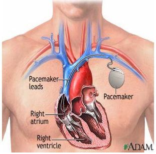
CS 228 : Logic in Computer Science

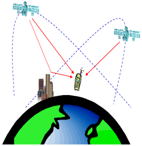
S. Krishna

So Far

- Dwelt on classical logics : propositional logic, FO and MSO on finite words
- Words : good abstraction for capturing properties to be checked on systems built
- Moving on to Temporal logics

Safety Critical Systems



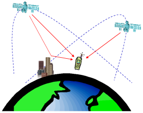


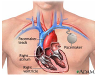
© AIRBUS S.A.S. 2010 - COMPUTER RENDERING BY FIXION - GWLNSD

The role of Automata and Logics

- Systems modeled as certain kinds of automata
- Safety critical properties written in some logic
- Check if the property is satisfied by all runs of the system

Verification through Model Checking





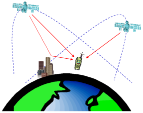
System

satisfy?

specification

good/bad properties

Verification through Model Checking



System

specification

satisfy?

System Model

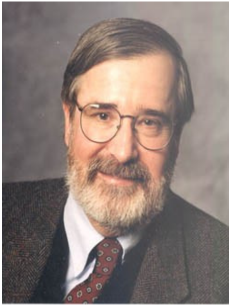
model-checking

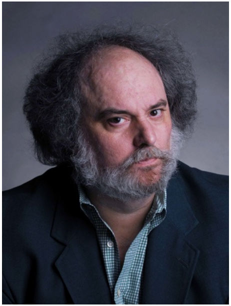
$\models$?

Spec Model

Logic formula ϕ

Model Checking : Pioneers





- Year 2008 : ACM confers the Turing Award to the pioneers of Model Checking: Ed Clarke, Allen Emerson, and Joseph Sifakis

Temporal Logics

- It will rain tomorrow and it does not rain today

Temporal Logics

- It will rain tomorrow and it does not rain today
- Linear Temporal Logic (LTL) for specification of programs (Amir Pnueli, 1977)
  - Turing Award 1996(Amir Pnueli)
  - For seminal work introducing temporal logic into computing science and for outstanding contributions to program and system verification.

Temporal Logics

- It will rain tomorrow and it does not rain today
- Linear Temporal Logic (LTL) for specification of programs (Amir Pnueli, 1977)
  - Turing Award 1996(Amir Pnueli)
  - For seminal work introducing temporal logic into computing science and for outstanding contributions to program and system verification.
- Temporal Logic CTL for program correctness; introduction of model-checking (Emerson and Clarke; Sifakis, 1982)
  - Turing Award 2008 (Clarke, Emerson and Sifakis).
  - For their role in developing model-checking into a highly effective verification technology that is widely adopted in the hardware and software industries.
  - See *http://www-verimag.imag.fr/ sifakis/TuringAwardPaper-Apr14.pdf*.

Transition Systems

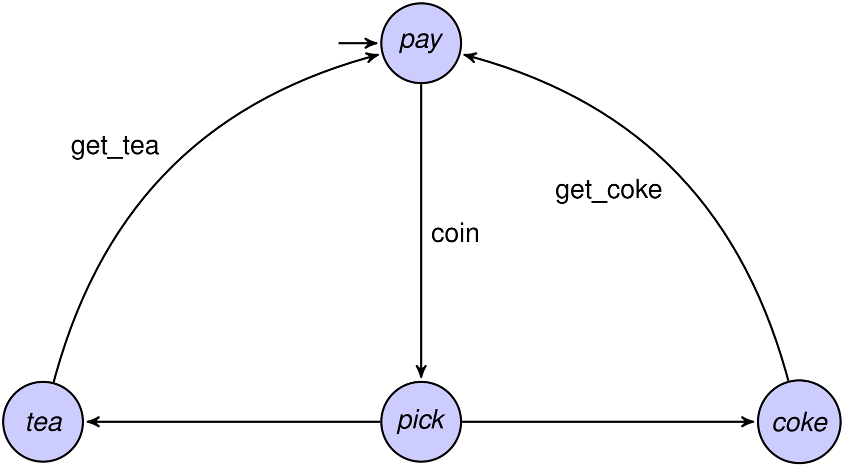
- model to capture the behaviour of systems
- Directed graph, vertices represent “states” of the system, edges represent “transitions” between states
- states? transitions? : system dependent

