LTL: Linear Temporal Logic

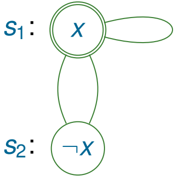
- Computation Tree
- Linear Temporal Logic
- Using Temporal Formulas
- Equivalences of Temporal Formulas

Computation Tree

Let $\mathbb{S} = (S, In, T, \mathcal{X}, dom)$ be a transition system and $s \in S$ be a state. The computation tree for $\mathbb{S}$ starting at s is the following (possibly infinite) tree.

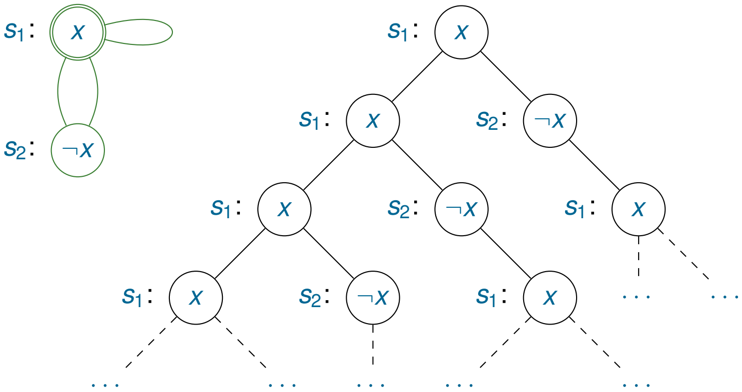
1. The nodes of the tree are labeled by states in S.
2. The root of the tree is labeled by s.
3. For every node s' in the tree, its children are exactly such nodes $s'' \in S$ that $(s', s'') \in T$.

Computation Trees and Paths



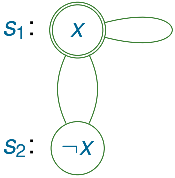
A computation path for $\mathbb{S}$: any branch $s_0, s_1, \ldots$ in the tree.

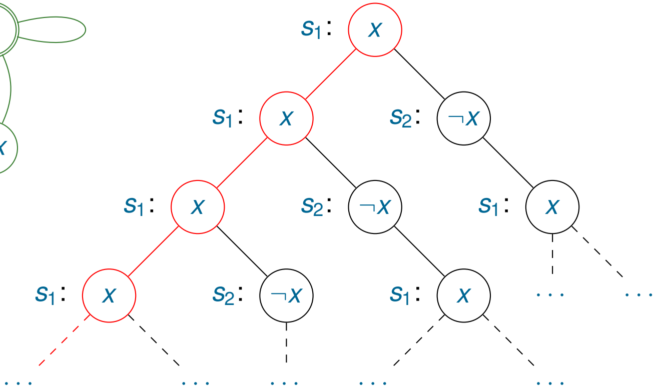
Computation Trees and Paths



A computation path for $\mathbb{S}$: any branch $s_0, s_1, \ldots$ in the tree.

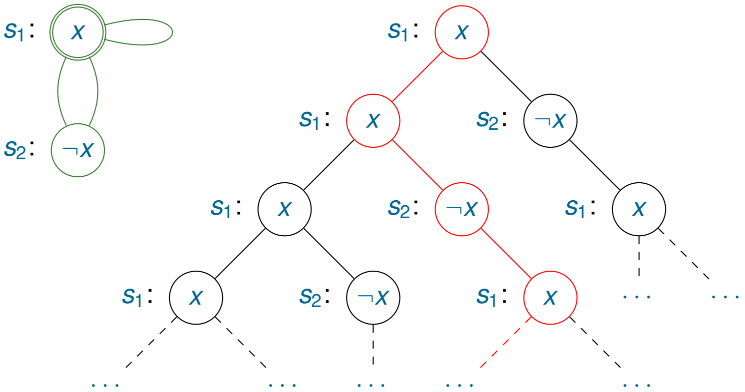
Computation Trees and Paths





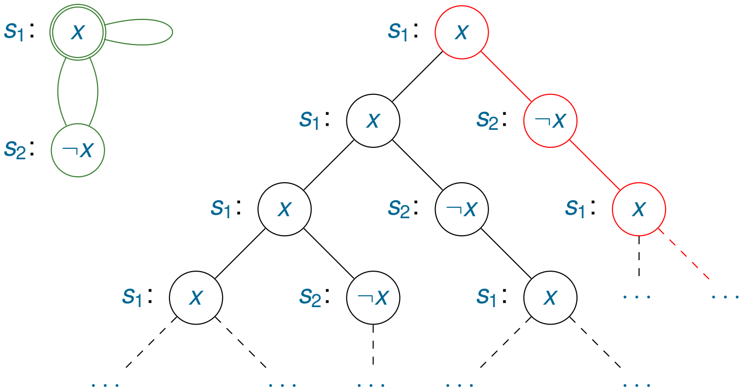
A computation path for $\mathbb{S}$: any branch $s_0, s_1, \ldots$ in the tree.

Computation Trees and Paths



A computation path for $\mathbb{S}$: any branch $s_0, s_1, \ldots$ in the tree.

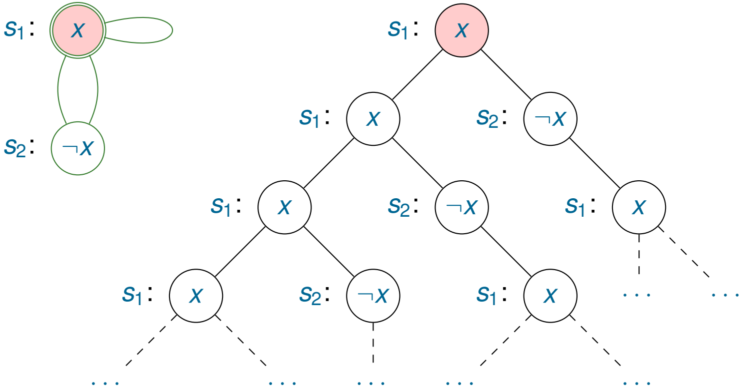
Computation Trees and Paths



A computation path for $\mathbb{S}$: any branch $s_0, s_1, \dots$ in the tree.

