Today’s class

Linear Temporal Logic
   W and R operators. LTL operator basis.

Computation Tree Logic
   Some practice, nested CTL formulas

LTL vs CTL
   Equivalence of two temporal formulas
Reactive System Code satisfies \( \models \) Requirements

Transition System satisfies \( \models \) Temporal Logic Formula \( \phi \)

Model Checking
LTL $W$ and $R$ operators

For a path $\pi$ of transition system $\mathcal{M}$,
LTL $W$ and $R$ operators

For a path $\pi$ of transition system $M$, $\pi \models \phi W \psi$ iff $(\exists i \geq 1 \, \pi^i \models \psi \land \forall 1 \leq j < i \, \pi^j \models \phi)$ $\lor \forall k \geq 1 \, \pi^k \models \phi$
LTL \( W \) and \( R \) operators

For a path \( \pi \) of transition system \( M \),

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

Weak Until

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

Release
LTL $W$ and $R$ operators

For a path $\pi$ of transition system $M$, 

$\pi \models \phi W \psi$ iff $(\exists i \geq 1 \; \pi^i \models \psi \land \forall 1 \leq j < i \; \pi^j \models \phi)$ 

$\lor \forall k \geq 1 \; \pi^k \models \phi$

Weak Until

or

$\pi \models \phi R \psi$ iff $(\exists i \geq 1 \; \pi^i \models \phi \land \forall 1 \leq j \leq i \; \pi^j \models \psi)$ 

$\lor \forall k \geq 1 \; \pi^k \models \psi$

Release
LTL $W$ and $R$ operators

For a path $\pi$ of transition system $M$, 

$\pi \models \phi W \psi$ iff $(\exists i \geq 1 \ \pi^i \models \psi \land \forall 1 \leq j < i \ \pi^j \models \phi) \lor \forall k \geq 1 \ \pi^k \models \phi$

Weak Until

or

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

Release
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.

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 $\nu$SMV specifications.

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

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 $\nu$SMV specifications.

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

On the other hand, when performing meta-analysis of LTL, we need to examine each operator.
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 \( \nu \)SMV 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.

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 \( \nu \)SMV 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.

\[
\{\land, \neg\} \quad \{\lambda, (f \ e)\} \quad \{S, K, I\}
\]

Propositional logic \quad Lambda calc \quad 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 $\mathcal{G}, \mathcal{F}, \mathcal{X}, \mathcal{U}, \mathcal{W}, \mathcal{R}$

$F\phi = T \ U \ \phi$

$G\phi = \bot \ R \ \phi$
An operator basis for LTL

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

$F\phi = T U \phi$

$G\phi = \bot R \phi$

$\phi W \psi = \phi U \psi \lor G\phi = \phi U \psi \lor \bot R \phi$
An operator basis for LTL

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

$F \phi = T \ U \ \phi$

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

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

$\phi \ R \ \psi = \neg (\neg \phi \ U \ \neg \psi)$
An operator basis for LTL

Let’s get rid of a bunch of operators \( G, F, X, U, W, R \)

\[
F \phi = T U \phi \\
G \phi = \bot R \phi \\
\phi W \psi = \phi U \psi \lor G \phi = \phi U \psi \lor \bot R \phi \\
\phi R \psi = \neg (\neg \phi U \neg \psi)
\]

\( X \) is special. Cannot be written in terms of the others.
An operator basis for LTL

Let’s get rid of a bunch of operators

\[ F\phi = T \land U \phi \]

\[ G\phi = \bot \land R \phi \]

\[ \phi W \psi = \phi U \psi \lor G\phi = \phi U \psi \lor \bot \land R \phi \]

\[ \phi R \psi = \neg (\neg \phi U \neg \psi) \]

\( X \) 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 \( \{G, F, X, U, W, R\} \)

\[ F\phi = T \quad U \phi \]
\[ G\phi = \bot \quad R \phi \]
\[ \phi \ W \psi = \phi \ U \psi \lor \quad G\phi = \phi \ U \psi \lor \bot \quad R \phi \]
\[ \phi \ R \psi = \neg (\neg \phi \ U \neg \psi) \]

\( X \) 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.
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

\[ \text{AG} \phi \quad \text{EG} \phi \quad \text{AF} \phi \quad \text{EF} \phi \quad \text{AX} \phi \quad \text{EX} \phi \]

\[ \text{EX} \phi \]
CTL review

\[ \text{EX} \phi \]

\[ \text{EG} \phi \]

\[ \text{EF} \phi \]

\[ \text{AF} \phi \]

\[ \text{AX} \phi \]

\[ \text{EX} \phi \]
CTL review

\(AX\phi\) \hspace{1cm} \(EG\phi\) \hspace{1cm} \(AF\phi\) \hspace{1cm} \(EF\phi\) \hspace{1cm} \(AX\phi\) \hspace{1cm} \(EX\phi\)
CTL review

$EX\phi$

$EG\phi$

$AG\phi$

$EF\phi$

$AX\phi$

$EX\phi$
CTL review

\[ EX\phi \quad EG\phi \quad AF\phi \quad EF\phi \quad AX\phi \quad EX\phi \]
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

\[ M \models p \land q ? \]
\[ M \models \neg r ? \]
\[ M \models \text{EX}(q \land r) ? \]
\[ M \models \text{AX}(q \land r) ? \]
\[ M \models \neg \text{AX}(q \land r) ? \]
\[ M \models \neg \text{EF}(p \land r) ? \]
\[ M \models p \cup r ? \]
\[ M \models \neg (p \land q) \cup r ? \]
\[ M \models \text{EG} \neg r ? \]
\[ M \models \text{AF} q ? \]
\[ M \models p \cup r ? \]
\[ M \models \neg (p \land q) \cup r ? \]
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.
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
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
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 from that point on, $p$ holds forever.

