



CentraleSupélec

université  
PARIS-SACLAY



CentraleSupélec

# CONCEPTION ET VÉRIFICATION DE SYSTÈMES CRITIQUES

## LA SPÉCIFICATION DES PROPRIÉTÉS AVEC LA LOGIQUE CTL

🎓 2A Cursus Ingénieurs - ST5 : Modélisation fonctionnelle et régulation

🏛️ CentraleSupélec - Université Paris-Saclay - 2024/2025



**Idir AIT SADOUNE**

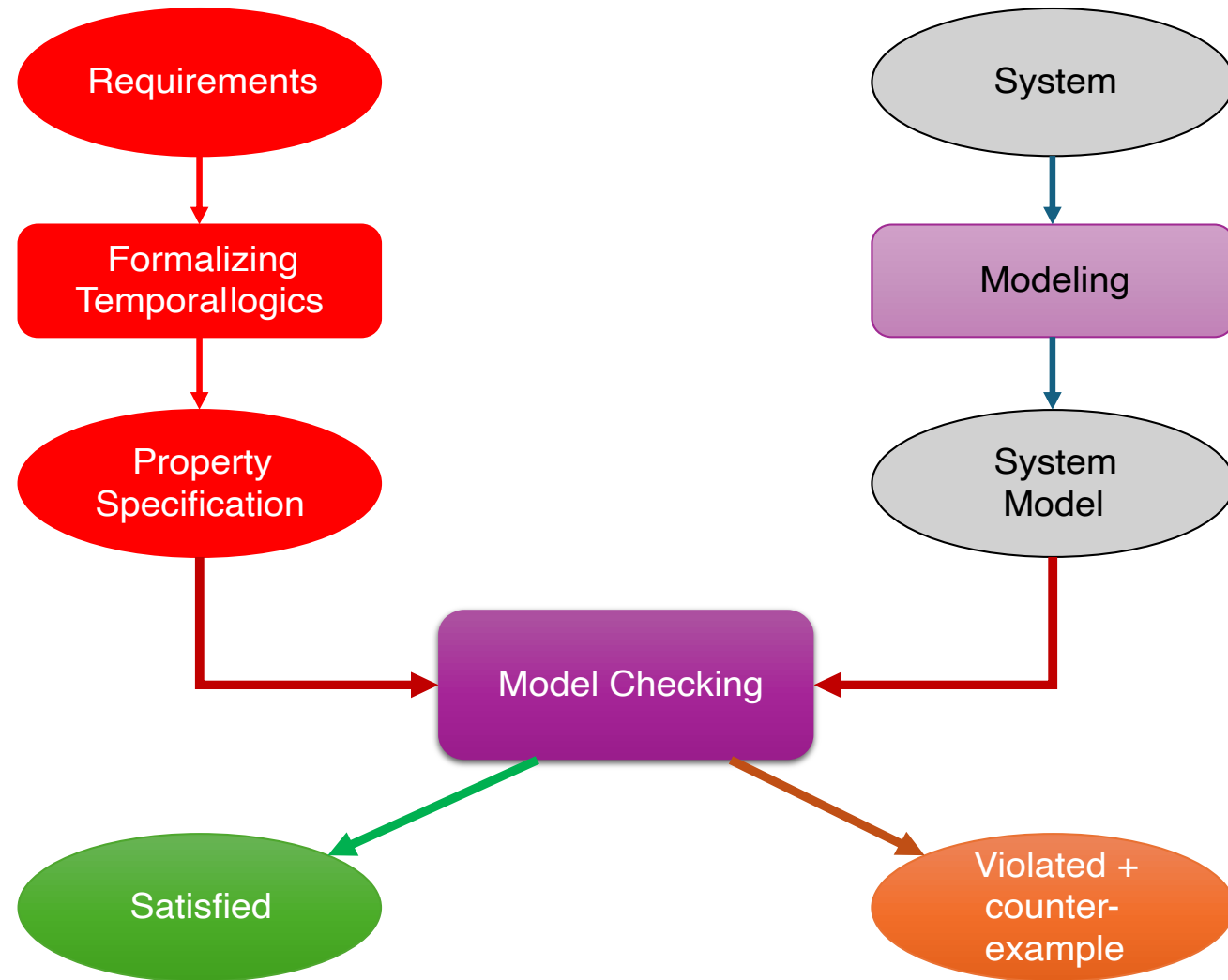
[idir.aitsadoune@centralesupelec.fr](mailto:idir.aitsadoune@centralesupelec.fr)

# OUTLINE

- Introducing CTL
- CTL Logics
- CTL running example
- Example : Dining Philosophers

[Back to the outline](#) - [Back to the begin](#)

# PRINCIPLE OF MODEL-CHECKING



# OUTLINE

- > Introducing CTL
- > CTL Logics
- > CTL running example
- > Example : Dining Philosophers

[Back to the outline](#) - [Back to the begin](#)

# LTL AND CTL

- **LTL - (linear-time logic)**
  - Describes properties of individual executions.
  - Semantics defined as a set of executions.
- **CTL - (computation tree logic)**
  - Describes properties of a computation tree: formulas can reason about many executions at once. (CTL belongs to the family of branching-time logics.)
  - Semantics defined in terms of states.

