CS477 Formal Software Dev Methods

Elsa L Gunter
2112 SC, UIUC
egunter@illinois.edu
http://courses.engr.illinois.edu/cs477

Slides based in part on previous lectures
by Mahesh Vishwanathan, and by Gul Agha

March 30, 2018

Labeled Transition System (LTS)

A labeled tranistion system (LTS) is a 4-tuple (Q, Σ, δ, I) where

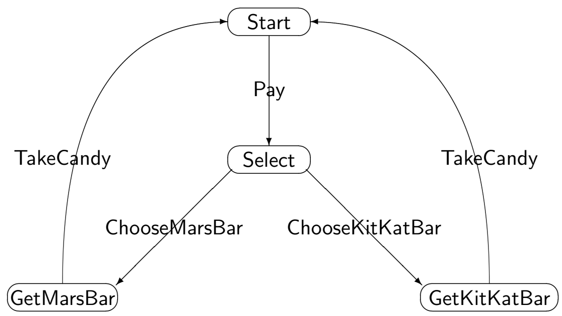
- Q set of states
  - Q finite or countably infinite
- Σ set of labels (aka actions)
  - Σ finite or countably infinite
- $\delta \subseteq Q \times \Sigma \times Q$ transition relation
- $I \subseteq Q$ initial states

Note: Write $q \xrightarrow{\alpha} q'$ for $(q, \alpha, q') \in \delta$.

Example: Candy Machine

- $Q = \{\text{Start}, \text{Select}, \text{GetMarsBar}, \text{GetKitKatBar}\}$
- $I = \{\text{Start}\}$
- $\Sigma = \{\text{Pay}, \text{ChooseMarsBar}, \text{ChooseKitKatBar}, \text{TakeCandy}\}$
- $\delta = \left\{ \begin{array}{l} (\text{Start}, \text{Pay}, \text{Select}) \\ (\text{Select}, \text{ChooseMarsBar}, \text{GetMarsBar}) \\ (\text{Select}, \text{ChooseKitKatBar}, \text{GetKitKatBar}) \\ (\text{GetMarsBar}, \text{TakeCandy}, \text{Start}) \\ (\text{GetKitKatBar}, \text{TakeCandy}, \text{Start}) \end{array} \right\}$

Example: Candy Machine



Predecessors, Successors and Determinism

Let (Q, Σ, δ, I) be a labeled transition system.

$$In(q, \alpha) = \{q' | q' \xrightarrow{\alpha} q\} \qquad In(q) = \bigcup_{\alpha\in\Sigma} In(q, \alpha)$$
$$Out(q, \alpha) = \{q' | q \xrightarrow{\alpha} q'\} \quad Out(q) = \bigcup_{\alpha\in\Sigma} Out(q, \alpha)$$

A labeled tranistion system (Q, Σ, δ, I) is deterministic if

$$|I| \leq 1 \text{ and } |Out(q, \alpha)| \leq 1$$

Labeled Transition Systems vs Finite State Automata

- LTS have no accepting states
  - Every FSA an LTS - just forget the accepting states
- Set of states and actions may be countably infinite
- May have infinite branching

Executions, Traces, and Runs

- A partial execution in an LTS is a finite or infinite alternating sequence of states and actions $\rho = q_0\alpha_1 q_1 \dots \alpha_n q_n \dots$ such that
  - $q_0 \in I$
  - $q_{i-1} \xrightarrow{\alpha_i} q_i$ for all i with q_i in sequence
- An execution is a maxial partial execution
- A finite or infinite sequence of actions $\alpha_1 \dots \alpha_n \dots$ is a trace if there exist states $q_0 \dots q_n \dots$ such that the sequence $q_0\alpha_1 q_1 \dots \alpha_n q_n \dots$ is a partial execution.
  - Let $\rho = q_0\alpha_1 q_1 \dots \alpha_n q_n \dots$ be a partial execution. Then $trace(\rho) = \alpha_1 \dots \alpha_n \dots$.