Transition Systems

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

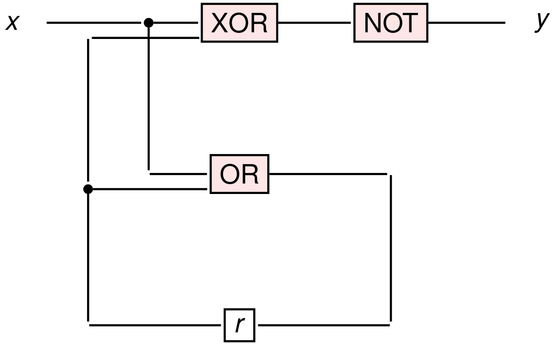
- S is a set of states
- Act is a set of actions
- $s \xrightarrow{\alpha} s'$ in $S \times Act \times S$ is the transition relation
- $I \subseteq S$ is the set of initial states
- AP is the set of atomic propositions
- $L : S \rightarrow 2^{AP}$ is the labeling function

A Model for a Vending Machine



- states, actions, transitions, initial states, atomic propositions

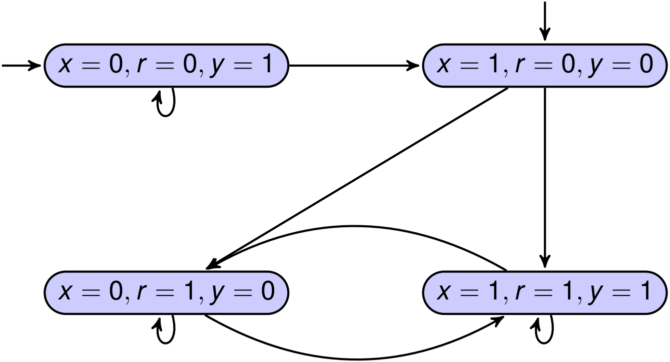
Sequential Circuits



- Input variable x, output variable y, register r
- Output $\neg(x \bigoplus r)$ and register evaluates to $x \vee r$

Transition System for the Circuit

Initially, assume $r = 0$



Traces of Transition Systems

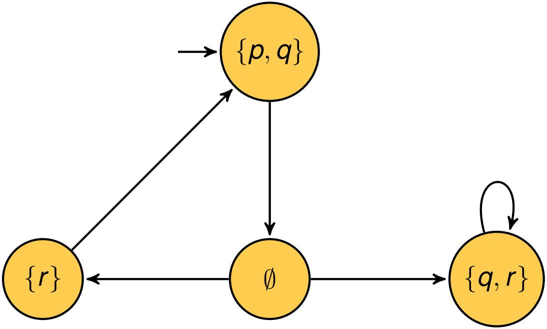
- Labels of the locations represent values of all observable propositions $\in AP$
- Captures system state
- Focus on sequences $L(s_0)L(s_1)\ldots$ of labels of locations
- Such sequences are called traces
- Assuming transition systems have no terminal states,
  - Traces are infinite words over 2^{AP}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

Traces of Transition Systems

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

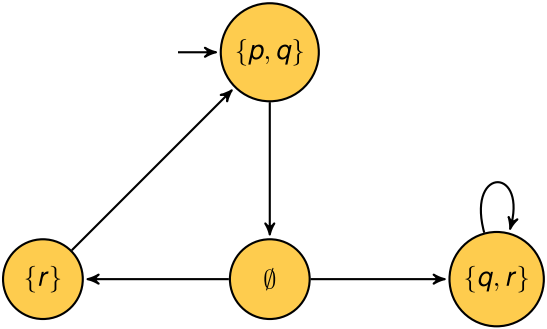
- All maximal executions/paths are infinite
- Path $\pi = s_0 s_1 s_2 \ldots$, $trace(\pi) = L(s_0)L(s_1)\ldots$
- For a set Π of paths, $Trace(\Pi) = \{trace(\pi) \mid \pi \in \Pi\}$
- For a location s, $Traces(s) = Trace(Paths(s))$
- $Traces(TS) = \bigcup_{s \in I} Traces(s)$

