Important note

Corrections have been made to the CTL part.

An important hypothesis was missing (see slide 15)

Crash course – Verification of Finite Automata
CTL model-checking

Exercise session - 08.12.2016

Romain Jacob

Reminders – Big picture

Objective Verify properties over DES models

Formal method ⇒ Absolute guarantee!

Problem Combinatorial explosion

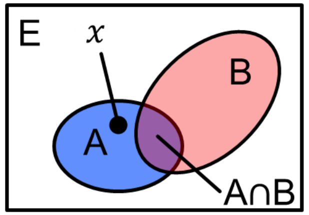
→ Huge amount of states, computationally intractable

Solution Work with sets of states

→ Symbolic Model-Checking

→ (O)BDDs

Reminders – First exercise session



Sets

⟺

$\psi_E = 1$
$\psi_A = f$
$\psi_B = g$
$\psi_{A\cap B} = f \cdot g$

Boolean functions/
Characteristic functions

- A
- $s \in A$

$\sigma(\cdot)$

- ψ_A
- $\psi_A(\sigma(s)) = 1$
  $\sigma(s) \vDash \psi_A$
  or just $s \vDash \psi_A$

Reads "s satisfies ψ_A"

Example:
$\sigma(s) = x_1\overline{x_0} = (1,0)$ and $\psi_A = x_1 + x_0$
$\rightarrow s \vDash \psi_A$?

Equivalence between sets and Boolean equations

$f: x_1 + \overline{x_1}\, x_2 + \overline{x_2}\, \overline{x_3}$

Fall $x_1 = 0$

$f_{|x_1=0}: x_2 + \overline{x_2}\, \overline{x_3}$

Fall $x_2 = 0$

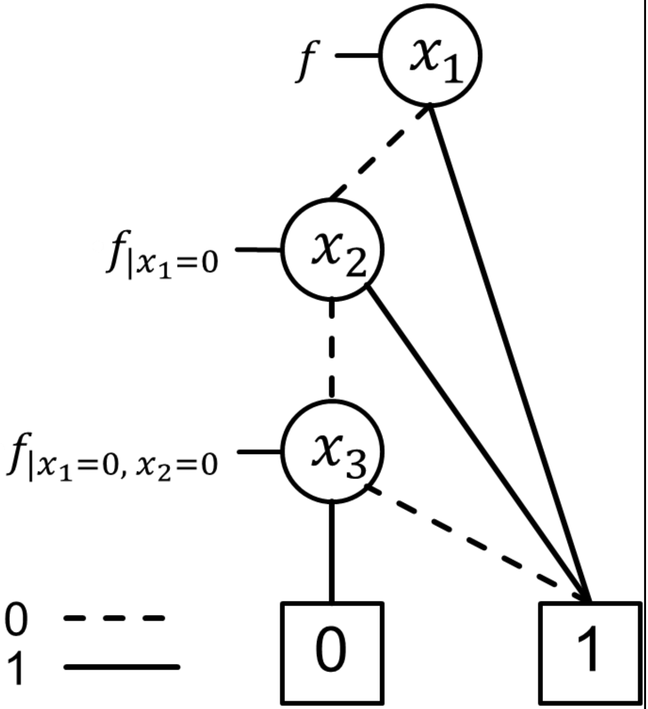
$f_{|x_1=0,\, x_2=0}: \overline{x_3}$

Fall $x_2 = 1$

$f_{|x_1=0,\, x_2=1}: 1$

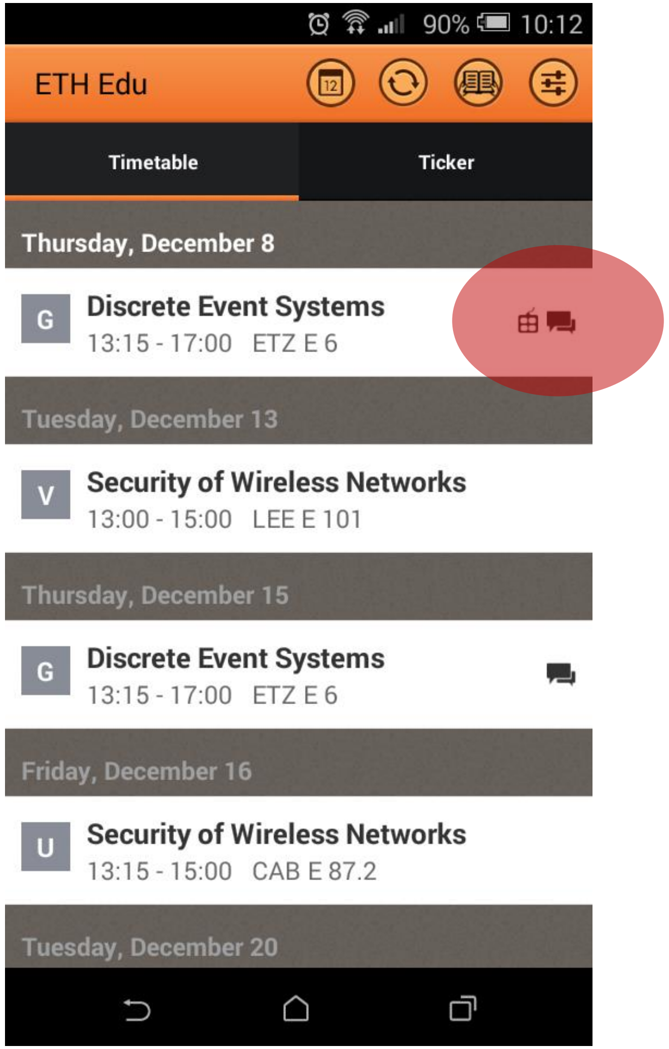
Fall $x_1 = 1$

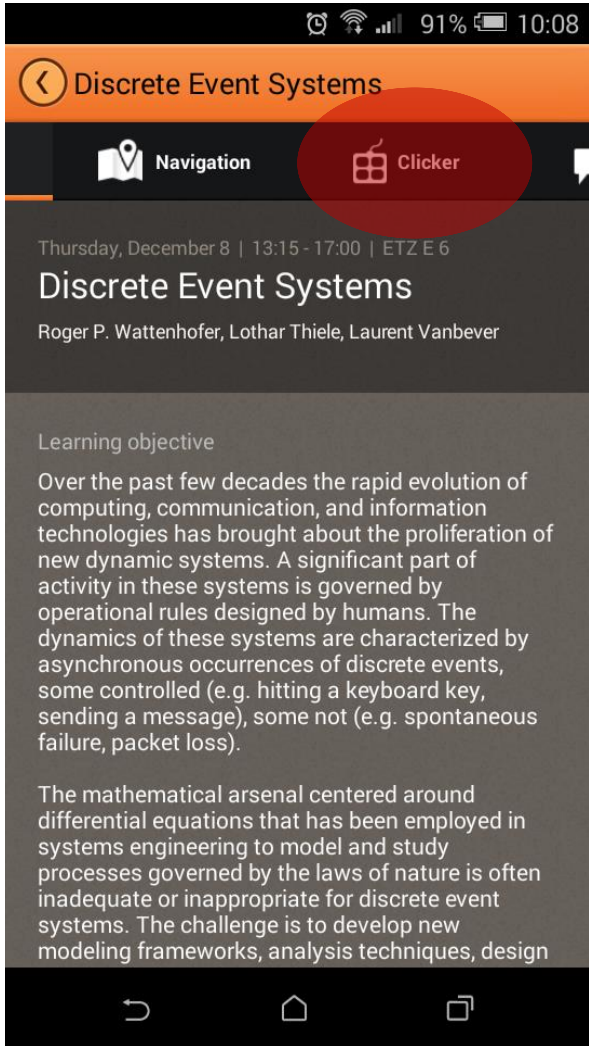
$f_{|x_1=1}: 1$

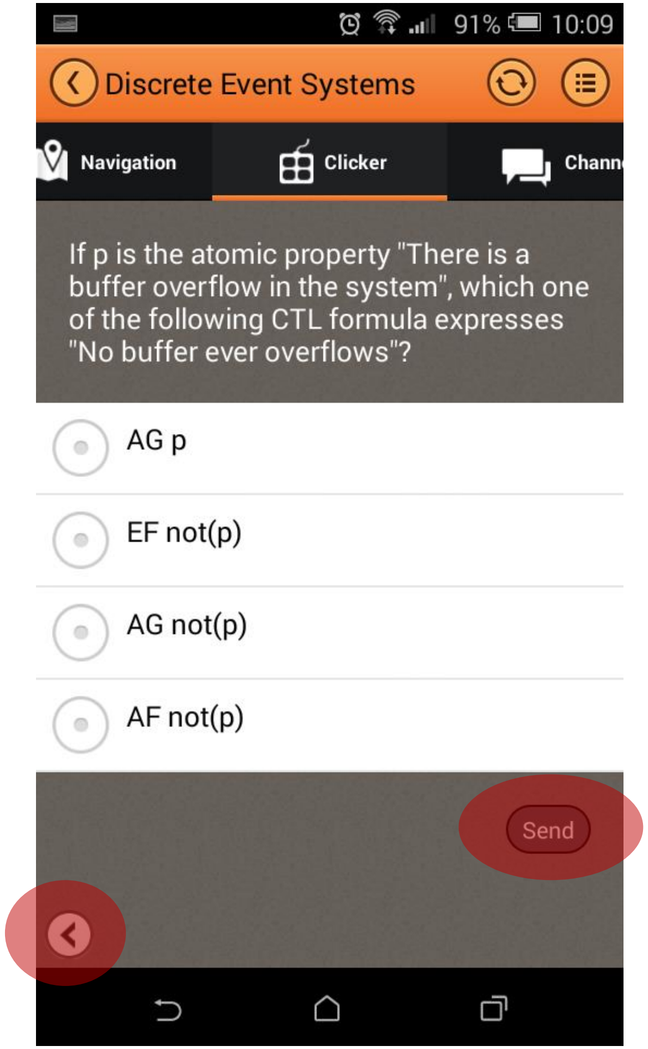


BBD representation of Boolean functions

Let see what you remember!


90% 10:12
ETH Edu
Timetable
Ticker
Thursday, December 8
G
Discrete Event Systems
13:15 - 17:00 ETZ E 6
Tuesday, December 13
V
Security of Wireless Networks
13:00 - 15:00 LEE E 101
Thursday, December 15
G
Discrete Event Systems
13:15 - 17:00 ETZ E 6
Friday, December 16
U
Security of Wireless Networks
13:15 - 15:00 CAB E 87.2
Tuesday, December 20


91% 10:08
Discrete Event Systems
Navigation
Clicker
Thursday, December 8 | 13:15 - 17:00 | ETZ E 6
Discrete Event Systems
Roger P. Wattenhofer, Lothar Thiele, Laurent Vanbever
Learning objective
Over the past few decades the rapid evolution of computing, communication, and information technologies has brought about the proliferation of new dynamic systems. A significant part of activity in these systems is governed by operational rules designed by humans. The dynamics of these systems are characterized by asynchronous occurrences of discrete events, some controlled (e.g. hitting a keyboard key, sending a message), some not (e.g. spontaneous failure, packet loss).
The mathematical arsenal centered around differential equations that has been employed in systems engineering to model and study processes governed by the laws of nature is often inadequate or inappropriate for discrete event systems. The challenge is to develop new modeling frameworks, analysis techniques, design


91% 10:09
Discrete Event Systems
Navigation
Clicker
Chann
If p is the atomic property "There is a buffer overflow in the system", which one of the following CTL formula expresses "No buffer ever overflows"?
AG p
EF not(p)
AG not(p)
AF not(p)
Send

Today's menu

1. Reachability of states
2. Comparison of automata
3. Formulation and verification of CTL properties

Can be formulated as reachability problems (items 2 and 3)

Reachability of states

Fairly simple

1. Start from the initial set of states,
2. Compute all states you can transition to in one hop (one transition),
   → The successor states,
3. Join the two sets,
4. Iterate from 2. until you reach a fix point.
5. Done !

Is this guarantee to terminate?

Reachability of states

Fairly simple

1. Start from the initial set of states,
2. Compute all states you can transition to in one hop (one transition),
   → The successor states,
3. Join the two sets,
4. Iterate from 2. until you reach a fix point.
5. Done !