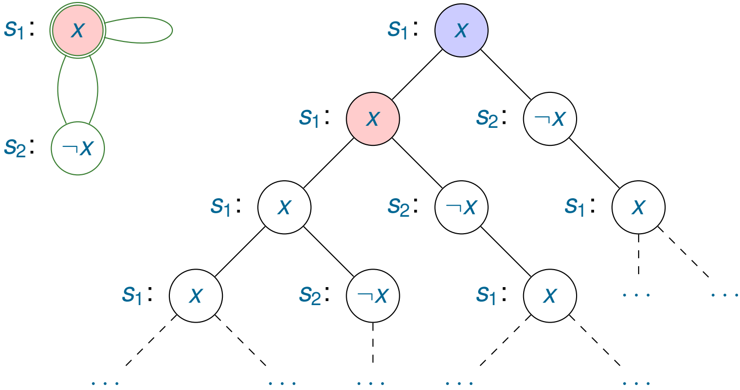
Computation

Every path in the computation tree corresponds to a computation:



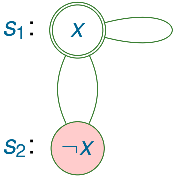
Computation

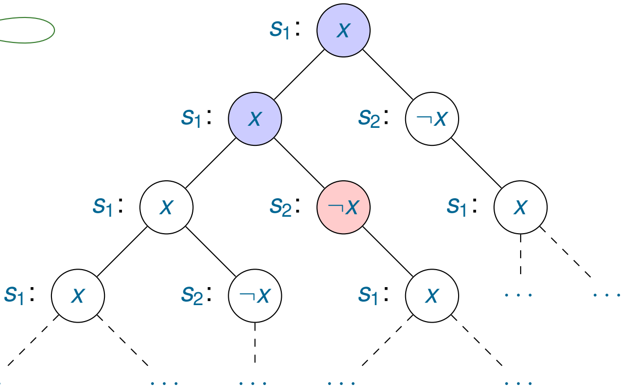
Every path in the computation tree corresponds to a computation:



Computation

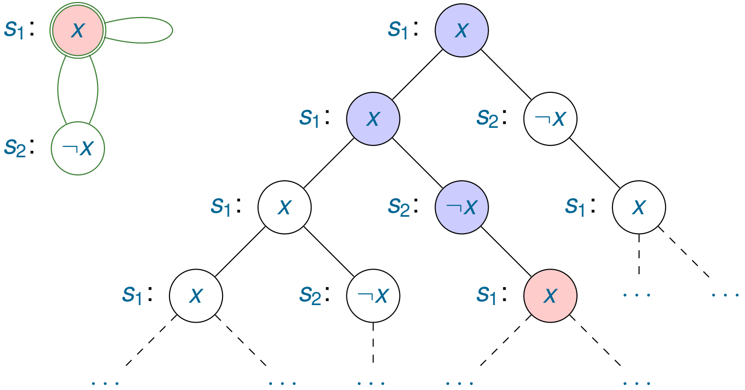
Every path in the computation tree corresponds to a computation:





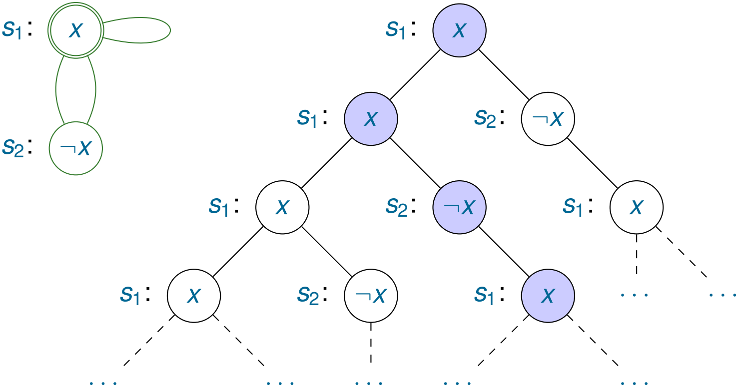
Computation

Every path in the computation tree corresponds to a computation:



Computation

Every path in the computation tree corresponds to a computation:



Properties

- Computation paths for a transition system are exactly all branches in the computation trees for this transition system.
- Let n be a node in a computation tree C for $\mathbb{S}$ labeled by s'. Then the subtree of C rooted at s' is the computation tree for $\mathbb{S}$ starting at s'. In other words, every subtree of a computation tree rooted at some node is itself a computation tree.
- For every transition system $\mathbb{S}$ and state s there exists a unique computation tree for $\mathbb{S}$ starting at s, up to the order of children.

Properties

- Computation paths for a transition system are exactly all branches in the computation trees for this transition system.
- Let n be a node in a computation tree C for $\mathbb{S}$ labeled by s'. Then the subtree of C rooted at s' is the computation tree for $\mathbb{S}$ starting at s'. In other words, every subtree of a computation tree rooted at some node is itself a computation tree.
- For every transition system $\mathbb{S}$ and state s there exists a unique computation tree for $\mathbb{S}$ starting at s, up to the order of children.

Properties

- Computation paths for a transition system are exactly all branches in the computation trees for this transition system.
- Let n be a node in a computation tree C for $\mathbb{S}$ labeled by s'. Then the subtree of C rooted at s' is the computation tree for $\mathbb{S}$ starting at s'. In other words, every subtree of a computation tree rooted at some node is itself a computation tree.
- For every transition system $\mathbb{S}$ and state s there exists a unique computation tree for $\mathbb{S}$ starting at s, up to the order of children.

LTL

Linear Temporal Logic is a logic for reasoning about properties of computation paths.

