CS 228 : Logic in Computer Science

Krishna. S

Moving On : Temporal Logics

- Starting Linear Temporal Logic (LTL)

Transition Systems

A Transition System is a tuple $(S, Act, \rightarrow, I, AP, L)$ where

- S is a set of states
- Act is a set of actions
- $s \stackrel{\alpha}{\rightarrow} s'$ in $S \times Act \times S$ is the transition relation
- $I \subseteq S$ is the set of initial states
- AP is the set of atomic propositions
- $L : S \rightarrow 2^{AP}$ is the labeling function

Traces of Transition Systems

- Labels of the locations represent values of all observable propositions $\in AP$
- Captures system state
- Focus on sequences $L(s_0)L(s_1)\ldots$ of labels of locations
- Such sequences are called traces
- Assuming transition systems have no terminal states,
  - Traces are infinite words over 2^{AP}
  - Traces $\in (2^{AP})^\omega$

Traces of Transition Systems

Given a transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states,

- All maximal executions/paths are infinite

Traces of Transition Systems

Given a transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states,

- All maximal executions/paths are infinite
- Path $\pi = s_0 s_1 s_2 \ldots$, $trace(\pi) = L(s_0)L(s_1)\ldots$

Traces of Transition Systems

Given a transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states,

- All maximal executions/paths are infinite
- Path $\pi = s_0 s_1 s_2 \ldots$, $trace(\pi) = L(s_0) L(s_1) \ldots$
- For a set Π of paths, $Trace(\Pi) = \{trace(\pi) \mid \pi \in \Pi\}$

Traces of Transition Systems

Given a transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states,

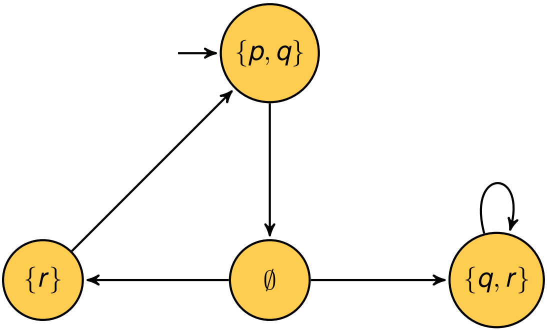
- All maximal executions/paths are infinite
- Path $\pi = s_0 s_1 s_2 \ldots$, $trace(\pi) = L(s_0)L(s_1)\ldots$
- For a set Π of paths, $Trace(\Pi) = \{trace(\pi) \mid \pi \in \Pi\}$
- For a location s, $Traces(s) = Trace(Paths(s))$

Traces of Transition Systems

Given a transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states,

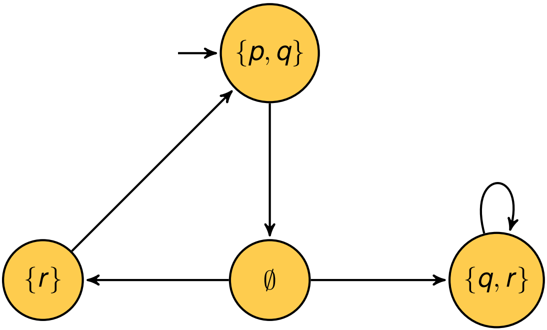
- All maximal executions/paths are infinite
- Path $\pi = s_0 s_1 s_2 \ldots$, $trace(\pi) = L(s_0)L(s_1)\ldots$
- For a set Π of paths, $Trace(\Pi) = \{trace(\pi) \mid \pi \in \Pi\}$
- For a location s, $Traces(s) = Trace(Paths(s))$
- $Traces(TS) = \bigcup_{s \in I} Traces(s)$