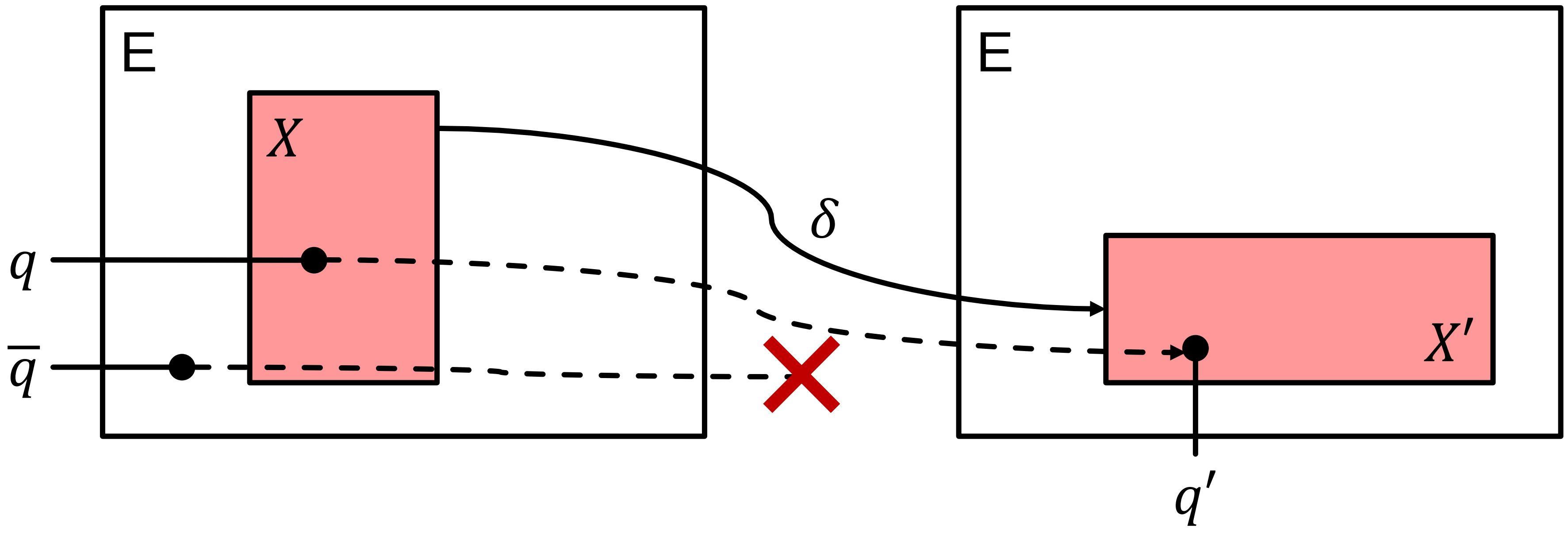
Is this guarantee to terminate?

→ Only if you have a finite model!!

How can we formalize this problem?

Formalization of reachable states

$$\delta : X \subseteq E \longrightarrow X' \subseteq E$$
$$q \;\longmapsto\; q'$$



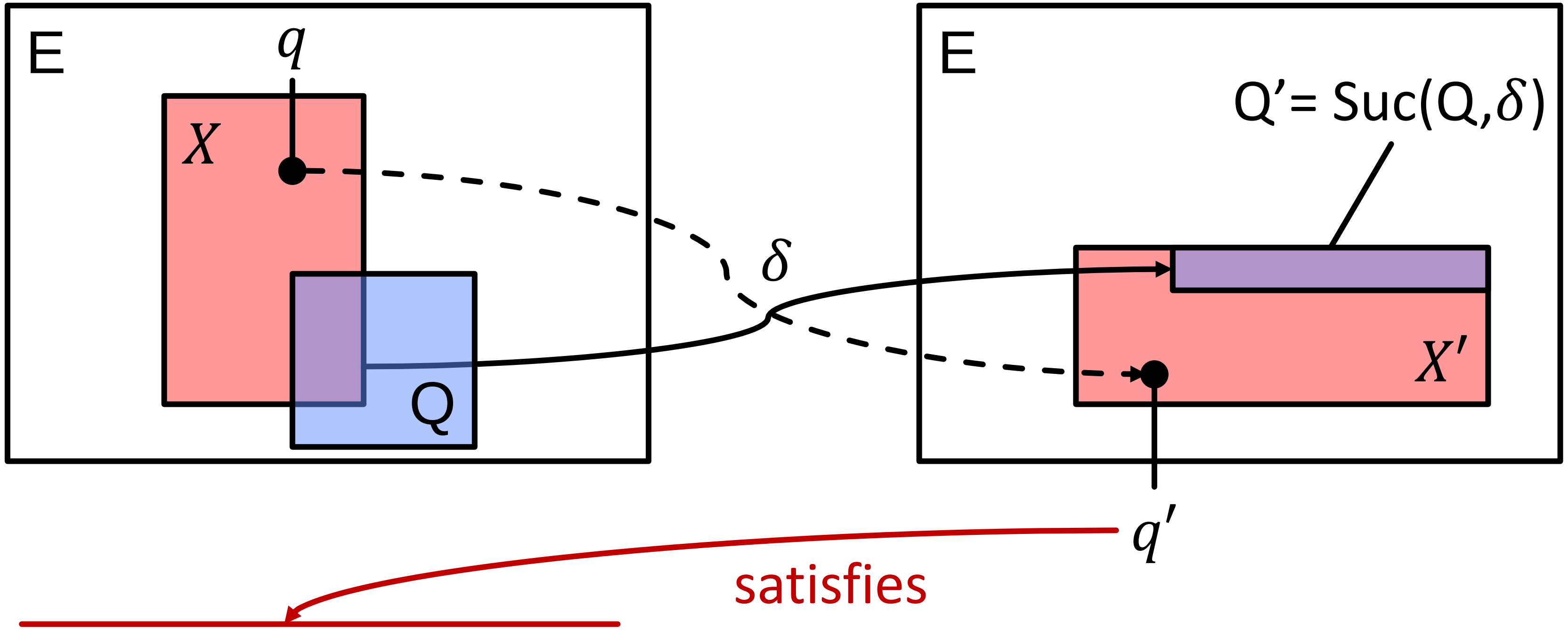
$q \in X \Leftrightarrow \exists\, q' \in X', \left| \begin{array}{l} \delta(q,q')\ is\ defined \\ \psi_\delta(q,q') = 1 \end{array} \right.$

$\overline{q} \notin X \Leftrightarrow \left| \begin{array}{l} \nexists\, q' \in X', \delta(\overline{q},q')\ is\ defined \\ \forall\, q' \in X\ , \psi_\delta(\overline{q},q') = 0 \end{array} \right.$

Formalization of reachable states

$$\delta : X \subseteq E \longrightarrow X' \subseteq E$$
$$q \;\longmapsto\; q'$$

What is Q'?



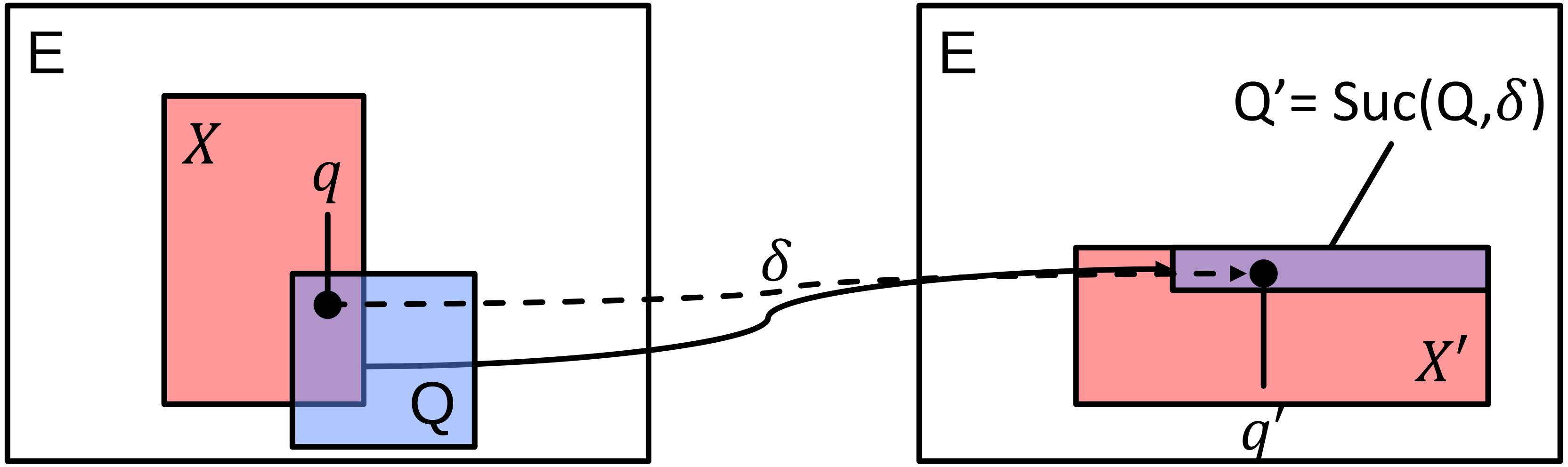
$$q' \in Q' \quad \Rightarrow \quad q' \in X' \quad \Rightarrow \quad \exists q \in X, \psi_\delta(q, q') = 1$$

Not sufficient !

We also need that q belongs to Q : $q \in Q$ or equivalently $\psi_Q(q) = 1$

Formalization of reachable states

$$\delta : X \subseteq E \longrightarrow X' \subseteq E$$
$$q \;\mapsto\; q'$$



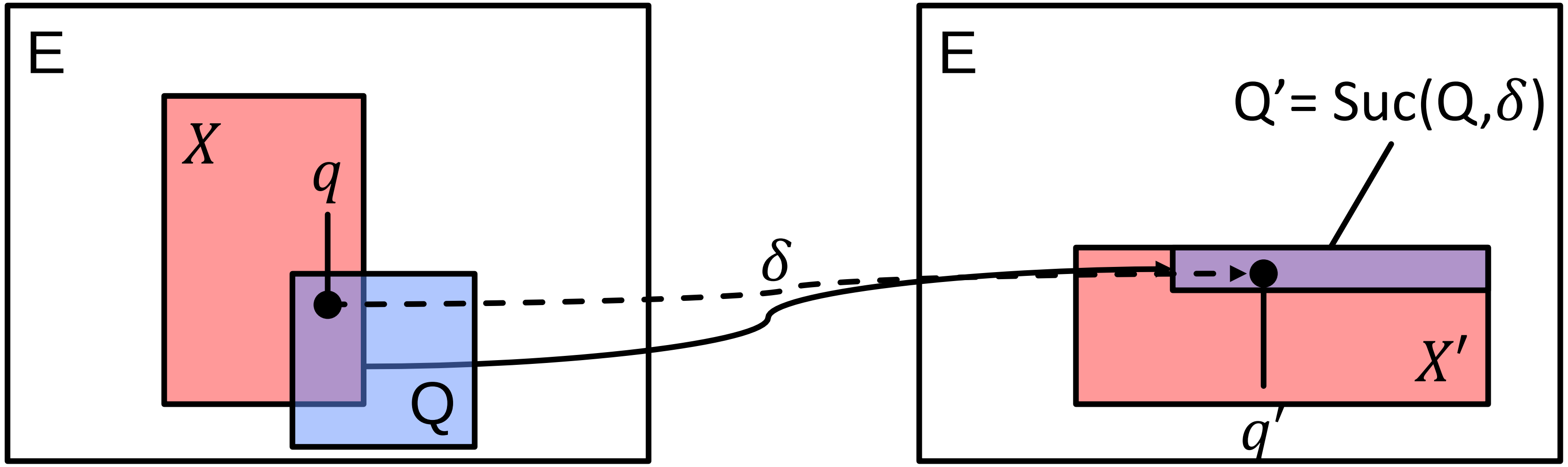
What is Q'?

$$q' \in Q' \;\Leftrightarrow \exists q \in X,\; \psi_Q(q) = 1 \text{ and } \psi_\delta(q, q') = 1$$
$$\Leftrightarrow \exists q \in X,\; \psi_Q(q) \cdot \psi_\delta(q, q') = 1$$

$$Q' = Suc(Q, \delta) = \{q' \,|\, \exists q \in X, \psi_Q(q) \cdot \psi_\delta(q, q') = 1\}$$

Formalization of reachable states

$$\delta : X \subseteq E \longrightarrow X' \subseteq E$$
$$q \longmapsto q'$$



$$Q' \quad = \quad Suc(Q,\delta) = \{q' \mid \exists q \in X, \psi_Q(q) \cdot \psi_\delta(q,q') = 1\}$$
$$\Leftrightarrow \quad \psi_{Q'} = \psi_Q \cdot \psi_\delta$$

Q_R: set of reachable states

$$Q_R \quad = Q_0 \cup_{i \geq 0} Suc(Q_i, \delta)$$
$$\Leftrightarrow \quad \psi_{Q_R} = \psi_{Q_0} \sum_{i \geq 0} \psi_{Q_i} \cdot \psi_\delta$$

Again, finite union if finite model

Comparison of automata

Two automata are equivalent $\Leftrightarrow$ Same <u>input</u> produces same <u>output</u>

→ Don't compare states!