Formulas are built in the same way as in propositional logic, with the following additions:

1. If F is a formula, then $\bigcirc F$, $\Box F$, and $\Diamond F$ are formulas;
2. If F and G are formulas, then $F \,\mathcal{U}\, G$ and $F \,\mathcal{R}\, G$ are formulas.

LTL

Linear Temporal Logic is a logic for reasoning about properties of computation paths.

Formulas are built in the same way as in propositional logic, with the following additions:

1. If F is a formula, then $\bigcirc F$, $\square F$, and $\lozenge F$ are formulas;
2. If F and G are formulas, then $F \, \mathfrak{U} \, G$ and $F \, \mathfrak{R} \, G$ are formulas.

LTL

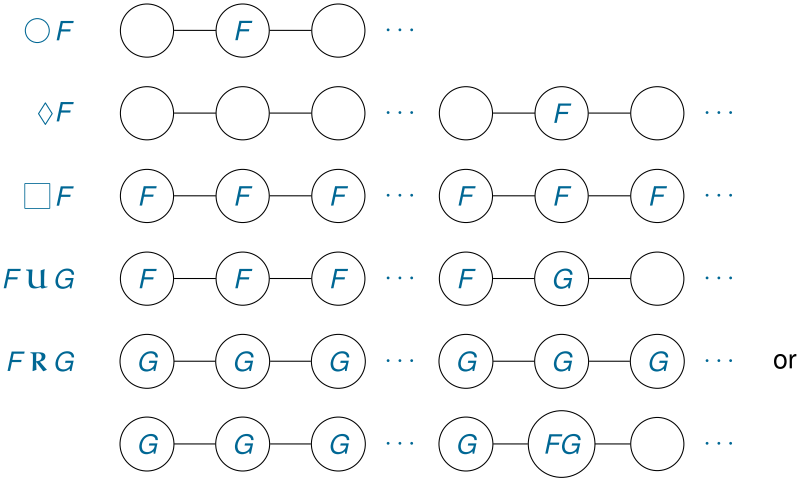
Linear Temporal Logic is a logic for reasoning about properties of computation paths.

Formulas are built in the same way as in propositional logic, with the following additions:

1. If F is a formula, then $\bigcirc F$, $\Box F$, and $\Diamond F$ are formulas;
2. If F and G are formulas, then $F \,\mathcal{U}\, G$ and $F \,\mathcal{R}\, G$ are formulas.

$\bigcirc$	next
$\Box$	always (in future)
$\Diamond$	sometimes (in future)
$\mathcal{U}$	until
$\mathcal{R}$	release

Semantics (intuitive)



Semantics

Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.

Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.

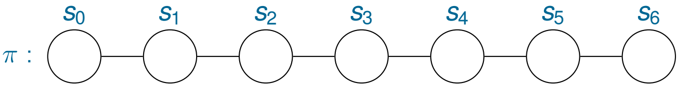
We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.

For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).

To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

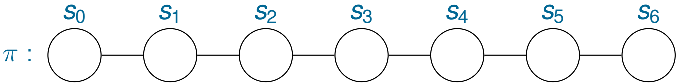
Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

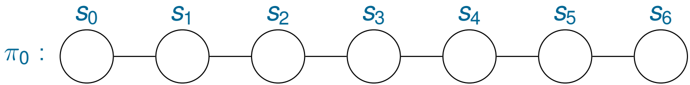
Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

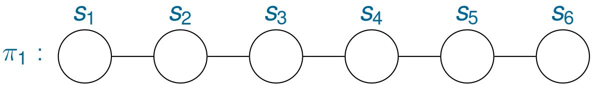
Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

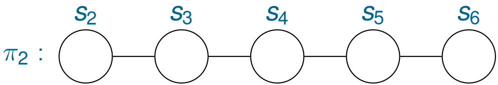
Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

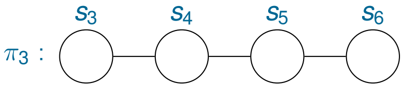
Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

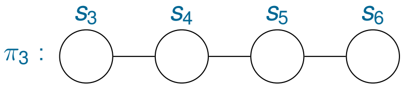
Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics

Unlike propositonal formulas, LTL formulas express properties of computations or computation paths.
Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and F be an LTL formula.



We define the notion F is true on π (or F holds on π), denoted by $\pi \models F$, by induction on F as follows.
For all $i = 0, 1, \ldots$ denote by π_i the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).
To define $\pi \models F$ we will use $\pi_i \models G$ for some i and G. We will sometimes (slightly informally) say that G is true in s_i or G holds in s_i to mean that G is true on π_i.

Semantics, formally

The semantics of propositional connectives is standard.

Atomic formulas are true iff they are true in s_0.

The semantics of formulas built using propositional connectives on π is the same as in propositional logic where all subformulas are also evaluated on π.

1. $\pi \models \top$ and $\pi \not\models \bot$.
2. $\pi \models x = v$ if $s_0 \models x = v$.
3. $\pi \models F_1 \land \ldots \land F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$;
   $\pi \models F_1 \lor \ldots \lor F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
4. $\pi \models \neg F$ if $\pi \not\models F$.
5. $\pi \models F \to G$ if either $\pi \not\models F$ or $\pi \models G$;
   $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.

Semantics, formally

The semantics of propositional connectives is standard.

Atomic formulas are true iff they are true in s_0.

The semantics of formulas built using propositional connectives on π is the same as in propositional logic where all subformulas are also evaluated on π.

1. $\pi \models \top$ and $\pi \not\models \bot$.
2. $\pi \models x = v$ if $s_0 \models x = v$.
3. $\pi \models F_1 \wedge \ldots \wedge F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$;
   $\pi \models F_1 \vee \ldots \vee F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
4. $\pi \models \neg F$ if $\pi \not\models F$.
5. $\pi \models F \rightarrow G$ if either $\pi \not\models F$ or $\pi \models G$;
   $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.

