ECE/CS 584: Verification of Embedded Computing Systems

Lecture 02

Sayan Mitra

Propositional Logic Summary

- **Syntax** (rules for constructing well formed sentences)
  - Countable set of (atomic) propositions PS: P1, P2, P3, …
  - S = $\mathrm{True} \mid p_1 \mid \neg S_1 \mid S_1 \wedge S_2$ | (S_1)
- **Semantics** defines a truth value functions or **valuations** v that maps each proposition PS to a truth value (T or F), v: PS→ {T, F} and by extension a valuation v':PROPS→{T,F}
- A proposition A is **valid** v'(A) = T for all valuations v. A is also called a **tautology**
- A proposition is **satisfiable** if there is a valuation (or truth assignment) v such that v(A) = T.
- Checking (un)satisfiability is called **boolean satisfiability problem** (SAT).
- SAT is (decidable) **NP-complete** problem

Predicate Logic or First Order Logic

- Syntax defined by a signature of **predicate** & **function** symbols
  - Variables
  - Predicate symbols with some **valence** or **arity**
    - a is predicate of 0-arity, like propositions
    - P(x) is a predicate of 1-arity
    - Q(x,y) is a predicate of 2-arity
  - Function symbols of some **valence**,
    - Function symbols of 0 arity are called constants
    - f(x) is a function of arity 1, e.g., -x
  - A **term** t ::= x | f(t1,t2,t3,…), where t1, t2, t3, … are terms
  - A **formula** φ ::= a | P(x) | Q(x,y) | t1 = t2 | $\neg\varphi$ | (φ1 ⇒ φ2) | … | … | $\forall x\varphi$ | $\exists x\varphi$
- Example of **Well Formed Formula**
  - $\exists x\, P(x)$, ∀x∀y(E(x, y) ⇒ E(y, x)), $\forall x\ y\ Q\big(x, f(y)\big) \equiv Q(f(y), x)$
- **Bounded** and **unbounded variables, closed formulas**

Existential

Universal

$\forall x \neg (f(x) = f(y))$

$\boxed{x = y}$

$f(f(x), y)$

Semantics

- An **interpretation or a model M** of a FOL formula assigns meaning to all the non-logical symbols and a domain for the variables (i.e., the variables, the predicate symbols, and the function symbols)
  - D: Domain of discourse
  - For each variable x, a valuation v(x) gives a value in D
  - Each function symbol f of arity n is assigned a function $D^n \rightarrow D$
  - Each predicate symbol P of atity n is assigned a predicate $D^n \rightarrow \{T, F\}$
- If formula φ evaluates to T with model M, then we say M **satisfies** φ, M $\vDash \varphi$ and φ is said to be **satisfiable**
- φ is **valid** if it is true for every interpretation

$\forall x\,(f(x) = g(x)) \wedge \neg(g(x) \wedge f(x) \ldots$

Example (Un)Decidable Classes

	Prefix	# of n-ary predicate symbols	# of n-ary function symbols	With Equality	Name
Undecidable	$\forall\exists\forall$	ω, 1	0	N	Kahr 1962
	$\forall^3\exists$	ω, 1	0	N	Suranyi 1959
	$\forall^*\exists$	0,1	0	N	Kalmar-Suranyi 1950
	$\forall\exists\forall\exists^*$	0,1	0	N	Gurevich 1966
	$\forall$	0	2	Y	Gurevich 1976
	$\forall$	0	0, 1	Y	Gurevich 1976
	$\forall^2\exists$	ω, 1	0	Y	Goldfarb 1984
Decidable	$\exists^*\forall^*$	all	0	Y	Ramsey 1930
	$\exists^*\forall\exists^*$	all	all	N	Maslov-Orevkov 1972
	$\exists^*$	all	all	Y	Gurevich 1976
	all	ω	ω	N	Lob 1967

Presberger Arithmetic **[1929]**

- First order theory of natural numbers with addition (no multiplication)
- Signature: Two constants 0 and 1, and a binary function +
- Axioms:
  - ~(0 = x + 1)
  - x + 1 = y + 1 ⟹ x = y
  - x + 0 = x
  - (x + y) + 1 = x + (y + 1)
  - (Infinity Axiom) For any first order formula P(x), P(0) ∧ ($\forall x\ P(x) \Rightarrow P(x+1)$) $\Longrightarrow$ $\forall y\ P(y)$
- Example: $\forall x\ \exists\ y\ ((y + y = x)\ \vee (y + y + 1 = x)$
- Cannot formalize divisibility or prime numbers
- Consistent: For any A, if A can be deduced from the axioms then ~A cannot be
- Complete: For any A, either A can be deduced or ~A can be deduced
- Decidable: There is an algorithm which decides for any A, whether A is true or ~A is true
  - Complexity $O(2^{2^{cn}})$, n : length of the formula c is some consant **[Fischer & Rabin 1974]**
  - 1954 Martin Davis implemented Presberger's decision procedure on "Johnniac" at IAS

