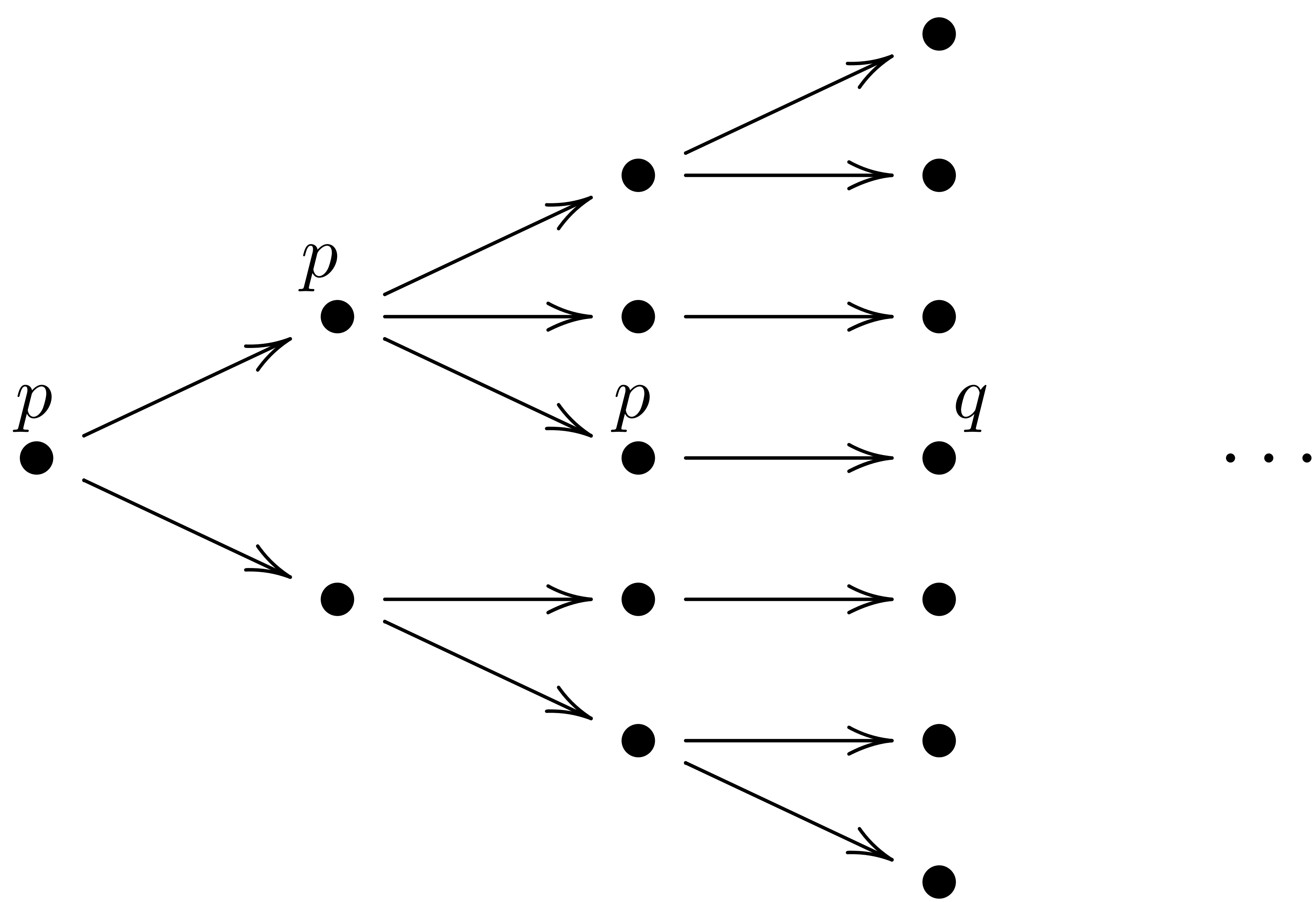
Example

A F p



Example

$\mathsf{E}\ (p\ \mathsf{U}\ q)$



CTL

CTL formulas

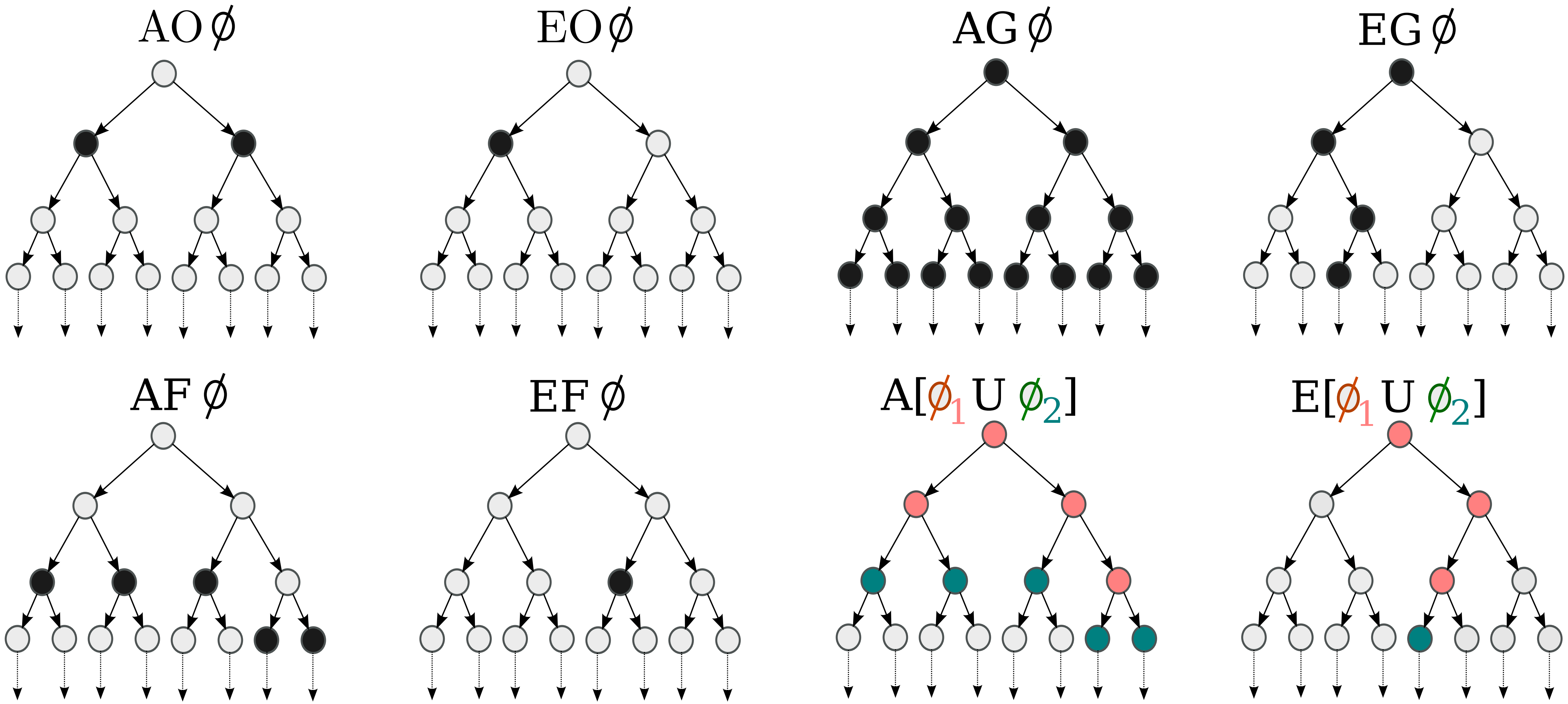
each path op (A/E) appears immediately before a linear op

each linear op (O/F/G/U) appears immediately after a path op

$\mathsf{E}\ \mathsf{O}\ \psi$	$\mathsf{E}\ \mathsf{F}\ \psi$	$\mathsf{E}\ \mathsf{G}\ \psi$	$\mathsf{E}\ (\psi_0\ \mathsf{U}\ \psi_1)$
$\mathsf{A}\ \mathsf{O}\ \psi$	$\mathsf{A}\ \mathsf{F}\ \psi$	$\mathsf{A}\ \mathsf{G}\ \psi$	$\mathsf{A}\ (\psi_0\ \mathsf{U}\ \psi_1)$

