CS 228 : Logic in Computer Science

Krishna. S

Recap

- Transition Systems as models of systems (read circuits, code, and so on)
- Traces of transition systems
- Properties as set of allowed traces
- These properties are certain languages over the alphabet 2^{AP}, and are called LT properties
- Writing properties in a language fashion
- Logic LTL to capture LT properties

Syntax of Linear Temporal Logic

Given *AP*, a set of propositions,

Syntax of Linear Temporal Logic

Given *AP*, a set of propositions,

- Propositional logic formulae over *AP*
  - $a \in AP$ (atomic propositions)
  - $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$

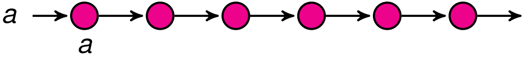
Syntax of Linear Temporal Logic

Given *AP*, a set of propositions,

- Propositional logic formulae over *AP*
  - $a \in AP$ (atomic propositions)
  - $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$
- Temporal Operators
  - $\bigcirc\varphi$ (Next φ)
  - $\varphi\, \mathsf{U} \psi$ (φ holds until a ψ-state is reached)
- LTL : Logic for describing LT properties

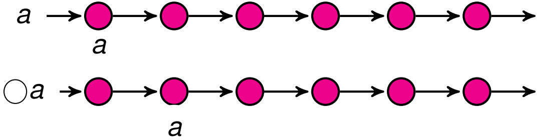
Semantics

LTL formulae over *AP* interpreted over words over Σ^ω, $\Sigma = 2^{AP}$



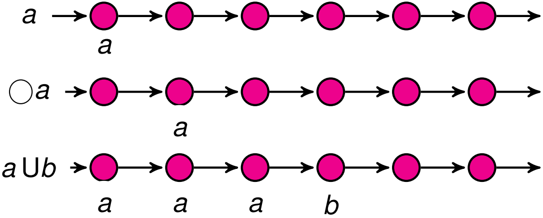
Semantics

LTL formulae over *AP* interpreted over words over Σ^ω, $\Sigma = 2^{AP}$



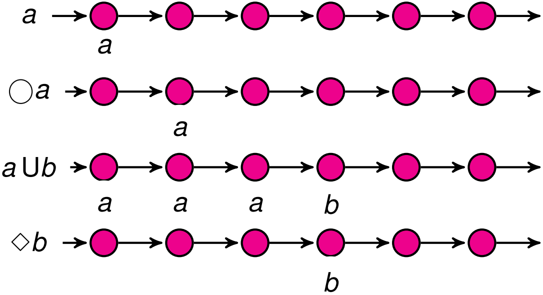
Semantics

LTL formulae over *AP* interpreted over words over Σ^ω, $\Sigma = 2^{AP}$



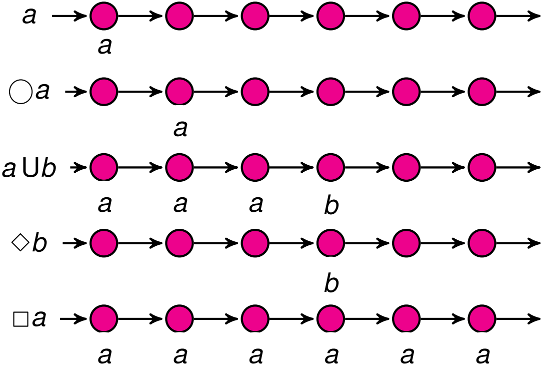
Semantics

LTL formulae over *AP* interpreted over words over Σ^ω, $\Sigma = 2^{AP}$



Semantics

LTL formulae over *AP* interpreted over words over Σ^ω, $\Sigma = 2^{AP}$



Derived Operators

- $\mathit{true} = \varphi \vee \neg\varphi$
- $\mathit{false} = \neg \mathit{true}$
- $\Diamond\varphi = \mathit{true}\, \mathsf{U} \varphi$ (Eventually φ)
- $\Box\varphi = \neg\Diamond\neg\varphi$ (Forever φ)

Precedence

- Unary Operators bind stronger than Binary
- $\bigcirc$ and $\neg$ equally strong
- U takes precedence over $\wedge, \vee, \rightarrow$
  - $a \vee b\, \mathsf{U} c \equiv a \vee (b\, \mathsf{U} c)$
  - $\bigcirc a\, \mathsf{U} \neg b \equiv (\bigcirc a)\, \mathsf{U} (\neg b)$

Examples

- Whenever the traffic light is red, it cannot become green immediately:

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\square(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$
- Once the traffic light becomes yellow, it will eventually become green

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$
- Once the traffic light becomes yellow, it will eventually become green
  $\Box(\mathit{yellow} \rightarrow \Diamond \mathit{green})$

