CS 181u Applied Logic

Lecture 10

Today's class

Linear Temporal Logic

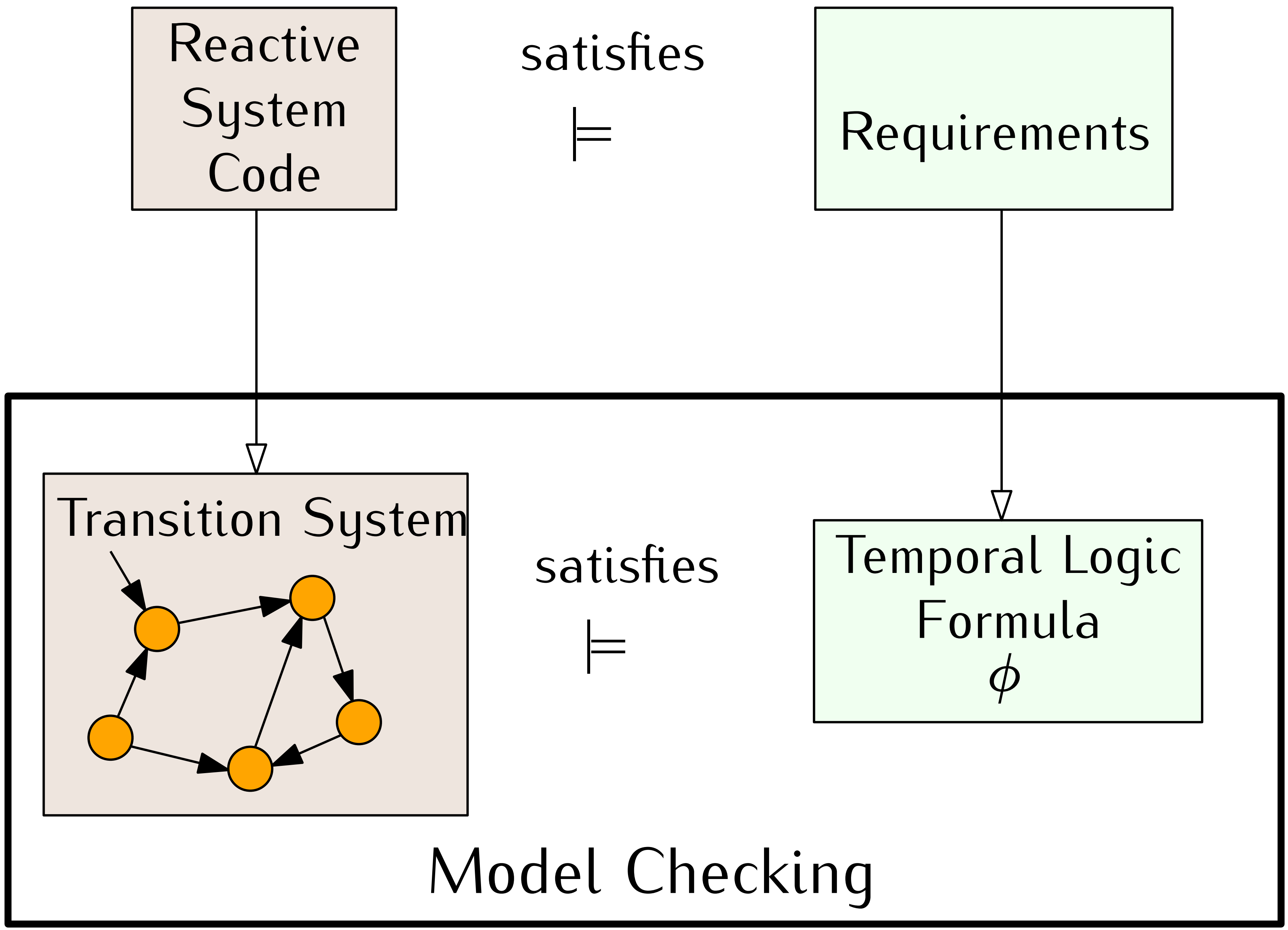
W and R operators. LTL operator basis.

Computation Tree Logic

Some practice, nested CTL formulas

LTL vs CTL

Equivalance of two temporal formulas


Reactive System Code
satisfies
⊨
Requirements
Transition System
satisfies
⊨
Temporal Logic Formula
ϕ
Model Checking

LTL W and R operators

For a path π of transition system $\mathcal{M}$,

LTL W and R operators

For a path π of transition system $\mathcal{M}$,

$$\pi \models \phi \; W \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \psi \wedge \forall \; 1 \leq j < i \;\; \pi^j \models \phi)$$
$$\vee \;\; \forall k \geq 1 \;\; \pi^k \models \phi$$

Weak Until

LTL W and R operators

For a path π of transition system $\mathcal{M}$,

$$\pi \models \phi \; W \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \psi \wedge \forall\, 1 \leq j < i \;\; \pi^j \models \phi) \\ \vee \;\; \forall k \geq 1 \;\; \pi^k \models \phi$$

Weak Until

$$\pi \models \phi \; R \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \phi \wedge \forall \;\; 1 \leq j \leq i \;\; \pi^j \models \psi) \\ \vee \;\; \forall k \geq 1 \;\; \pi^k \models \psi$$

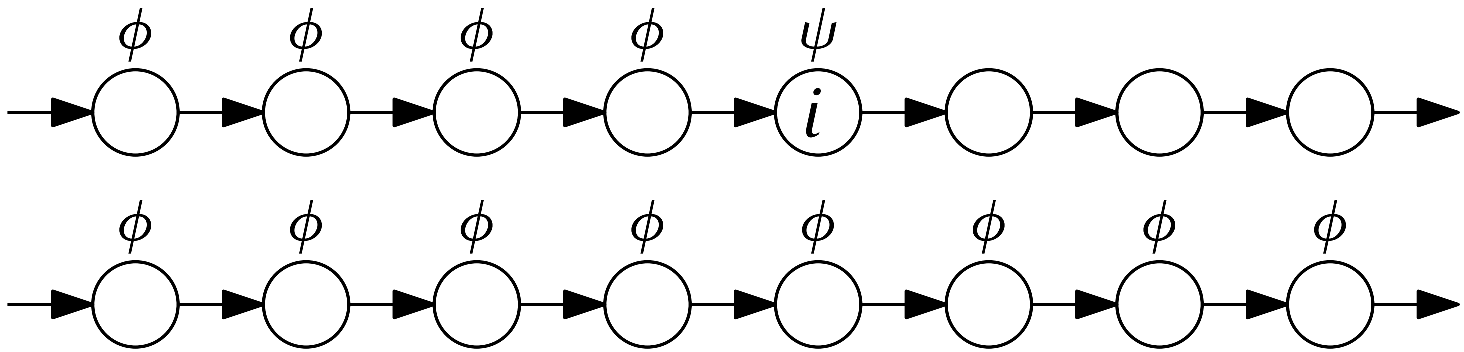
Release

LTL W and R operators

For a path π of transition system $\mathcal{M}$,

$$\pi \models \phi \; W \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \psi \land \forall\, 1 \leq j < i \;\; \pi^j \models \phi)$$
$$\lor \;\; \forall k \geq 1 \;\; \pi^k \models \phi$$

Weak Until



or

$$\pi \models \phi \; R \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \phi \land \forall\; 1 \leq j \leq i \;\; \pi^j \models \psi)$$
$$\lor \;\; \forall k \geq 1 \;\; \pi^k \models \psi$$

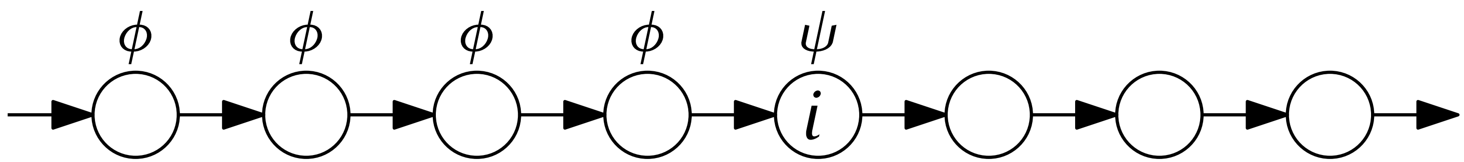
Release

LTL W and R operators

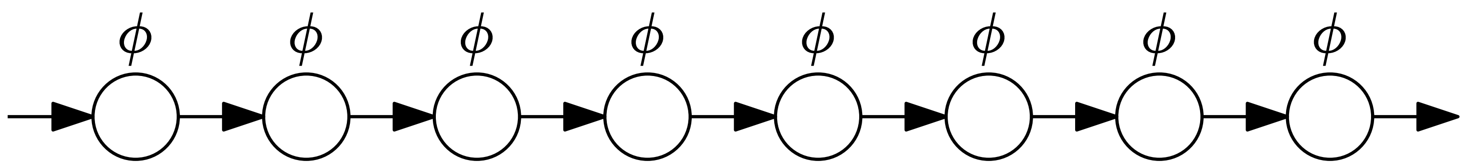
For a path π of transition system $\mathcal{M}$,

$$\pi \models \phi \; W \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \psi \wedge \forall \, 1 \leq j < i \;\; \pi^j \models \phi) \\ \vee \;\; \forall k \geq 1 \;\; \pi^k \models \phi$$

Weak Until

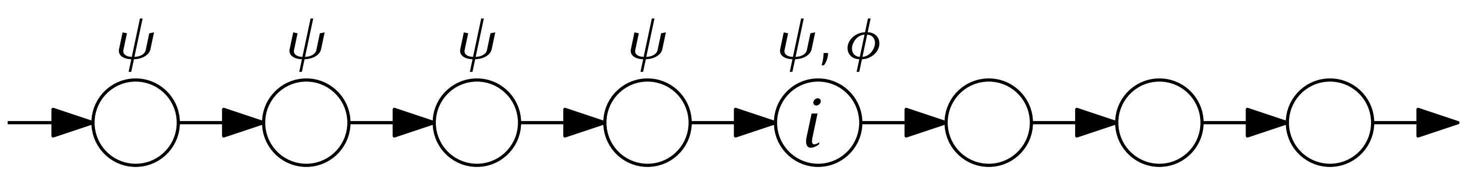


or

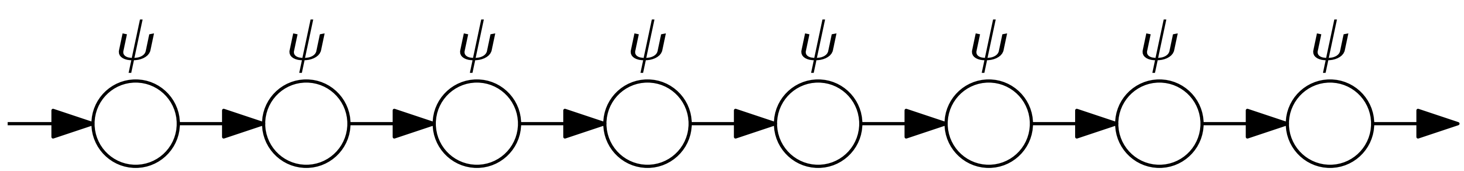


$$\pi \models \phi \; R \; \psi \quad \text{iff} \quad (\exists i \geq 1 \;\; \pi^i \models \phi \wedge \forall \; 1 \leq j \leq i \;\; \pi^j \models \psi) \\ \vee \;\; \forall k \geq 1 \;\; \pi^k \models \psi$$

Release



or



Why so many operators? G, X, U, F, W, R

Why so many operators? G, X, U, F, W, R

The acts of the mind, wherein it exerts its power over simple ideas, are chiefly these three: Combining several simple ideas into one compound one, and thus all complex ideas are made. The second is bringing two ideas, whether simple or complex, together, and setting them by one another so as to take a view of them at once, without uniting them into one, by which it gets all its ideas of relations. The third is separating them from all other ideas that accompany them in their real existence: this is called abstraction, and thus all its general ideas are made.