Semantics, formally

The semantics of propositional connectives is standard.

Atomic formulas are true iff they are true in s_0.

The semantics of formulas built using propositional connectives on π is the same as in propositional logic where all subformulas are also evaluated on π.

1. $\pi \models \top$ and $\pi \not\models \bot$.
2. $\pi \models x = v$ if $s_0 \models x = v$.
3. $\pi \models F_1 \wedge \ldots \wedge F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$;
   $\pi \models F_1 \vee \ldots \vee F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
4. $\pi \models \neg F$ if $\pi \not\models F$.
5. $\pi \models F \rightarrow G$ if either $\pi \not\models F$ or $\pi \models G$;
   $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.

Semantics, formally

The semantics of propositional connectives is standard.

Atomic formulas are true iff they are true in s_0.

The semantics of formulas built using propositional connectives on π is the same as in propositional logic where all subformulas are also evaluated on π.

1. $\pi \models \top$ and $\pi \not\models \bot$.
2. $\pi \models x = v$ if $s_0 \models x = v$.
3. $\pi \models F_1 \land \ldots \land F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$;
   $\pi \models F_1 \lor \ldots \lor F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
4. $\pi \models \neg F$ if $\pi \not\models F$.
5. $\pi \models F \rightarrow G$ if either $\pi \not\models F$ or $\pi \models G$;
   $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.

Semantics, formally

The semantics of propositional connectives is standard.

Atomic formulas are true iff they are true in s_0.

The semantics of formulas built using propositional connectives on π is the same as in propositional logic where all subformulas are also evaluated on π.

1. $\pi \models \top$ and $\pi \not\models \bot$.
2. $\pi \models x = v$ if $s_0 \models x = v$.
3. $\pi \models F_1 \wedge \ldots \wedge F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$;
   $\pi \models F_1 \vee \ldots \vee F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
4. $\pi \models \neg F$ if $\pi \not\models F$.
5. $\pi \models F \rightarrow G$ if either $\pi \not\models F$ or $\pi \models G$;
   $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.

Semantics, formally

The semantics of propositional connectives is standard.

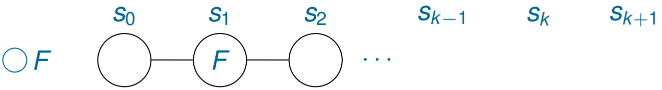
Atomic formulas are true iff they are true in s_0.

The semantics of formulas built using propositional connectives on π is the same as in propositional logic where all subformulas are also evaluated on π.

1. $\pi \models \top$ and $\pi \not\models \bot$.
2. $\pi \models x = v$ if $s_0 \models x = v$.
3. $\pi \models F_1 \land \ldots \land F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$;
   $\pi \models F_1 \lor \ldots \lor F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
4. $\pi \models \neg F$ if $\pi \not\models F$.
5. $\pi \models F \rightarrow G$ if either $\pi \not\models F$ or $\pi \models G$;
   $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.

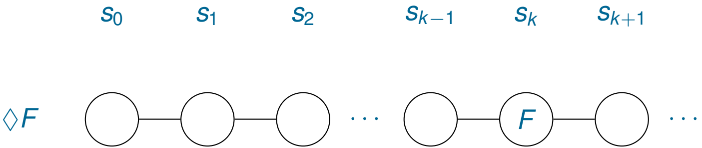
Semantics of temporal operators

6. $\pi \models \bigcirc F$ if $\pi_1 \models F$;
   $\pi \models \Diamond F$ if for some $k \geq 0$ we have $\pi_k \models F$;
   $\pi \models \Box F$ if for all $i \geq 0$ we have $\pi_i \models F$.
7. $\pi \models F \mathbin{\mathrm{U}} G$ if for some $k \geq 0$ we have $\pi_k \models G$ and
   $\pi_0 \models F, \ldots, \pi_{k-1} \models F$;
   $\pi \models F \mathbin{\mathrm{R}} G$ if for all $k \geq 0$, either $\pi_k \models G$ or there exists $j < k$ such that $\pi_j \models F$.



Semantics of temporal operators

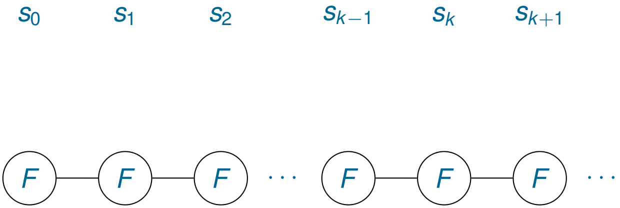
6. $\pi \models \bigcirc F$ if $\pi_1 \models F$;
   $\pi \models \Diamond F$ if for some $k \geq 0$ we have $\pi_k \models F$;
   $\pi \models \Box F$ if for all $i \geq 0$ we have $\pi_i \models F$.
7. $\pi \models F \mathbin{\mathcal{U}} G$ if for some $k \geq 0$ we have $\pi_k \models G$ and $\pi_0 \models F, \ldots, \pi_{k-1} \models F$;
   $\pi \models F \mathbin{\mathcal{R}} G$ if for all $k \geq 0$, either $\pi_k \models G$ or there exists $j < k$ such that $\pi_j \models F$.



Semantics of temporal operators

6. $\pi \models \bigcirc F$ if $\pi_1 \models F$;
   $\pi \models \Diamond F$ if for some $k \geq 0$ we have $\pi_k \models F$;
   $\pi \models \Box F$ if for all $i \geq 0$ we have $\pi_i \models F$.
7. $\pi \models F \,\mathbf{U}\, G$ if for some $k \geq 0$ we have $\pi_k \models G$ and $\pi_0 \models F, \ldots, \pi_{k-1} \models F$;
   $\pi \models F \,\mathbf{R}\, G$ if for all $k \geq 0$, either $\pi_k \models G$ or there exists $j < k$ such that $\pi_j \models F$.