Example Traces



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

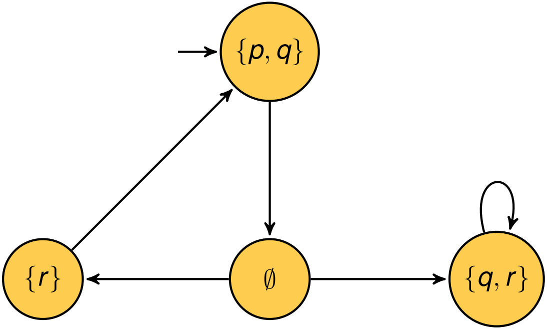
Example Traces



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

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

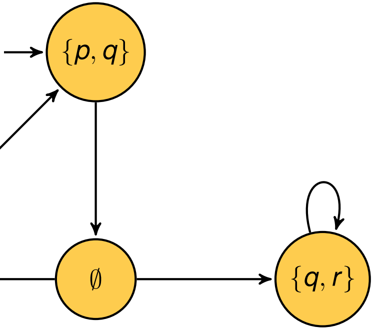
Example Traces



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

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

Example Traces



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

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

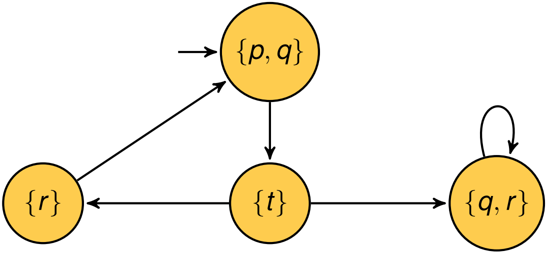
Linear Time Properties

- 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

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

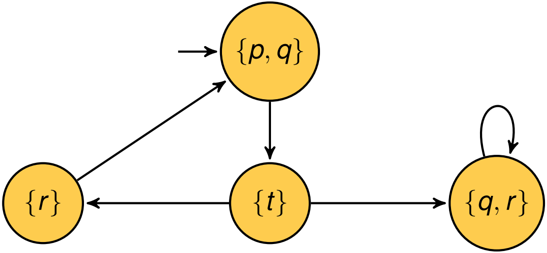
- $s \in S$ satisfies LT property P (denoted $s \models P$) iff $Traces(s) \subseteq P$

Specifying Traces



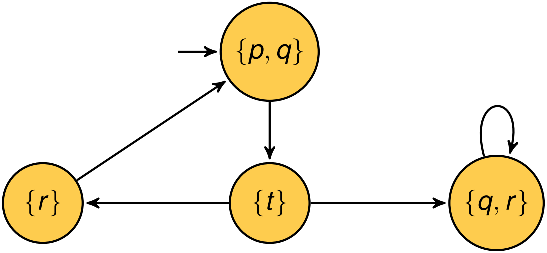
- Whenever p is true, r will eventually become true

Specifying Traces



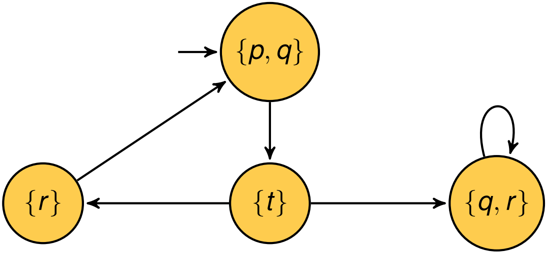
- Whenever p is true, r will eventually become true
    - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$