- Computation of the joint transition function,

$$\psi_\delta(q_1, q_2, q_1', q_2') = (\exists u \ : \ \psi_{\omega_1}(u, q_1, q_1') \cdot \psi_{\omega_2}(u, q_2, q_2'))$$

➢ Get rid of the input

- Computation of the reachable states (method according to previous slides),

$$\psi_Q(q_1, q_2)$$

➢ Compute Q_R

- Computation of the reachable output values,

$$\psi_Y(y_1, y_2) = (\exists q_1, q_2 \ : \ \psi_Q(q_1, q_2) \cdot \psi_{\omega_1}(q_1, y_1) \cdot \psi_{\omega_2}(q_2, y_2))$$

➢ Deduce reachable outputs

- The automata are not equivalent if the following term is true,

$$\exists y_1, y_2 \ : \ \psi_Y(y_1, y_2) \cdot (y_1 \neq y_2)$$

➢ Test for equivalence

Formulation of CTL properties

Based on atomic propositions (ϕ) and quantifiers

Aϕ	→ «**All** ϕ»,	ϕ holds on all paths	Quantifiers over paths
Eϕ	→ «**E**xists ϕ»,	ϕ holds on at least one path	
Xϕ	→ «Ne**X**t ϕ»,	ϕ holds on the next state	Path-specific quantifiers
Fϕ	→ «**F**inally ϕ»,	ϕ holds at some state along the path	
Gϕ	→ «**G**lobally ϕ»,	ϕ holds on all states along the path	
ϕ_1Uϕ_2	→ «ϕ_1**U**ntil ϕ_2»,	ϕ_1 holds until ϕ_2 holds	

Formulation of CTL properties

Proper CTL formula: {A,E} {X,F,G,U}ϕ

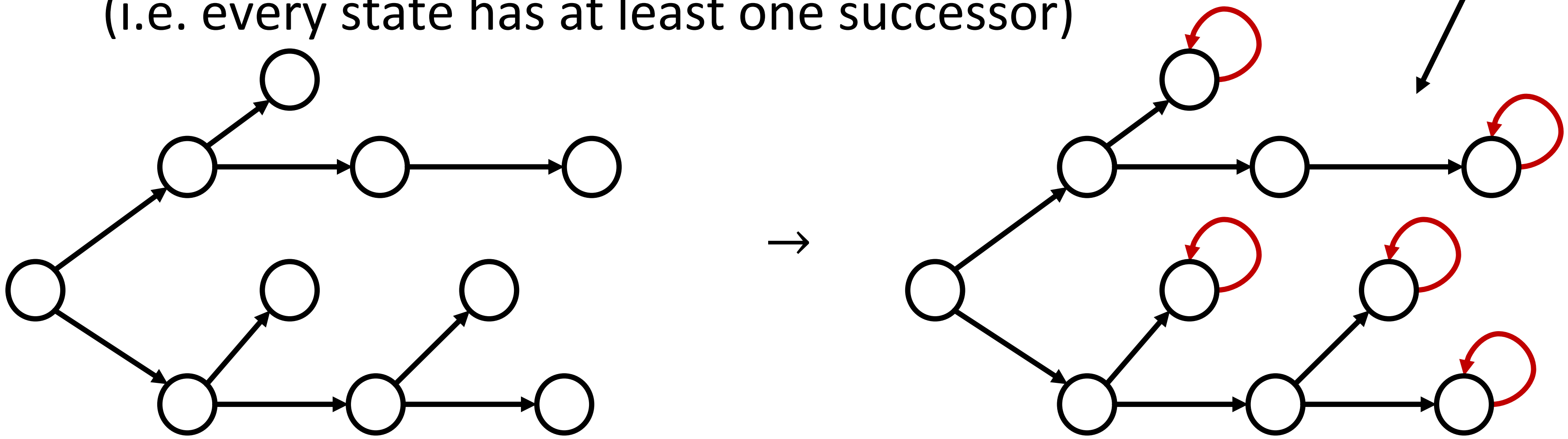
→ Quantifiers **go by pairs**, you need one of each.

Missing Hypothesis

Interpretation on CTL formula

→ Transition functions are **fully defined** (i.e. every state has at least one successor)

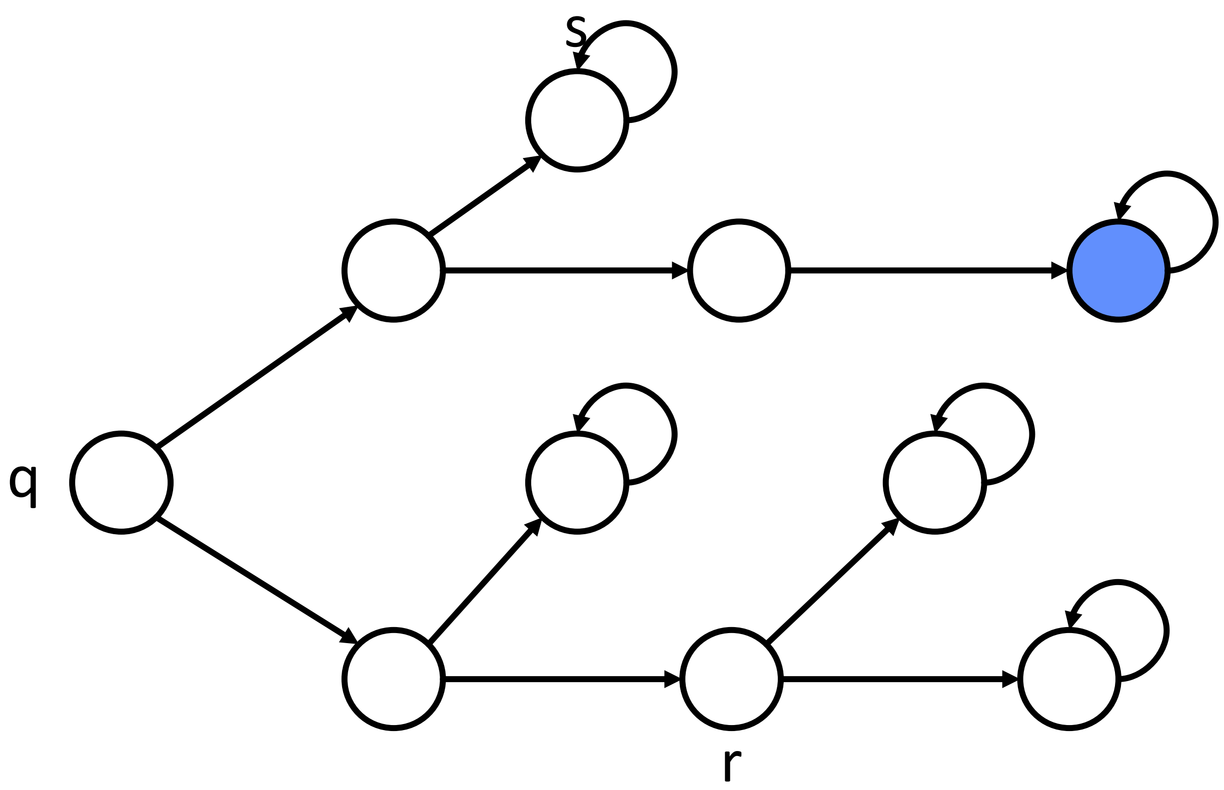
Simple "means" that we get rid of leaf nodes… → They transition to themselves

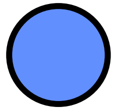


Automaton of interest → **Automaton to work with**

Formulation of CTL properties

EF ϕ : "There exists a path along which at some state ϕ holds."




$\models \phi$

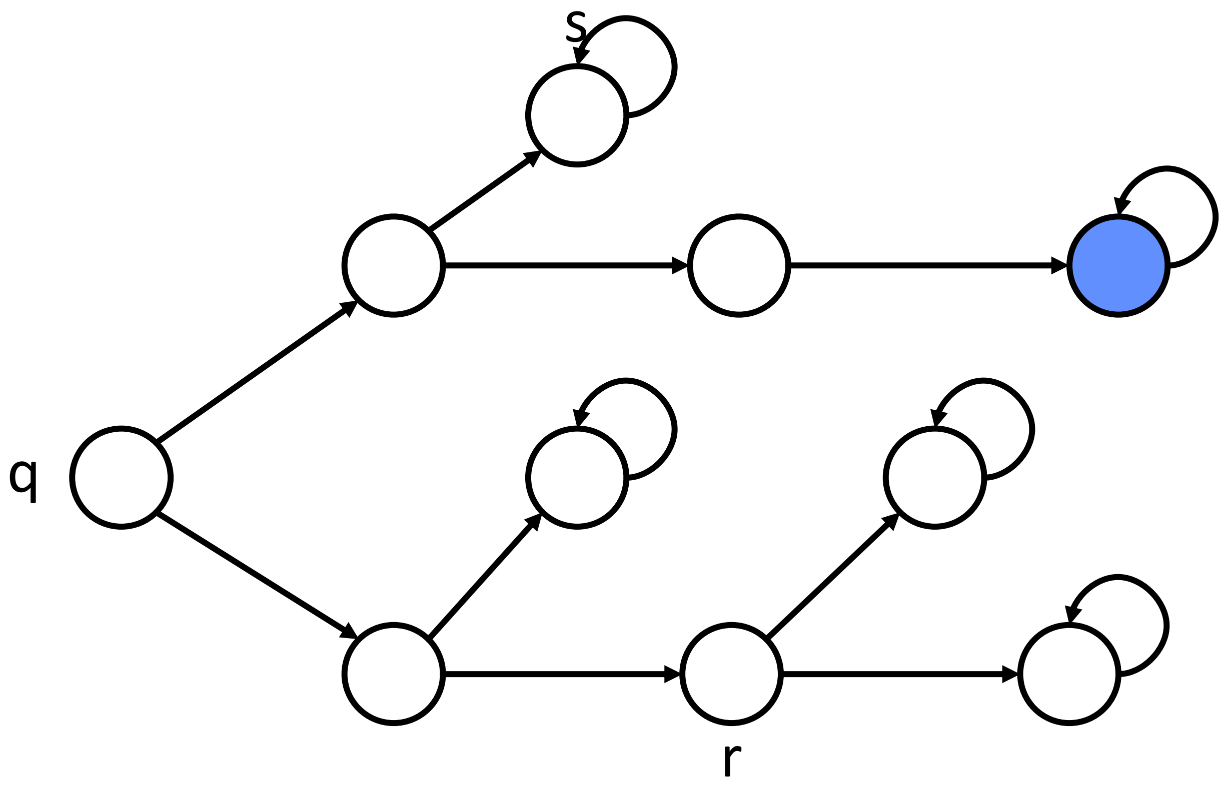
$q \models EF\ \phi$

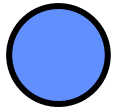
$r \models ?$

$s \models ?$

Formulation of CTL properties

EF ϕ : "There exists a path along which at some state ϕ holds."





$\vDash \phi$

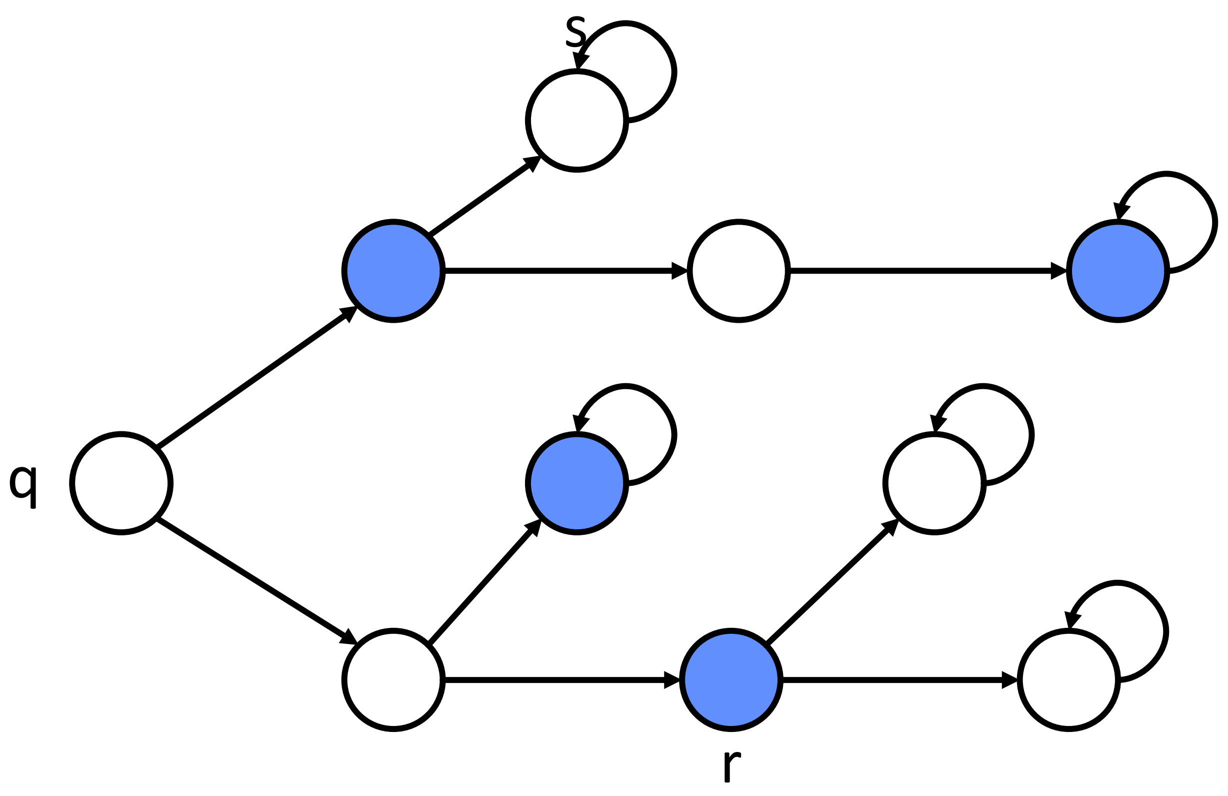
$q \vDash EF\ \phi$

$r \nvDash EF\ \phi$

$s \nvDash EF\ \phi$

Formulation of CTL properties

AF ϕ : “On all paths, at some state ϕ holds .”