  A finite or inifnite sequence of states $q_0 \dots q_n \dots$ is a run if there exist actions $\alpha_1 \dots \alpha_n \dots$ such that the sequence $q_0\alpha_1 q_1 \dots \alpha_n q_n \dots$ is a partial execution.
  - Let $\rho = q_0\alpha_1 q_1 \dots \alpha_n q_n \dots$ be a partial execution. Then $run(\rho) = q_0 \dots q_n \dots$.

Example: Candy Machine

- Partial execution:
  $\rho = Start \cdot Pay \cdot Select \cdot ChooseMarsBar \cdot GetMarsBar \cdot TakeCandy \cdot Start$
- Trace: $trace(\rho) = Pay \cdot ChooseMarsBar \cdot TakeCandy$
- Run: $run(\rho) = Start \cdot Select \cdot GetMarsBar \cdot Start$

Program Transition System

A Program Transition System is a triple $(\mathcal{S}, T, init)$

- $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{F}, \phi, \mathcal{R}, \rho)$ is a first-order structure over signature $\mathcal{G} = (V, F, af, R, ar)$, used to interpret expressions and conditionals
- T is a finite set of conditional transitions of the form

$$g \rightarrow (v_1, \dots, v_n) := (e_1, \dots, e_n)$$

  where $v_i \in V$ distinct, and e_i term in $\mathcal{G}$, for $i = 1 \dots n$
- $init$ initial condition asserted to be true at start of program

Example: Traffic Light

$V = \{Turn, NSC, EWC\}$, $F = \{NS, EW, Red, Yellow, Green\}$ (all arity 0), $R = \{=\}$

NSG	$Turn = NS \wedge NSC = Red \rightarrow NSC := Green$
NSY	$Turn = NS \wedge NSC = Green \rightarrow NSC := Yellow$
NSR	$Turn = NS \wedge NSC = Yellow \rightarrow (Turn, NSC) := (EW, Red)$
EWG	$Turn = EW \wedge EWC = Red \rightarrow EWC := Green$
EWY	$Turn = EW \wedge EWC = Green \rightarrow EWC := Yellow$
EWR	$Turn = EW \wedge EWC = Yellow \rightarrow (Turn, EWC) := (NS, Red)$

$init = (NSC = Red \wedge EWC = Red \wedge (Turn = NS \vee Turn = EW)$

Mutual Exclusion (Attempt)

P1 ::		P2 ::	
$m1$:	while true do	$n1$:	while true do
$m2$:	$p11(*not\ in\ crit\ sect*)$	$n2$:	$p21(*not\ in\ crit\ sect*)$
$m3$:	$c1 := 0$	$n3$:	$c2 := 0$
$m4$:	$wait(c2 = 1)$	$n4$:	$wait(c1 = 1)$
$m5$:	$r1(*in\ crit\ sect*)$	$n5$:	$r2(*in\ crit\ sect*)$
$m6$:	$c1 := 1$	$n6$:	$c2 := 1$
$m7$:	od	$n7$:	od

Mutual Exclusion PTS

$V = \{pc1, pc2, c1, c2\}$, $F = \{m1, \dots, m6, n1, \dots, n6, 0, 1\}$

$$
\begin{aligned}
T = \quad & pc1 = m1 \rightarrow pc1 := m2 \\
& pc1 = m2 \rightarrow pc1 := m3 \\
& pc1 = m3 \rightarrow (pc1, c1) := (m4, 0) \\
& pc1 = m4 \wedge c2 = 1 \ \ to \ \ pc1 := m5 \\
& pc1 = m5 \rightarrow pc1 := m6 \\
& pc1 = m6 \rightarrow (pc1, c1) := (m1, 1) \\
& pc2 = n1 \rightarrow pc2 := n2 \\
& pc2 = n2 \rightarrow pc2 := n3 \\
& pc2 = n3 \rightarrow (pc2, c2) := (n4, 0) \\
& pc2 = n4 \wedge c1 = 1 \ \ to \ \ pc2 := n5 \\
& pc2 = n5 \rightarrow pc2 := n6 \\
& pc2 = n6 \rightarrow (pc2, c2) := (n1, 1)
\end{aligned}
$$

$init = (pc1 = m1 \wedge pc2 = n1 \wedge c1 = 1 \wedge c2 = 1)$

Interpreting PTS as LTS

Let $(\mathcal{S}, T, init)$ be a program transition system. Assume V finite, $\mathcal{D}$ at most countable.