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$
- Once the traffic light becomes yellow, it will eventually become green
  $\Box(\mathit{yellow} \rightarrow \Diamond \mathit{green})$
- Whenever the traffic light is red, it will eventually become green, but it must be yellow for sometime in between the red and the green

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(red \rightarrow \neg \bigcirc green)$
- Eventually the traffic light will become yellow
  $\Diamond yellow$
- Once the traffic light becomes yellow, it will eventually become green
  $\Box(yellow \rightarrow \Diamond green)$
- Whenever the traffic light is red, it will eventually become green, but it must be yellow for sometime in between the red and the green
  $\Box(red \rightarrow \bigcirc(red\ \mathrm{U}[yellow$

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$
- Once the traffic light becomes yellow, it will eventually become green
  $\Box(\mathit{yellow} \rightarrow \Diamond \mathit{green})$
- Whenever the traffic light is red, it will eventually become green, but it must be yellow for sometime in between the red and the green
  $\Box(\mathit{red} \rightarrow \bigcirc(\mathit{red}\ \mathsf{U}[\mathit{yellow} \wedge \bigcirc (\mathit{yellow}\ \mathsf{U} \mathit{green})]))$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$.

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$.

- $\sigma \models a$ iff $a \in A_0$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$.

- $\sigma \models a$ iff $a \in A_0$
- $\sigma \models \varphi_1 \wedge \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^{\omega} \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$.

- $\sigma \models a$ iff $a \in A_0$
- $\sigma \models \varphi_1 \wedge \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^{\omega} \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$.

- $\sigma \models a$ iff $a \in A_0$
- $\sigma \models \varphi_1 \wedge \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$
- $\sigma \models \bigcirc\varphi$ iff $A_1 A_2 \ldots \models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$.

- $\sigma \models a$ iff $a \in A_0$
- $\sigma \models \varphi_1 \wedge \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$
- $\sigma \models \bigcirc\varphi$ iff $A_1 A_2 \ldots \models \varphi$
- $\sigma \models \varphi \, \mathsf{U} \psi$ iff
  $\exists j \geqslant 0$ such that $A_j A_{j+1} \ldots \models \psi \wedge \forall 0 \leqslant i < j, A_i A_{i+1} \ldots \models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

- $\sigma \models \Diamond\varphi$ iff $\exists j \geqslant 0, A_j A_{j+1} \ldots \models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

- $\sigma \models \Diamond\varphi$ iff $\exists j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\varphi$ iff $\forall j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

- $\sigma \models \Diamond\varphi$ iff $\exists j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\varphi$ iff $\forall j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\Diamond\varphi$ iff $\forall j \geqslant 0$, $\exists i \geqslant j$, $A_i A_{i+1} \ldots \models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

- $\sigma \models \Diamond\varphi$ iff $\exists j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\varphi$ iff $\forall j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\Diamond\varphi$ iff $\forall j \geqslant 0$, $\exists i \geqslant j$, $A_i A_{i+1} \ldots \models \varphi$
- $\sigma \models \Diamond\Box\varphi$ iff $\exists j \geqslant 0$, $\forall i \geqslant j$, $A_i A_{i+1} \ldots \models \varphi$

If $\sigma = A_0 A_1 A_2 \ldots$, $\sigma \models \varphi$ is also written as $\sigma, 0 \models \varphi$. This simply means $A_0 A_1 A_2 \ldots \models \varphi$. One can also define $\sigma, i \models \varphi$ to mean $A_i A_{i+1} A_{i+2} \ldots \models \varphi$ to talk about a suffix of the word σ satisfying a property.

Transition System Semantics

Let $TS = (S, S_0, \rightarrow, AP, L)$ be a transition system, and φ an LTL formula over AP

- For an infinite path fragment π of TS,

$$\pi \models \varphi \text{ iff } \mathit{trace}(\pi) \models \varphi$$

Transition System Semantics

Let $TS = (S, S_0, \to, AP, L)$ be a transition system, and φ an LTL formula over AP

- For an infinite path fragment π of TS,

$$\pi \models \varphi \text{ iff } \mathit{trace}(\pi) \models \varphi$$

- For $s \in S$,

$$s \models \varphi \text{ iff } \forall \pi \in \mathit{Paths}(s), \pi \models \varphi$$

Transition System Semantics

Let $TS = (S, S_0, \rightarrow, AP, L)$ be a transition system, and φ an LTL formula over AP

- For an infinite path fragment π of TS,

$$\pi \models \varphi \text{ iff } \mathit{trace}(\pi) \models \varphi$$

- For $s \in S$,

$$s \models \varphi \text{ iff } \forall \pi \in \mathit{Paths}(s), \pi \models \varphi$$

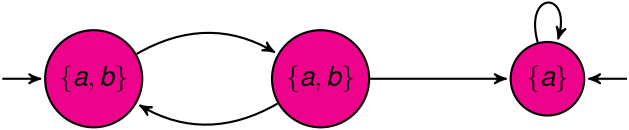
- $TS \models \varphi$ iff $\mathit{Traces}(TS) \subseteq L(\varphi)$

Transition System Semantics

Assume all states in TS are reachable from S_0.

