Contents & Goals

Last Lecture:
• System configuration cont'd
• Action language and transformer

This Lecture:
• Educational Objectives:
  • What does this State Machine mean? What happens if I inject this event?
  • Can you please model the following behaviour.
  • What is: Signal, Event, Ether, Transformer, Step, RTC.

Content:
• Step, Run-to-Completion Step

Definition.
Let $A$ be a set of labels and $S$ a (not necessarily finite) set of states. We call $\rightarrow \subseteq S \times A \times S$ a (labelled) transition relation.

Let $S_0 \subseteq S$ be a set of initial states. A (finite or infinite) sequence $s_0 a_0 \rightarrow s_1 a_1 \rightarrow s_2 a_2 \rightarrow \ldots$ with $s_i \in S$, $a_i \in A$ is called computation of the labelled transition system $(S, A, \rightarrow, S_0)$ if and only if

• initiation: $s_0 \in S_0$
• consecution: $(s_i, a_i, s_{i+1}) \in \rightarrow$ for $i \in \mathbb{N}^0$.

Active vs. Passive Classes/Objects

• Note: From now on, for simplicity, assume that all classes are active.
  We'll later briefly discuss the Rhapsody framework which proposes a way how to integrate non-active objects.

• Note: The following RTC "algorithm" follows Harel and Gery (1997) (i.e. the one realised by the Rhapsody code generation) if the standard is ambiguous or leaves choices.

From Core State Machines to LTS

Definition.
Let $S_0 = (T_0, C_0, V_0, \text{atr}_0, E)$ be a signature with signals (all classes in $C_0$ active), $D_0$ a structure of $S_0$, and $(\mathcal{E}_0 \mathbin{\&} \text{ready}, \text{\&}, \text{\&}, \cdot)$ an ether over $S_0$ and $D_0$.

Assume there is one core state machine $M_C$ per class $C \in C_0$.

We say, the state machines induce the following labelled transition relation on states $S := \Sigma D \times \mathcal{E}_0 \cup \{\#\}$ with labels $A := 2(D \mathbin{\&} \mathcal{E}_0) \times 2(D \mathbin{\&} \mathcal{E}_0 \cup \{*, +\}) \times D(C) \times D(C)$:

• $((\sigma, \varepsilon), (\text{cons, Snd})) \rightarrow u((\sigma', \varepsilon'))$ if and only if
  (i) an event with destination $u$ is discarded,
  (ii) an event is dispatched to $u$, i.e. stable object processes an event, or
  (iii) run-to-completion processing by $u$ continues, i.e. object $u$ is not stable and continues to process an event,
  (iv) the environment interacts with object $u$.

• $((\text{cons, } \emptyset)) \rightarrow \#$ if and only if
  (v) an error condition occurs during consumption of $\text{cons}$, or
  $s = \#$ and $\text{cons} = \emptyset$. 

---

Define Core State Machines in LTS

---

Transition Relation, Computation

---

Active vs. Passive Classes/Objects

---

From Core State Machines to LTS

---

Transition Relation, Computation

---

Active vs. Passive Classes/Objects

---

From Core State Machines to LTS

---

Transition Relation, Computation

---

Active vs. Passive Classes/Objects

---

From Core State Machines to LTS

---
\(\sigma\)) \[Snd, u = (\sigma \land \{D | b, u.\text{stable}\}) = 1\]

\(\sigma, \epsilon (\sigma \uparrow \epsilon, u.(\sigma \land \epsilon))\]

where \(\epsilon\) is the empty expression, i.e., \(\epsilon = \emptyset\). Example: Discarding An Event

\(\)