- Let $Q = V \to \mathcal{D}$, interpretted as all assingments of values to variables
  - Can restrict to mappings q where v and $q(v)$ have same type
- Let $\Sigma = T$
- Let $\delta = \{(q, g \to (v_1, \ldots, v_n) := (e_1, \ldots, e_n), q') \mid$
  $\mathcal{M}_q(g) \wedge$
  $(\forall i \leq n. q'(v_i) = \mathcal{T}_q(e_i)) \wedge$
  $(\forall v \notin \{v_1, \ldots, v_n\}. \; q'(v) = q(v))\}$
- $I = \{q \mid \mathcal{T}_q(init) = \mathbf{T}\}$

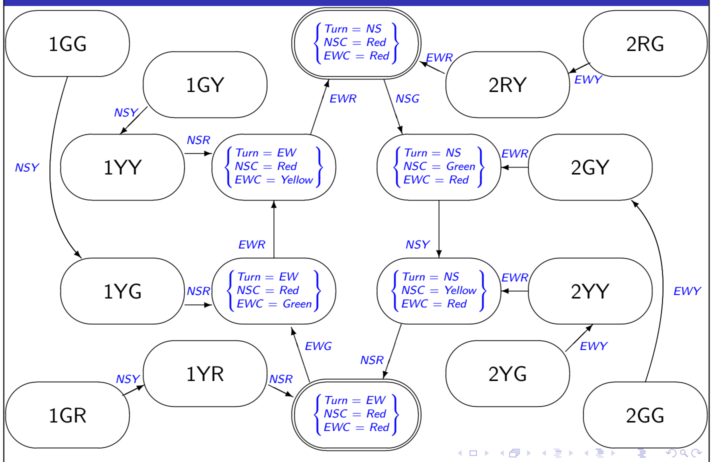
Example: Traffic Light

$V = \{Turn, NSC, EWC\}$, $F = \{NS, EW, Red, Yellow, Green\}$ (all arity 0), $R = \{=\}$

NSG	$Turn = NS \wedge NSC = Red \to NSC := Green$
NSY	$NSC = Green \to NSC := Yellow$
NSR	$NSC = Yellow \to (Turn, NSC) := (EW, Red)$
EWG	$Turn = EW \wedge EWC = Red \to EWC := Green$
EWY	$EWC = Green \to EWC := Yellow$
EWR	$EWC = Yellow \to (Turn, EWC) := (NS, Red)$

$init = (NSC = Red \wedge EWC = Red \wedge (Turn = NS \vee Turn = EW)$

Example: Traffic Lights



Examples (cont)

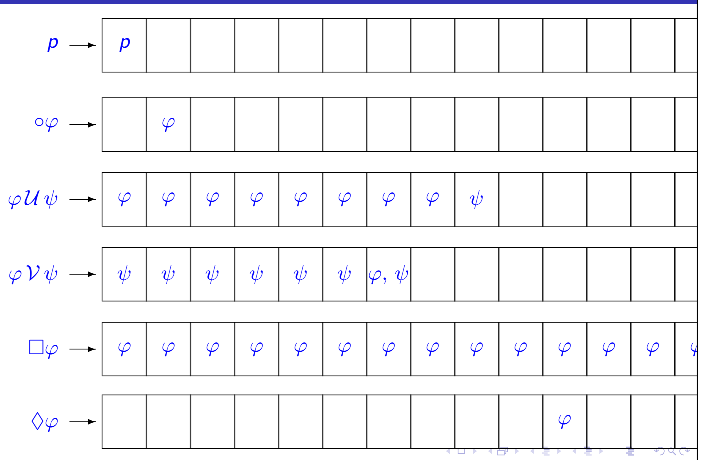
- LTS for traffic light has $3 \times 3 \times 2 = 18$ possible well typed states
  - Is is possible to reach a state where $NSC \neq Red \wedge EWC \neq Red$ from an initial state?
  - If so, what sequence of actions allows this?
  - Do all the immediate predecessors of a state where $NSC = Green \vee EWC = Green$ satisfy $NSC = Red \wedge EWC = Red$?
  - If not, are any of those offend states reachable from and initial state, and if so, how?
- LTS for Mutual Exclusion has $6 \times 6 \times 2 \times 2 = 144$ posible well-tped states.
  - Is is possible to reach a state where $pc1 = m5 \wedge pc2 = n5$?
- How can we state these questions rigorously, formally?
- Can we find an algorithm to answer these types of questions?