Theory of Time Input/Output Automata

Lecture 02

Sayan Mitra

Roadmap

- Syntax
- Semantics
- Abstraction, Implementation
- Simulations
- Composition
- Substitutivity

v

Variables and Valuations

- A variable *x* is a name for a state component
- *type(x)*
- A set of variables X
- A valuation for X maps each x ∈ X to an element in type(x)
- $val(X)$: set of all valuations of X

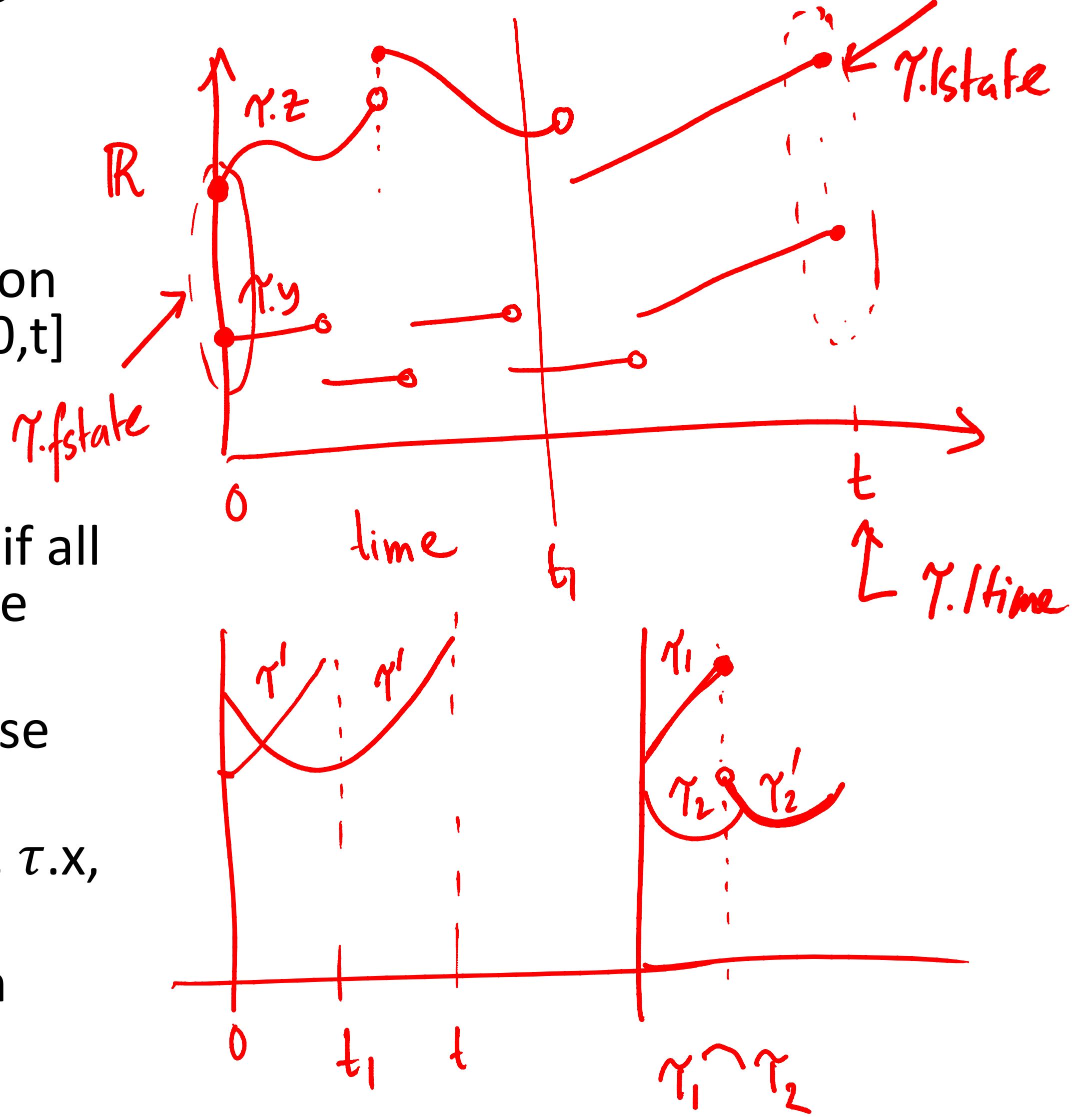
$\mathbf{x} \in val(X)$

- x:$\mathbb{R}$
- color:{R,G,B}
- clock:$\mathbb{R}^{\geq 0}$
- X = {x,color,clock}
- **x** = <x → 5.5, color →G, clock → 12>
- **y** = <x → 7.90, color →G, clock → 1>
- **x.**color = G, **x.**x = 5.5, **y.**x = 7.90