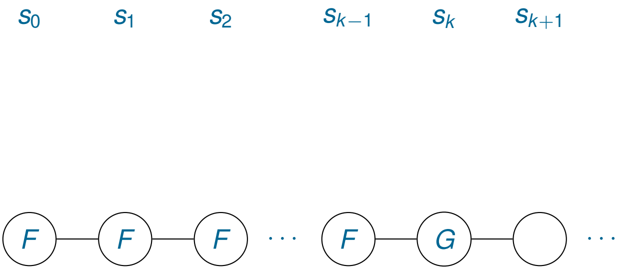
$\Box F$



Semantics of temporal operators

6. $\pi \models \bigcirc F$ if $\pi_1 \models F$;
   $\pi \models \Diamond F$ if for some $k \geq 0$ we have $\pi_k \models F$;
   $\pi \models \Box F$ if for all $i \geq 0$ we have $\pi_i \models F$.
7. $\pi \models F \mathbf{U} G$ if for some $k \geq 0$ we have $\pi_k \models G$ and
   $\pi_0 \models F, \ldots, \pi_{k-1} \models F$;
   $\pi \models F \mathbf{R} G$ if for all $k \geq 0$, either $\pi_k \models G$ or there exists $j < k$ such that $\pi_j \models F$.

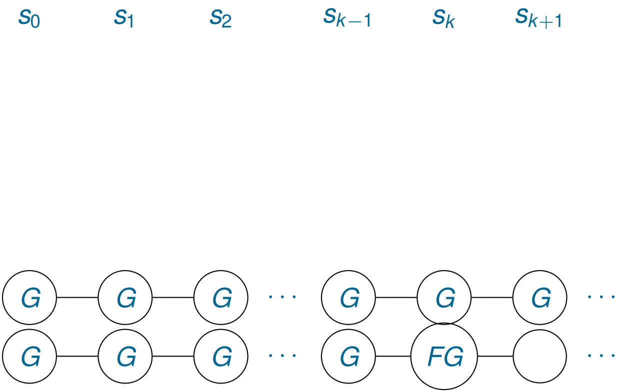
$F \mathbf{U} G$



Semantics of temporal operators

6. $\pi \models \bigcirc F$ if $\pi_1 \models F$;
   $\pi \models \Diamond F$ if for some $k \geq 0$ we have $\pi_k \models F$;
   $\pi \models \Box F$ if for all $i \geq 0$ we have $\pi_i \models F$.
7. $\pi \models F \,\mathcal{U}\, G$ if for some $k \geq 0$ we have $\pi_k \models G$ and
   $\pi_0 \models F, \ldots, \pi_{k-1} \models F$;
   $\pi \models F \,\mathcal{R}\, G$ if for all $k \geq 0$, either $\pi_k \models G$ or there exists $j < k$
   such that $\pi_j \models F$.

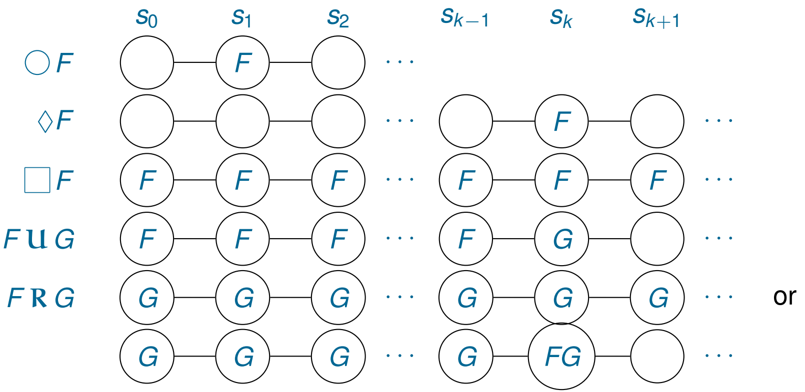
$F \,\mathcal{R}\, G$



or

Semantics of temporal operators

6. $\pi \models \bigcirc F$ if $\pi_1 \models F$;
   $\pi \models \Diamond F$ if for some $k \geq 0$ we have $\pi_k \models F$;
   $\pi \models \Box F$ if for all $i \geq 0$ we have $\pi_i \models F$.
7. $\pi \models F \mathbf{U} G$ if for some $k \geq 0$ we have $\pi_k \models G$ and
   $\pi_0 \models F, \ldots, \pi_{k-1} \models F$;
   $\pi \models F \mathbf{R} G$ if for all $k \geq 0$, either $\pi_k \models G$ or there exists $j < k$
   such that $\pi_j \models F$.



Standard properties

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Consider a transition system $\mathbb{S}$. We are interested in checking the following properties of LTL formulas

For an LTL formula F we can consider two kinds of properties of $\mathbb{S}$:

1. does F hold on some computation path for $\mathbb{S}$ from an initial state?
2. does F hold on all computation paths for $\mathbb{S}$ from an initial state?

Standard properties

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Consider a transition system $\mathbb{S}$. We are interested in checking the following properties of LTL formulas

For an LTL formula F we can consider two kinds of properties of $\mathbb{S}$:

1. does F hold on some computation path for $\mathbb{S}$ from an initial state?
2. does F hold on all computation paths for $\mathbb{S}$ from an initial state?

Standard properties

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Consider a transition system $\mathbb{S}$. We are interested in checking the following properties of LTL formulas

For an LTL formula F we can consider two kinds of properties of $\mathbb{S}$:

1. does F hold on some computation path for $\mathbb{S}$ from an initial state?
2. does F hold on all computation paths for $\mathbb{S}$ from an initial state?

Precedences of Connectives and Temporal Operators

Connective	Precedence
$\neg, \bigcirc, \Diamond, \Box$	6
$\mathbf{U}, \mathbf{R}$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Expressing Some Properties

1. F never holds in two consecutive states.

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \to \bigcirc \neg F)$

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\Box(F \rightarrow \Box F)$

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \to \bigcirc \neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\Box(F \to \Box F)$
3. F holds in at most one state.