SICP by Abelson, Sussman, and Sussman quoting John Locke from his *Essay Concerning Human Understanding*

Why so many operators? G, X, U, F, W, R

We don't actually need any of them if we are OK with always writing temporal properties using first order logic and quantifying over paths.

Why so many operators? G, X, U, F, W, R

We don't actually need any of them if we are OK with always writing temporal properties using first order logic and quantifying over paths.

However, they are useful to have on hand to state things concisely, like when writing νSMV specifications.

$$GF(pc = w) \quad \text{vs} \quad \forall i \geq 1 \;\; \exists j \geq i \;\; \pi^j \models (pc = w)$$

Why so many operators? G, X, U, F, W, R

We don't actually need any of them if we are OK with always writing temporal properties using first order logic and quantifying over paths.

However, they are useful to have on hand to state things concisely, like when writing νSMV specifications.

$$GF(pc = w) \qquad \text{vs} \qquad \forall i \geq 1 \;\; \exists j \geq i \;\; \pi^j \models (pc = w)$$

On the other hand, when performing meta-analysis of LTL, we need to examine each operator.

Why so many operators? G, X, U, F, W, R

We don't actually need any of them if we are OK with always writing temporal properties using first order logic and quantifying over paths.

However, they are useful to have on hand to state things concisely, like when writing νSMV specifications.

$$GF(pc = w) \quad \text{vs} \quad \forall i \geq 1 \;\; \exists j \geq i \;\; \pi^j \models (pc = w)$$

On the other hand, when performing meta-analysis of LTL, we need to examine each operator.

Hence, it is good to reduce everything down to a smallest set of sufficiently expressive operators.

Why so many operators? G, X, U, F, W, R

We don't actually need any of them if we are OK with always writing temporal properties using first order logic and quantifying over paths.

However, they are useful to have on hand to state things concisely, like when writing νSMV specifications.

$$GF(pc = w) \quad \text{vs} \quad \forall i \geq 1 \ \ \exists j \geq i \ \ \pi^j \models (pc = w)$$

On the other hand, when performing meta-analysis of LTL, we need to examine each operator.

Hence, it is good to reduce everything down to a smallest set of sufficiently expressive operators.

$\{\wedge, \neg\}$	$\{\lambda, (f \quad e)\}$	$\{S, K, I\}$
Propositional logic	Lambda calc	Combinatory Logic

An operator basis for LTL

Let's get rid of a bunch of operators G, F, X, U, W, R

An operator basis for LTL

Let's get rid of a bunch of operators $\cancel{G}, \cancel{F}, X, U, W, R$

$$F\phi = \top \; U \; \phi$$

$$G\phi = \bot \; R \; \phi$$

An operator basis for LTL

Let's get rid of a bunch of operators $\cancel{G}, \cancel{F}, X, U, \cancel{W}, R$

$$F\phi = \top\ U\ \phi$$

$$G\phi = \bot\ R\ \phi$$

$$\phi\ W \psi = \phi\ U \psi\ \vee\ G\phi = \phi\ U\ \psi\ \vee\ \bot\ R\ \phi$$

An operator basis for LTL

Let's get rid of a bunch of operators $\cancel{G}, \cancel{F}, X, U, \cancel{W}, \cancel{R}$

$F\phi = \top\ U\ \phi$

$G\phi = \bot\ R\ \phi$

$\phi\ W\psi = \phi\ U\psi\ \lor\ G\phi = \phi\ U\ \psi\ \lor\ \bot\ R\ \phi$

$\phi\ R\ \psi = \neg(\neg\phi\ U\ \neg\psi)$

An operator basis for LTL

Let's get rid of a bunch of operators $\cancel{G}, \cancel{F}, X, U, \cancel{W}, \cancel{R}$

$$F\phi = \top \; U \; \phi$$

$$G\phi = \bot \; R \; \phi$$

$$\phi \; W \psi = \phi \; U \psi \;\vee\; G\phi = \phi \; U \; \psi \;\vee\; \bot \; R \; \phi$$

$$\phi \; R \; \psi = \neg(\neg\phi \; U \; \neg\psi)$$

X is special. Cannot be written in terms of the others.

An operator basis for LTL

Let's get rid of a bunch of operators $\cancel{G}, \cancel{F}, X, U, \cancel{W}, \cancel{R}$

$F\phi = \top \; U \; \phi$

$G\phi = \bot \; R \; \phi$

$\phi \; W \psi = \phi \; U \psi \;\vee\; G\phi = \phi \; U \; \psi \;\vee\; \bot \; R \; \phi$

$\phi \; R \; \psi = \neg(\neg\phi \; U \; \neg\psi)$

X is special. Cannot be written in terms of the others.

Hence, $\{U, X\}$ is a basis for LTL.

An operator basis for LTL

Let's get rid of a bunch of operators $\cancel{G}, \cancel{F}, X, U, \cancel{W}, \cancel{R}$

$$F\phi = \top \; U \; \phi$$

$$G\phi = \bot \; R \; \phi$$

$$\phi \; W \psi = \phi \; U \psi \; \vee \; G\phi = \phi \; U \; \psi \; \vee \; \bot \; R \; \phi$$

$$\phi \; R \; \psi = \neg(\neg\phi \; U \; \neg\psi)$$

X is special. Cannot be written in terms of the others.

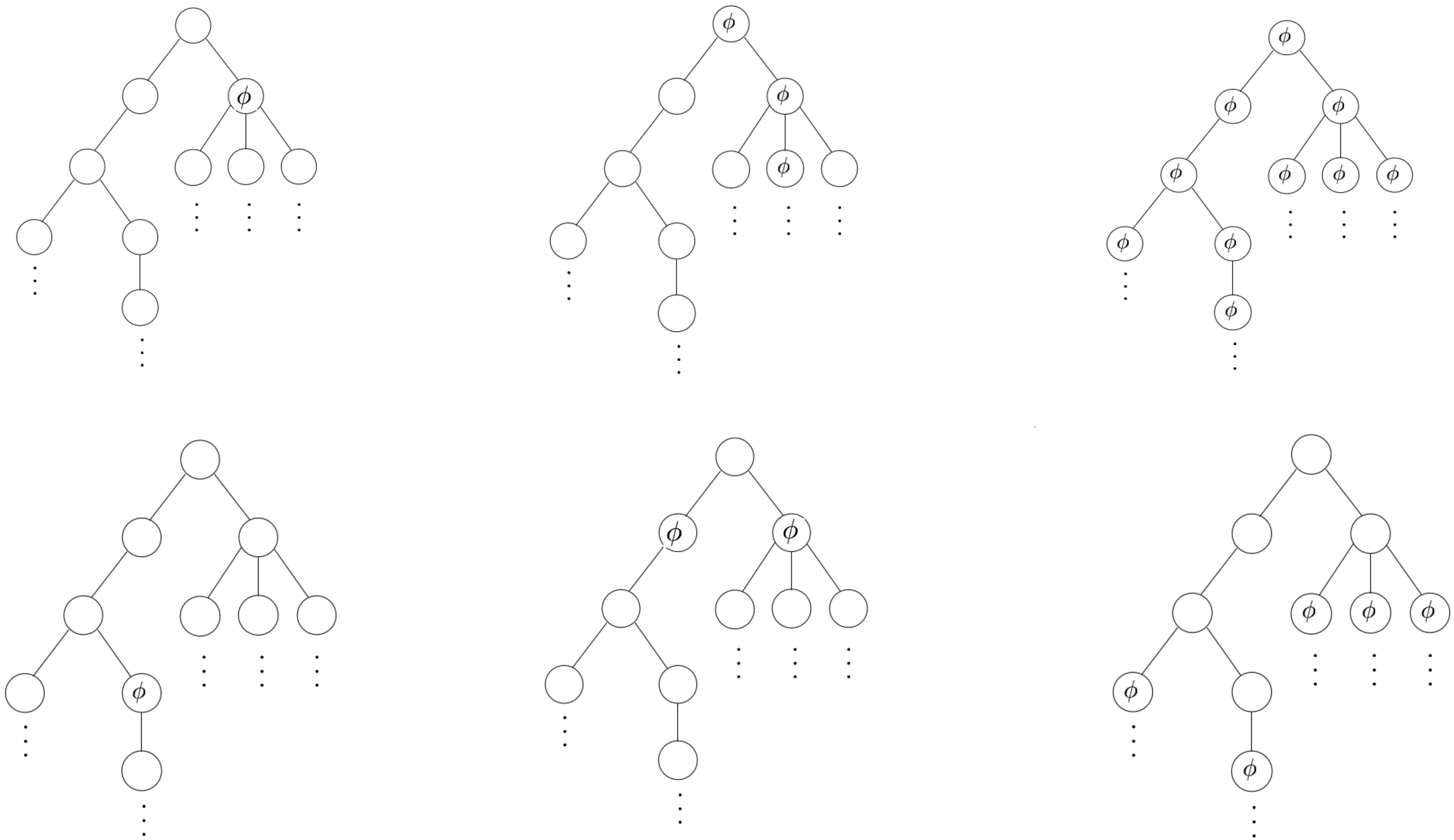
Hence, $\{U, X\}$ is a basis for LTL.

Similar reasoning shows that $\{R, X\}$
and $\{W, X\}$ are also bases.