Linear Temporal Logic - Syntax

$$\begin{aligned}\varphi \; ::= \; & p \mid (\varphi) \mid \not\varphi \mid \varphi \wedge \varphi' \mid \varphi \vee \varphi' \\ \mid \; & \circ\varphi \mid \varphi\mathcal{U}\varphi' \mid \varphi\mathcal{V}\varphi' \mid \Box\varphi \mid \Diamond\varphi\end{aligned}$$

- p – a propostion over state variables
- $\circ\varphi$ – "next"
- $\varphi\mathcal{U}\varphi'$ – "until"
- $\varphi\mathcal{V}\varphi'$ – "releases"
- $\Box\varphi$ – "box", "always", "forever"
- $\Diamond\varphi$ – "diamond", "eventually", "sometime"

LTL Semantics: The Idea



Formal LTL Semantics

Given:

- $\mathcal{G} = (V, F, af, R, ar)$ signature expressing state propositions
- Q set of states,
- $\mathcal{M}$ modeling function over Q and $\mathcal{G}$: $\mathcal{M}(q, p)$ is true iff q models p. Write $q \models p$.
- $\sigma = q_0 q_1 \ldots q_n \ldots$ infinite sequence of state from Q.
- $\sigma^i = q_i q_{i+1} \ldots q_n \ldots$ the i^{th} tail of σ

Say σ models LTL formula φ, write $\sigma \models \varphi$ as follows:

- $\sigma \models p$ iff $q_0 \models p$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$
- $\sigma \models \varphi \wedge \psi$ iff $\sigma \models \varphi$ and $\sigma \models \psi$.
- $\sigma \models \varphi \vee \psi$ iff $\sigma \models \varphi$ or $\sigma \models \psi$.

Formal LTL Semantics

- $\sigma \models \circ\varphi$ iff $\sigma^1 \models \varphi$
- $\sigma \models \varphi \mathcal{U} \psi$ iff for some k, $\sigma^k \models \psi$ and for all $i < k$, $\sigma^i \models \varphi$
- $\sigma \models \varphi \mathcal{V} \psi$ iff for some k, $\sigma^k \models \varphi$ and for all $i \leq k$, $\sigma^i \models \psi$, or for all i, $\sigma^i \models \psi$.
- $\sigma \models \Box\varphi$ if for all i, $\sigma^i \models \psi$
- $\sigma \models \Diamond\varphi$ if for some i, $\sigma^i \models \psi$

Some Common Combinations

- $\Box\Diamond p$ "p will hold infinitely often"
- $\Diamond\Box p$ "p will continuously hold from some point on"
- $(\Box p) \Rightarrow (\Box q)$ "if p happens infinitely often, then so does q

Some Equivalences

- $\Box(\varphi \wedge \psi) = (\Box\varphi) \wedge (\Box\psi)$
- $\Diamond(\varphi \vee \psi) = (\Diamond\varphi) \vee (\Diamond\psi)$
- $\Box\varphi = \mathbf{F}\, \mathcal{V}\, \varphi$
- $\Diamond\varphi = \mathbf{T}\, \mathcal{U}\, \varphi$
- $\varphi\, \mathcal{V}\, \psi = \neg((\neg\varphi)\, \mathcal{U}\, (\neg\psi))$
- $\varphi\, \mathcal{U}\, \psi = \neg((\neg\varphi)\, \mathcal{V}\, (\neg\psi))$
- $\neg(\Diamond\varphi) = \Box(\neg\varphi)$
- $\neg(\Box\varphi) = \Diamond(\neg\varphi)$