Example Traces



$AP = \{p, q, r, t\}$

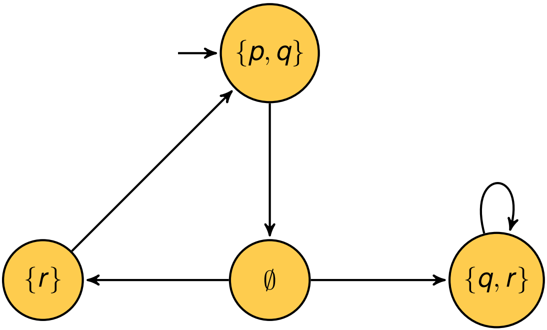
Example Traces



$AP = \{p, q, r, t\}$

- $\{p, q\}\emptyset\ \{q, r\}^{\omega}$

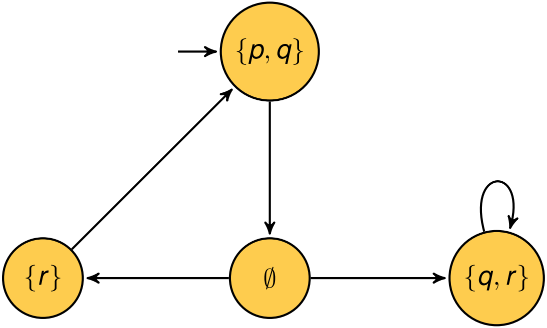
Example Traces



$AP = \{p, q, r, t\}$

- $\{p, q\}\emptyset\ \{q, r\}^\omega$
- $(\{p, q\}\emptyset\{r\})^\omega$

Example Traces



$AP = \{p, q, r, t\}$

- $\{p, q\}\emptyset\ \{q, r\}^\omega$
- $(\{p, q\}\emptyset\{r\})^\omega$
- $(\{p, q\}\emptyset\{r\})^*\ \{p, q\}\emptyset\ \{q, r\}^\omega$

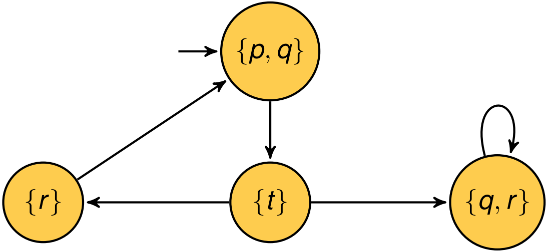
Linear Time Properties

- Linear-time properties specify traces that a *TS* must have
- A LT property P over AP is a subset of $(2^{AP})^\omega$
- TS over AP satisfies a LT property P over AP

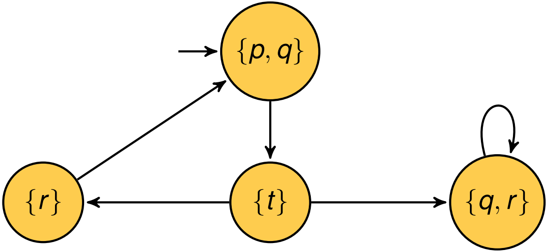
$$TS \models P \text{ iff } Traces(TS) \subseteq P$$

- $s \in S$ satisfies LT property P (denoted $s \models P$) iff $Traces(s) \subseteq P$

Specifying Traces

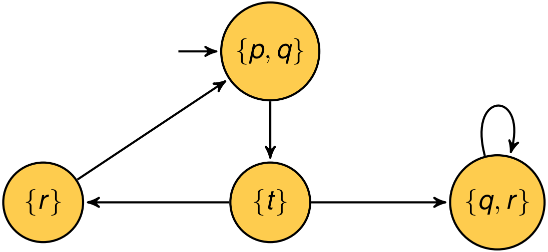


Specifying Traces



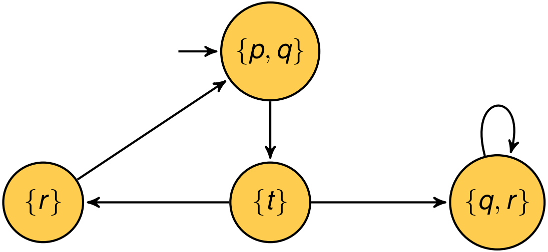
- Whenever p is true, r will eventually become true

Specifying Traces



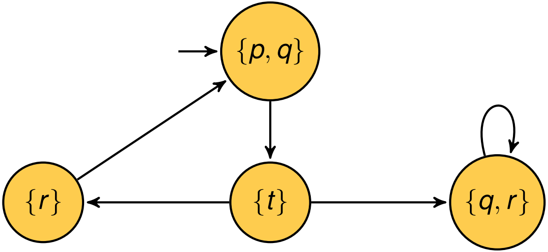
- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$