CTL vs LTL

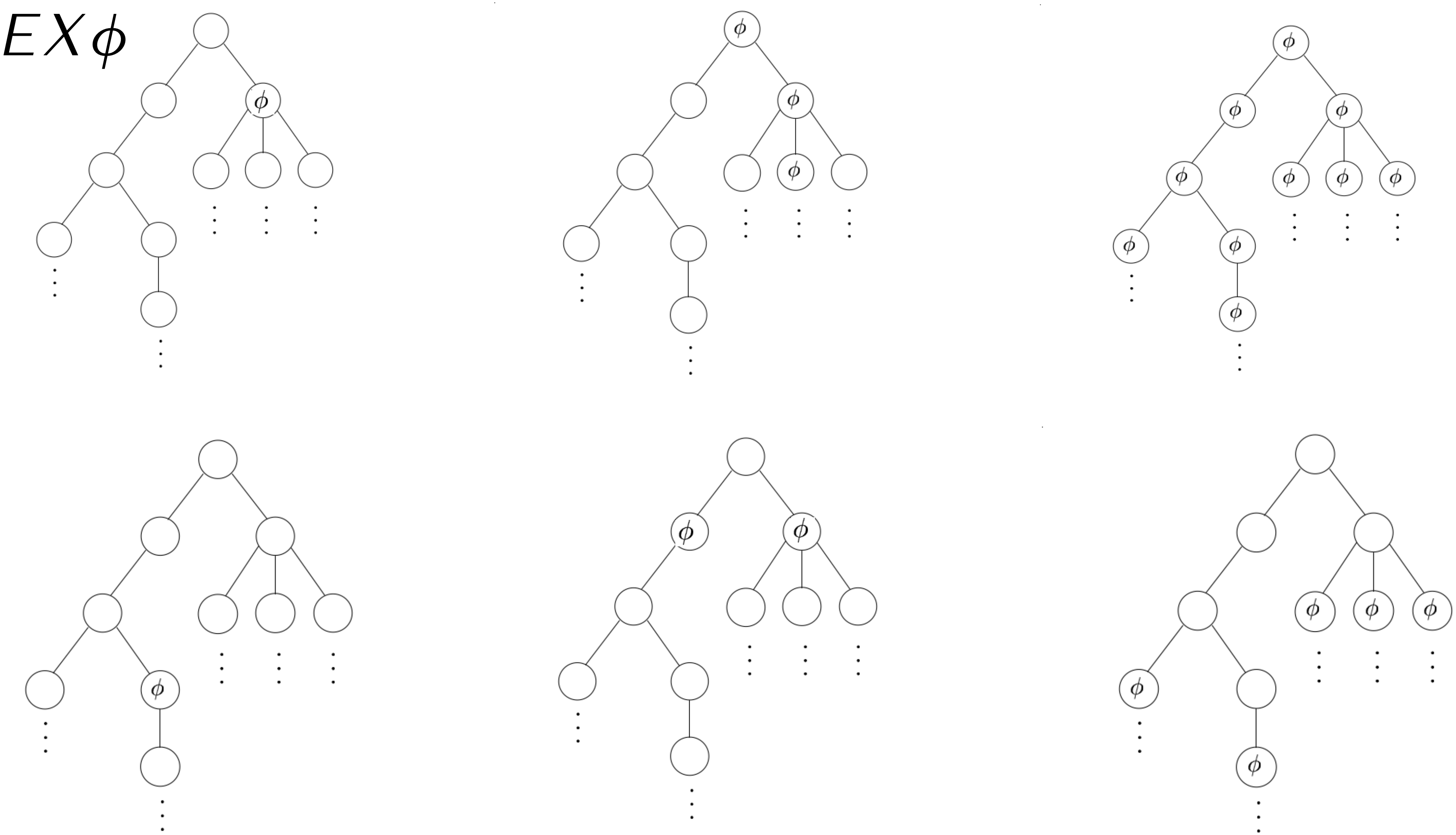
We would like to say something about the difference in expressiveness between CTL and LTL. Let's put that on hold for the moment and get a better grip on CTL first.

CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$



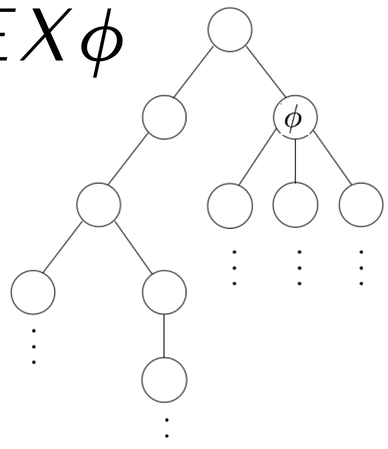
CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$

$EX\phi$

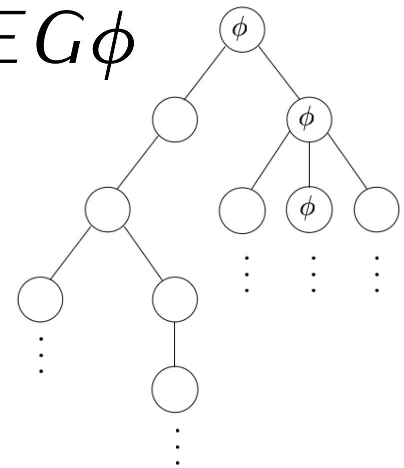


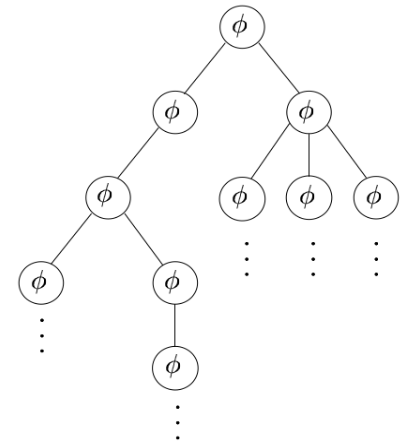
CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$