Some More Eqivalences

- $\Box\varphi = \varphi \wedge \circ\Box\varphi$
- $\Diamond\varphi = \varphi \vee \circ\Diamond\varphi$
- $\varphi\, \mathcal{V}\, \psi = (\varphi \wedge \psi) \vee (\psi \wedge \circ(\varphi\, \mathcal{V}\, \psi))$
- $\varphi\, \mathcal{U}\, \psi = \psi \vee (\varphi \wedge \circ(\varphi\, \mathcal{V}\, \psi))$
- $\Box$, $\Diamond$, $\mathcal{U}$, $\mathcal{V}$ may all be understood recursively, by what they state about right now, and what they state about the future
- Caution: $\Box$ vs $\Diamond$, $\mathcal{U}$ vs $\mathcal{V}$ differ in there limit behavior

Traffic Light Example

Basic Behavior:

- $\Box((NSC = Red) \vee (NSC = Green) \vee (NSC = Yellow))$
- $\Box((NSC = Red) \Rightarrow ((NSC \neq Green) \wedge (NSC \neq Yellow))$
- Similarly for *Green* and *Red*
- $\Box(((NCS = Red) \wedge \circ(NCS \neq Red)) \Rightarrow \circ(NCS = Green))$
- Same as $\Box((NCS = Red) \Rightarrow ((NCS = Red)\, \mathcal{U}\, (NCS = Green)))$
- $\Box(((NCS = Green) \wedge \circ(NCS \neq Green)) \Rightarrow \circ(NCS = Yellow))$
- $\Box(((NCS = Yellow) \wedge \circ(NCS \neq Yellow)) \Rightarrow \circ(NCS = Red))$
- Same for *EWC*

Traffic Light Example

Basic Safety

- $\Box((NSC = Red) \vee (EWC = Red)$
- $\Box(\ ((NSC = Red) \wedge (EWC = Red))\,\mathcal{V}$
  $((NSC \neq Green) \Rightarrow (\circ(NSC = Green))))$

Basic Liveness

- $(\Diamond(NSC = Red)) \wedge (\Diamond(NSC = Green)) \wedge (\Diamond(NSC = Yellow))$
- $(\Diamond(EWC = Red)) \wedge (\Diamond(EWC = Green)) \wedge (\Diamond(EWC = Yellow))$

Proof System for LTL

- First step: View $\varphi\,\mathcal{V}\,\psi$ as moacro: $\varphi\,\mathcal{V}\,\psi = \neg((\neg\varphi)\,\mathcal{U}\,(\neg\psi))$
- Second Step: Extend all rules of Prop Logic to LTL
- Third Step: Add one more rule: $\dfrac{\varphi}{\Box\varphi}$ Gen
- Fourth Step: Add a collection of axioms (a sufficient set of 8 exists)
  - A1: $\Box\varphi \Leftrightarrow \neg(\Diamond(\neg\varphi))$
  - A2: $\Box(\varphi \Rightarrow \psi) \Rightarrow (\Box\varphi \Rightarrow \Box\psi)$
  - A3: $\Box\varphi \Rightarrow (\varphi \wedge \circ\Box\varphi)$
  - A4: $\circ\neg\varphi \Leftrightarrow \neg\circ\varphi$
  - A5: $\circ(\varphi \Rightarrow \psi) \Rightarrow (\circ\varphi \Rightarrow \circ\psi)$
  - A6: $\Box(\varphi \Rightarrow \circ\varphi) \Rightarrow (\varphi \Rightarrow \Box\varphi)$
  - A7: $\varphi\,\mathcal{U}\,\psi \Leftrightarrow (\varphi \wedge \psi) \vee (\varphi \wedge \circ(\varphi\,\mathcal{V}\,\psi)$
  - A8: $\varphi\,\mathcal{U}\,\psi \Rightarrow \Diamond\psi$
- Result: a sound and relatively complete proof system
- Can implement in Isabelle in much the same way as we did Hoare Logic