$\vDash \phi$

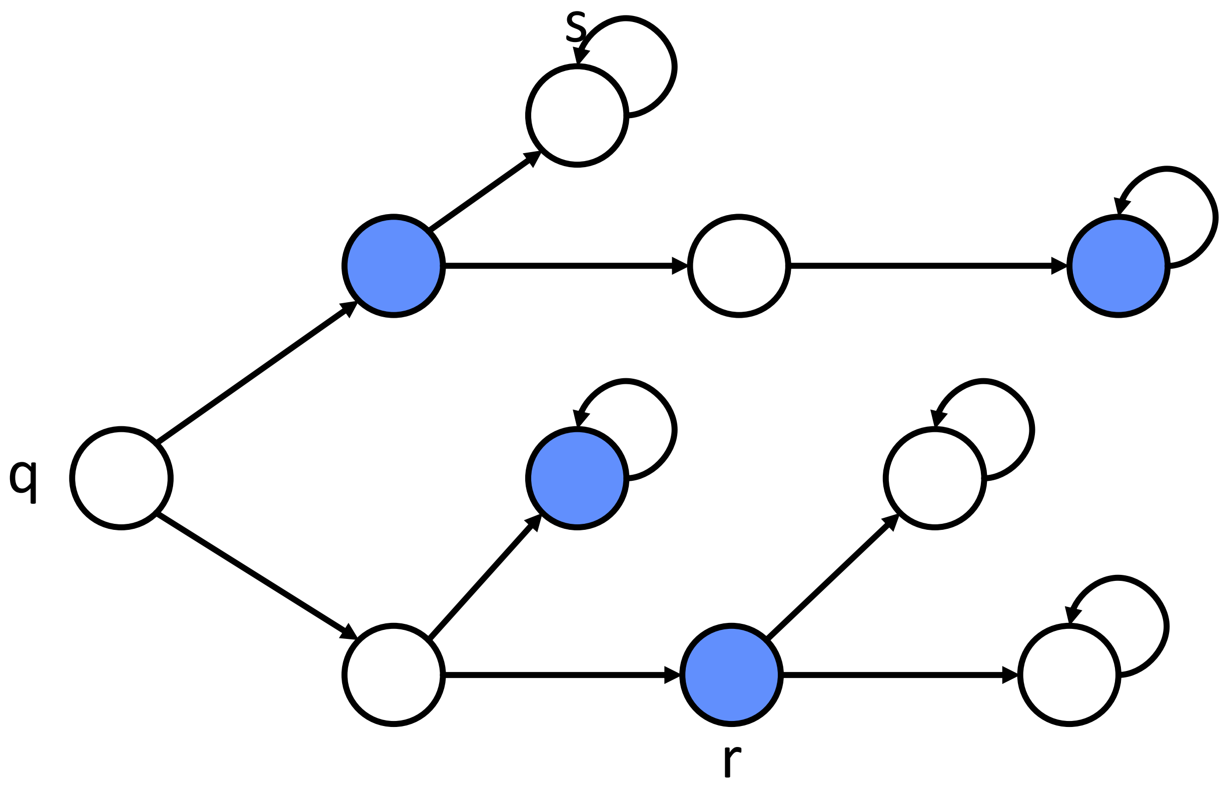
$q \vDash AF\ \phi$

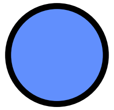
$r \vDash ?$

$s \vDash ?$

Formulation of CTL properties

AF ϕ : "On all paths, at some state ϕ holds ."





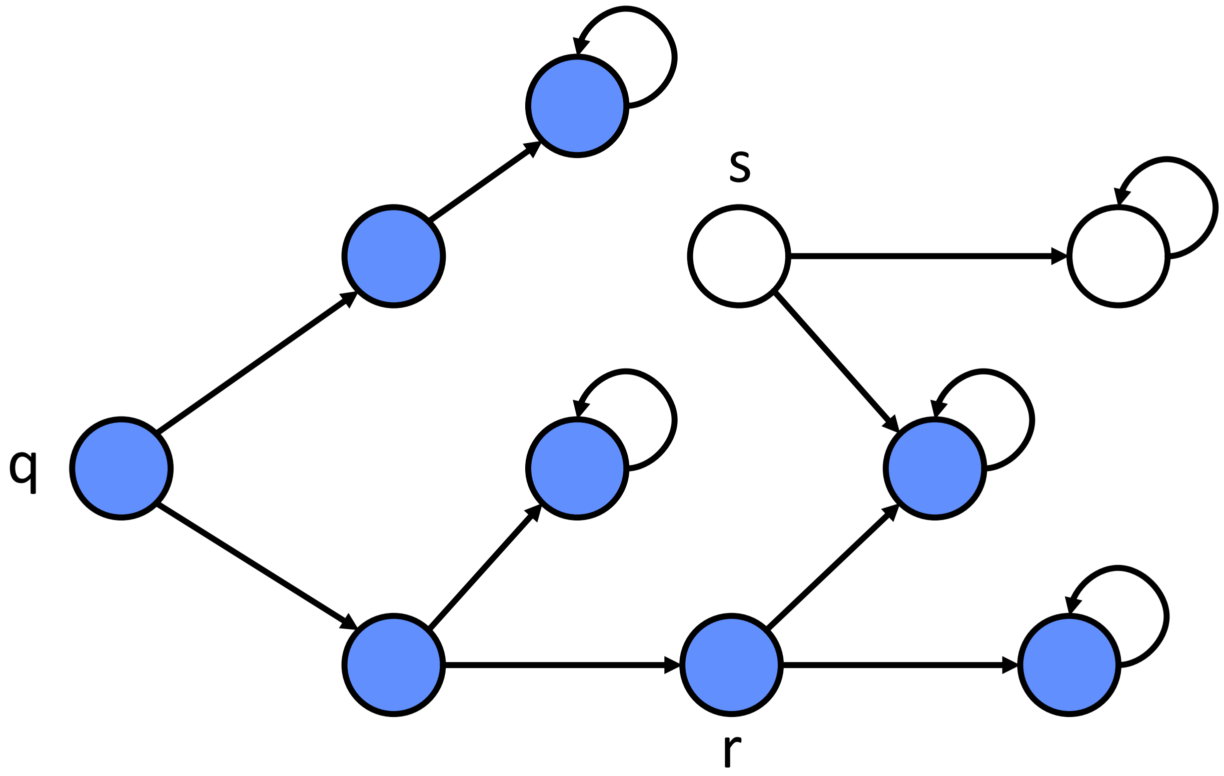
$q \models AF\ \phi$

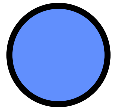
$r \models AF\ \phi$

$s \not\models AF\ \phi$

Formulation of CTL properties

AG ϕ : "On all paths, for all states ϕ holds."





$\models \phi$

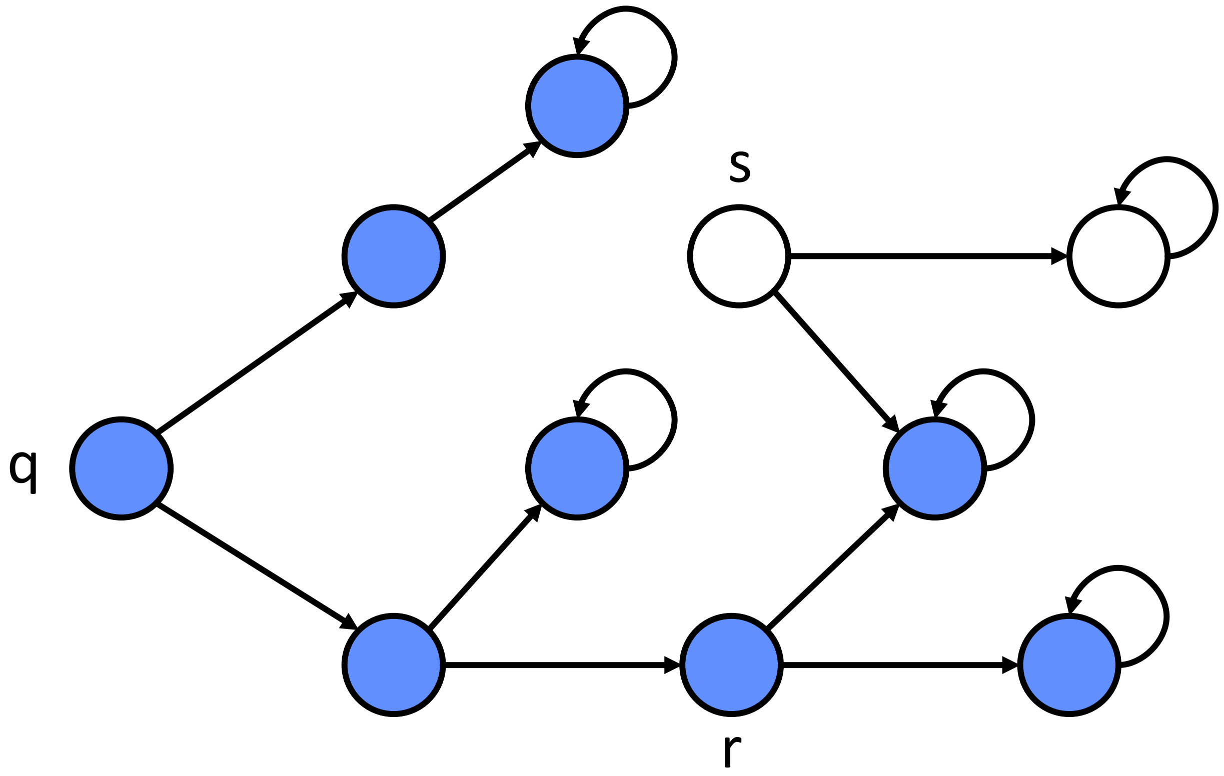
$q \models AG\ \phi$

$r \models ?$

$s \models ?$

Formulation of CTL properties

AG ϕ : "On all paths, for all states ϕ holds."



● $\vDash \phi$

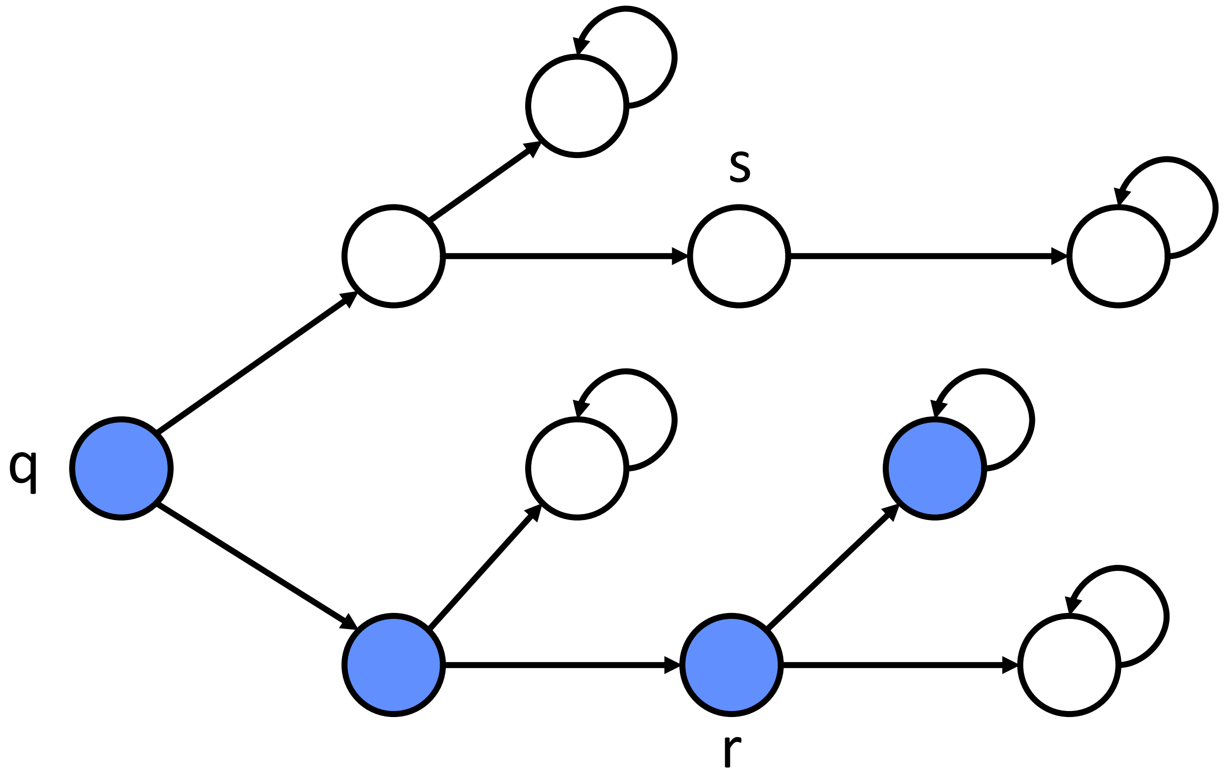
$q \vDash AG\ \phi$

$r \vDash AG\ \phi$

$s \nvDash AG\ \phi$

Formulation of CTL properties

EG ϕ : "There exists a path along which for all states ϕ holds ."