$X = \{z, y\}$ type(y) = $\mathbb{R}$

Trajectories

- Time = $\mathbb{R}^{\geq 0}$
- Time interval = [a,b]
- A trajectory for X is a function $\tau: [0, t] \rightarrow val(X)$, where [0,t] is an interval
- $\tau.dom = [0, t]$
- x is **continuous (or analog)** if all its trajectories are piecewise continuous
- **Discrete** if they are piecewise constant
- Notations: τ.fstate, τ.lstate, τ.x, τ.X
- Prefix, suffix, concatenation



Hybrid Automata (a.k.a Timed Automata Kaynar, et al. 2005)

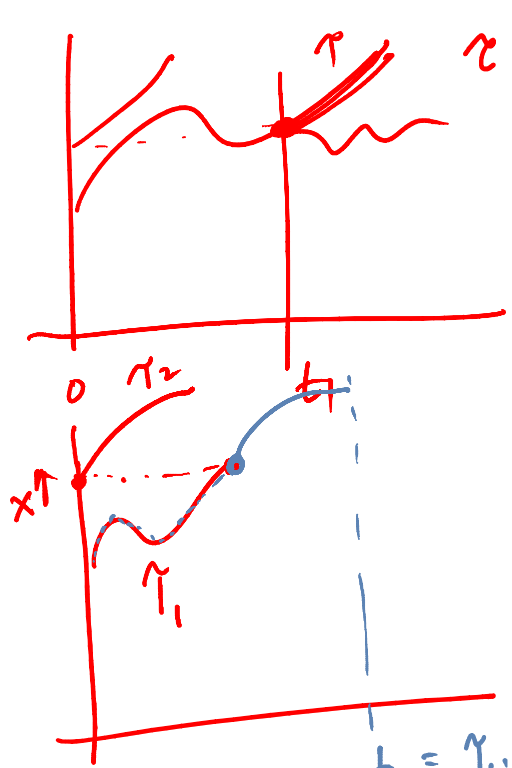
$\mathcal{A} = (X, Q, \Theta, E, H, \mathcal{D}, \mathcal{T})$

$(x, a, x') \in \mathcal{D}$ $x, x' \in Q$ $a \in A$

$x \xrightarrow{a} x'$

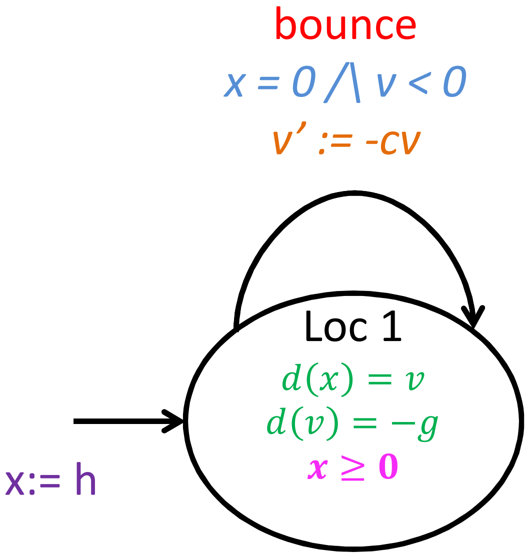
- X: set of internal variables
- $Q \subseteq val(X)$ set of states
- $\Theta \subseteq Q$ set of start states
- *E,H* sets of internal and external actions, A= E ∪ H
- $\mathcal{D} \subseteq Q \times A \times Q$ transitions
- $\mathcal{T}$*:* set of trajectories for X which is closed under prefix, suffix, and concatenation

$\tau \in \mathcal{T}$



$t = \tau_1.ltime + \tau_2.ltime$

Bouncing Ball



initial value

A

Automaton Bouncingball(c,h,g)

X **variables: analog** x: Reals := h, v: Reals := 0

S **states:** ~~True~~ $\mathbf{x}.x \geq 0$

actions: external bounce

transitions:

bounce

pre $x = 0 \wedge v < 0$ assign

eff $v := -cv$

$$\mathcal{D} = \{(\mathbf{x}, a, \mathbf{x}') \mid \mathbf{x}.x = 0 \wedge \mathbf{x}.v < 0 \wedge a = \text{bounce} \wedge \mathbf{x}'.v = -c(\mathbf{x}.v)\}$$

trajectories:

evolve d(x) = v; d(v) = -g

invariant $x \geq 0$

Graphical Representation used in many articles

TIOA Specification Language (close to PHAVer & UPPAAL's language)

Trajectory Semantics

$\dot{x} = f(x,u)$

$d(x) = v$

$d(v) = -g$

inv $x \geq 0$

$\mathcal{T}$ for $X = \{x, v\}$

$\tau \in \mathcal{T}$ iff (1) $\forall t \in \tau.dom$ $\quad \tau(t).v = \tau(0).v + \int_0^t -g \, ds$

$\tau(t).x = \tau(0).x + \int_0^t \tau(s).v \, ds$

(2) $\forall t \in \tau.dom$ $\quad \tau(t).x \geq 0$