Expressing Some Properties

1. F never holds in two consecutive states. $\square(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\square(F \rightarrow \square F)$
3. F holds in at most one state. $\square(F \rightarrow \bigcirc\square\neg F)$

Expressing Some Properties

1. F never holds in two consecutive states. $\square(F \to \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\square(F \to \square F)$
3. F holds in at most one state. $\square(F \to \bigcirc\square\neg F)$
4. F holds in at least two states.

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\Box(F \rightarrow \Box F)$
3. F holds in at most one state. $\Box(F \rightarrow \bigcirc\Box\neg F)$
4. F holds in at least two states. $\Diamond(F \wedge \bigcirc\Diamond F)$

Expressing Some Properties

1. F never holds in two consecutive states. $\square(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\square(F \rightarrow \square F)$
3. F holds in at most one state. $\square(F \rightarrow \bigcirc \square\neg F)$
4. F holds in at least two states. $\lozenge(F \wedge \bigcirc\lozenge F)$
5. Unless s_i is the first state of the path, if F holds in state s_i, then G must hold in at least one of the two states just before s_i, that is s_{i-1} and s_{i-2}.

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\Box(F \rightarrow \Box F)$
3. F holds in at most one state. $\Box(F \rightarrow \bigcirc\Box\neg F)$
4. F holds in at least two states. $\Diamond(F \land \bigcirc\Diamond F)$
5. Unless s_i is the first state of the path, if F holds in state s_i, then G must hold in at least one of the two states just before s_i, that is s_{i-1} and s_{i-2}. $(\bigcirc F \rightarrow G) \land \Box(\bigcirc\bigcirc F \rightarrow G \lor \bigcirc G)$

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s. $\Box(F \rightarrow \Box F)$
3. F holds in at most one state. $\Box(F \rightarrow \bigcirc\Box\neg F)$
4. F holds in at least two states. $\Diamond(F \wedge \bigcirc\Diamond F)$
5. Unless s_i is the first state of the path, if F holds in state s_i, then G must hold in at least one of the two states just before s_i, that is s_{i-1} and s_{i-2}. $(\bigcirc F \rightarrow G) \wedge \Box(\bigcirc\bigcirc F \rightarrow G \vee \bigcirc G)$
6. F happens infinitely often.

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s.
   $\Box(F \rightarrow \Box F)$
3. F holds in at most one state. $\Box(F \rightarrow \bigcirc\Box\neg F)$
4. F holds in at least two states. $\Diamond(F \wedge \bigcirc\Diamond F)$
5. Unless s_i is the first state of the path, if F holds in state s_i, then G must hold in at least one of the two states just before s_i, that is s_{i-1} and s_{i-2}. $(\bigcirc F \rightarrow G) \wedge \Box(\bigcirc\bigcirc F \rightarrow G \vee \bigcirc G)$
6. F happens infinitely often. $\Box\Diamond F$

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \to \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s. $\Box(F \to \Box F)$
3. F holds in at most one state. $\Box(F \to \bigcirc\Box\neg F)$
4. F holds in at least two states. $\Diamond(F \land \bigcirc\Diamond F)$
5. Unless s_i is the first state of the path, if F holds in state s_i, then G must hold in at least one of the two states just before s_i, that is s_{i-1} and s_{i-2}. $(\bigcirc F \to G) \land \Box(\bigcirc\bigcirc F \to G \lor \bigcirc G)$
6. F happens infinitely often. $\Box\Diamond F$
7. F holds in each even state and does not hold in each odd state (states are counted from 0).

Expressing Some Properties

1. F never holds in two consecutive states. $\Box(F \rightarrow \bigcirc\neg F)$
2. If F holds in a state s, it also holds in all states after s. $\Box(F \rightarrow \Box F)$
3. F holds in at most one state. $\Box(F \rightarrow \bigcirc\Box\neg F)$
4. F holds in at least two states. $\Diamond(F \wedge \bigcirc\Diamond F)$
5. Unless s_i is the first state of the path, if F holds in state s_i, then G must hold in at least one of the two states just before s_i, that is s_{i-1} and s_{i-2}. $(\bigcirc F \rightarrow G) \wedge \Box(\bigcirc\bigcirc F \rightarrow G \vee \bigcirc G)$
6. F happens infinitely often. $\Box\Diamond F$
7. F holds in each even state and does not hold in each odd state (states are counted from 0). $F \wedge \Box(F \leftrightarrow \bigcirc\neg F)$.

Meaning of Some Formulas

- $\Diamond \Box F$;

Meaning of Some Formulas

- $\Diamond \Box F$;
- $\Box(F \rightarrow \bigcirc F)$;

Meaning of Some Formulas

- $\Diamond \Box F$;
- $\Box (F \to \bigcirc F)$;
- $\neg F \, \mathcal{U} \, \Box F$;

Meaning of Some Formulas

- $\Diamond \Box F$;
- $\Box (F \rightarrow \bigcirc F)$;
- $\neg F \, \mathcal{U} \, \Box F$;
- $F \, \mathcal{U} \, \neg F$;

Meaning of Some Formulas

- $\Diamond \Box F$;
- $\Box(F \rightarrow \bigcirc F)$;
- $\neg F \,\mathcal{U}\, \Box F$;
- $F \,\mathcal{U}\, \neg F$;
- $\Diamond F \wedge \Box(F \rightarrow \bigcirc F)$;

Meaning of Some Formulas

- $\Diamond\Box F$;
- $\Box(F \to \bigcirc F)$;
- $\neg F\,\mathcal{U}\,\Box F$;
- $F\,\mathcal{U}\,\neg F$;
- $\Diamond F \land \Box(F \to \bigcirc F)$;
- $\Box\Diamond F$;