# COMPUTATION TREE

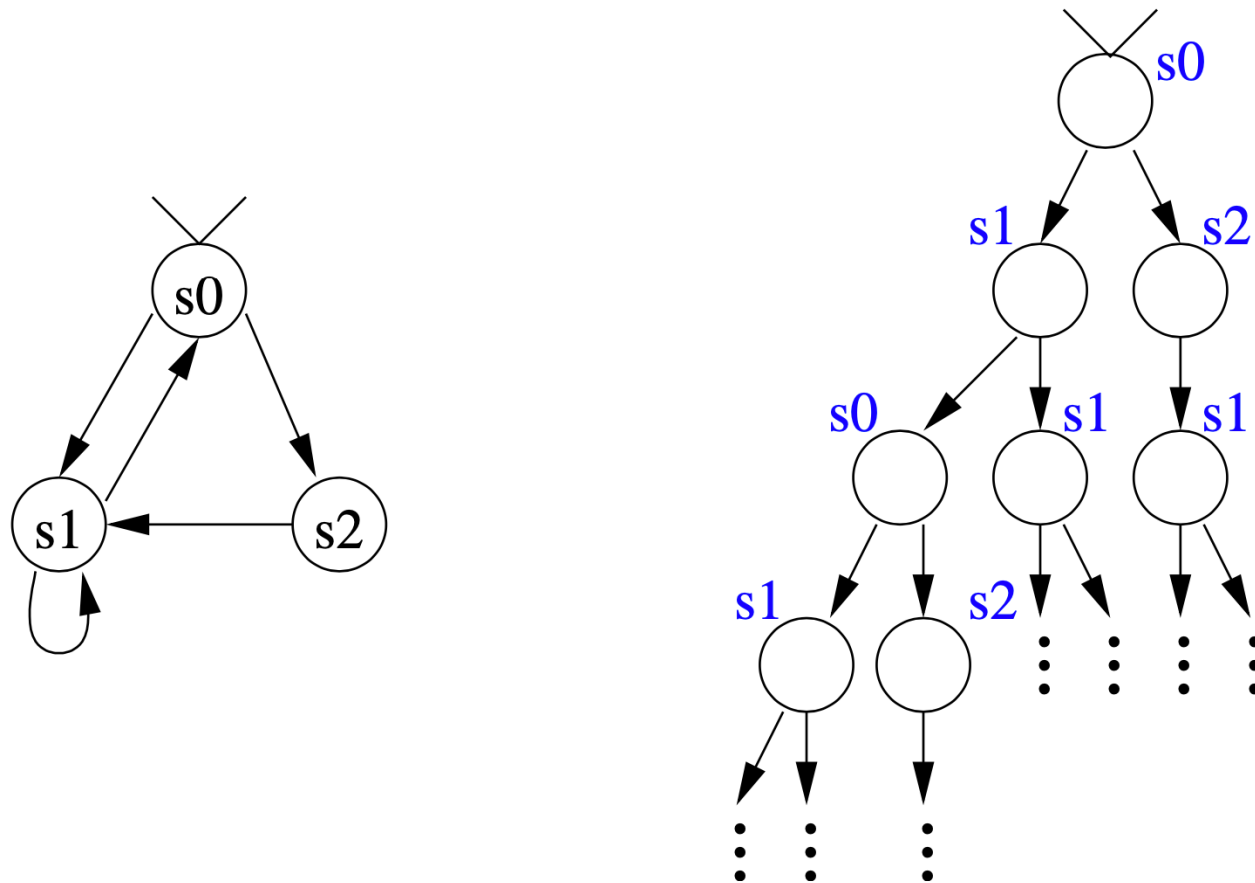
- Let  $\mathcal{T} = (S, \rightarrow, s^0)$  be a transition system.  
Intuitively, the **computation tree** of  $\mathcal{T}$  is the **acyclic unfolding** of  $\mathcal{T}$ .
- Formally, we can define the unfolding as the least (possibly infinite) transition system  $(U, \rightarrow', u^0)$  with a labelling  $l : U \rightarrow S$  such that:
  - $u^0 \in U$  and  $l(u^0) = s^0$
  - if  $u \in U$ ,  $l(u) = s$ , and  $s \rightarrow s'$  for some  $u, s, s'$   
then there is  $u' \in U$  with  $u \rightarrow' u'$  and  $l(u') = s'$
  - $u^0$  does not have a direct predecessor, and all other states in  $U$  have exactly one direct predecessor

**Note:** For model checking CTL, the construction of the computation tree will not be necessary. However, this definition serves to clarify the concepts behind CTL.

# COMPUTATION TREE

## EXAMPLE

A transition system and its computation tree (labelling  $l$  given in blue)



# OUTLINE

- Introducing CTL
- CTL Logics
- CTL running example
- Example : Dining Philosophers

[Back to the outline](#) - [Back to the begin](#)



# CTL - OVERVIEW

- Combines temporal operators with quantification over runs
- Operators have the following form:  $Q T$ 
  - $Q$ 
    - ⇒  $E$  : there exists an execution
    - ⇒  $A$  : for all executions
  - $T$ 
    - ⇒  $X \equiv \bigcirc$  : next
    - ⇒  $F \equiv \Diamond$  : finally
    - ⇒  $G \equiv \Box$  : globally
    - ⇒  $U \equiv \bigcup$  : until
    - ⇒ (and possibly others)

# CTL - SYNTAX

- We define a minimal syntax first. Later we define additional operators with the help of the minimal syntax.
- Let  $AP$  be a set of atomic propositions: The set of CTL formulas over  $AP$  is as follows:

if  $a \in AP$ , then  $a$  is a CTL formula;

if  $\phi_1, \phi_2$  are CTL formulas, then so are

$\neg\phi_1, \quad \phi_1 \vee \phi_2, \quad EX \phi_1, \quad EG \phi_1, \quad E (\phi_1 U \phi_2)$

# CTL - SEMANTICS

- let  $\mathcal{K} = (S, \rightarrow, s^0, AP, v)$  be a **Kripke** structure.
- We define the semantic of every CTL formula  $\phi$  over  $AP$  with respect to  $\mathcal{K}$  as a set of states  $\llbracket \phi \rrbracket_{\mathcal{K}}$ , as follows :

$$\llbracket a \rrbracket_{\mathcal{K}} = v(a) \quad a \in AP$$

$$\llbracket \neg \phi_1 \rrbracket_{\mathcal{K}} = S \setminus \llbracket \phi_1 \rrbracket_{\mathcal{K}}$$

$$\llbracket \phi_1 \vee \phi_2 \rrbracket_{\mathcal{K}} = \llbracket \phi_1 \rrbracket_{\mathcal{K}} \cup \llbracket \phi_2 \rrbracket_{\mathcal{K}}$$

$$\llbracket EX \phi_1 \rrbracket_{\mathcal{K}} = \{s \mid \text{there is a state } t \text{ with } s \rightarrow t \text{ and } t \in \llbracket \phi_1 \rrbracket_{\mathcal{K}}\}$$

$$\llbracket EG \phi_1 \rrbracket_{\mathcal{K}} = \{s \mid \text{there is a run } \sigma \text{ with } \sigma(0) = s \text{ and } \sigma(i) \in \llbracket \phi_1 \rrbracket_{\mathcal{K}} \forall i \geq 0\}$$

$$\llbracket E (\phi_1 U \phi_2) \rrbracket_{\mathcal{K}} = \{s \mid \text{there is a run } \sigma \text{ with } \sigma(0) = s \text{ and } k \geq 0, \sigma(i) \in \llbracket \phi_1 \rrbracket_{\mathcal{K}} \forall i < k, \sigma(k) \in \llbracket \phi_2 \rrbracket_{\mathcal{K}}\}$$

# CTL - EXTENDED SYNTAX

$$\textit{false} \equiv \neg \textit{true}$$

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

$$EF \phi \equiv E(\textit{true} U \phi)$$

$$AX \phi \equiv \neg EX \neg\phi$$

$$AG \phi \equiv \neg EF \neg\phi$$

$$AF \phi \equiv \neg EG \neg\phi$$

$$A(\phi_1 U \phi_2) \equiv \neg E \neg(\phi_1 U \phi_2)$$

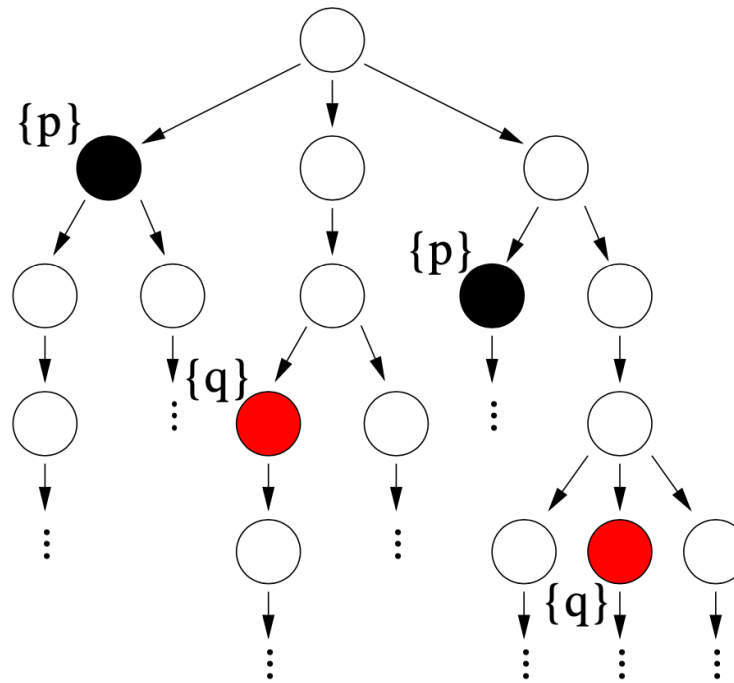
# OUTLINE

- Introducing CTL
- CTL Logics
- CTL running example
- Example : Dining Philosophers