$\mathsf{A}\ \mathsf{G}\ \mathsf{F}\ \psi$ CTL*, not CTL

CTL formulas



CTL: minimal set of ops

$$\neg\,\cdot \qquad \cdot \vee \cdot \qquad \mathsf{EO}\,\cdot \qquad \mathsf{EG}\,\cdot \qquad \mathsf{E}(\cdot \mathsf{U} \cdot)$$

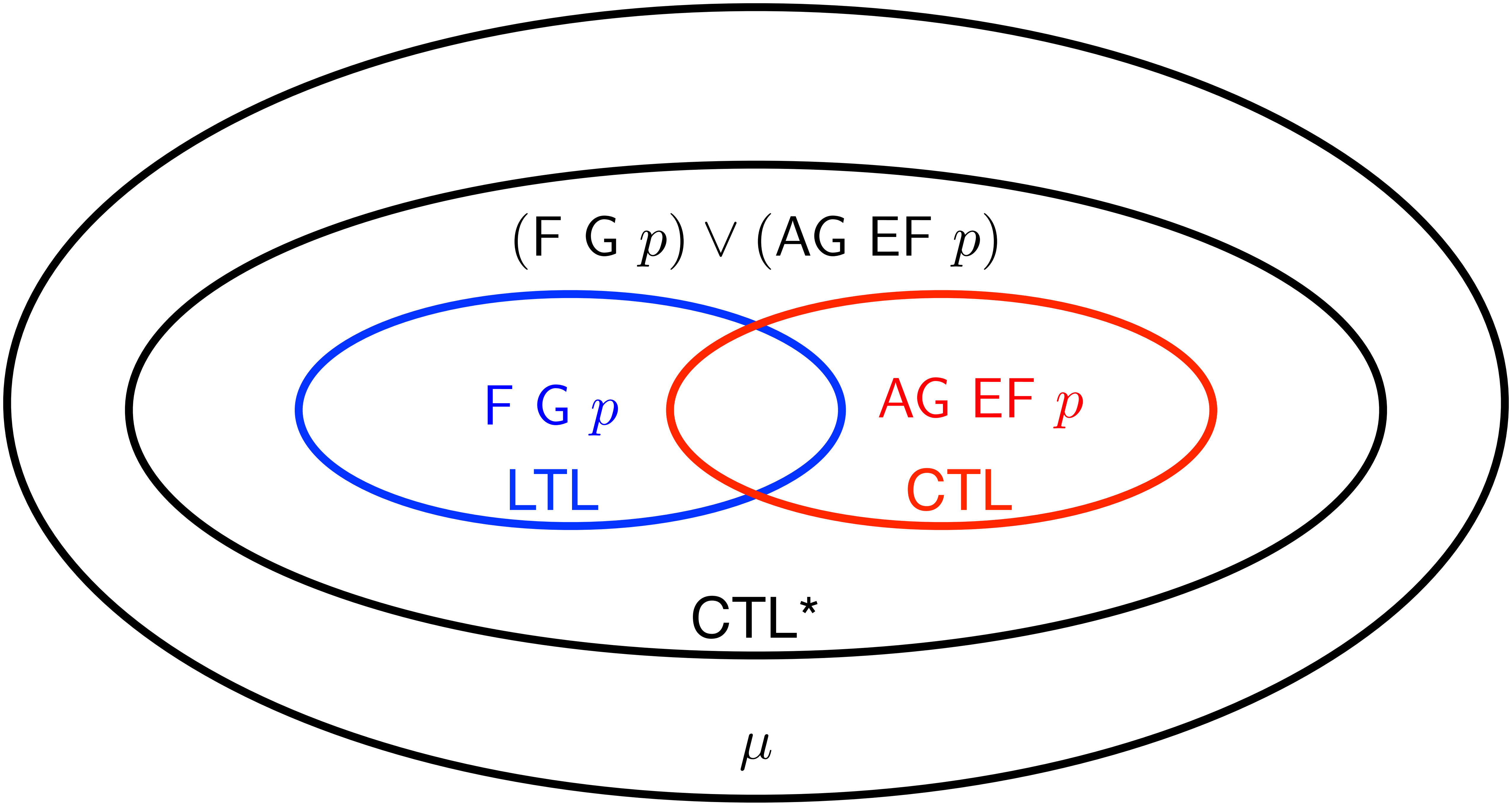
$$\mathsf{AO}\ \psi \equiv \neg(\mathsf{EO}\ \neg\psi)$$