$\models \phi$

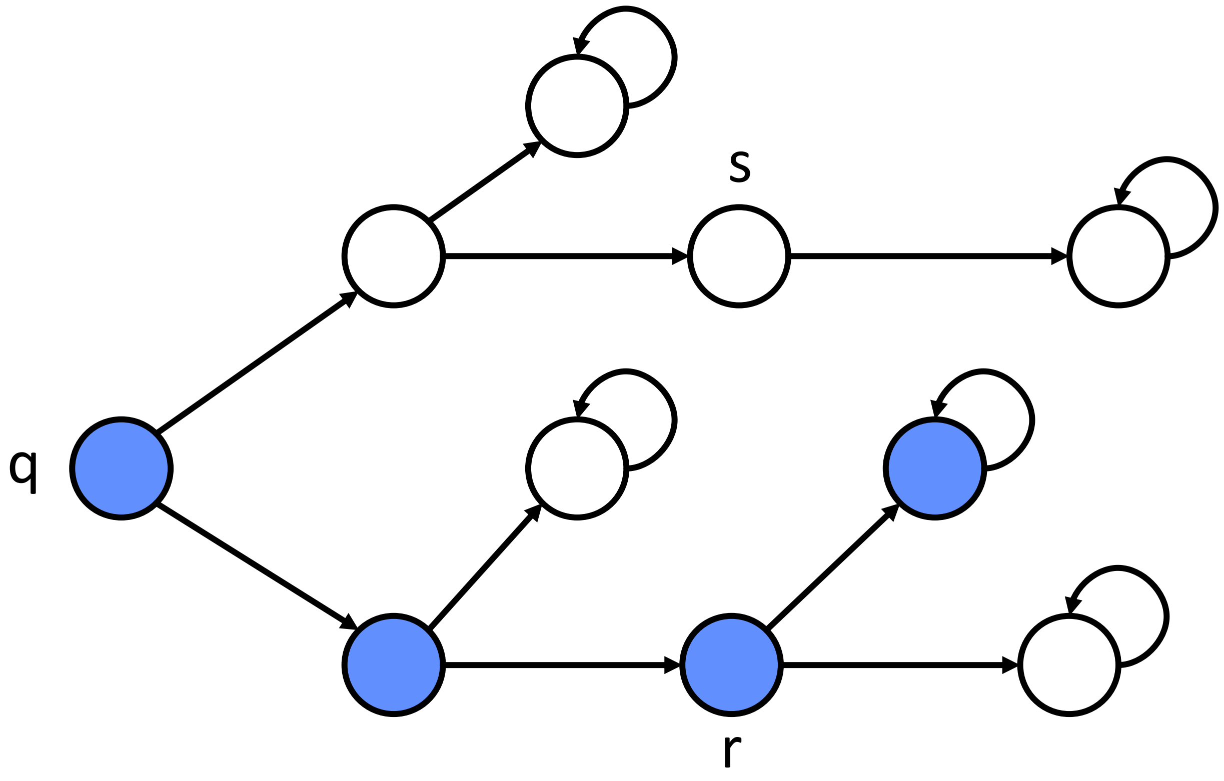
$q \models EG\ \phi$

$r \models ?$

$s \models ?$

Formulation of CTL properties

EG ϕ : "There exists a path along which for all states ϕ holds ."



$\bullet \vDash \phi$

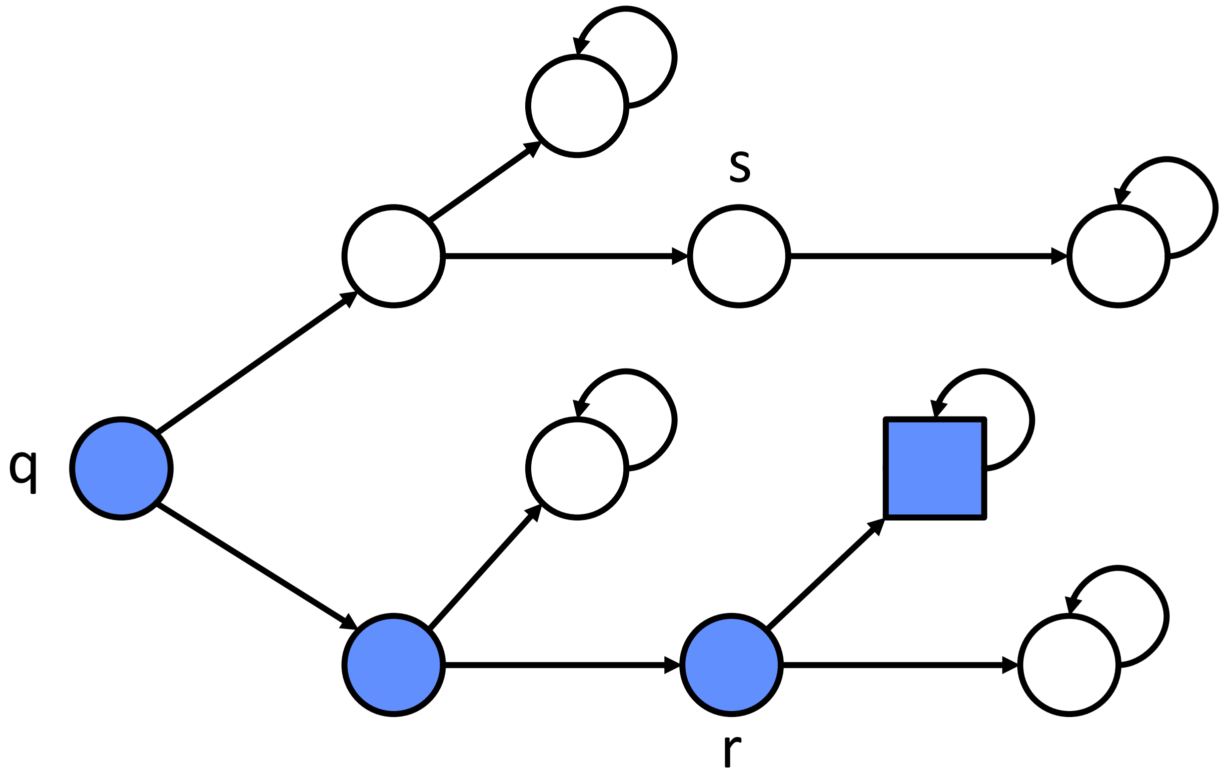
$q \vDash EG\ \phi$

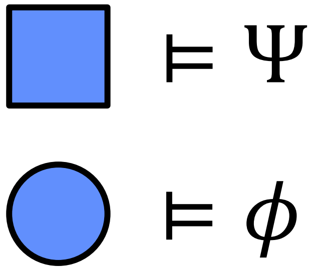
$r \vDash EG\ \phi$

$s \nvDash EG\ \phi$

Formulation of CTL properties

EϕUΨ : "There exists a path along which ϕ holds until Ψ holds."





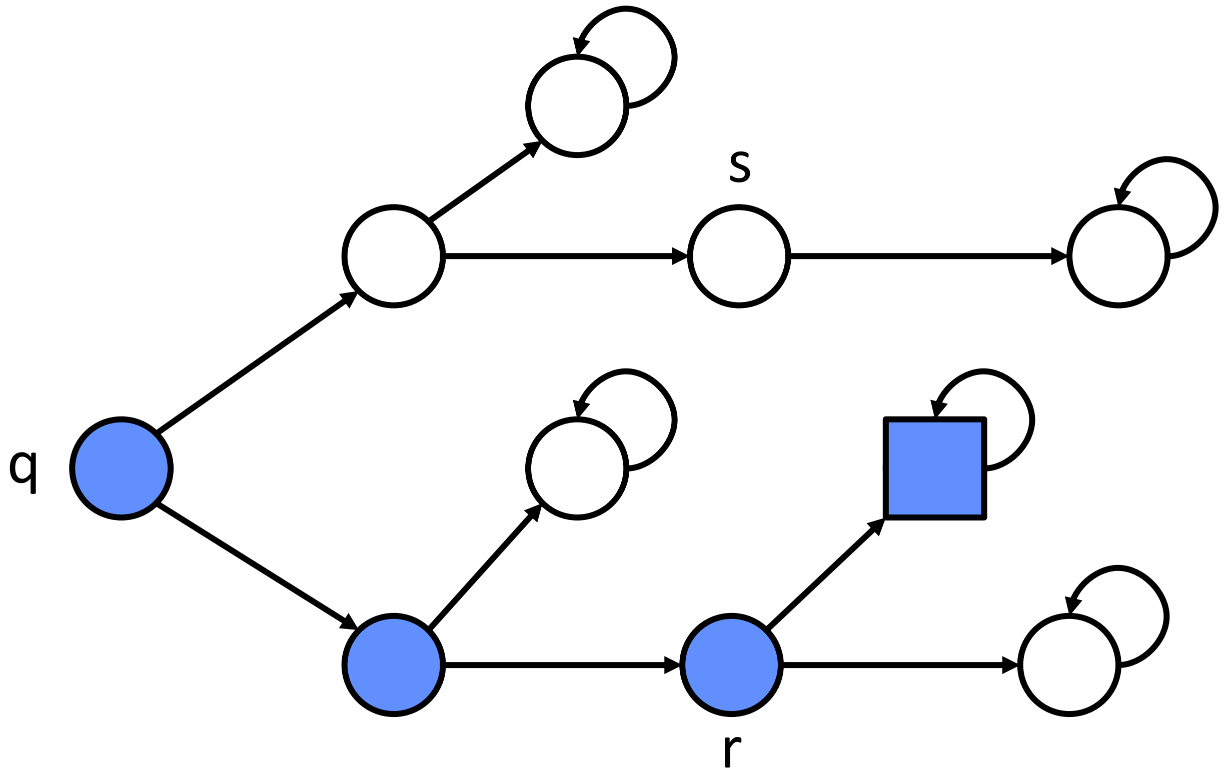
$q \models \phi EU\Psi$

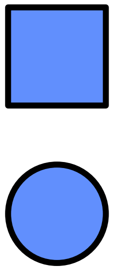
$r \models ?$

$s \models ?$

Formulation of CTL properties

$E\phi U\Psi$: "There exists a path along which ϕ holds until Ψ holds."





$\vDash \Psi$

$\vDash \phi$

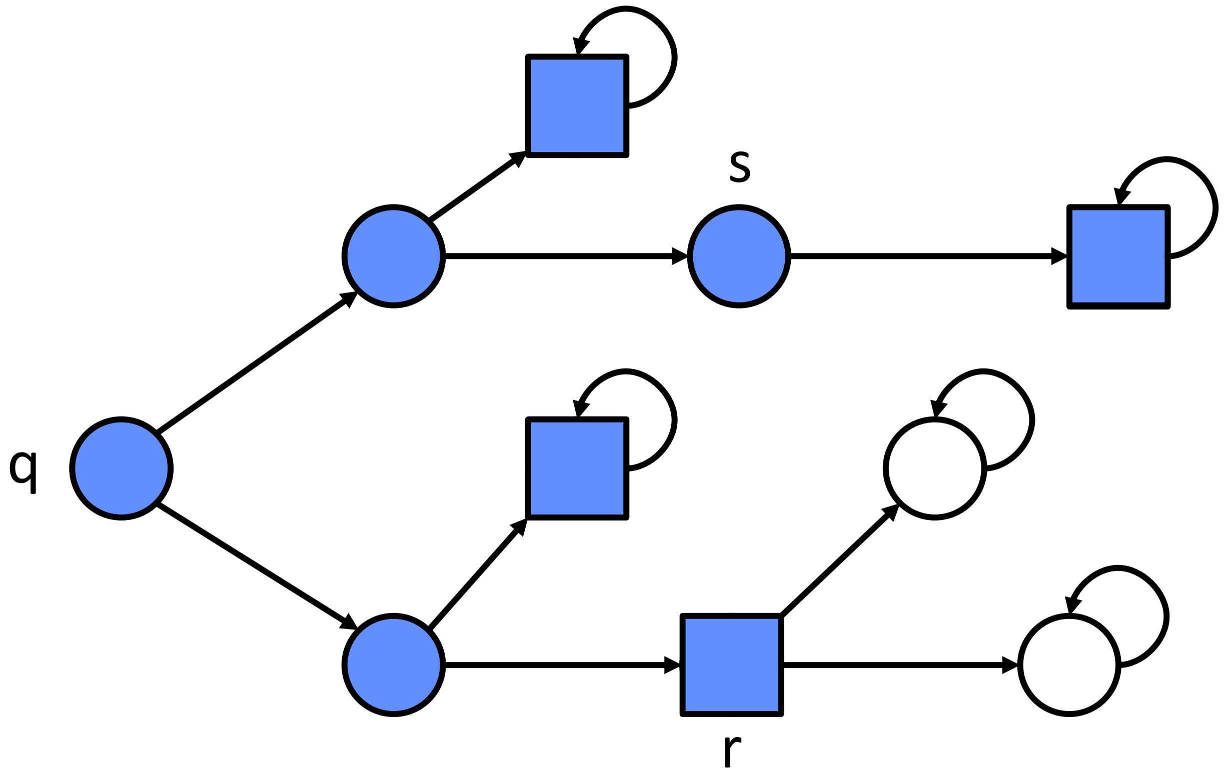
$q \vDash \phi EU\Psi$

$r \vDash \phi EU\Psi$

$s \nvDash \phi EU\Psi$

Formulation of CTL properties

$A\phi U\Psi$: "On all paths, ϕ holds until Ψ holds."