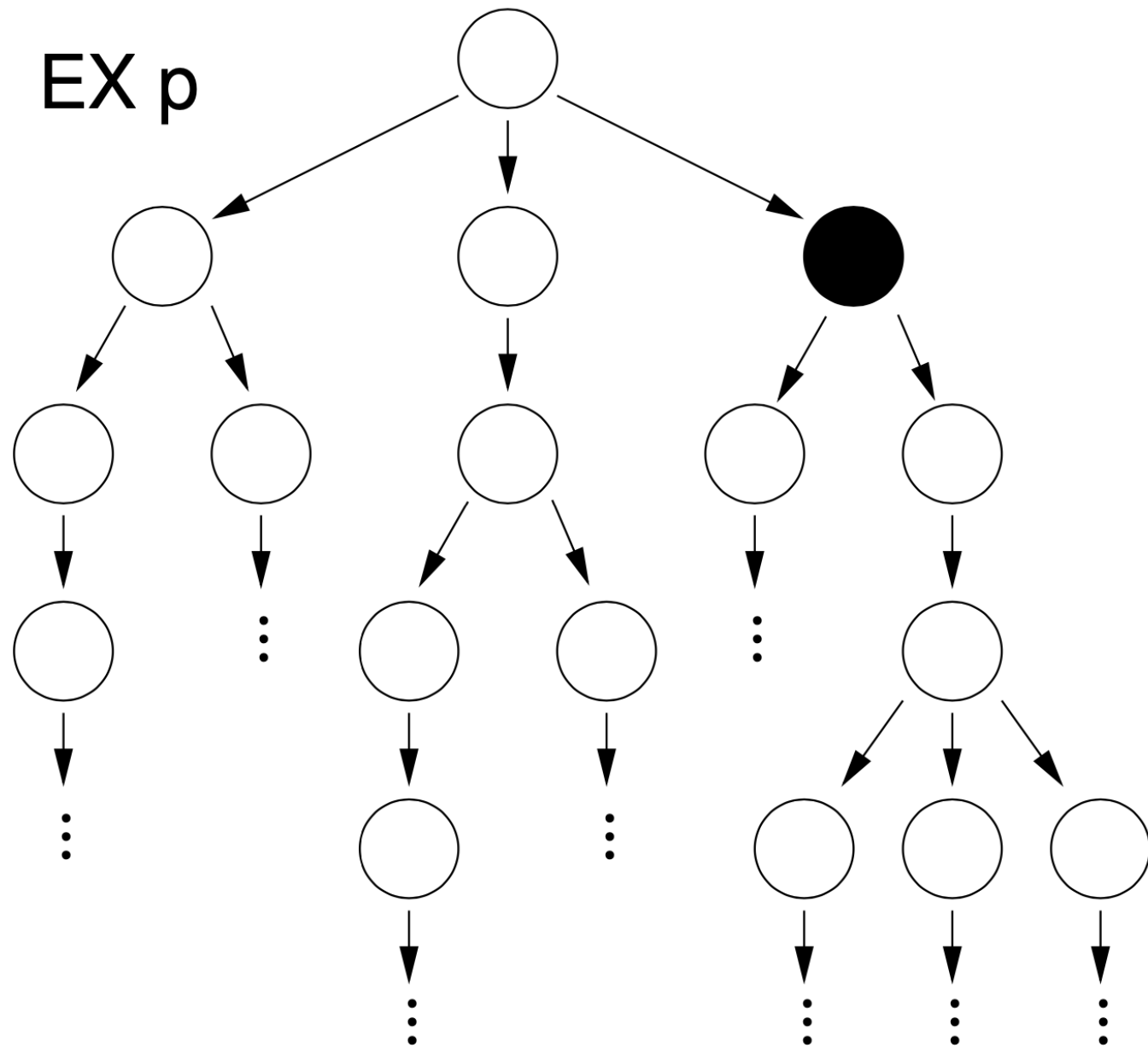
[Back to the outline](#) - [Back to the begin](#)