Meaning of Some Formulas

- $\Diamond \Box F$;
- $\Box (F \rightarrow \bigcirc F)$;
- $\neg F \,\mathcal{U}\, \Box F$;
- $F \,\mathcal{U}\, \neg F$;
- $\Diamond F \wedge \Box (F \rightarrow \bigcirc F)$;
- $\Box \Diamond F$;
- $F \wedge \Box (F \leftrightarrow \neg \bigcirc F)$;

Some useful properties

- Reachability and safety properties.
  Let unsafe describe states which are unsafe.
  Then $\Box\neg$unsafe express a safety requirement.
  Ex: $\Box\neg(\text{disp} = \text{beer} \wedge \text{customer} = \text{prof})$
- Mutual exclusion. Two processes are not in the critical section.
  Ex: $\Box\neg(\text{critical}_1 \wedge \text{critical}_2)$
- Fairness. Ex: $\Box(\text{customer} = \text{student} \rightarrow \Diamond\text{customer} = \text{prof})$
- Responsiveness: every request will be eventually acknowledged
  Ex: $\Box(\text{request} \rightarrow (\text{request}\,\mathcal{U}\,\text{ack}))$
- Alternation. Ex: $A \wedge \Box(A \leftrightarrow \neg\bigcirc A)$

Some useful properties

- Reachability and safety properties.
  Let unsafe describe states which are unsafe.
  Then $\Box\neg\text{unsafe}$ express a safety requirement.
  Ex: $\Box\neg(\text{disp} = \text{beer} \wedge \text{customer} = \text{prof})$
- Mutual exclusion. Two processes are not in the critical section.
  Ex: $\Box\neg(\text{critical}_1 \wedge \text{critical}_2)$
- Fairness. Ex: $\Box(\text{customer} = \text{student} \rightarrow \Diamond\text{customer} = \text{prof})$
- Responsiveness: every request will be eventually acknowledged
  Ex: $\Box(\text{request} \rightarrow (\text{request}\ \mathcal{U}\ \text{ack}))$
- Alternation. Ex: $A \wedge \Box(A \leftrightarrow \neg\bigcirc A)$

Some useful properties

- Reachability and safety properties.
  Let unsafe describe states which are unsafe.
  Then $\Box \neg \mathrm{unsafe}$ express a safety requirement.
  Ex: $\Box \neg(\mathrm{disp} = \mathrm{beer} \wedge \mathrm{customer} = \mathrm{prof})$

- Mutual exclusion. Two processes are not in the critical section.
  Ex: $\Box \neg(\mathrm{critical}_1 \wedge \mathrm{critical}_2)$

- Fairness. Ex: $\Box(\mathrm{customer} = \mathrm{student} \rightarrow \Diamond \mathrm{customer} = \mathrm{prof})$

- Responsiveness: every request will be eventually acknowledged
  Ex: $\Box(\mathrm{request} \rightarrow (\mathrm{request}\, \mathcal{U}\, \mathrm{ack}))$

- Alternation. Ex: $A \wedge \Box(A \leftrightarrow \neg \bigcirc A)$

Some useful properties

- Reachability and safety properties.
  Let unsafe describe states which are unsafe.
  Then $\Box \neg unsafe$ express a safety requirement.
  Ex: $\Box \neg (disp = beer \wedge customer = prof)$
- Mutual exclusion. Two processes are not in the critical section.
  Ex: $\Box \neg (critical_1 \wedge critical_2)$
- Fairness. Ex: $\Box (customer = student \rightarrow \Diamond customer = prof)$
- Responsiveness: every request will be eventually acknowledged
  Ex: $\Box (request \rightarrow (request \, \mathcal{U} \, ack))$
- Alternation. Ex: $A \wedge \Box (A \leftrightarrow \neg \bigcirc A)$

Some useful properties

- Reachability and safety properties.
  Let unsafe describe states which are unsafe.
  Then $\Box \neg \mathsf{unsafe}$ express a safety requirement.
  Ex: $\Box \neg(\mathsf{disp} = \mathsf{beer} \wedge \mathsf{customer} = \mathsf{prof})$
- Mutual exclusion. Two processes are not in the critical section.
  Ex: $\Box \neg(\mathsf{critical}_1 \wedge \mathsf{critical}_2)$
- Fairness. Ex: $\Box(\mathsf{customer} = \mathsf{student} \rightarrow \Diamond \mathsf{customer} = \mathsf{prof})$
- Responsiveness: every request will be eventually acknowledged
  Ex: $\Box(\mathsf{request} \rightarrow (\mathsf{request}\, \mathcal{U}\, \mathsf{ack}))$
- Alternation. Ex: $A \wedge \Box(A \leftrightarrow \neg \bigcirc A)$

Some Equivalences

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Negation:

$$
\begin{aligned}
\neg \bigcirc A &\equiv \\
\neg \Diamond A &\equiv \\
\neg \Box A &\equiv \\
\neg (A \,\mathcal{U}\, B) &\equiv
\end{aligned}
$$

Some Equivalences

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Negation:

$$\begin{array}{rcl}
\neg \bigcirc A & \equiv & \bigcirc \neg A \\
\neg \Diamond A & \equiv & \\
\neg \Box A & \equiv & \\
\neg (A \, \mathcal{U} \, B) & \equiv &
\end{array}$$

Some Equivalences

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Negation:

$$
\begin{array}{rcl}
\neg \bigcirc A & \equiv & \bigcirc \neg A \\
\neg \Diamond A & \equiv & \Box \neg A \\
\neg \Box A & \equiv & \\
\neg (A \, \mathcal{U} \, B) & \equiv &
\end{array}
$$

Some Equivalences

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Negation:

$$
\begin{array}{rcl}
\neg \bigcirc A & \equiv & \bigcirc \neg A \\
\neg \Diamond A & \equiv & \Box \neg A \\
\neg \Box A & \equiv & \Diamond \neg A \\
\neg (A \,\mathbf{U}\, B) & \equiv &
\end{array}
$$

Some Equivalences

Two LTL formulas F and G are called equivalent, denoted $F \equiv G$, if for every path π we have $\pi \models F$ if and only if $\pi \models G$.