■ $\vDash \Psi$

● $\vDash \phi$

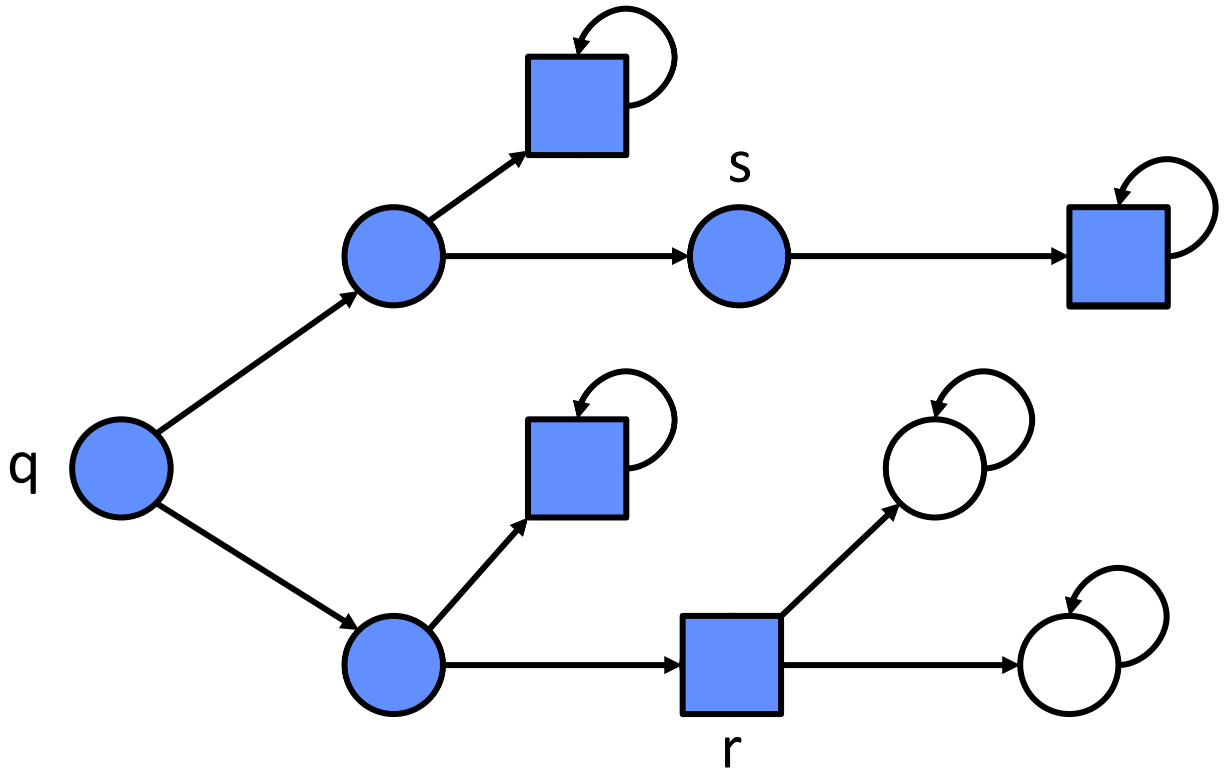
$q \vDash \phi AU\Psi$

$r \vDash ?$

$s \vDash ?$

Formulation of CTL properties

$A\phi U\Psi$: "On all paths, ϕ holds until Ψ holds."



■ $\models \Psi$

● $\models \phi$

$q \models \phi A U \Psi$

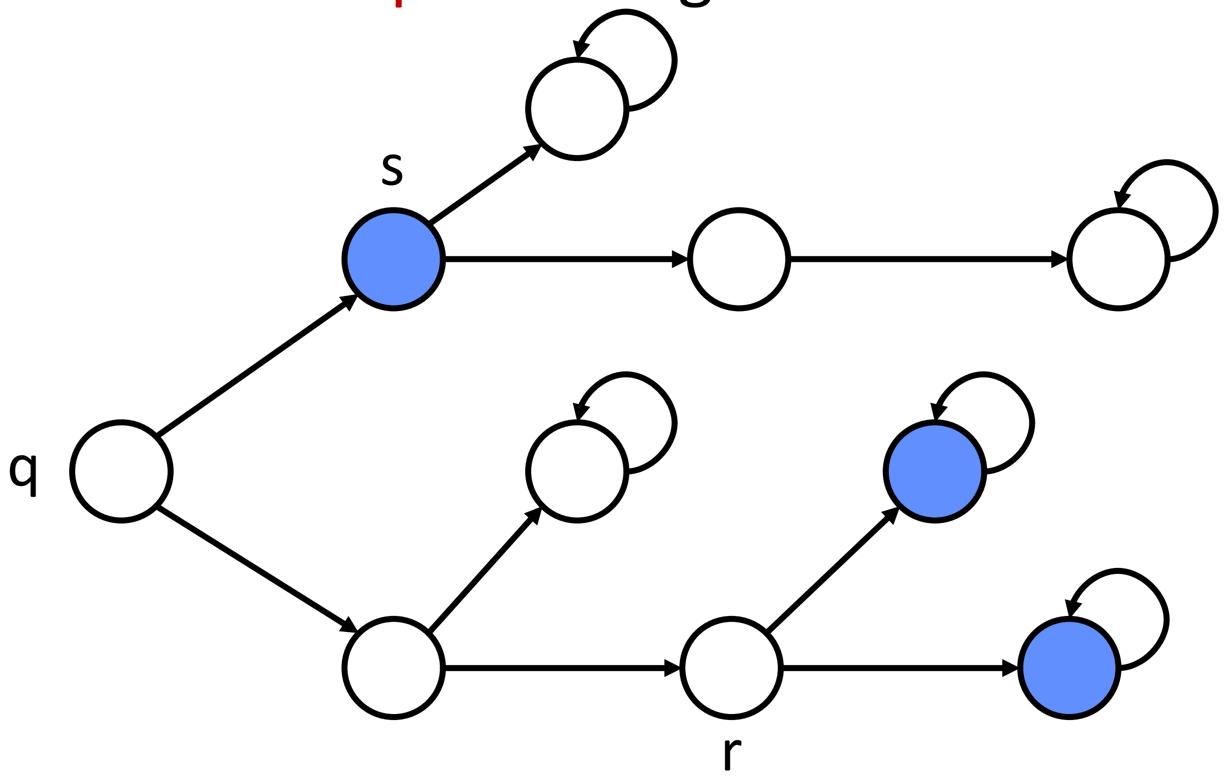
$r \models \phi A U \Psi$

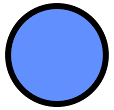
$s \models \phi A U \Psi$

Formulation of CTL properties

AXϕ : “On all paths, the next state satisfies ϕ.”

EXϕ : “There exists a path along which the next state satisfies ϕ.”



 $\models \phi$

$q \models EX\phi$

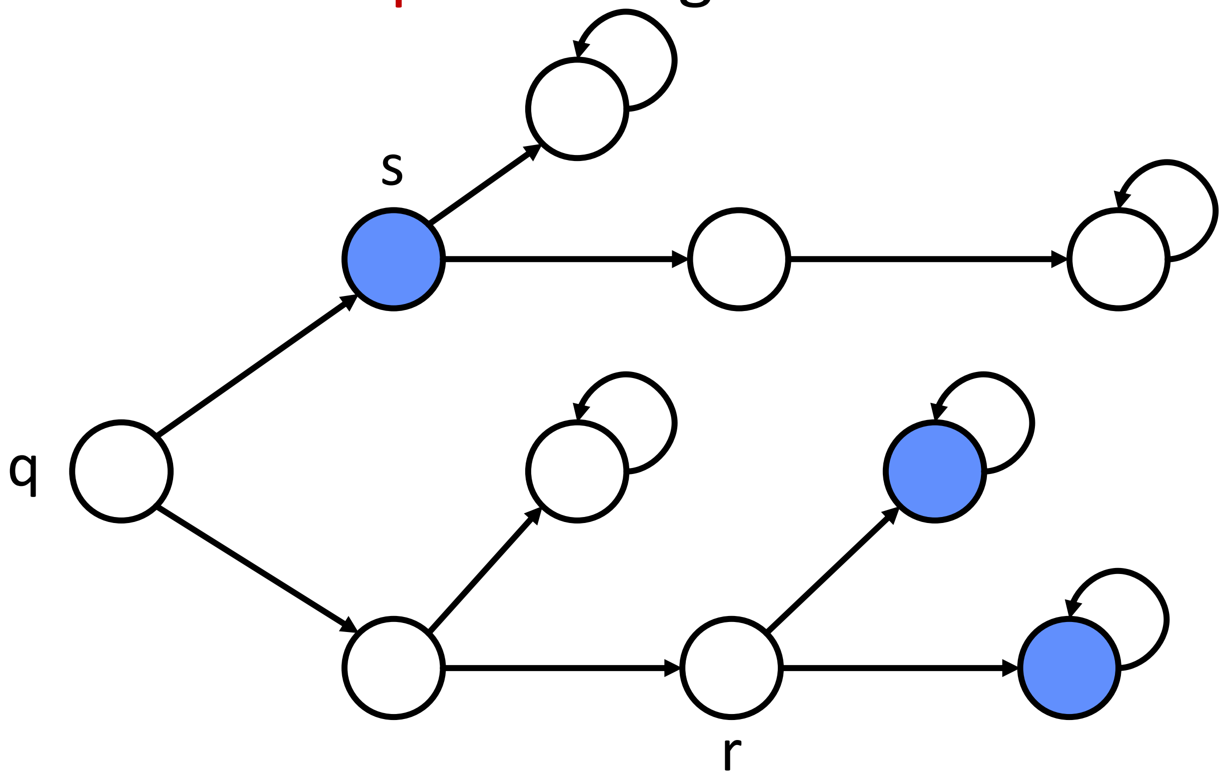
$r \models ?$

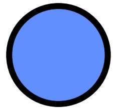
$s \models ?$

Formulation of CTL properties

AXϕ : “On all paths, the next state satisfies ϕ.”

EXϕ : “There exists a path along which the next state satisfies ϕ.”





$q \models EX\phi$

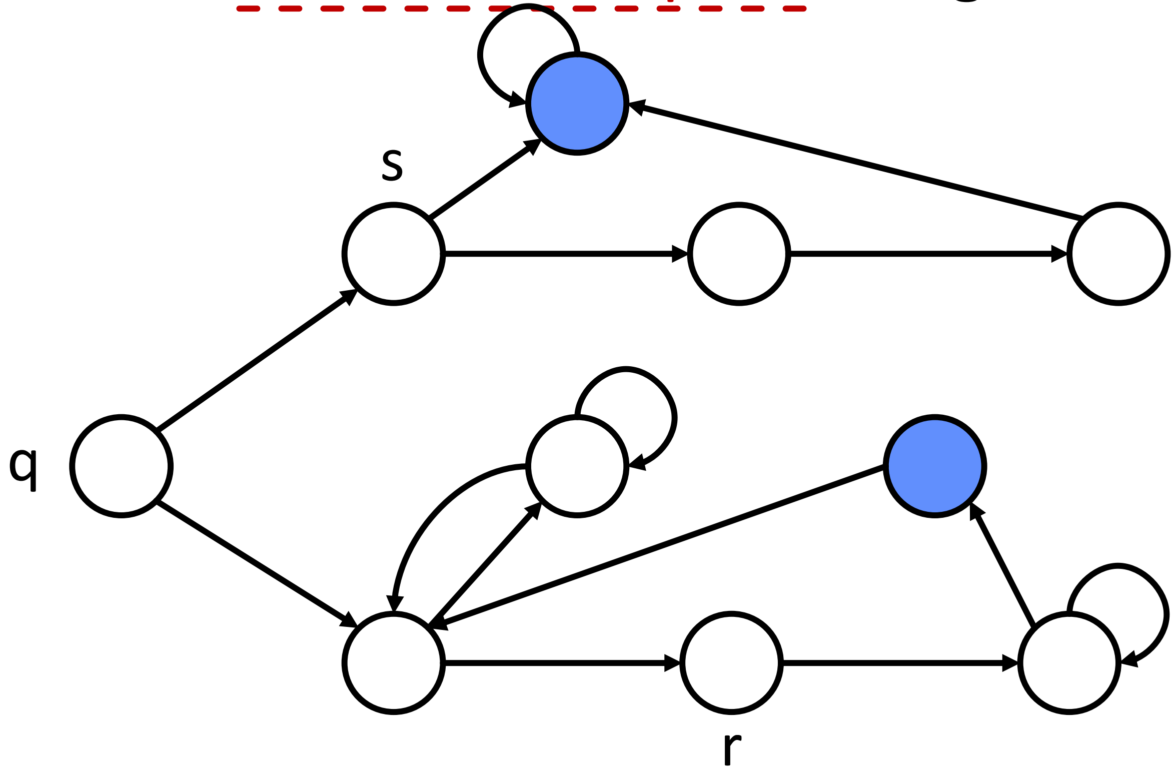
$r \models EX\phi$

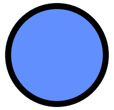
$s \not\models EX\phi$

Formulation of CTL properties

AG EF ϕ : "On all paths and for all states,

there exists a path along which at some state ϕ holds."