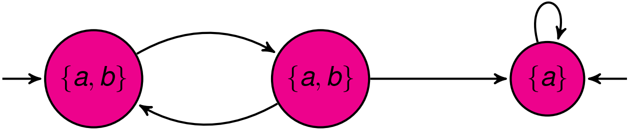
- $TS \models \varphi$ iff $TS \models L(\varphi)$ iff $Traces(TS) \subseteq L(\varphi)$
- $TS \models L(\varphi)$ iff $\pi \models \varphi \; \forall \pi \in Paths(TS)$
- $\pi \models \varphi \; \forall \pi \in Paths(TS)$ iff $s_0 \models \varphi \; \forall s_0 \in S_0$

Example



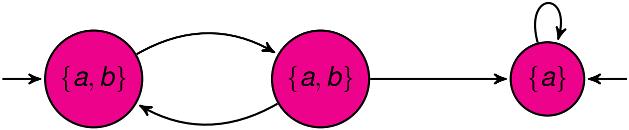
- $TS \models \Box a$,

Example



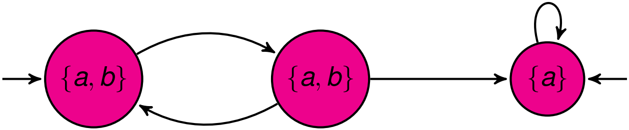
- $TS \models \Box a$,
- $TS \not\models \bigcirc(a \wedge b)$

Example



- $TS \models \Box a$,
- $TS \not\models \bigcirc(a \wedge b)$
- $TS \not\models (b \, \mathsf{U} (a \wedge \neg b))$

Example



- $TS \models \Box a$,
- $TS \not\models \bigcirc(a \wedge b)$
- $TS \not\models (b \,\mathrm{U}(a \wedge \neg b))$
- $TS \models \Box(\neg b \rightarrow \Box(a \wedge \neg b))$

More Semantics

- For paths π, $\pi \models \varphi$ iff $\pi \not\models \neg\varphi$

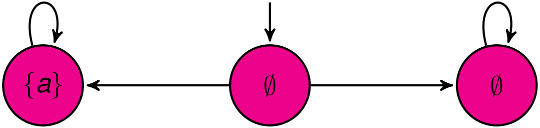
More Semantics

- For paths π, $\pi \models \varphi$ iff $\pi \not\models \neg\varphi$
  $\mathit{trace}(\pi) \in L(\varphi)$ iff $\mathit{trace}(\pi) \notin L(\neg\varphi) = \overline{L(\varphi)}$
- $TS \not\models \varphi$ iff $TS \models \neg\varphi$?
  - $TS \models \neg\varphi \rightarrow \forall$ paths π of TS, $\pi \models \neg\varphi$
  - Thus, $\forall\pi$, $\pi \not\models \varphi$. Hence, $TS \not\models \varphi$

More Semantics

- For paths π, $\pi \models \varphi$ iff $\pi \not\models \neg\varphi$
  $trace(\pi) \in L(\varphi)$ iff $trace(\pi) \notin L(\neg\varphi) = \overline{L(\varphi)}$
- $TS \not\models \varphi$ iff $TS \models \neg\varphi$?
  - $TS \models \neg\varphi \rightarrow \forall$ paths π of TS, $\pi \models \neg\varphi$
  - Thus, $\forall\pi$, $\pi \not\models \varphi$. Hence, $TS \not\models \varphi$
  - Now assume $TS \not\models \varphi$
  - Then $\exists$ some path π in TS such that $\pi \models \neg\varphi$
  - However, there could be another path π' such that $\pi' \models \varphi$
  - Then $TS \not\models \neg\varphi$ as well
- Thus, $TS \not\models \varphi \not\equiv TS \models \neg\varphi$.

An Example



$TS \not\models \Diamond a$ and $TS \not\models \Box \neg a$

Equivalence of LTL Formulae

Equivalence

φ and ψ are equivalent ($\varphi \equiv \psi$) iff $L(\varphi) = L(\psi)$.