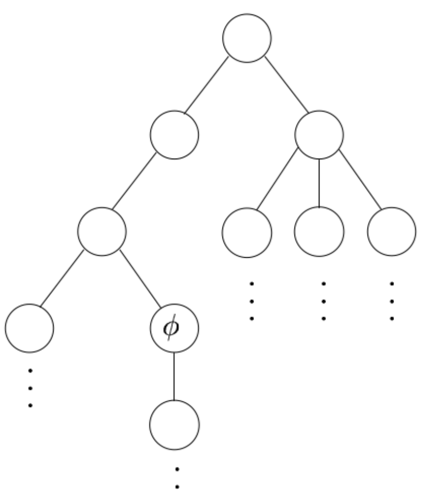
$EX\phi$

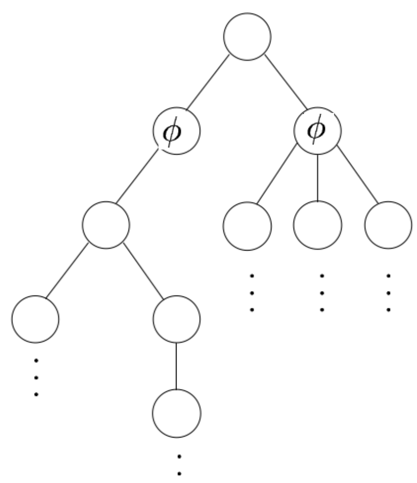


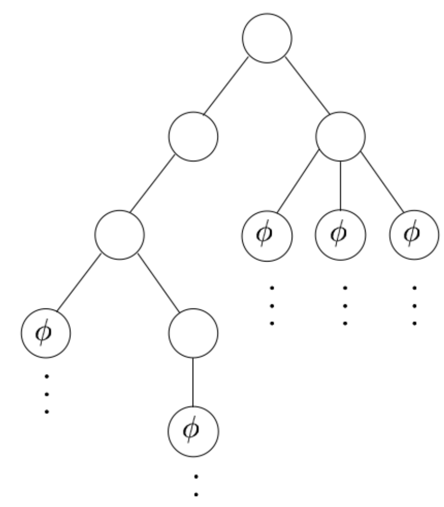
$EG\phi$





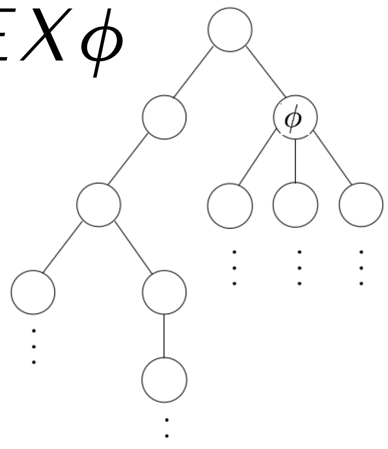




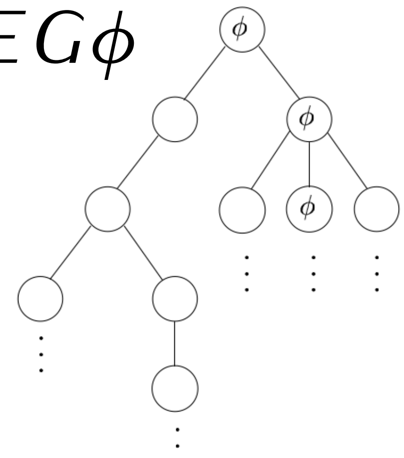


CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$

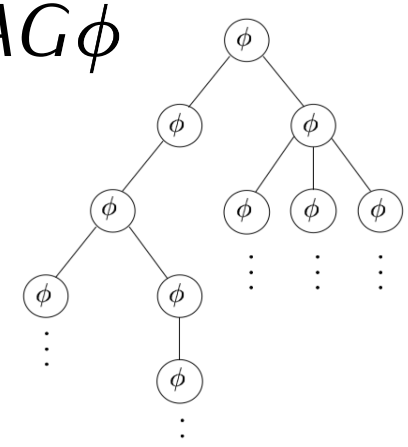
$EX\phi$

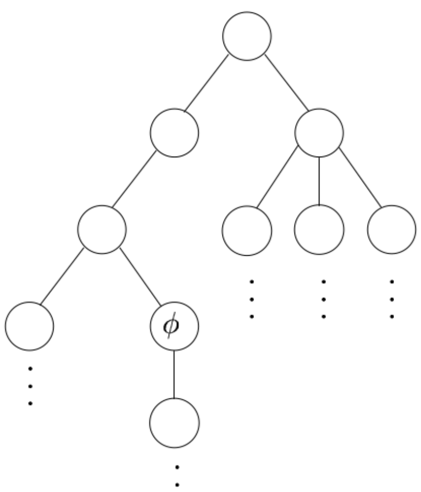


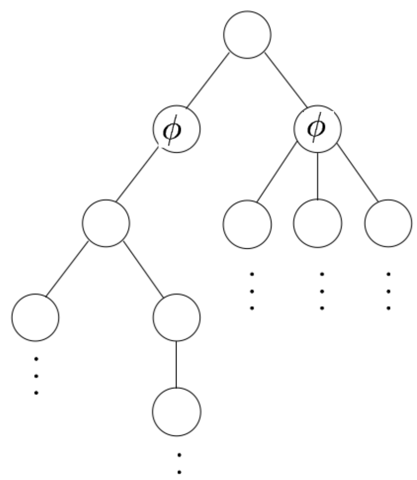
$EG\phi$

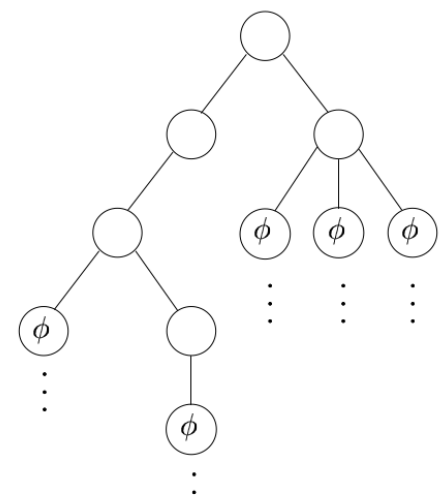


$AG\phi$



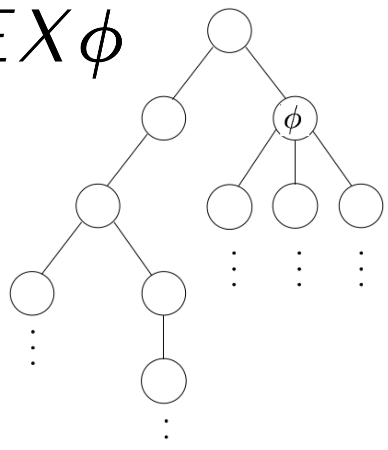




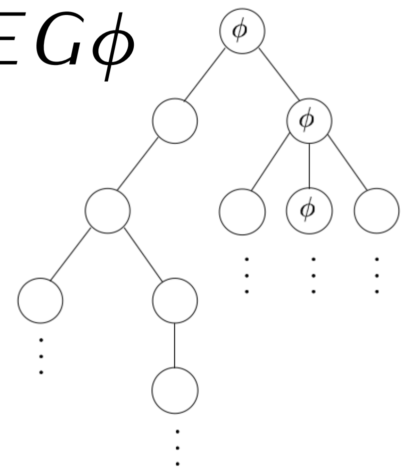


CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$

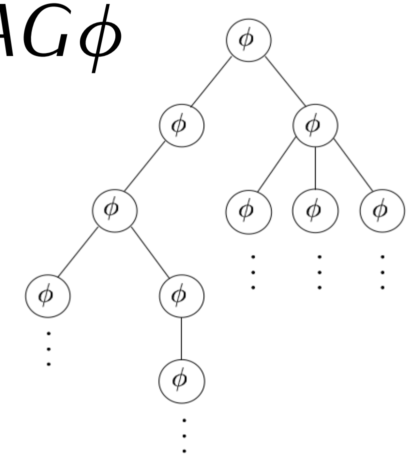
$EX\phi$



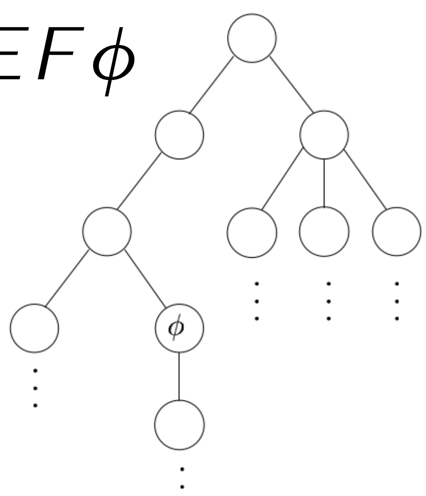
$EG\phi$

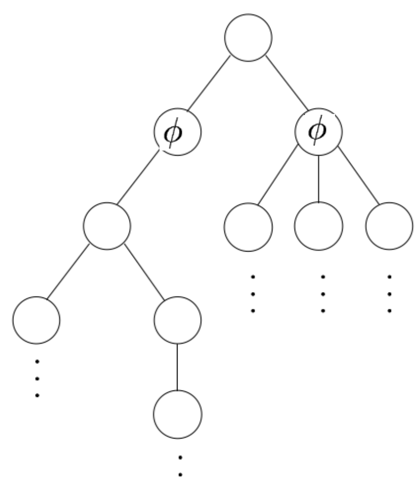


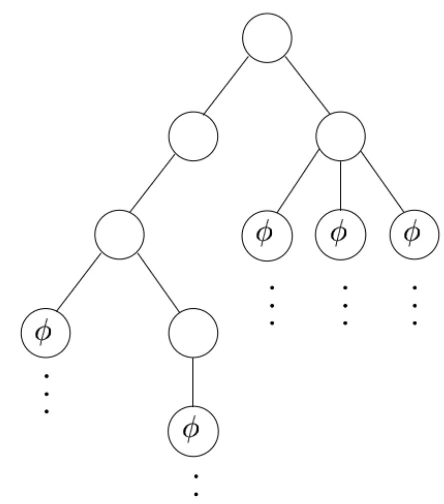
$AG\phi$



$EF\phi$

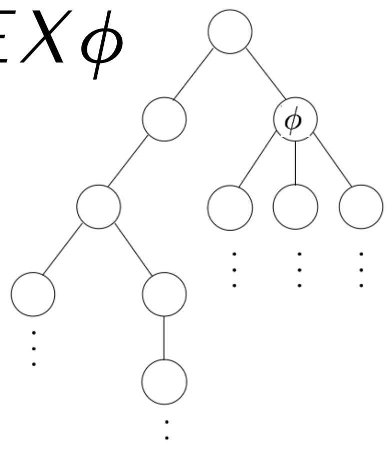




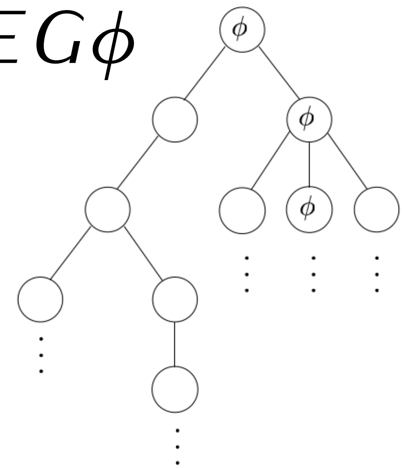


CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$

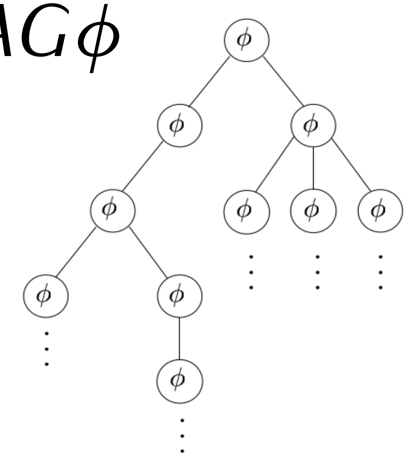
$EX\phi$



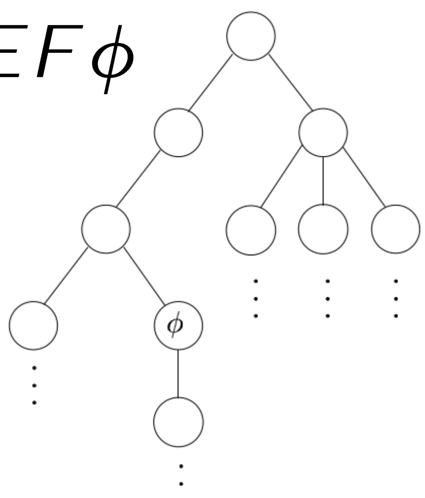
$EG\phi$



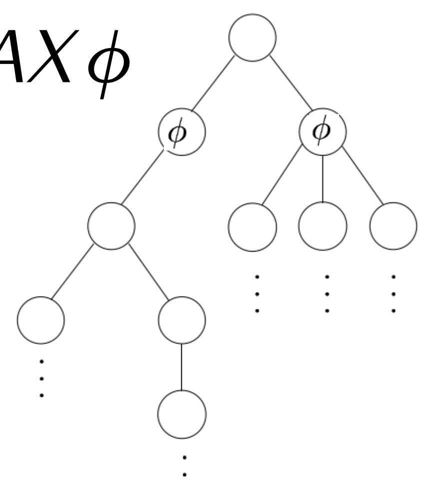
$AG\phi$

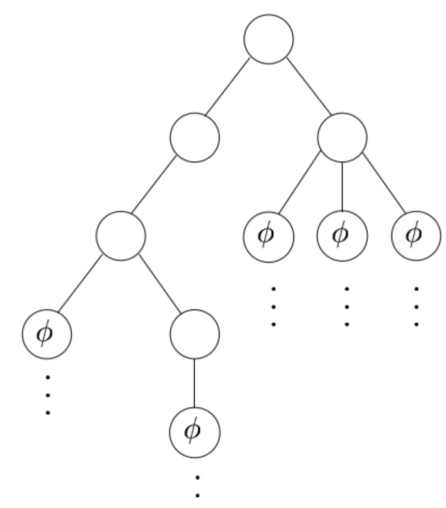


$EF\phi$



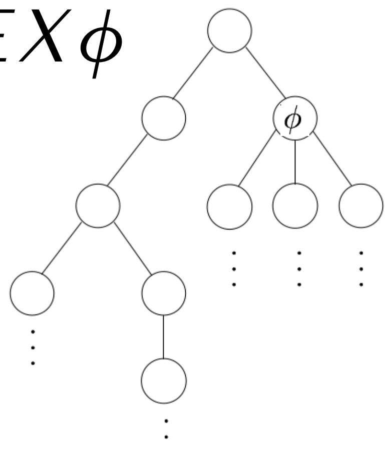
$AX\phi$



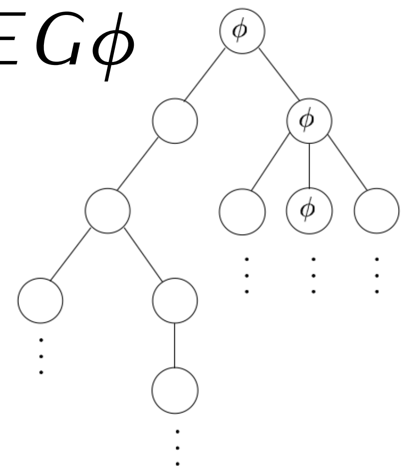


CTL review $AG\phi$ $EG\phi$ $AF\phi$ $EF\phi$ $AX\phi$ $EX\phi$

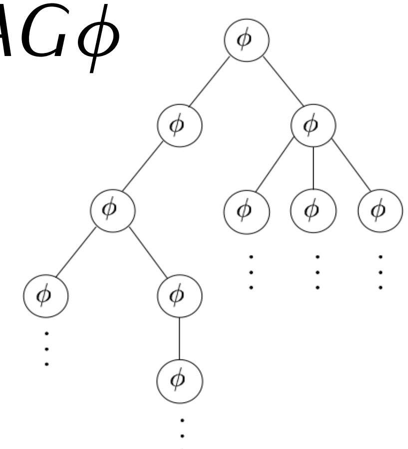
$EX\phi$



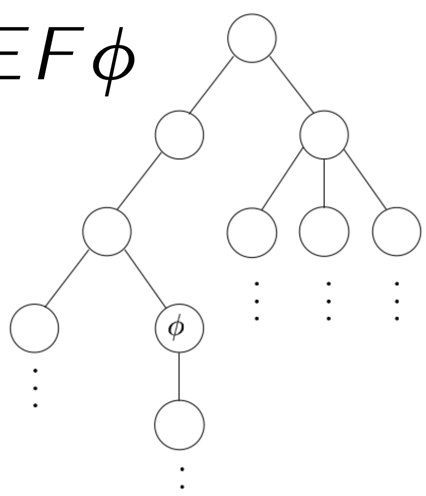
$EG\phi$



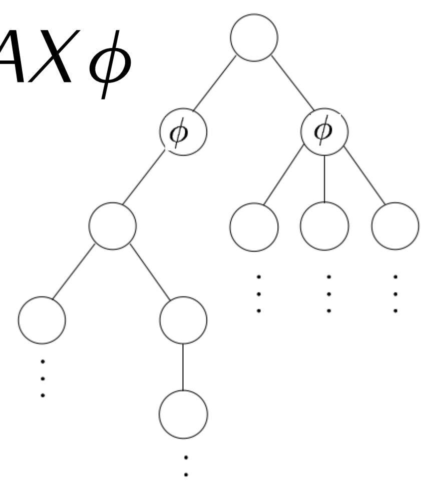
$AG\phi$



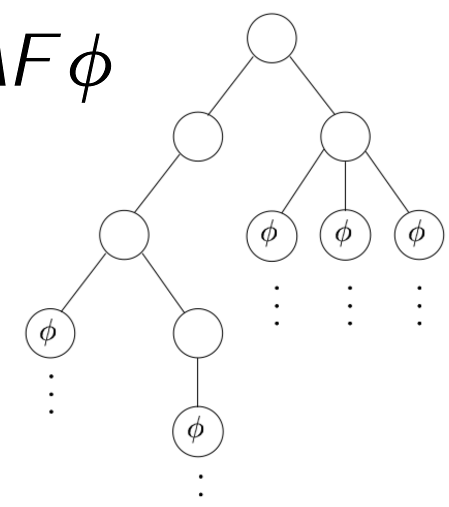
$EF\phi$



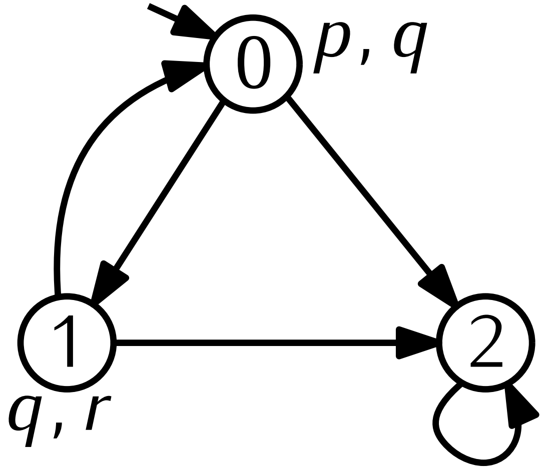
$AX\phi$



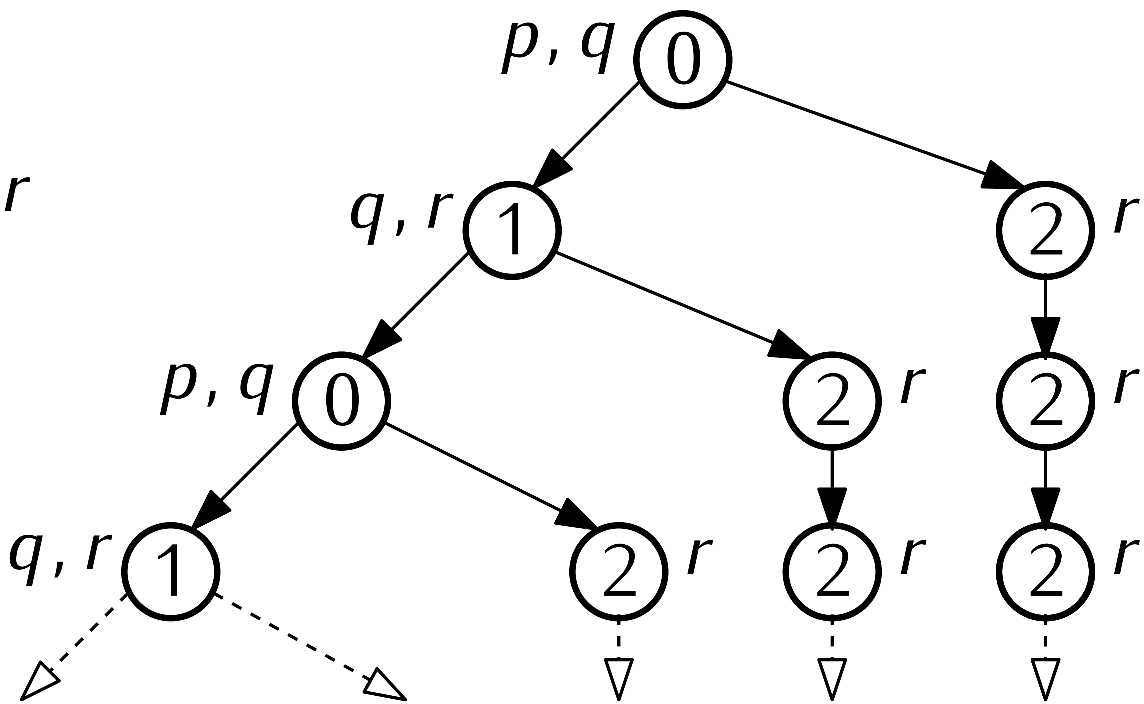
$AF\phi$



Computation Tree Logic Example Properties



Computation tree for $\mathcal{M}$



$\mathcal{M} \models p \land q$?

$\mathcal{M} \models \neg r$?

$\mathcal{M} \models EX(q \land r)$?

$\mathcal{M} \models AX(q \land r)$?

$\mathcal{M} \models \neg AX(q \land r)$?

$\mathcal{M} \models \neg EF(p \land r)$?

$\mathcal{M} \models EG\neg r$?

$\mathcal{M} \models AFq$?

$\mathcal{M} \models p\ AU\ r$?

$\mathcal{M} \models \neg(p \land q)\ EU\ r$?

Nested CTL operators

Consider the following sentence:

> No matter what, at some point the transition system reaches a state where there exists a path such that from that point on, p holds forever.

Nested CTL operators

Consider the following sentence:

> $\boxed{\text{No matter what,}}$ at some point the transition system reaches a state where there exists a path such that from that point on, *p* holds forever.

Inevitably

Nested CTL operators

Consider the following sentence:

> No matter what, at some point the transition system reaches a state where there exists a path such that from that point on, p holds forever.

Inevitably Eventually

Nested CTL operators

Consider the following sentence:

> No matter what, at some point the transition system reaches a state where there exists a path such that from that point on, p holds forever.

Inevitably Eventually Possibly

Nested CTL operators

Consider the following sentence:

> No matter what, at some point the transition system reaches a state where there exists a path such that $\boxed{\text{from that point on, } p \text{ holds forever.}}$

Inevitably Eventually Possibly Always p

Nested CTL operators

Consider the following sentence:

> No matter what, at some point the transition system reaches a state where there exists a path such that $\boxed{\text{from that point on, } p \text{ holds forever.}}$

Inevitably Eventually Possibly Always p

$$AF\ EG\ p$$

Nested CTL operators

Consider the following sentence:

> No matter what, at some point the transition system reaches a state where there exists a path such that from that point on, p holds forever.

Inevitably Eventually Possibly Always p

$$AF\ EG\ p$$

A typical HW problem: translate a few sentences into CTL.

A basis for CTL

The CTL operators AG, AF, AX, EF, and EG can all be written using only negations ($\neg$), and EX, EU, and AU.

A basis for CTL

The CTL operators AG, AF, AX, EF, and EG can all be written using only negations ($\neg$), and EX, EU, and AU.

$$AX\phi \equiv \text{ ????}$$

A basis for CTL

The CTL operators AG, AF, AX, EF, and EG can all be written using only negations ($\neg$), and EX, EU, and AU.

$$AX\phi \equiv \ ????$$

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

A basis for CTL

The CTL operators AG, AF, AX, EF, and EG can all be written using only negations ($\neg$), and EX, EU, and AU.

$$AX\phi \equiv \ ????$$

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

A typical HW problem: show that CTL operators can be written using only $\{\neg, EX, EU, AU\}$.

Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Another interesting property of CTL formulas

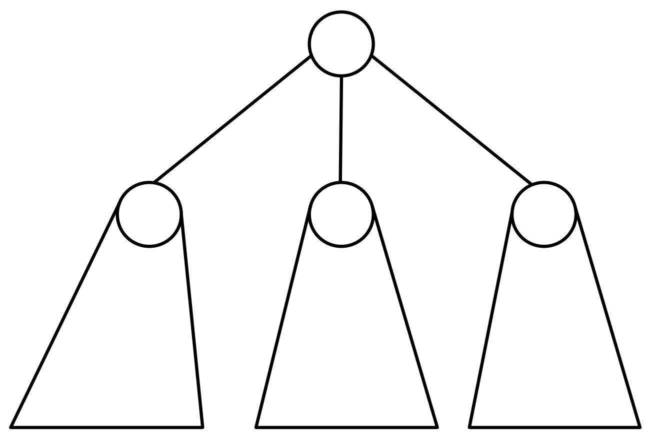
Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