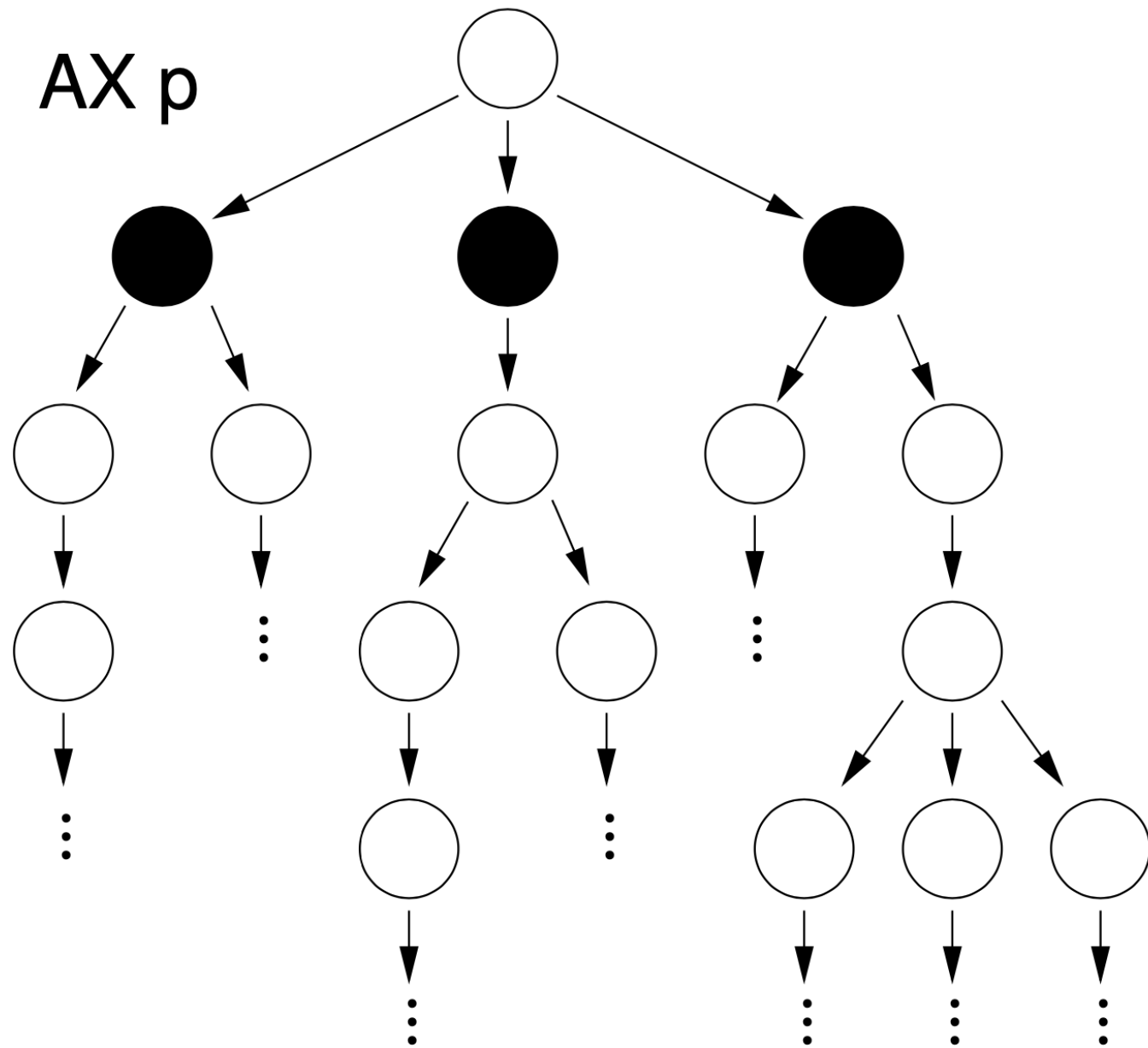
# CTL EXAMPLES

We use the following computation tree as a running example (with varying distributions of **red** and **black** states)