 $\vDash \phi$

$q \vDash AG\ EF\phi$

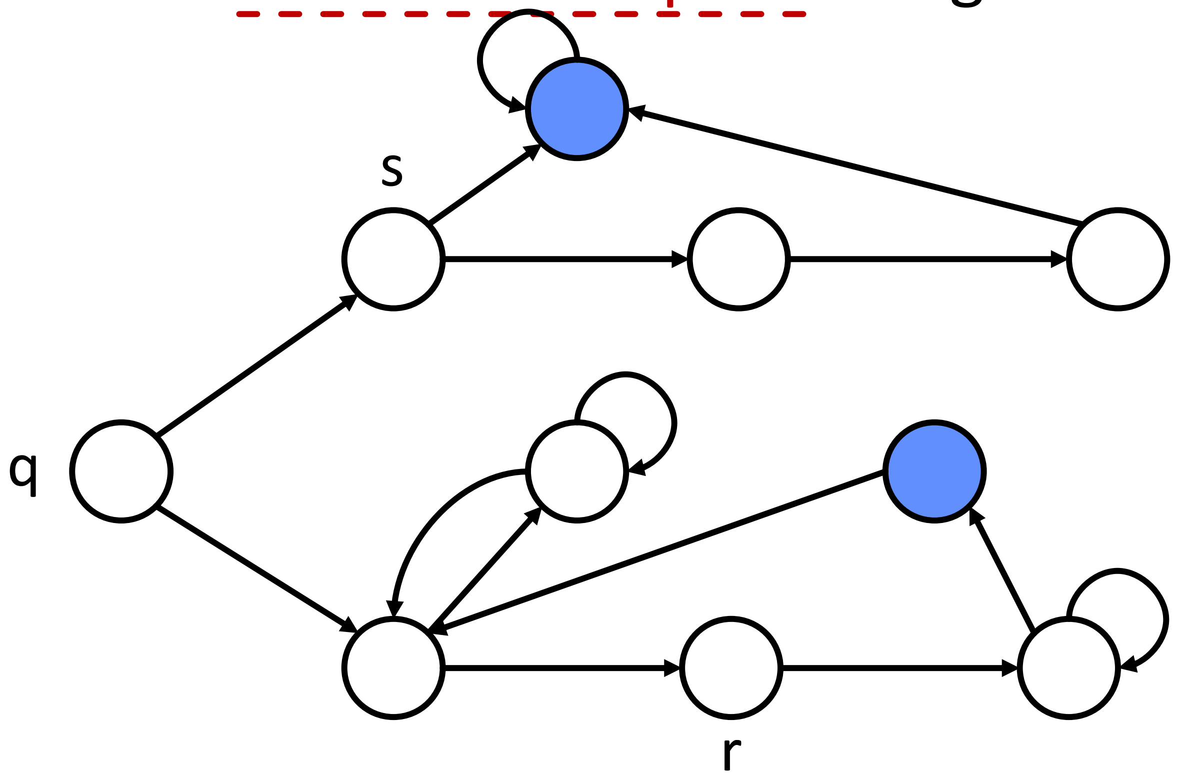
$r \vDash ?$

$s \vDash ?$

Formulation of CTL properties

AG EF ϕ : "On all paths and for all states,

there exists a path along which at some state ϕ holds."



$\vDash \phi$

q $\vDash$ AG EFϕ

r $\vDash$ AG EFϕ

s $\vDash$ AG EFϕ

Inverting properties is sometimes useful!

$\text{AG}\ \phi \equiv \neg\ \text{EF}\ \neg\phi$

$\text{AF}\ \phi \equiv \neg\ \text{EG}\ \neg\phi$

$\text{EF}\ \phi \equiv \neg\ \text{AG}\ \neg\phi$

$\text{EG}\ \phi \equiv \neg\ \text{AF}\ \neg\phi$

→ "On all paths, for all states ϕ holds."

≡

"There exists no path along which at some state ϕ doesn't hold."

→ …

Remark There exists other temporal logics

→ LTL (Linear Tree Logic)

→ CTL* = {CTL,LTL}

→ …

How to verify CTL properties?

Convert the property verification into a reachability problem

1. Start from states in which the property holds;
2. Compute all predecessor states for which the property still holds true;
   (same as for computing successor, with the inverse the transition function)
3. If initial states set is a subset, the property is satisfied by the model.

Computation specifics are described in the lecture slides.

So… what is Model-Checking exactly?

An algorithm

Input

- A DES model, **M**
  - Finite automata,
  - Petri nets,
  - Kripke machine, …
- A logic property, ϕ
  - CTL,
  - LTL, …

Output

- $\mathbf{M} \vDash \phi$?
- A trace for which the property does not hold!

Crash course – Verification of Finite Automata
CTL model-checking

Your turn to work!

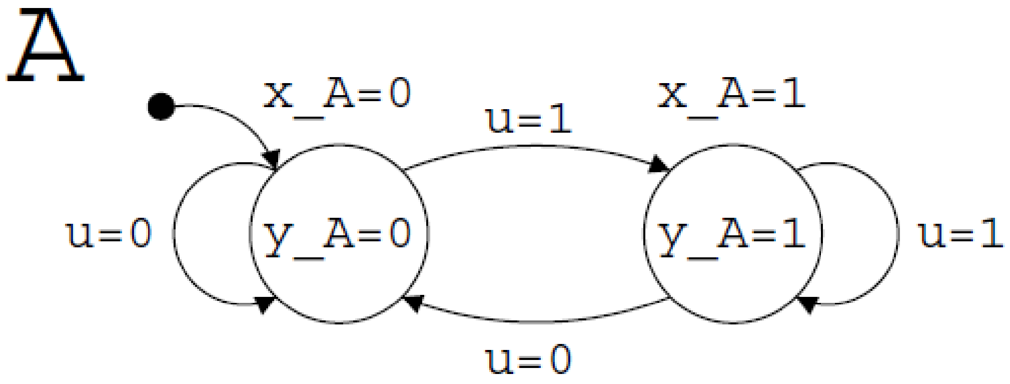
Slides online on my webpage:

http://people.ee.ethz.ch/~jacobr/

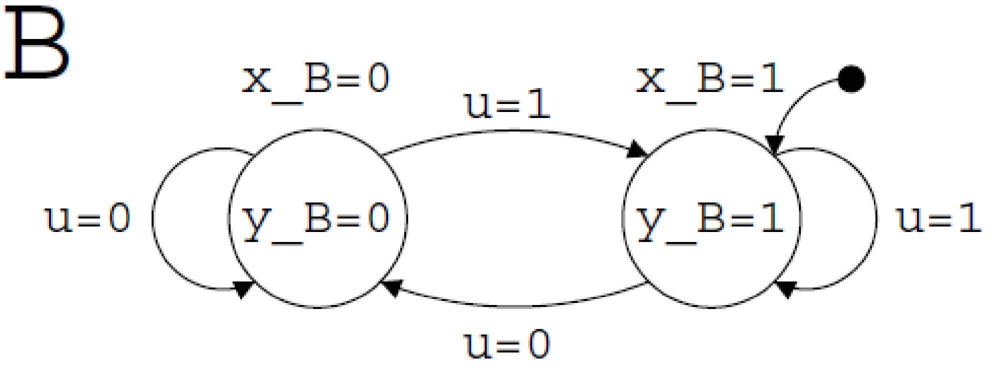
Comparison of Finite Automata

a) Express the characteristic function of the transition relation for both automaton, $\psi_r(x, x', u)$.

$$\psi_A(x_A, x'_A, u) = \overline{x_A}\,\overline{x'_A}\,\overline{u} + \overline{x_A}\,x'_A u + x_A x'_A u + x_A \overline{x'_A}\,\overline{u}$$



$$\psi_B(x_B, x'_B, u) = \overline{x_B}\,\overline{x'_B}\,\overline{u} + \overline{x_B}\,x'_B u + x_B x'_B u + x_B \overline{x'_B}\,\overline{u}$$



Comparison of Finite Automata

b) Express the joint transition function, ψ_f.

$$\psi_f(x_A, x'_A, x_B, x'_B) = (\exists u : \psi_A(x_A, x'_A, u) \cdot \psi_B(x_B, x'_B, u))$$

$$\psi_f(x_A, x'_A, x_B, x'_B)$$

$$= (\overline{x_A}x'_A + x_A x'_A) \cdot (\overline{x_B}x'_B + x_B x'_B) +$$
$$(\overline{x_A}\,\overline{x'_A} + x_A \overline{x'_A}) \cdot (\overline{x_B}\,\overline{x'_B} + x_B \overline{x'_B})$$

$$= \overline{x_A}x'_A\overline{x_B}x'_B + \overline{x_A}x'_A x_B x'_B + x_A x'_A \overline{x_B}x'_B + x_A x'_A x_B x'_B +$$
$$\overline{x_A}\,\overline{x'_A}\,\overline{x_B}\,\overline{x'_B} + \overline{x_A}\,\overline{x'_A} x_B \overline{x'_B} + x_A \overline{x'_A}\,\overline{x_B}\,\overline{x'_B} + x_A \overline{x'_A} x_B \overline{x'_B}$$

Comparison of Finite Automata

c) Express the characteristic function of the reachable states, $\psi_X(x_A, x_B)$.

$$\psi_{X_0}(x_A, x_B) = \overline{x_A} x_B$$

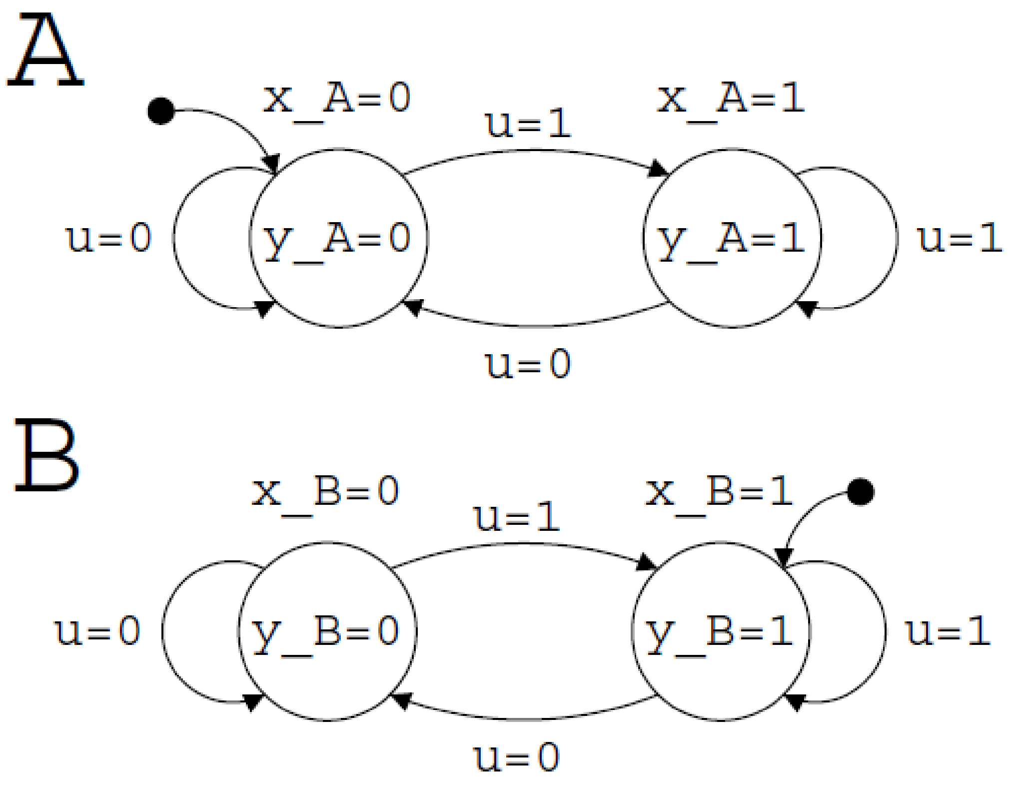
$$\psi_{X_1} = \overline{x_A} x_B + \overline{x_A x_B} + x_A x_B$$

$$\psi_{X_2} = \overline{x_A} x_B + \overline{x_A x_B} + x_A x_B$$

$$= \psi_{X_1}$$

$\rightarrow$ the fix-point is reached!

$$\boxed{\psi_X = \overline{x_A} x_B + \overline{x_A x_B} + x_A x_B}$$



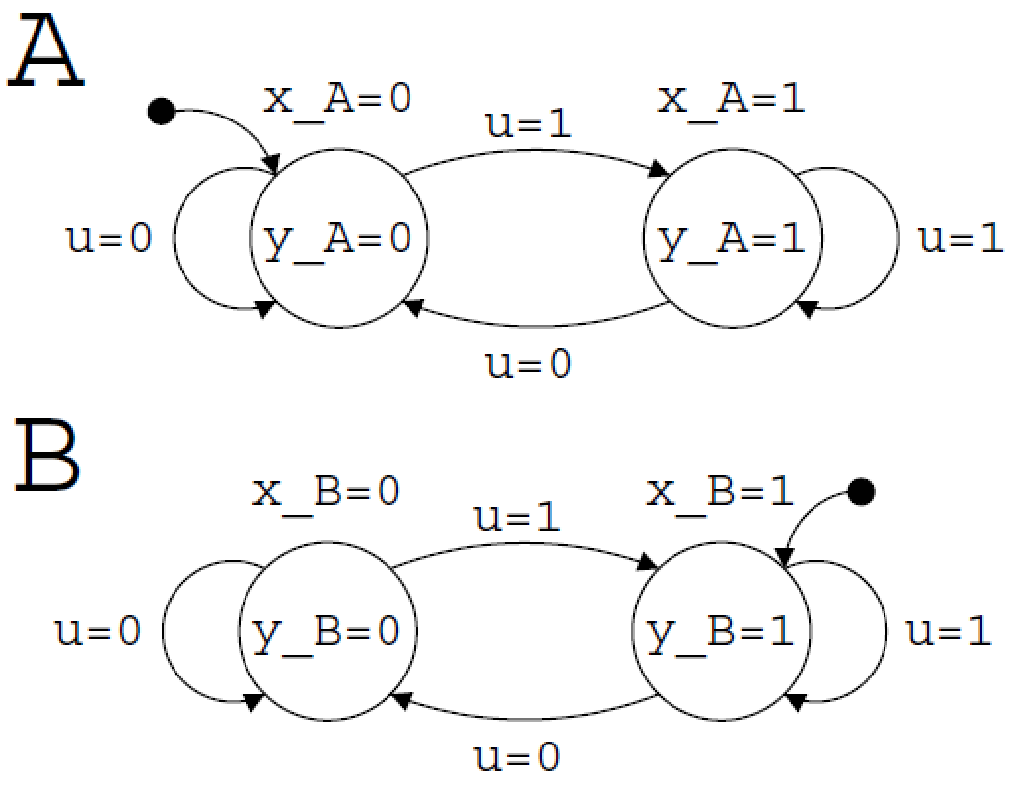
Comparison of Finite Automata

d) Express the characteristic function of the reachable output, $\psi_Y(x_A, x_B)$.