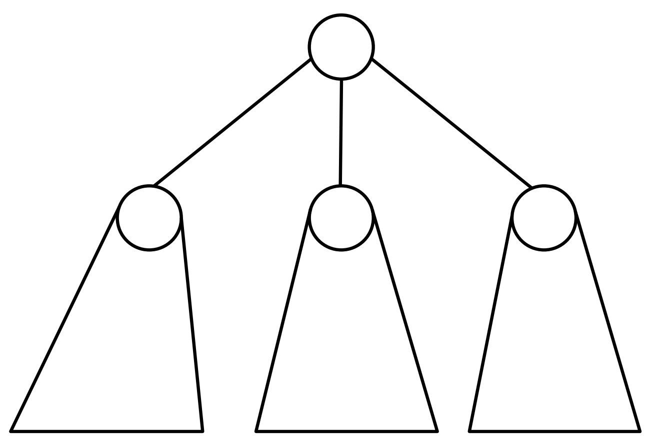


Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

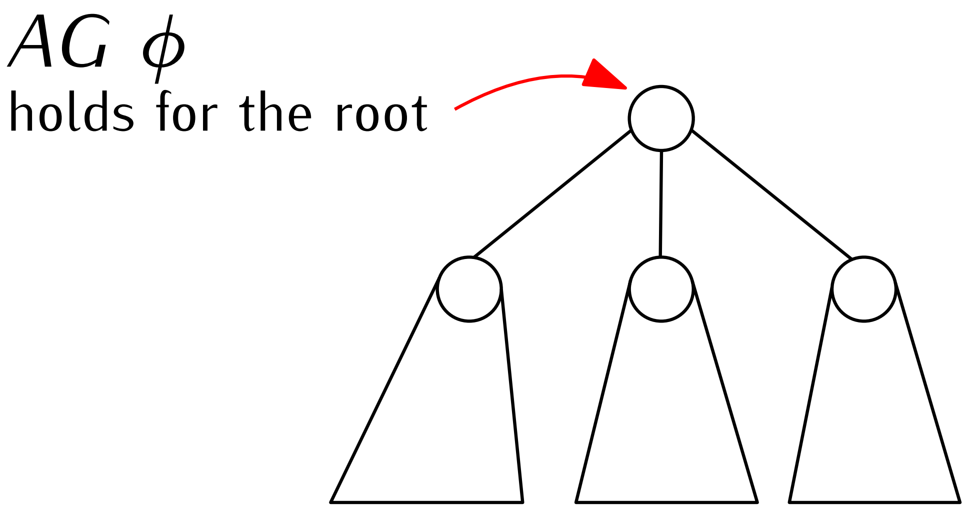
$AG\ \phi$



Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

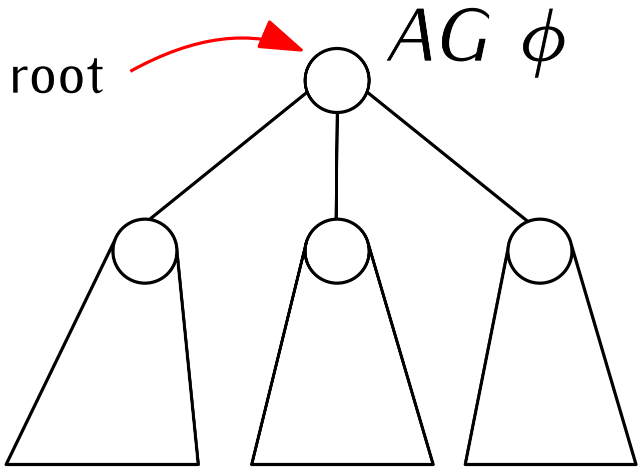


Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

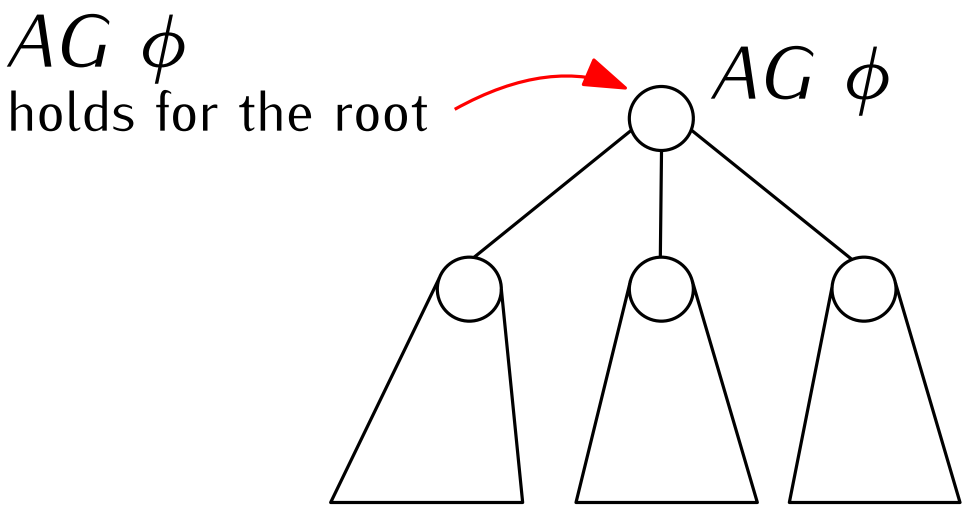
$AG\ \phi$
holds for the root



Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

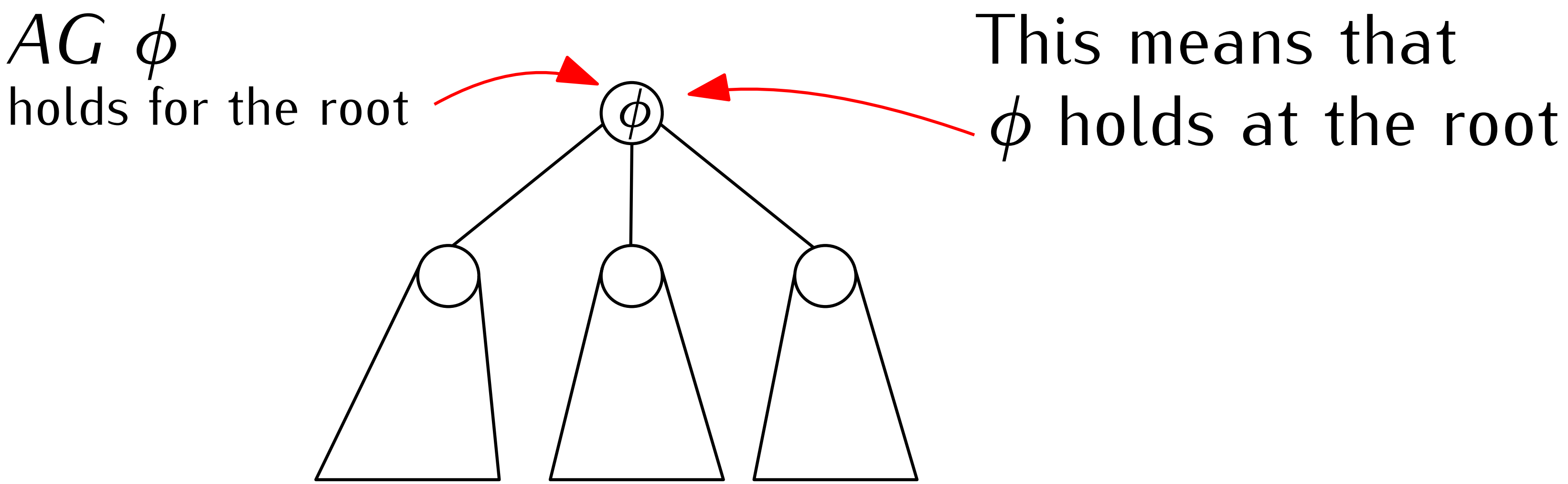


This means that

Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

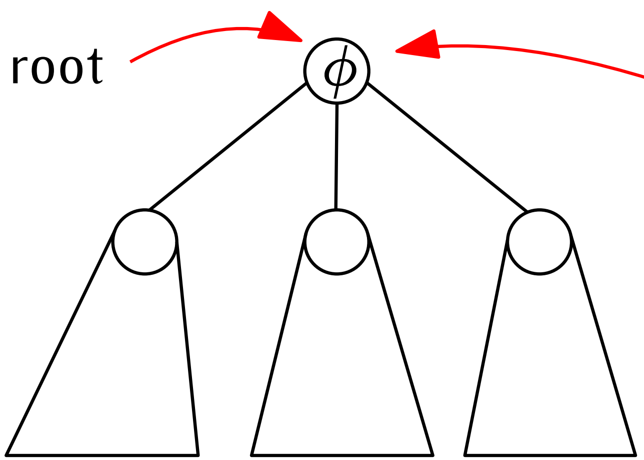


Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

$AG\ \phi$
holds for the root



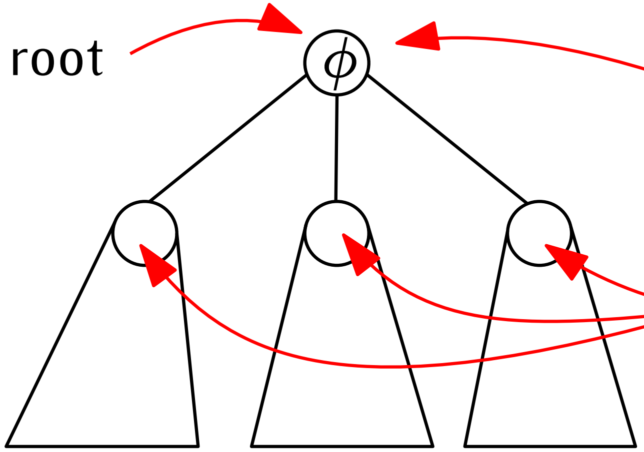
This means that
ϕ holds at the root
and

Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

$AG\ \phi$
holds for the root



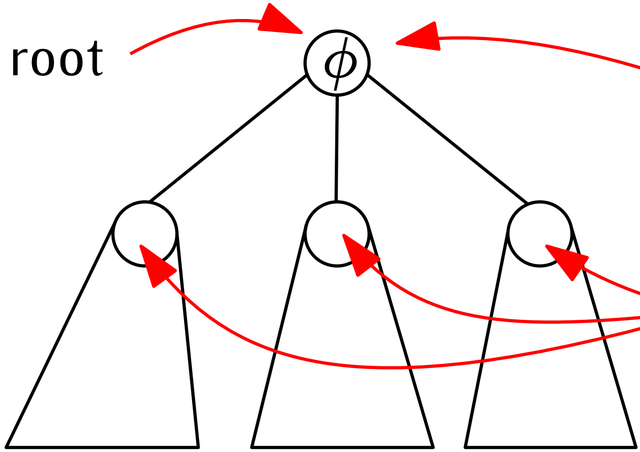
This means that
ϕ holds at the root
and
for all next states
$AG\phi$ holds

Another interesting property of CTL formulas

Any CTL operator $\oplus$ can be written using only $\oplus$, AX, EX, and Boolean connectives.

Example: Operator AG can be written using only AG, AX, EX, and Boolean connectives.

$AG\ \phi$
holds for the root



This means that
ϕ holds at the root
and
for all next states
$AG\phi$ holds

$$AG\ \phi \equiv \phi \wedge (AX\ AG\ \phi)$$

A typical HW problem: Do the same for EG, EF, AF, EU, AU.

Equivalence of properties

Given two temporal logic formulas α and β, when can we say that the two formulas are equivalent?

Equivalence of properties

Given two temporal logic formulas α and β, when can we say that the two formulas are equivalent?

We say that $\alpha \equiv \beta$ iff

Equivalence of properties

Given two temporal logic formulas α and β, when can we say that the two formulas are equivalent?

We say that $\alpha \equiv \beta$ iff

$$\forall \mathcal{M} \quad (\mathcal{M} \models \alpha \Leftrightarrow \mathcal{M} \models \beta)$$

Equivalence of properties

Given two temporal logic formulas α and β, when can we say that the two formulas are equivalent?

We say that $\alpha \equiv \beta$ iff

$$\forall \mathcal{M} \quad (\mathcal{M} \models \alpha \Leftrightarrow \mathcal{M} \models \beta)$$

We say that $\alpha \not\equiv \beta$ iff

Equivalence of properties

Given two temporal logic formulas α and β, when can we say that the two formulas are equivalent?

We say that $\alpha \equiv \beta$ iff

$$\forall \mathcal{M} \quad (\mathcal{M} \models \alpha \Leftrightarrow \mathcal{M} \models \beta)$$

We say that $\alpha \not\equiv \beta$ iff

$$\exists \mathcal{M} \quad ((\mathcal{M} \models \alpha \wedge \mathcal{M} \not\models \beta) \vee (\mathcal{M} \not\models \alpha \wedge \mathcal{M} \models \beta))$$

Equivalence of properties

Given two temporal logic formulas α and β, when can we say that the two formulas are equivalent?

We say that $\alpha \equiv \beta$ iff

$$\forall \mathcal{M} \quad (\mathcal{M} \models \alpha \Leftrightarrow \mathcal{M} \models \beta)$$

We say that $\alpha \not\equiv \beta$ iff

$$\exists \mathcal{M} \quad ((\mathcal{M} \models \alpha \wedge \mathcal{M} \not\models \beta) \vee (\mathcal{M} \not\models \alpha \wedge \mathcal{M} \models \beta))$$

In words, two formulas are not equivalent if we can find a transition system that satisifes one formula but not the other.