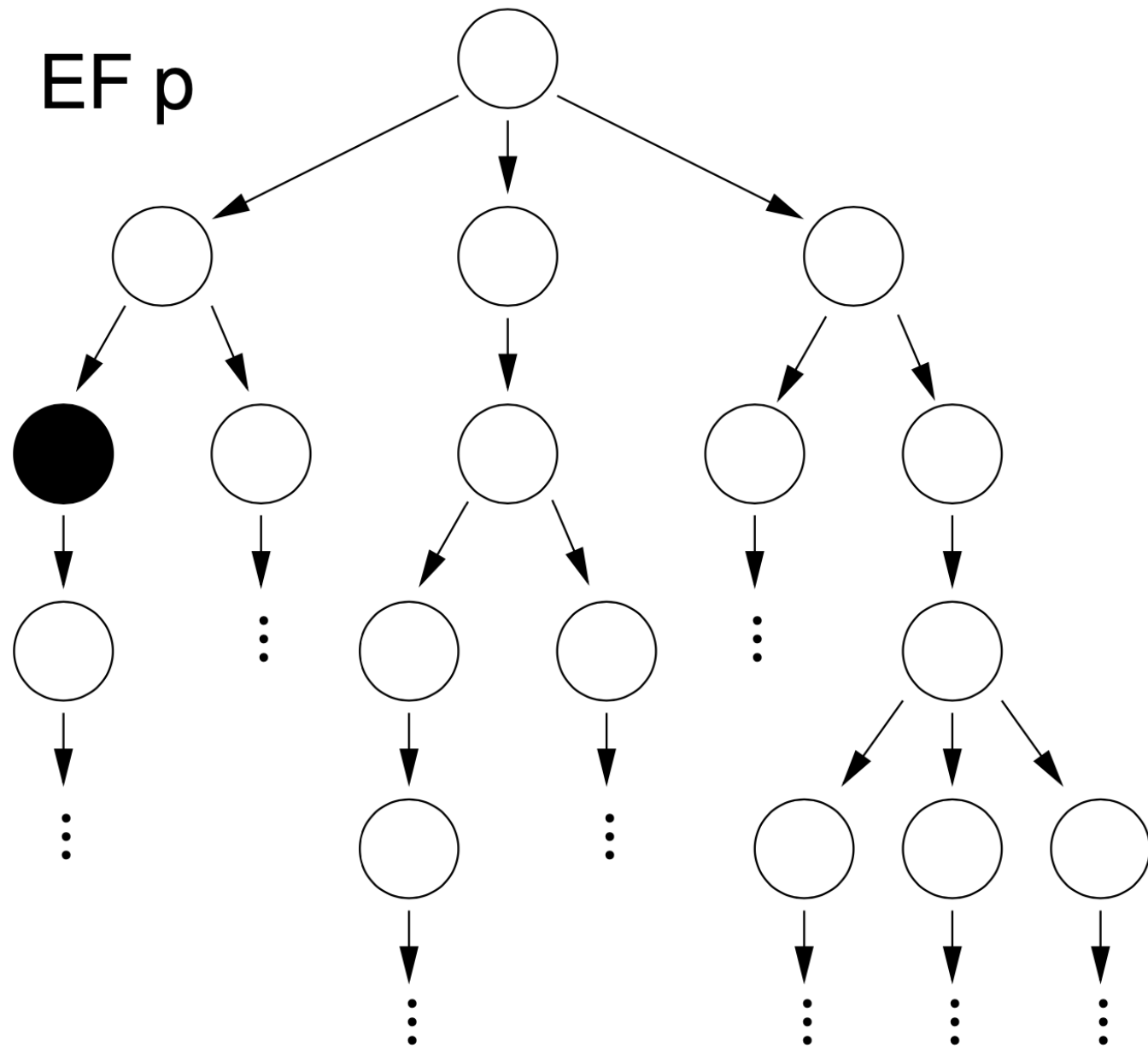


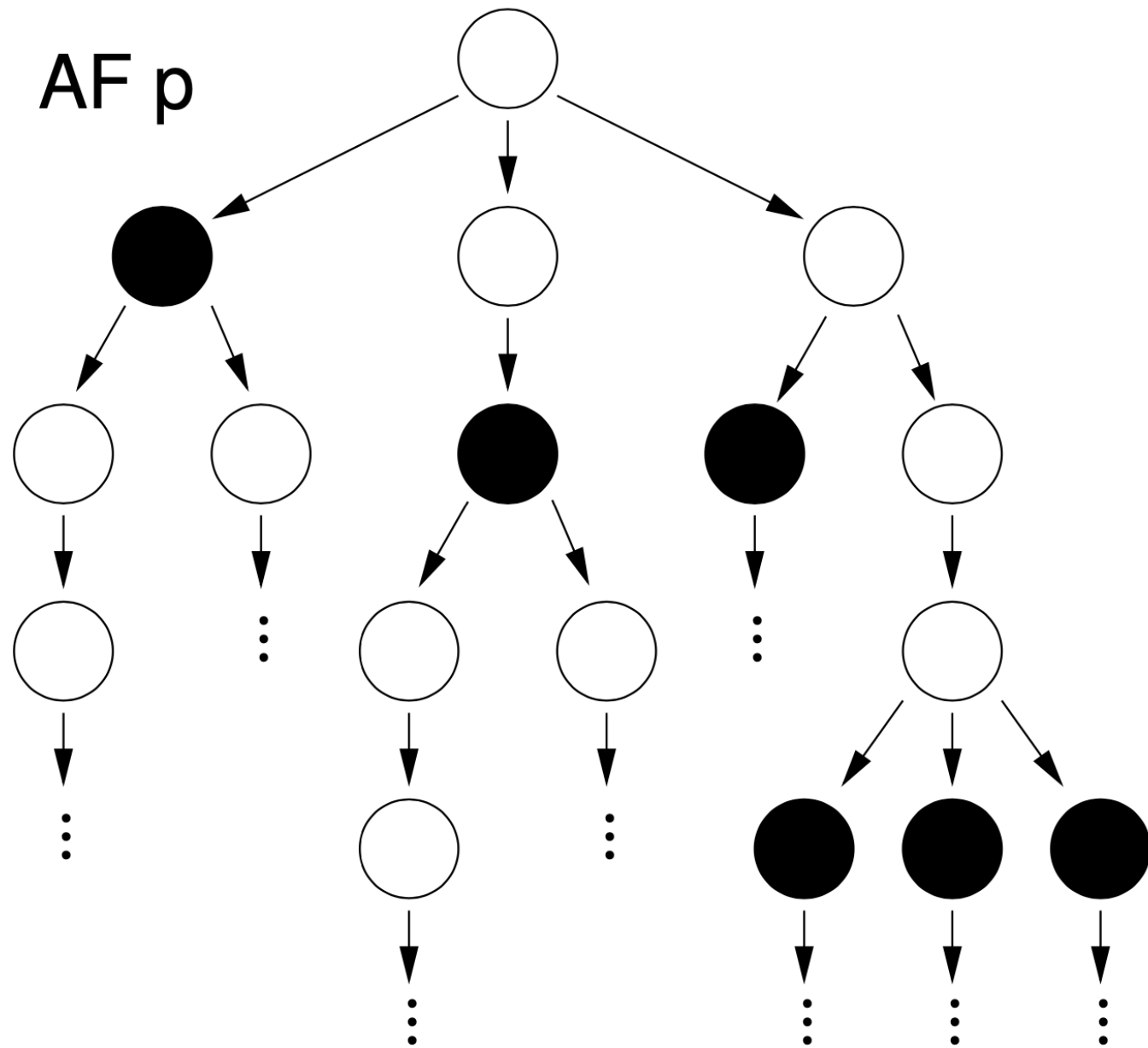
In the following slides, the topmost state satisfies the given formula if the black states satisfy  $p$  and the red states satisfy  $q$ .