Negation:

$$
\begin{aligned}
\neg \bigcirc A &\equiv \bigcirc \neg A \\
\neg \Diamond A &\equiv \Box \neg A \\
\neg \Box A &\equiv \Diamond \neg A \\
\neg (A \, \mathbf{U} \, B) &\equiv \neg A \, \mathbf{R} \, \neg B
\end{aligned}
$$

Expressing operators through $\mathcal{U}$.

Expressing operators through $\mathcal{U}$.

$$
\begin{aligned}
\Diamond A &\equiv \\
\Box A &\equiv \\
A \,\mathcal{R}\, B &\equiv
\end{aligned}
$$

LTL with only temporal operators $\mathcal{U}, \bigcirc$ has the same expressive power as LTL.

Expressing operators through $\mathcal{U}$.

Expressing operators through $\mathcal{U}$.

$$
\begin{aligned}
\Diamond A &\equiv \top\, \mathcal{U}\, A \\
\Box A &\equiv \\
A\, \mathcal{R}\, B &\equiv
\end{aligned}
$$

LTL with only temporal operators $\mathcal{U}, \bigcirc$ has the same expressive power as LTL.

Expressing operators through $\mathbf{U}$.

Expressing operators through $\mathbf{U}$.

$$
\begin{array}{rcl}
\Diamond A & \equiv & \top \, \mathbf{U} \, A \\
\Box A & \equiv & \neg(\top \, \mathbf{U} \, \neg A) \\
A \, \mathbf{R} \, B & \equiv &
\end{array}
$$

LTL with only temporal operators $\mathbf{U}, \bigcirc$ has the same expressive power as LTL.

Expressing operators through $\mathbf{U}$.

Expressing operators through $\mathbf{U}$.

$$
\begin{array}{rcl}
\Diamond A & \equiv & \top \,\mathbf{U}\, A \\
\Box A & \equiv & \neg(\top \,\mathbf{U}\, \neg A) \\
A \,\mathbf{R}\, B & \equiv & \neg(\neg A \,\mathbf{U}\, \neg B)
\end{array}
$$

LTL with only temporal operators $\mathbf{U}$, $\bigcirc$ has the same expressive power as LTL.

Equivalences: Unwinding Properties

$$
\begin{array}{rcl}
\Diamond F & \equiv & F \vee \bigcirc \Diamond F \\
\Box F & \equiv & F \wedge \bigcirc \Box F \\
F \,\mathbf{U}\, G & \equiv & G \vee (F \wedge \bigcirc (F \,\mathbf{U}\, G)) \\
F \,\mathbf{R}\, G & \equiv & G \wedge (F \vee \bigcirc (F \,\mathbf{R}\, G))
\end{array}
$$

Other Equivalences

$$\begin{aligned}\Diamond(F \vee G) &\equiv \Diamond F \vee \Diamond G \\ \Box(F \wedge G) &\equiv \Box F \wedge \Box G\end{aligned}$$

Other Equivalences

$$\begin{aligned} \Diamond(F \lor G) &\equiv \Diamond F \lor \Diamond G \\ \Box(F \land G) &\equiv \Box F \land \Box G \end{aligned}$$

But

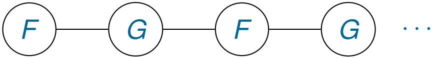
$$\begin{aligned} \Box(F \lor G) &\not\equiv \Box F \lor \Box G \\ \Diamond(F \land G) &\not\equiv \Diamond F \land \Diamond G \end{aligned}$$

How to Show that Two Formulas are not Equivalent?

Find a path that satisfies one of the formulas but not the other. For example for $\Box(F \vee G)$ and $\Box F \vee \Box G$.

How to Show that Two Formulas are not Equivalent?

Find a path that satisfies one of the formulas but not the other. For example for $\Box(F \vee G)$ and $\Box F \vee \Box G$.



Standard properties

Consider a transition system $\mathbb{S}$. We are interested in checking the following properties of LTL formulas

For an LTL formula F we can consider two kinds of properties of $\mathbb{S}$:

1. does F hold on some computation path for $\mathbb{S}$ from an initial state?
2. does F hold on all computation paths for $\mathbb{S}$ from an initial state?

Standard properties

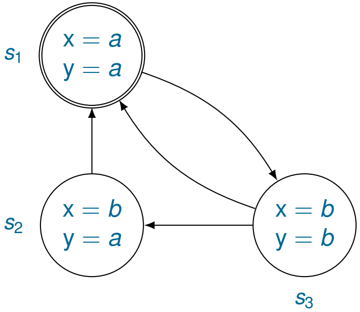
Consider a transition system $\mathbb{S}$. We are interested in checking the following properties of LTL formulas

For an LTL formula F we can consider two kinds of properties of $\mathbb{S}$:

1. does F hold on some computation path for $\mathbb{S}$ from an initial state?
2. does F hold on all computation paths for $\mathbb{S}$ from an initial state?

Example

Consider a transition system with the following state transition graph.



Are the following formulas true on all/some paths (from the initial state)?

$\Box(x = a \leftrightarrow y = a)$

$\Box(x = b \rightarrow \Diamond y = a)$

$\Box\Diamond y = b$

$\Box\Diamond y \neq b$

Summary LTL

Linear temporal logic (LTL) – expressing properties of computations.

- Computation tree, path
- LTL Syntax $\bigcirc$, $\Box$, $\Diamond$, $\mathcal{U}$, $\mathcal{R}$
- LTL Semantics
- Equivalences of temporal formulas
- expressing properties of state-changing systems