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

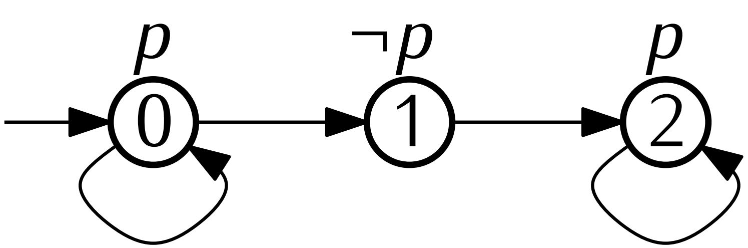
$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:

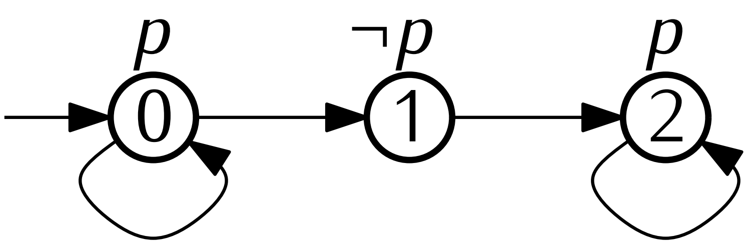


Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Paths of $\mathcal{M}$ look like:

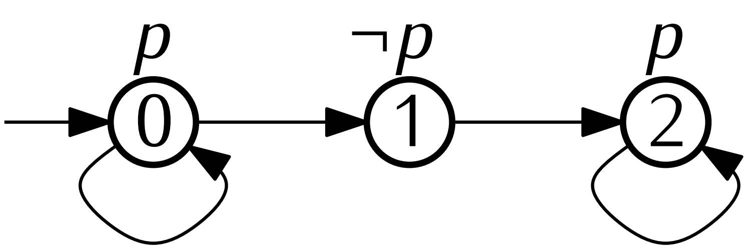
0^ω or $0^*1\ 2^\omega$

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

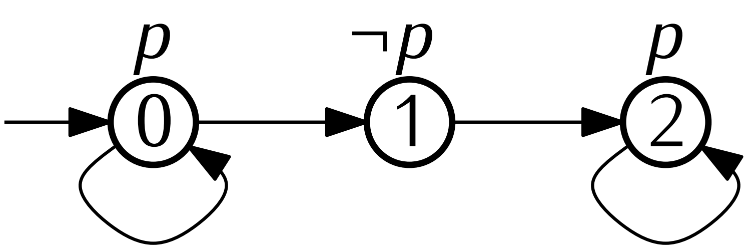
$p, p, p, \ldots, \neg p, p, p, p, \ldots$

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Paths of $\mathcal{M}$ look like:

0^ω $\quad$ or $\quad$ $0^* 1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

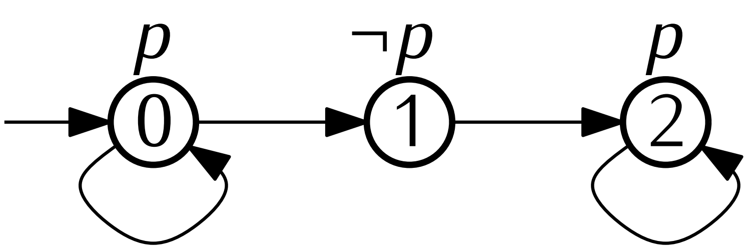
$$\mathcal{M} \models\ F\ G\ p$$

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:

Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

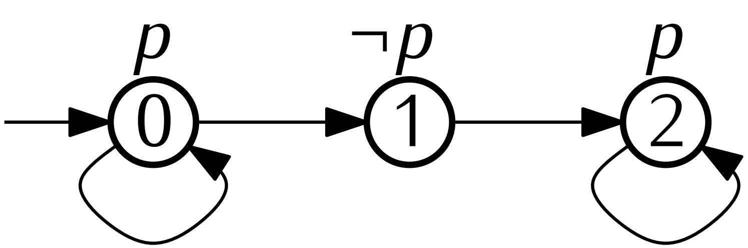
$$\mathcal{M} \models F\ G\ p$$

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:

Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

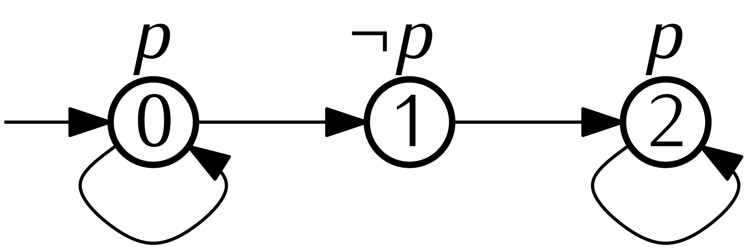
$$\mathcal{M} \models\ F\ G\ p$$

Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F\ G\ p$ $\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Paths of $\mathcal{M}$ look like:

$0^\omega \quad$ or $\quad 0^*1\ 2^\omega$

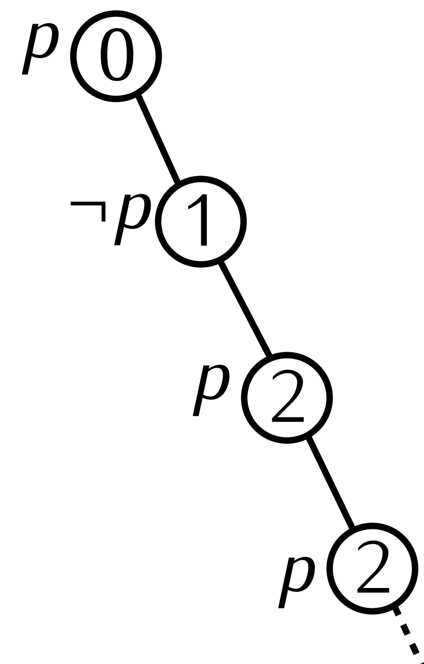
Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

$\mathcal{M} \models\ F\ G\ p$

Computation tree:

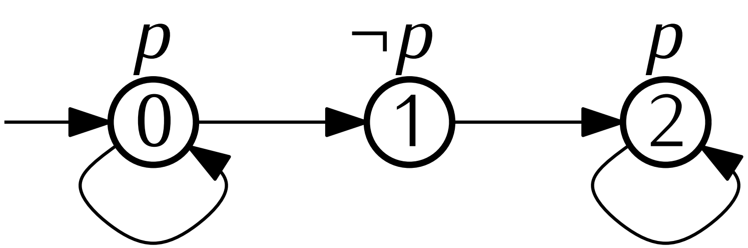


Showing that $\alpha \not\equiv \beta$

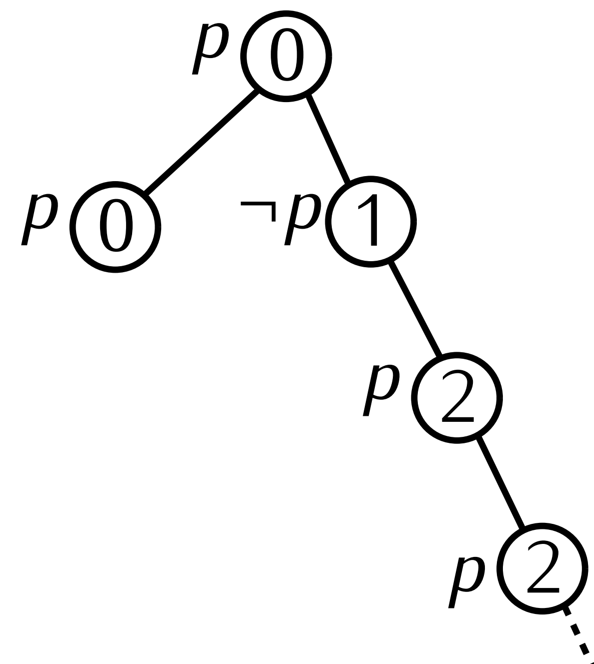
Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:



Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

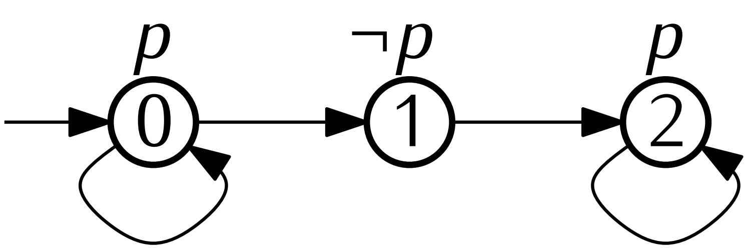
$$\mathcal{M} \models F\ G\ p$$

Showing that $\alpha \not\equiv \beta$

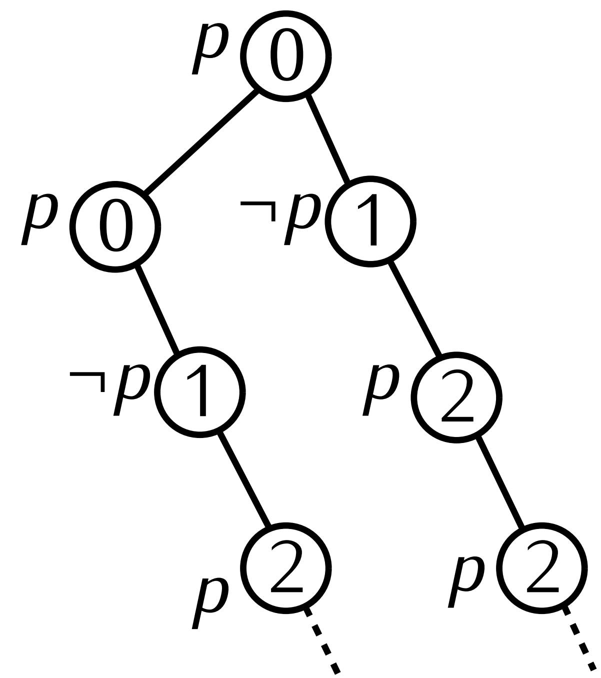
Consider these two temporal formulas

$F\ G\ p$ $\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:



Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

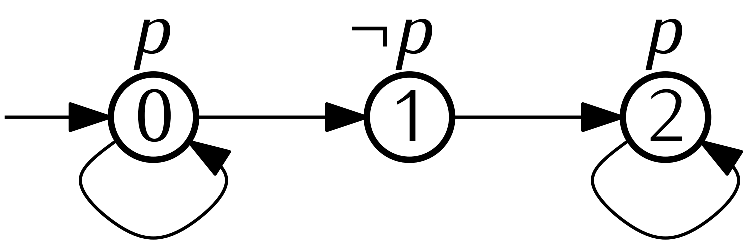
$$\mathcal{M} \models\ F\ G\ p$$

Showing that $\alpha \not\equiv \beta$

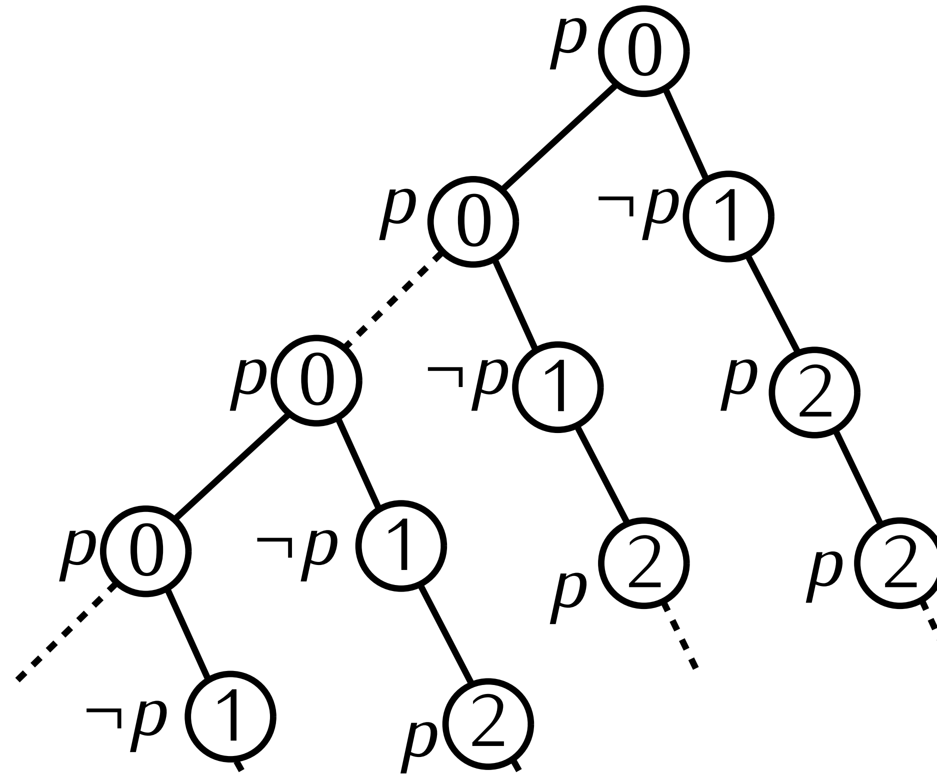
Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:



Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

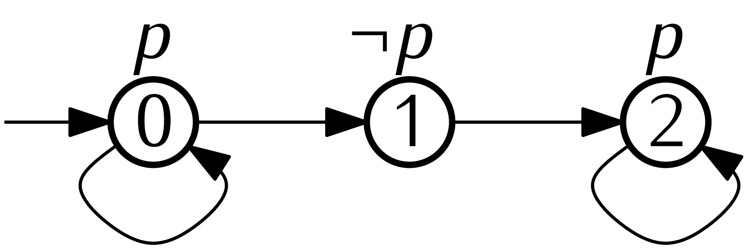
$\mathcal{M} \models\ F\ G\ p$

Showing that $\alpha \not\equiv \beta$

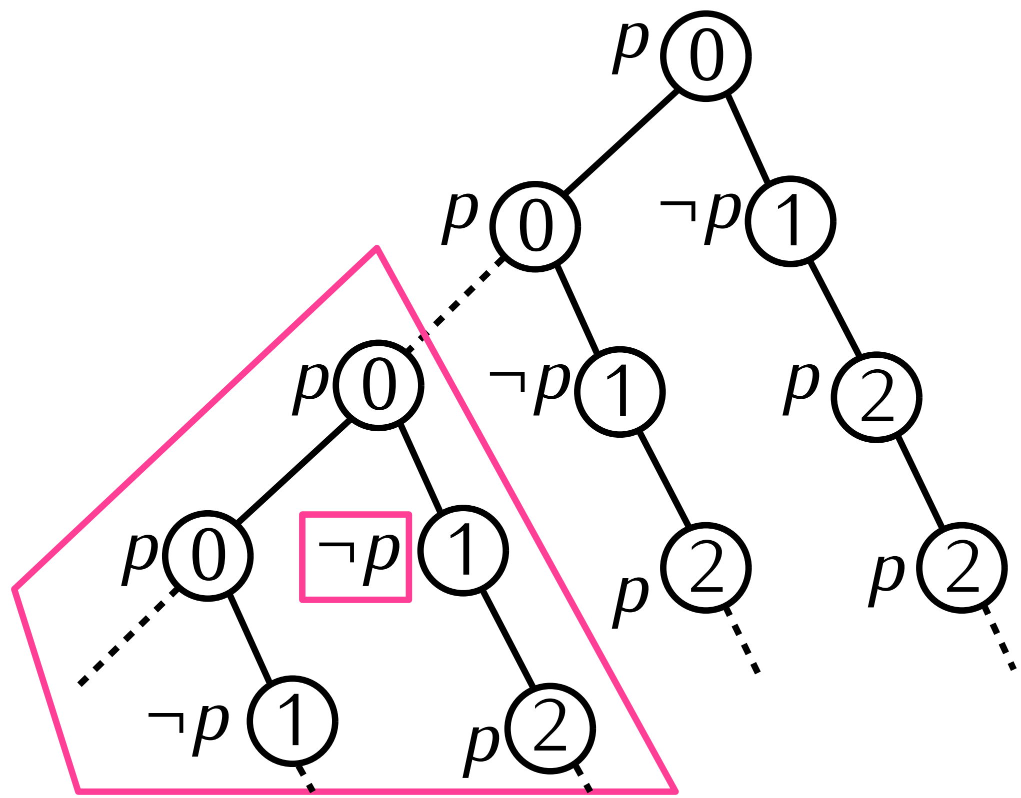
Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:



Paths of $\mathcal{M}$ look like:

0^ω or $0^* 1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

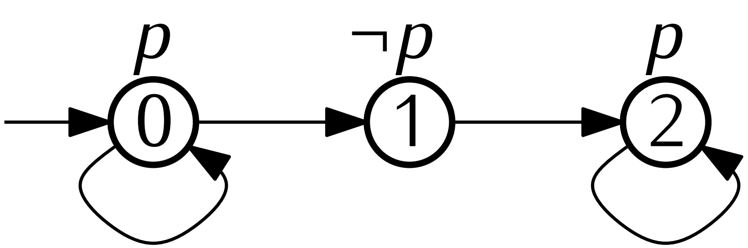
$\mathcal{M} \models\ F\ G\ p$

Showing that $\alpha \not\equiv \beta$

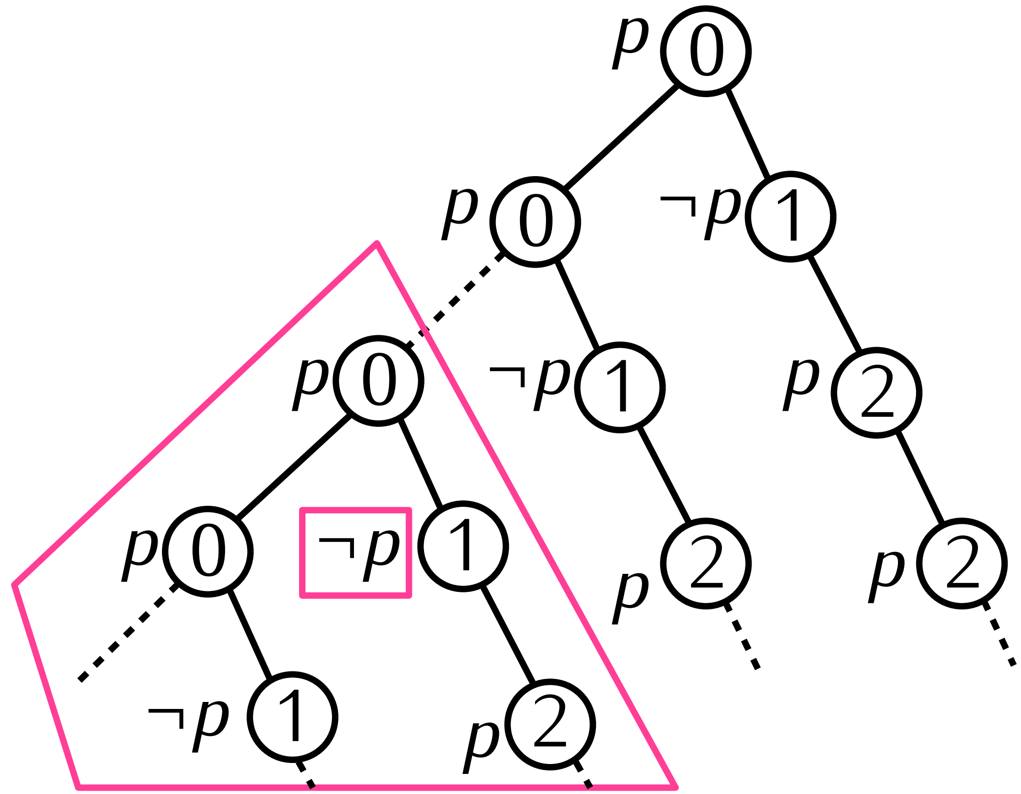
Consider these two temporal formulas

$F\ G\ p$ $\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:



Paths of $\mathcal{M}$ look like:

0^ω or $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

$\mathcal{M} \models F\ G\ p$

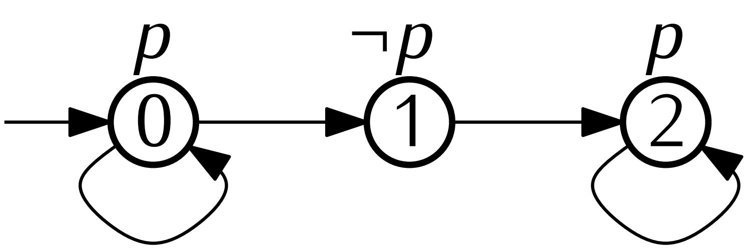
$\mathcal{M} \not\models AF\ AG\ p$

Showing that $\alpha \not\equiv \beta$

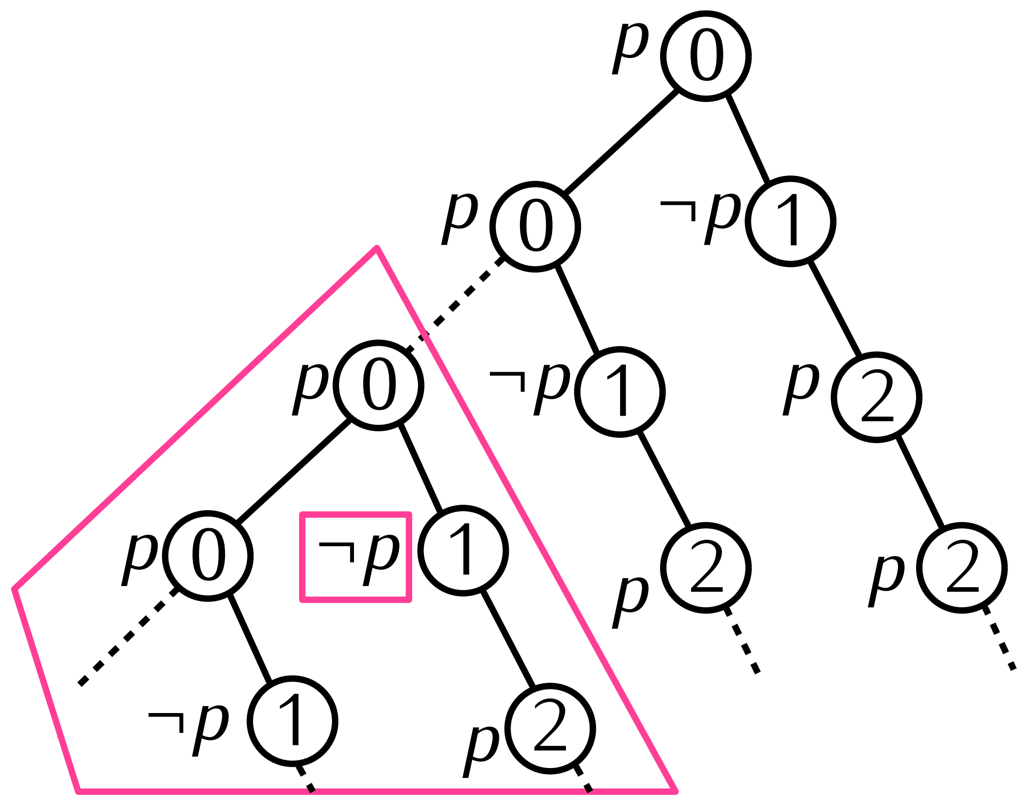
Consider these two temporal formulas

$F\ G\ p$ $\qquad\qquad$ $AF\ AG\ p$

Consider this transition system, $\mathcal{M}$:



Computation tree:



Paths of $\mathcal{M}$ look like:

0^ω $\quad$ or $\quad$ $0^*1\ 2^\omega$

Sequences of propositions:

$p, p, p, p, p, \ldots$

$p, p, p, \ldots, \neg p, p, p, p, \ldots$

$\mathcal{M} \models\ F\ G\ p$ $\qquad\qquad$ $\mathcal{M} \not\models\ AF\ AG\ p$

Typical HW problem: Show two formulas not equivalent.