Specifying Traces



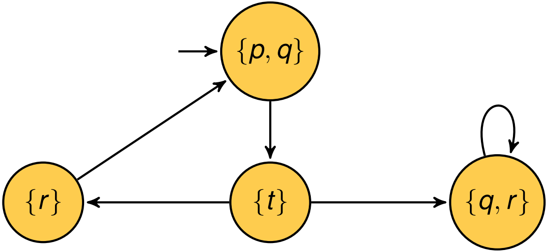
- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often

Specifying Traces



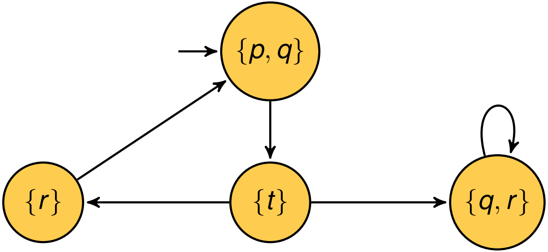
- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$

Specifying Traces



- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$
- Whenever r is true, so is q

Specifying Traces



- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \to \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$
- Whenever r is true, so is q
  - $\{A_0 A_1 \cdots \mid \forall i \geqslant 0, r \in A_i \to q \in A_i\}$

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

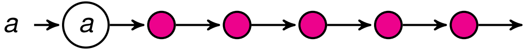
Given *AP*, a set of propositions,

- 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, $\Sigma = 2^{AP}$

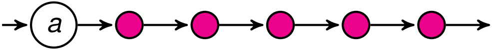
This is not an automaton, but a word over Σ

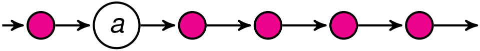


Semantics

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

This is not an automaton, but a word over Σ

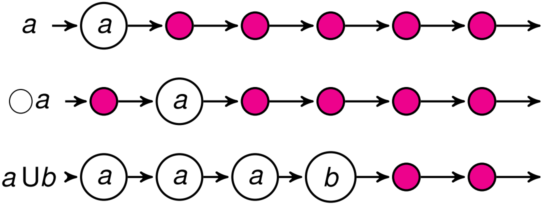
a 

$\bigcirc a$ 

Semantics

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

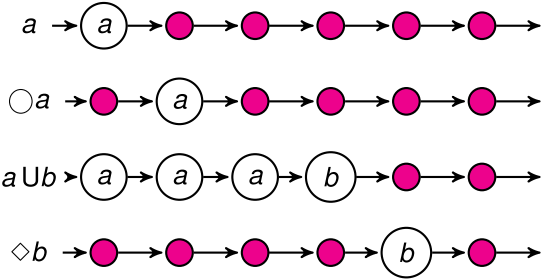
This is not an automaton, but a word over Σ



Semantics

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

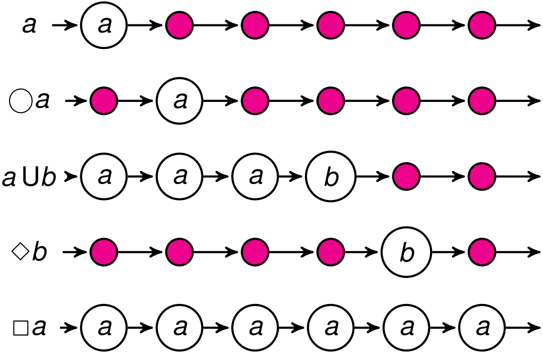
This is not an automaton, but a word over Σ



Semantics

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

This is not an automaton, but a word over Σ



Derived Operators

- $\mathit{true} = \varphi \vee \neg\varphi$
- $\mathit{false} = \neg\mathit{true}$
- $\Diamond\varphi = \mathit{true}\,\mathrm{U}\varphi$ (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 $\wedge, \vee, \rightarrow$
  - $a \vee b\,\mathrm{U}c \equiv a \vee (b\,\mathrm{U}c)$
  - $\bigcirc a\,\mathrm{U}\neg b \equiv (\bigcirc a)\,\mathrm{U}(\neg b)$