Expansion Laws

- $\varphi \, \mathsf{U} \psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi \, \mathsf{U} \psi))$
- $\Diamond\varphi \equiv \varphi \vee \bigcirc\Diamond\varphi$
- $\Box\varphi \equiv \varphi \wedge \bigcirc\Box\varphi$

Equivalence of LTL Formulae

φ and ψ are equivalent iff $L(\varphi) = L(\psi)$.

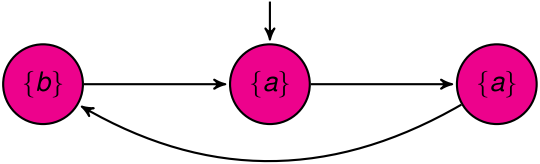
Equivalence of LTL Formulae

φ and ψ are equivalent iff $L(\varphi) = L(\psi)$.

Distribution

$\bigcirc(\varphi \vee \psi) \equiv \bigcirc\varphi \vee \bigcirc\psi$,
$\bigcirc(\varphi \wedge \psi) \equiv \bigcirc\varphi \wedge \bigcirc\psi$,
$\bigcirc(\varphi \,\mathsf{U}\, \psi) \equiv (\bigcirc\varphi) \,\mathsf{U}\, (\bigcirc\psi)$,
$\Diamond(\varphi \vee \psi) \equiv \Diamond\varphi \vee \Diamond\psi$,
$\Box(\varphi \wedge \psi) \equiv \Box\varphi \wedge \Box\psi$

Equivalence of LTL Formulae



$$TS \models \Diamond a \wedge \Diamond b, \; TS \not\models \Diamond(a \wedge b)$$

$$TS \models \Box(a \vee b), \; TS \not\models \Box a \vee \Box b$$

Satisfiability, Model Checking of LTL

Two Questions

Given transition system *TS*, and an LTL formula φ. Does $TS \models \varphi$?
Given an LTL formula φ, is $L(\varphi) = \emptyset$?

How we go about this:

- Translate φ into an automaton A_φ that accepts infinite words such that $L(A_\varphi) = L(\varphi)$.
- Check for emptiness of A_φ to check satisfiability of φ.
- Check if $TS \cap \overline{A_\varphi}$ is empty, to answer the model-checking problem.

Notations for Infinite Words

- Σ is a finite alphabet
- Σ^* set of finite words over Σ
- An infinite word is written as $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots$, where $\alpha(i) \in \Sigma$
- Such words are called ω-words
- $\mathit{Inf}(\alpha) = \{a \in \Sigma \mid \alpha(i) = a \text{ for infinitely many } i\}$. $\mathit{Inf}(\alpha)$ gives the set of symbols occurring infinitely often in α.

ω-automata

An ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where

- Q is a finite set of states
- Σ is a finite alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a state transition function (if non-deterministic, otherwise, $\delta : Q \times \Sigma \to Q$)
- $q_0 \in Q$ is an initial state and Acc is an acceptance condition

ω-automata

An ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where

- Q is a finite set of states
- Σ is a finite alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a state transition function (if non-deterministic, otherwise, $\delta : Q \times \Sigma \to Q$)
- $q_0 \in Q$ is an initial state and Acc is an acceptance condition

Run

A run ρ of $\mathcal{A}$ on an ω-word $\alpha = a_1 a_2 \cdots \in \Sigma^\omega$ is an infinite state sequence $\rho(0)\rho(1)\rho(2)\ldots$ such that

- $\rho(0) = q_0$,
- $\rho(i) = \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is deterministic,
- $\rho(i) \in \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is non-deterministic,

ω-automata

An ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where

- Q is a finite set of states
- Σ is a finite alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a state transition function (if non-deterministic, otherwise, $\delta : Q \times \Sigma \to Q$)
- $q_0 \in Q$ is an initial state and Acc is an acceptance condition

Run

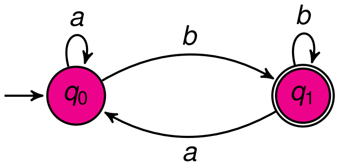
A run ρ of $\mathcal{A}$ on an ω-word $\alpha = a_1 a_2 \cdots \in \Sigma^\omega$ is an infinite state sequence $\rho(0)\rho(1)\rho(2)\ldots$ such that

- $\rho(0) = q_0$,
- $\rho(i) = \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is deterministic,
- $\rho(i) \in \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is non-deterministic,

Büchi Acceptance

For Büchi Acceptance, Acc is specified as a set of states, $G \subseteq Q$. The ω-word α is accepted if there is a run ρ of α such that $Inf(\rho) \cap G \neq \emptyset$.

ω-Automata with Büchi Acceptance



$L(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid \alpha$ has a run ρ such that $Inf(\rho) \cap G \neq \emptyset$

Language accepted=Infinitely many b's.

Comparing NFA and NBA

Comparing NFA and NBA