$$\psi_{g_A} = \overline{x_A y_A} + x_A y_A$$
$$\psi_{g_B} = \overline{x_B y_B} + x_B y_B$$

and $$\boxed{\psi_X = \overline{x_A} x_B + \overline{x_A x_B} + x_A x_B}$$

$$\psi_Y(y_A, y_B)$$
$$= (\exists (x_A, x_B) : \psi_X \cdot \psi_{g_A} \cdot \psi_{g_B})$$
$$= y_A y_B + \overline{y_A y_B} + \overline{y_A} y_B$$



Comparison of Finite Automata

e) Are the automata equivalent? **Hint**: Evaluate, for example, $\psi_Y(0,1)$.

$$\psi_Y((y_A, y_B) = (0,1)) = 1$$

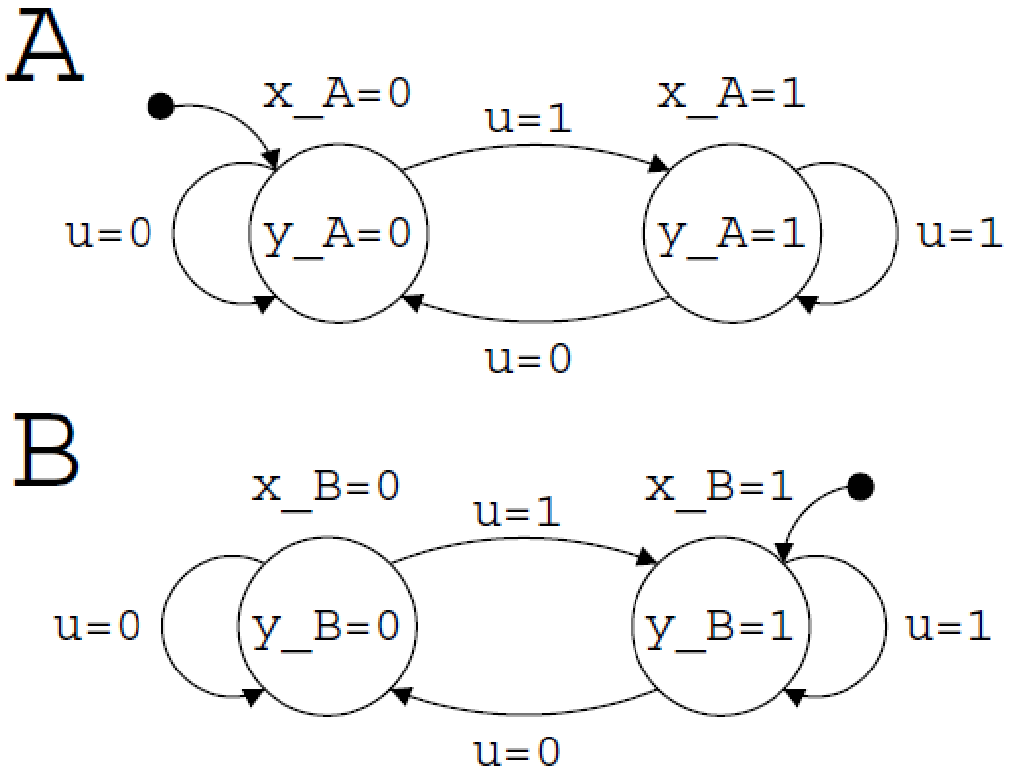
Or, in a more general way,

$$\psi_Y(y_A, y_B) = y_A y_B + \overline{y_A y_B} + \overline{y_A} y_B$$

and $(y_A \neq y_B) = \overline{y_A} y_B + y_A \overline{y_B}$

implies $\psi_Y \cdot (y_A \neq y_B) \neq 0$

→ Automata are not equivalent.



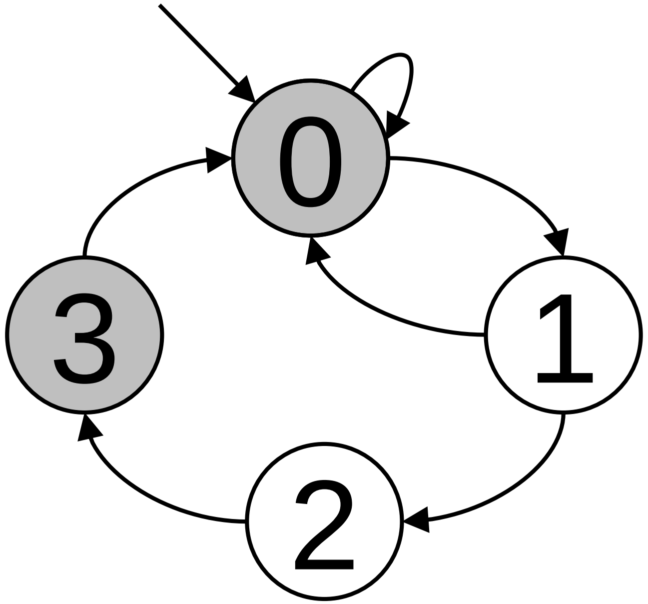
Temporal Logic

i. EF a

ii. EG a

iii. EX AX a

iv. EF (a AND EX NOT(a))



Temporal Logic

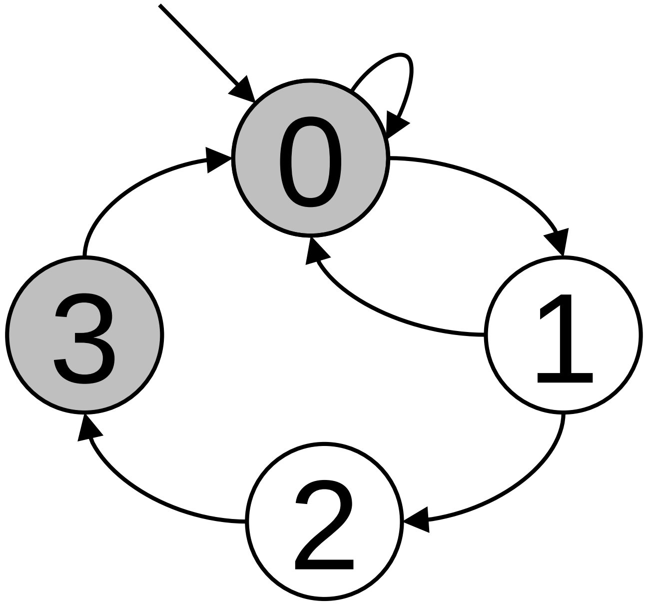
i. EF a

$Q = \{0, 1, 2, 3\}$

ii. EG a

iii. EX AX a

iv. EF (a AND EX NOT(a))



Temporal Logic

i. EF a

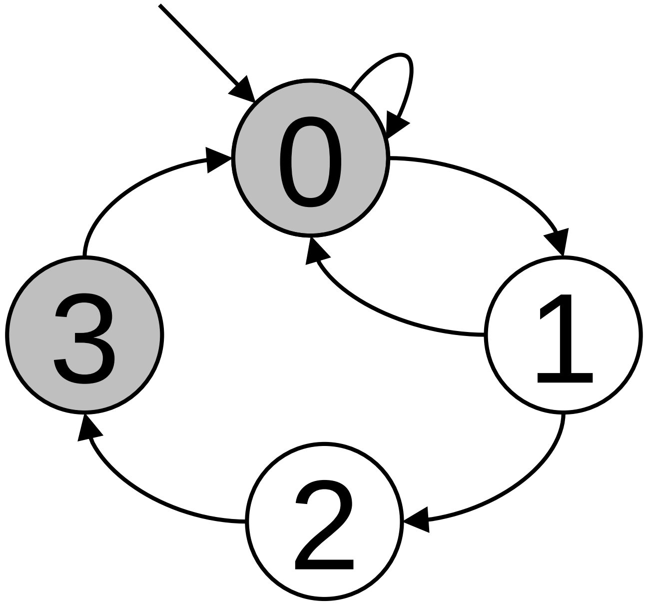
$$Q = \{0, 1, 2, 3\}$$

ii. EG a

$$Q = \{0, 3\}$$

iii. EX AX a

iv. EF (a AND EX NOT(a))



Temporal Logic

i. EF a

$$Q = \{0, 1, 2, 3\}$$

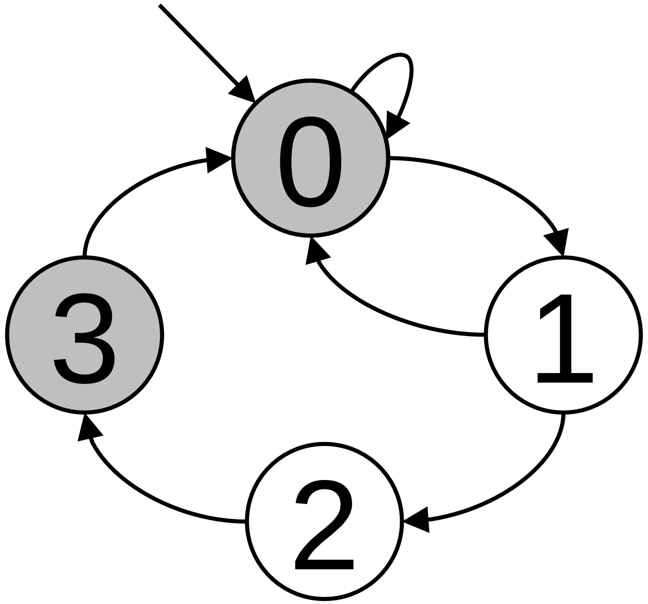
ii. EG a

$$Q = \{0, 3\}$$

iii. EX AX a

$$Q = \{1, 2\}$$

iv. EF (a AND EX NOT(a))



Temporal Logic

i. EF a

$Q = \{0, 1, 2, 3\}$

ii. EG a

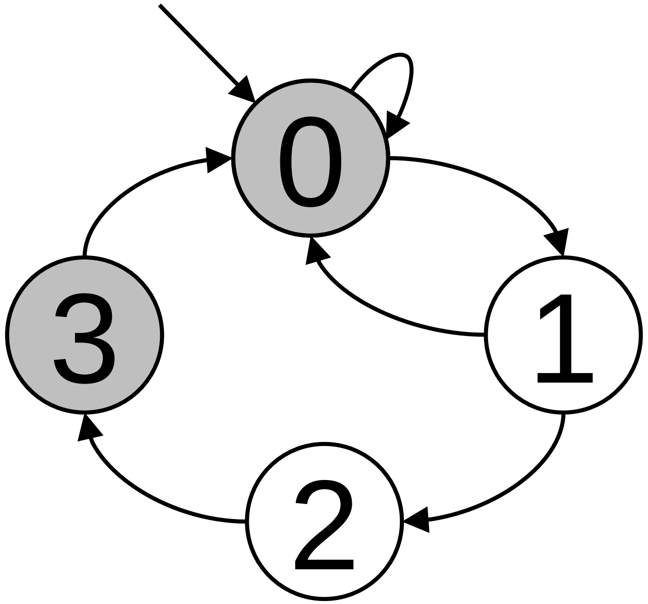
$Q = \{0, 3\}$

iii. EX AX a

$Q = \{1, 2\}$

iv. EF (a AND EX NOT(a))

$Q = \{0, 1, 2, 3\}$



Temporal Logic

Trick AF Z not(EG not(Z))

Require: ψ_Z, ψ_f $\triangleright$ Equivalence in term of sets:

current $=$ NOT(ψ_Z); $\triangleright\ X_0$

next $=$ current AND $\psi_{\texttt{PRE}(current,f)}$; $\triangleright\ X_1 = X_0 \cap Pre(X_0, f)$

while next != current **do** $\triangleright\ X_i\ != X_{i-1}$

  current $=$ next;

  next $=$ current AND $\psi_{\texttt{PRE}(current,f)}$; $\triangleright\ X_i = X_{i-1} \cap Pre(X_{i-1}, f)$

end while $\triangleright\ X_f \models \text{EG NOT}(Z)$

return $\psi_{\text{AF }Z}$ =NOT(current); $\triangleright\ \overline{X_f} \models \text{AF } Z = \text{NOT(EG NOT}(Z))$

Crash course – Verification of Finite Automata

CTL model-checking

See you next week!

Slides online on my webpage:

http://people.ee.ethz.ch/~jacobr/