Specifying Traces



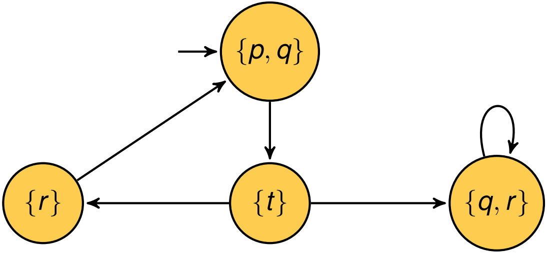
- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often

Specifying Traces



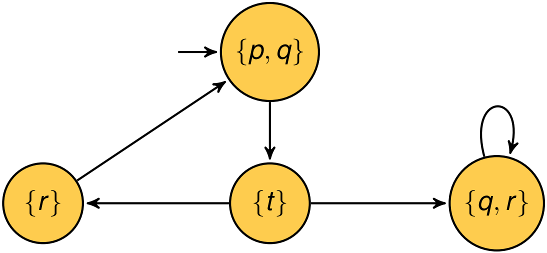
- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$

Specifying Traces



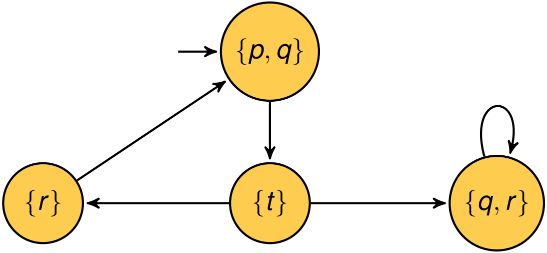
- Whenever p is true, r will eventually become true
  - $\{A_0A_1A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \to \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0A_1A_2 \cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$
- Whenever r is true, so is q

Specifying Traces



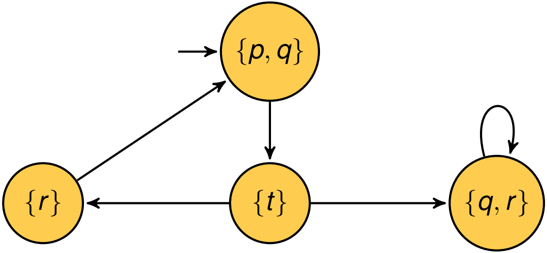
- Whenever p is true, r will eventually become true
  - $\{A_0A_1A_2\cdots \mid \forall i \geqslant 0, p \in A_i \to \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0A_1A_2\cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$
- Whenever r is true, so is q
  - $\{A_0A_1\cdots \mid \forall i \geqslant 0, r \in A_i \to q \in A_i\}$

Specifying Traces



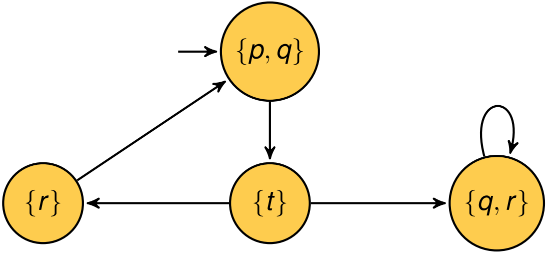
- It is never the case that p, r are true together

Specifying Traces



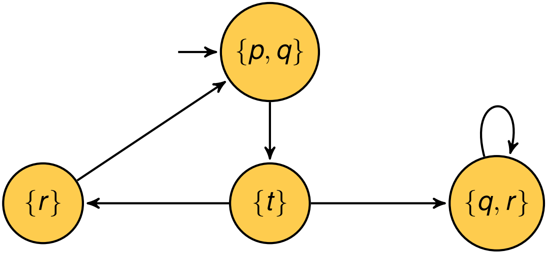
- It is never the case that p, r are true together
  - $\{A_0 A_1 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow r \notin A_i\}$

Specifying Traces



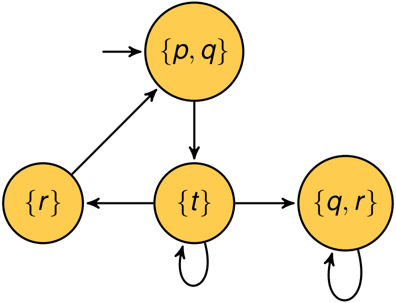
- It is never the case that p, r are true together
  - $\{A_0A_1 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow r \notin A_i\}$
- t and r are false until r becomes true