$$\mathsf{AF}\ \psi \equiv \neg(\mathsf{EG}\ \neg\psi) \qquad\qquad \mathsf{EF}\ \psi \equiv \mathsf{E}(\mathbf{tt}\ \mathsf{U}\ \psi)$$

$$\begin{aligned}\mathsf{AG}\ \psi &\equiv \neg(\mathsf{EF}\ \neg\psi)\\ &\equiv \neg\mathsf{E}(\mathbf{tt}\ \mathsf{U}\neg\psi)\end{aligned}$$

$$\mathsf{A}\ (\psi_0\ \mathsf{U}\ \psi_1) \equiv \neg(\mathsf{EG}\ \neg\psi_1 \vee \mathsf{E}(\neg\psi_1\ \mathsf{U}\ \neg(\psi_0 \vee \psi_1)))$$

Expressiveness



Temporal logics exercises

Shared resource

Two processes p_1 and p_2 want to access a single shared resource r. Consider the atomic propositions:

req_i: holds when process p_i is requesting access to r;
use_i: holds when process p_i has had access to r;
rel_i: holds when process p_i has released r.

with $i \in [1, 2]$. Use LTL formulas to specify the following properties:

1. *mutual exclusion*: r is accessed by only one process at a time;

2. *release*: every time p_1 accesses r, it releases r after some time;

3. *priority*: whenever both p_1 and p_2 require r, p_1 is granted access first;

4. *no starvation*: whenever p_1 requires r, it is eventually granted access.

Shared resource

req_i — p_i requests r

use_i — p_i has access to r

rel_i — p_i releases r

mutual exclusion	$\mathsf{G}\ \neg(use_1 \wedge use_2)$
release	$\mathsf{G}\ (use_1 \Rightarrow \mathsf{F}\ rel_1)$
priority	$\mathsf{G}\ ((req_1 \wedge req_2) \Rightarrow ((\neg use_2)\ \mathsf{U}\ (use_1 \wedge \neg use_2)))$
no starvation	$\mathsf{G}\ (req_1 \Rightarrow \mathsf{F}\ use_1)$

1. *mutual exclusion*: r is accessed by only one process at a time;
2. *release*: every time p_1 accesses r, it releases r after some time;
3. *priority*: whenever both p_1 and p_2 require r, p_1 is granted access first;
4. *no starvation*: whenever p_1 requires r, it is eventually granted access.

Exercise: dogs

Three dogs live in a house with two couches and a front garden. Let $couch_{i,j}$ represent the predicate "the dog i sits on couch j" and $garden_i$ represent the predicate "the dog i plays in the front garden".

1. Write an LTL formula expressing the fact that whenever dog 1 plays in the garden then he keeps playing until he sits on some couch (but he may also play forever).

2. Write a CTL formula expressing the fact that dog 2 eventually plays in the garden whenever couch 1 is occupied by another dog.

Exercise: dogs

$couch_{i,j}$ — i sits on j

$garden_i$ — i plays in the garden

LTL

$\mathsf{G}\ (garden_1 \Rightarrow ((\mathsf{G}\ garden_1) \vee (garden_1\ \mathsf{U}\ (couch_{1,1} \vee couch_{1,2}))))$

CTL $\mathsf{AG}\ ((couch_{1,1} \vee couch_{3,1}) \Rightarrow \mathsf{AF}\ garden_2)$

μ-calculus $\nu x.\ ((\neg couch_{3,1} \vee \neg couch_{3,2}) \wedge \Box x)$

1. Write an LTL formula expressing the fact that whenever dog 1 plays in the garden then he keeps playing until he sits on some couch (but he may also play forever).
2. Write a CTL formula expressing the fact that dog 2 eventually plays in the garden whenever couch 1 is occupied by another dog.
3. Write a μ-calculus formula expressing the fact that no more than one couch is occupied at any time by dog 3.