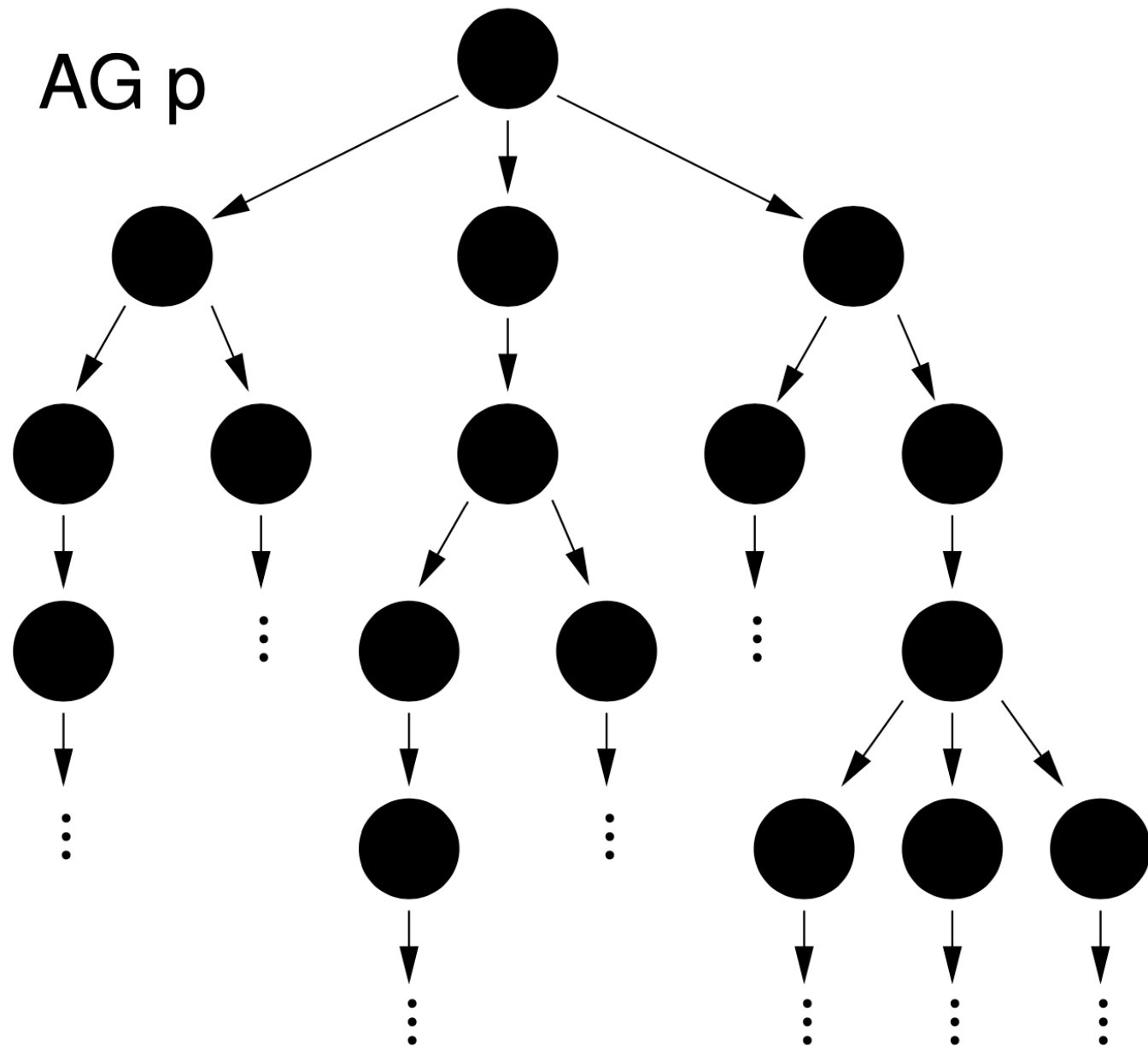


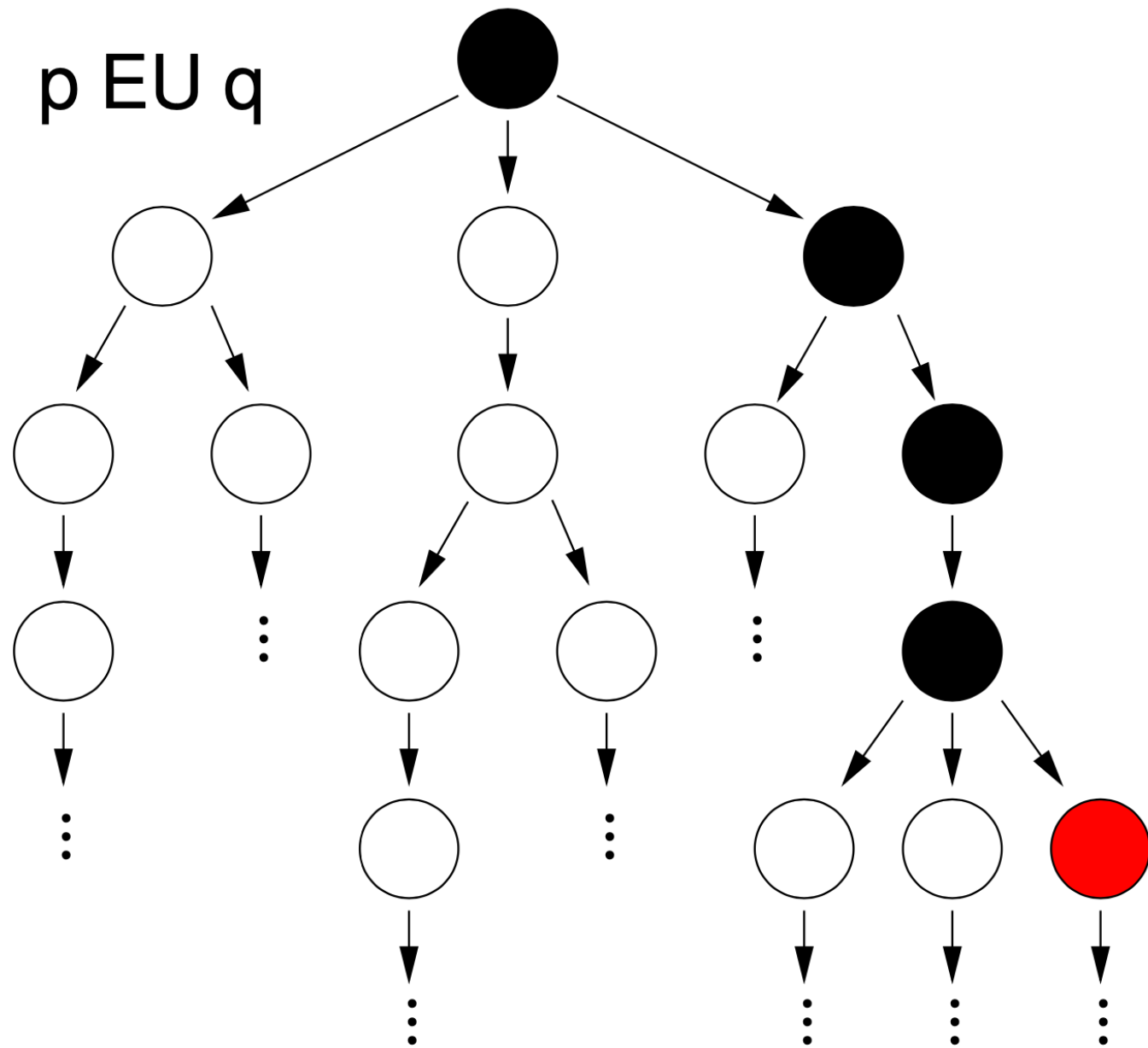


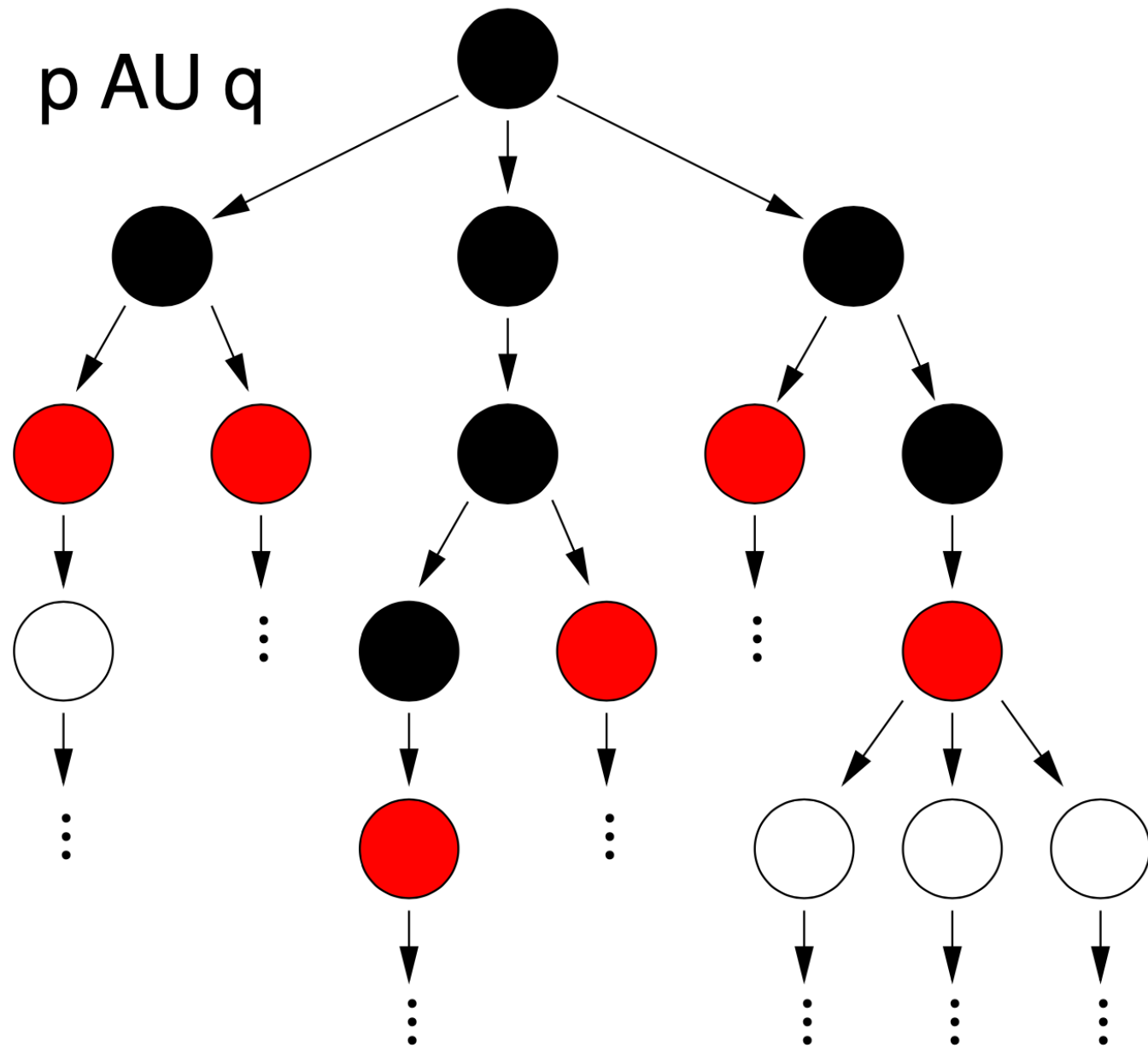






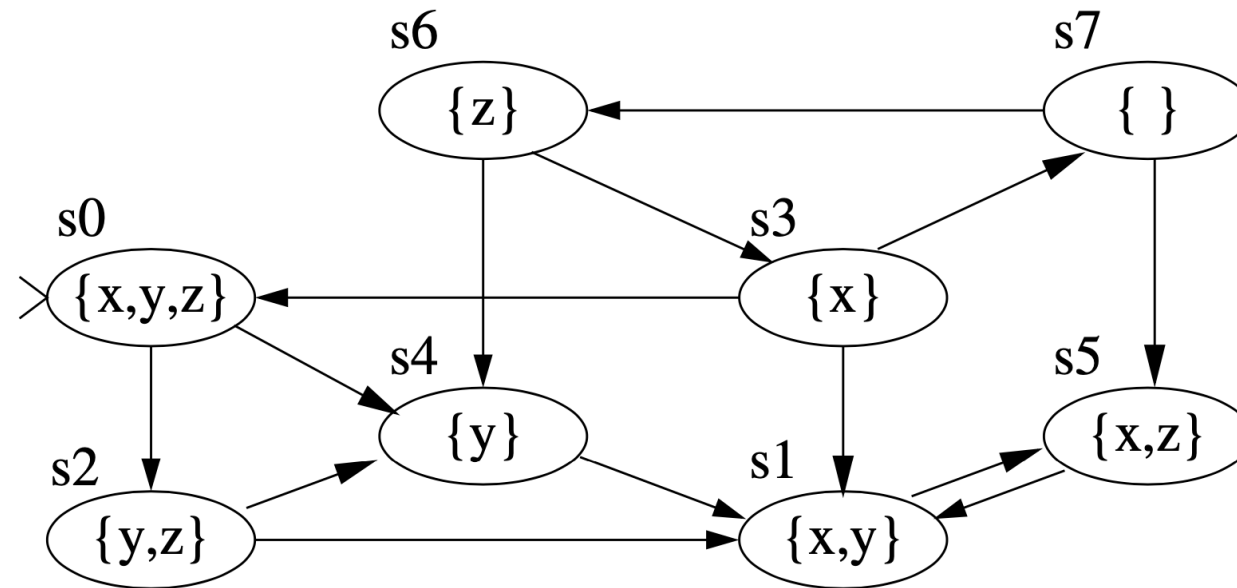






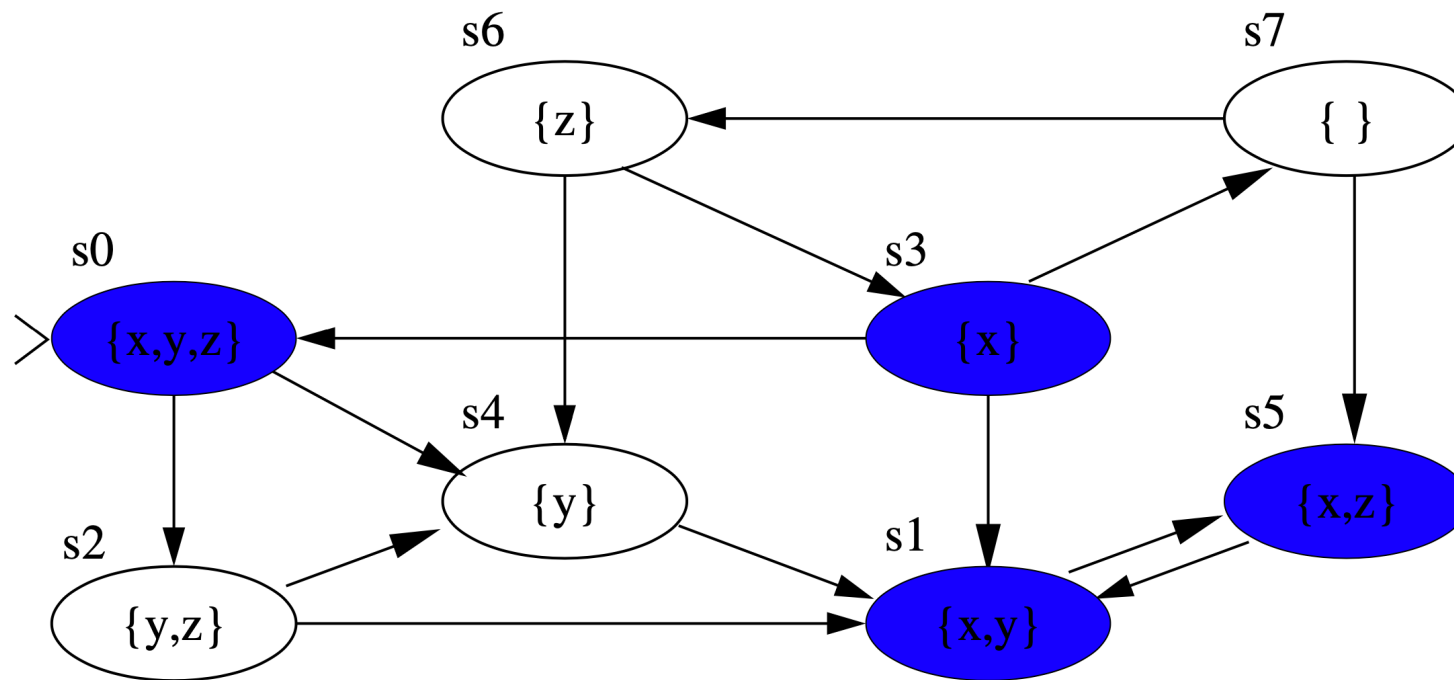
# SOLVING NESTED FORMULAS

$$s^0 \in \llbracket AFA G x \rrbracket$$



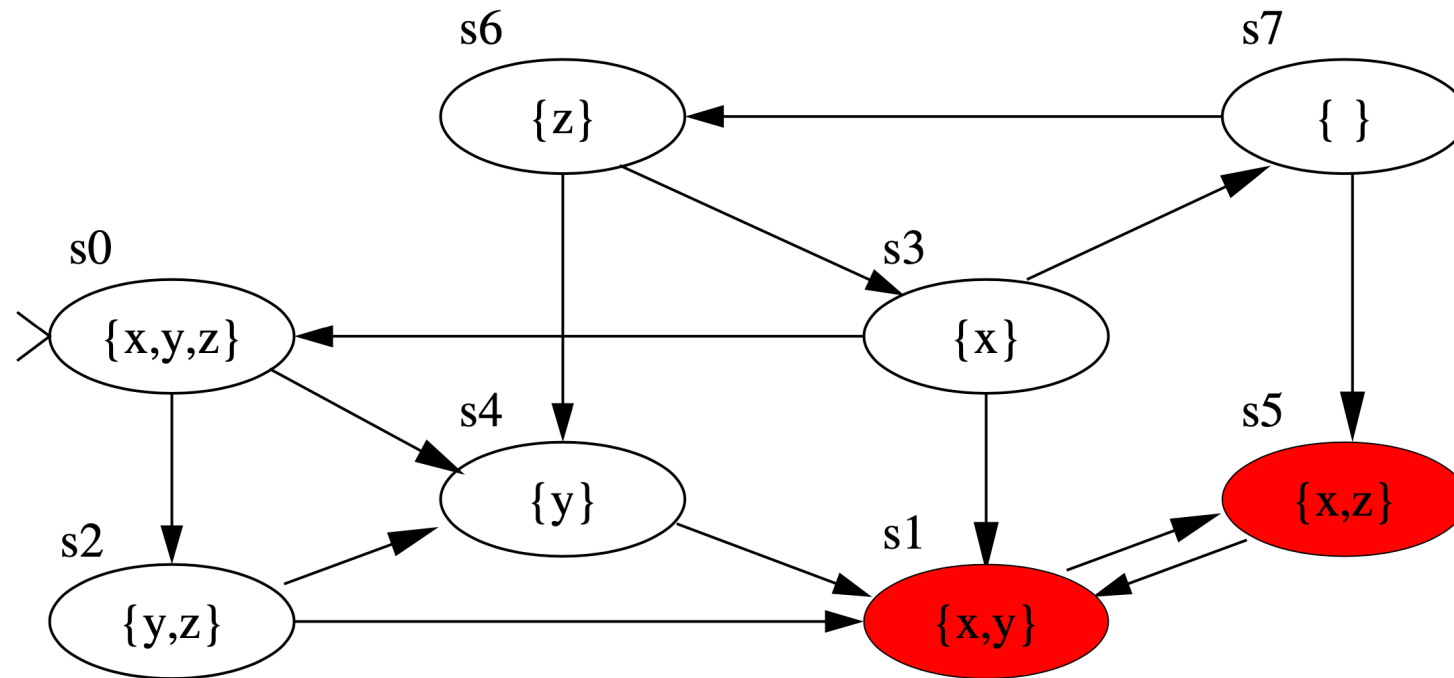
- To compute the semantics of formulas with nested operators,
  - we first compute the states satisfying the innermost formulas;
  - then we use those results to solve progressively more complex formulas.
- In this example, we compute  $\llbracket x \rrbracket$ ,  $\llbracket AG x \rrbracket$ , and  $\llbracket AFA G x \rrbracket$ , in that order