Inevitably Eventually Possibly Always $p$
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 \( p \)

\[ AF \ EG \ p \]
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.
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 $AU$.

$$AX\phi \equiv \ ????$$
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 $\Theta$ can be written using only $\Theta, AX, EX$, 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.

$AG \phi$

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.

$AG \phi$
holds for the root

$AG \phi$

$\oplus$, $AX$, $EX$, 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.

$AG \phi$ holds for the root

This means that
Another interesting property of CTL formulas

Any CTL operator $\bigoplus$ can be written using only
$\bigoplus$, $AX$, $EX$, 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

$\phi$ holds at 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.

$AG \, \phi$
holds for the root

This means that $\phi$ 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 $\phi$ 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.

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

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

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

Given two temporal logic formulas $\alpha$ and $\beta$, when can we say that the two formulas are equivalent?
Equivalence of properties

Given two temporal logic formulas $\alpha$ and $\beta$, 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 $\alpha$ and $\beta$, when can we say that the two formulas are equivalent?

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

$$\forall M \ (M \models \alpha \iff M \models \beta)$$
Equivalence of properties

Given two temporal logic formulas $\alpha$ and $\beta$, when can we say that the two formulas are equivalent?

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

$$\forall M \quad (M \models \alpha \iff M \models \beta)$$

We say that $\alpha \not\equiv \beta$ iff
Equivalence of properties

Given two temporal logic formulas $\alpha$ and $\beta$, when can we say that the two formulas are equivalent?

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

$$\forall M \ (M \models \alpha \iff M \models \beta)$$

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

$$\exists M \ ((M \models \alpha \land M \not\models \beta) \lor (M \not\models \alpha \land M \models \beta))$$
Equivalence of properties

Given two temporal logic formulas $\alpha$ and $\beta$, when can we say that the two formulas are equivalent?

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

$$\forall M \ (M \models \alpha \iff M \models \beta)$$

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

$$\exists M \ ((M \models \alpha \land M \not\models \beta) \lor (M \not\models \alpha \land M \models \beta))$$

In words, two formulas are not equivalent if we can find a transition system that satisfies one formula but not the other.
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

\[ F \; G \; p \quad \quad \quad AF \; AG \; p \]
Showing that \( \alpha \neq \beta \)

Consider these two temporal formulas
\[
F G p \quad AF \ AG \ p
\]

Consider this transition system, \( \mathcal{M} \):

\[
\begin{array}{ccc}
p & \neg p & p \\
0 & 1 & 2
\end{array}
\]
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F G p \quad AF AG p$

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

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

$0^\omega$ or $0^*1 \ 2^\omega$
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F G p$ $AF AG p$

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

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

$0^\omega$ 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

$F G p$ \hspace{1cm} $AF AG p$

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

![Transition System Diagram]

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

$0^\omega$ or $0^*12^\omega$

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

\[ F \, G \, p \quad \text{and} \quad AF \, AG \, p \]

Consider this transition system, $M$:

Paths of $M$ look like:

\[ 0^\omega \quad \text{or} \quad 0^*1 \, 2^\omega \]

Sequences of propositions:

\[ p, p, p, p, p, \ldots \]
\[ p, p, p, \ldots, \neg p, p, p, p, \ldots \]

$M \models F \, G \, p$
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

\[ F \ G \ p \]

\[ AF \ AG \ p \]

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

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

\[ 0^\omega \quad \text{or} \quad 0^*1 \ 2^\omega \]

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

\[
F G p
\]

\[
AF AG p
\]

Consider this transition system, \( \mathcal{M} \):

```
0 \rightarrow p \rightarrow 1 \rightarrow \neg p \rightarrow 2 \rightarrow p
```

Paths of \( \mathcal{M} \) look like:

\[0^\omega \text{ or } 0^*12^\omega\]

Sequences of propositions:

\[p, p, p, p, p, p, \ldots\]

\[p, p, p, \ldots, \neg p, p, p, p, \ldots\]

\( \mathcal{M} \models F G p \)
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F \ G \ p$

$AF \ AG \ p$

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

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

$0^\omega$ 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

$$F \; G \; p$$

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

![Transition System Diagram]

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

$$0^\omega \text{ or } 0^*1\; 2^\omega$$

Sequences of propositions:

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

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

$\mathcal{M} \models F \; G \; p$
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas:

- $F G p$
- $AF AG p$

Consider this transition system, $M$:

Paths of $M$ look like:

- $0^\omega$ or $0^*12^\omega$

Sequences of propositions:

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

$M \models F G p$

Computation tree:
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas:

$F \, G \, p$

$AF \, AG \, p$

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

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

$0^\omega$ 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

\[ F \ G \ p \]

Consider this transition system, \( \mathcal{M} \):

Paths of \( \mathcal{M} \) look like:

\[ 0^\omega \quad \text{or} \quad 0^*1 \ 2^\omega \]

Sequences of propositions:

\[ p, p, p, p, p, \ldots \]
\[ p, p, p, \ldots, \neg p, p, p, p, \ldots \]

\[ \mathcal{M} \models F \ G \ p \]

\[ \mathcal{M} \not\models AF \ AG \ p \]
Showing that $\alpha \not\equiv \beta$

Consider these two temporal formulas

$F G p$

$AF AG p$

Consider this transition system, $M$:

 Paths of $M$ look like:

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

Sequences of propositions:

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

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

$M \models F G p$

$M \not\models AF AG p$

Typical HW problem: Show two formulas not equivalent.