Specifying Traces



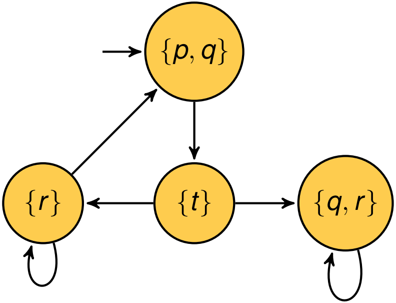
- It is never the case that p, r are true together
  - $\{A_0 A_1 \cdots \mid \forall i \geqslant 0, p \in A_i \rightarrow r \notin A_i\}$
- t and r are false until r becomes true
  - $\{A_0 A_1 \cdots \mid \exists i \geqslant 0, r \in A_i, \text{ and } \forall j < i, t \notin A_j \wedge r \notin A_j\}$

Safety Properties



- P=Whenever p is true, r is true within the next 5 steps.
- This property is violated by the bad prefix $\{p, q\}\{t\}^6$
- Safety properties = Nothing bad happens

Liveness Properties



- P=r is seen infinitely often.
- This property is satisfied by all traces
- Liveness properties = Something good happens again and again

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)
    - $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$

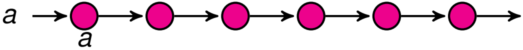
Syntax of Linear Temporal Logic

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)
  - φ Uψ (φ holds until a ψ-state is reached)
- LTL : Logic for describing LT properties

Semantics

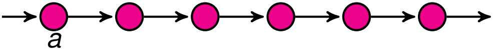
LTL formulae over *AP* interpreted over words over Σ^ω, $\Sigma = 2^{AP}$



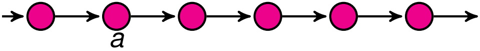
Semantics

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

a

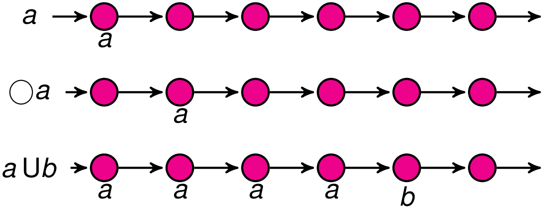


$\bigcirc a$



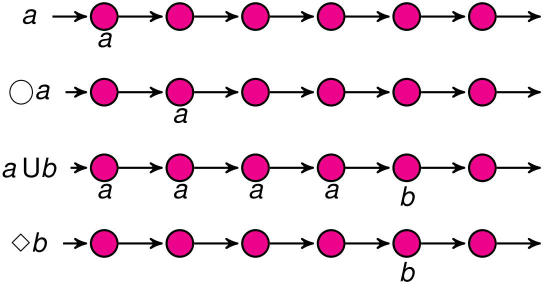
Semantics

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



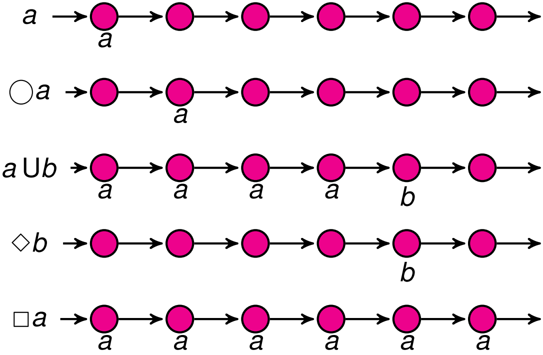
Semantics

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 $\wedge, \vee, \rightarrow$
  - $a \vee b\,\mathsf{U}c \equiv a \vee (b\,\mathsf{U}c)$
  - $\bigcirc a\,\mathsf{U}\neg b \equiv (\bigcirc a)\,\mathsf{U}(\neg b)$