Compute  $\llbracket x \rrbracket$

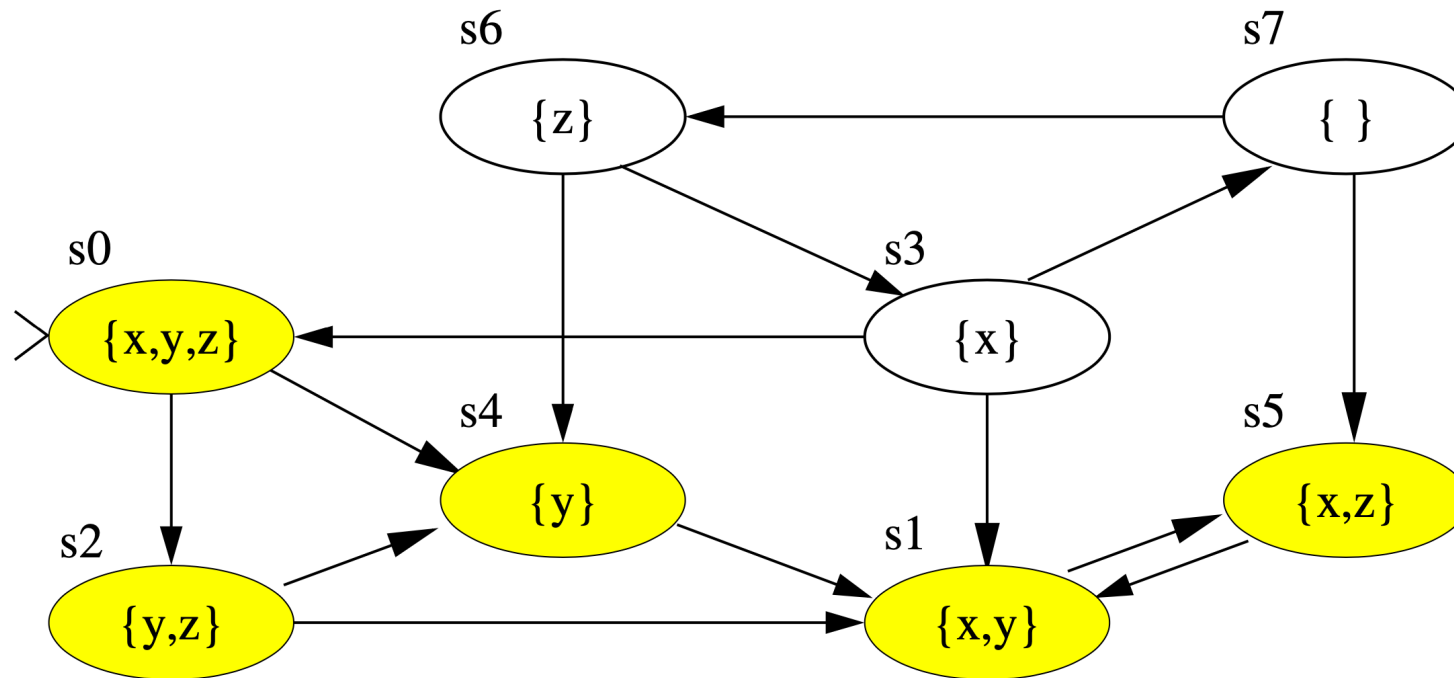




Compute  $\llbracket AG\ x \rrbracket$



Compute  $\llbracket AFAG\ x \rrbracket$

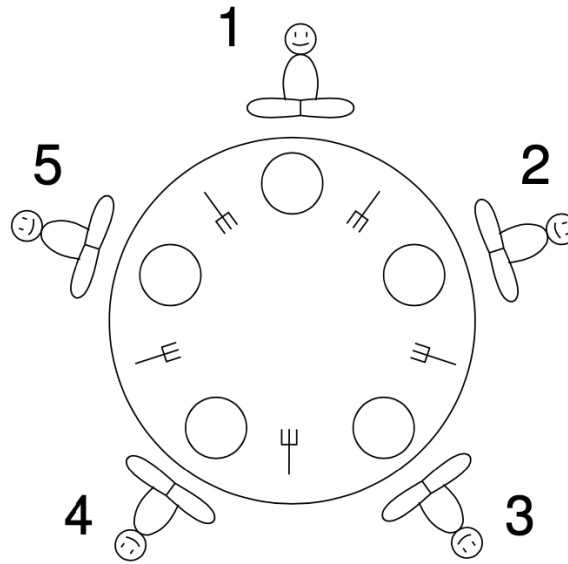


# OUTLINE

- Introducing CTL
- CTL Logics
- CTL running example
- Example : Dining Philosophers

[Back to the outline](#) - [Back to the begin](#)

# EXAMPLE : DINING PHILOSOPHERS



- Five philosophers are sitting around a table, taking turns at thinking and eating.
- We shall express a couple of properties in CTL. Let us assume the following atomic propositions:
  - $e_i \rightarrow$  philosopher  $i$  is currently eating
  - $f_i \rightarrow$  philosopher  $i$  has just finished eating

# EXAMPLE : DINING PHILOSOPHERS

- Philosophers 1 and 4 will never eat at the same time.

$$AG \neg(e_1 \wedge e_4)$$

- Whenever philosopher 4 has finished eating, he cannot eat again until philosopher 3 has eaten.

$$AG (f_4 \Rightarrow A (\neg e_4 W e_3))$$

- Philosopher 2 will be the first to eat.

$$A(\neg(e_1 \vee e_3 \vee e_4 \vee e_5) U e_2)$$

# THANK YOU

[PDF version of the slides](#)

[Back to the begin](#) - [Back to the outline](#)