Software Design, Modelling and Analysis in UML

Lecture 13: Core State Machines III

2015-12-17

Prof. Dr. Andreas Podelski, **Dr. Bernd Westphal**

Albert-Ludwigs-Universität Freiburg, Germany

Contents & Goals

Last Lecture:

- System configuration cont'd
- Action language and transformer

This Lecture:

- **Educational Objectives:** Capabilities for following tasks/questions.
  - 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

Transition Relation

Transition Relation, Computation

Definition. Let A be a set of **labels** and S a (not necessarily finite) set of 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

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots$$

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 $\mathscr{S}_0 = (\mathscr{T}_0, \mathscr{C}_0, V_0, atr_0, \mathscr{E})$ be a signature with signals (all classes in $\mathscr{C}_0$ **active**), $\mathscr{D}_0$ a structure of $\mathscr{S}_0$, and $(Eth, ready, \oplus, \ominus, [\,\cdot\,])$ an ether over $\mathscr{S}_0$ and $\mathscr{D}_0$. Assume there is one core state machine M_C per class $C \in \mathscr{C}$.

dest

We say, the state machines induce the following labelled transition relation on states $S := (\Sigma_{\mathscr{S}}^{\mathscr{D}} \times Eth) \,\dot\cup\, \{\#\}$ with labels $A := \underbrace{2^{\mathscr{D}(\mathscr{E})}}_{\text{which sig. instance consumed}} \times \underbrace{2^{(\mathscr{D}(\mathscr{E}) \,\dot\cup\, \{*,+\}) \times \mathscr{D}(\mathscr{C})}}_{\text{observation, what has been sent out + cons/dests.}} \times \mathscr{D}(\mathscr{C})$:

error state

- $(\sigma, \varepsilon) \xrightarrow[u]{(cons, Snd)} (\sigma', \varepsilon')$

  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,

- $s \xrightarrow[u]{(cons, \emptyset)} \#$

  if and only if

  (v) an **error condition** occurs during consumption of $cons$, or
  ($s = \#$ and) $cons = \emptyset$.

(i) Discarding An Event

$$(\sigma, \varepsilon) \xrightarrow[u]{(cons, Snd)} (\sigma', \varepsilon')$$

if

condition on (σ, ε)

and

conditions on (σ', ε')

(i) Discarding An Event

$$(\sigma, \varepsilon) \xrightarrow[u]{(cons, Snd)} (\sigma', \varepsilon')$$

if

- an E-event (instance of signal E) is ready in ε for object u of a class $\mathscr{C}$, i.e. if

$$u \in \mathrm{dom}(\sigma) \cap \mathscr{D}(C) \wedge \exists\, u_E \in \mathscr{D}(E) : u_E \in \mathit{ready}(\varepsilon, u)$$

- u is stable and in state machine state s, i.e. $\sigma(u)(\mathit{stable}) = 1$ and $\sigma(u)(\mathit{st}) = s$,
- but there is no corresponding transition enabled (all transitions incident with current state of u either have other triggers or the guard is not satisfied)

$$\forall\, (s, F, \mathit{expr}, \mathit{act}, s') \in \to (\mathcal{SM}_C) : F \neq E \vee I[\![\mathit{expr}]\!](\sigma, u) = 0$$

and

update value of b of object u to b

- in the system configuration, stability may change, u_E goes away, i.e.

$$\sigma' = \sigma[u.\mathit{stable} \mapsto b] \setminus \{u_E \mapsto \sigma(u_E)\}$$

where $b = 0$ if and only if there is a transition **with trigger '_'** enabled for u in (σ', ε').

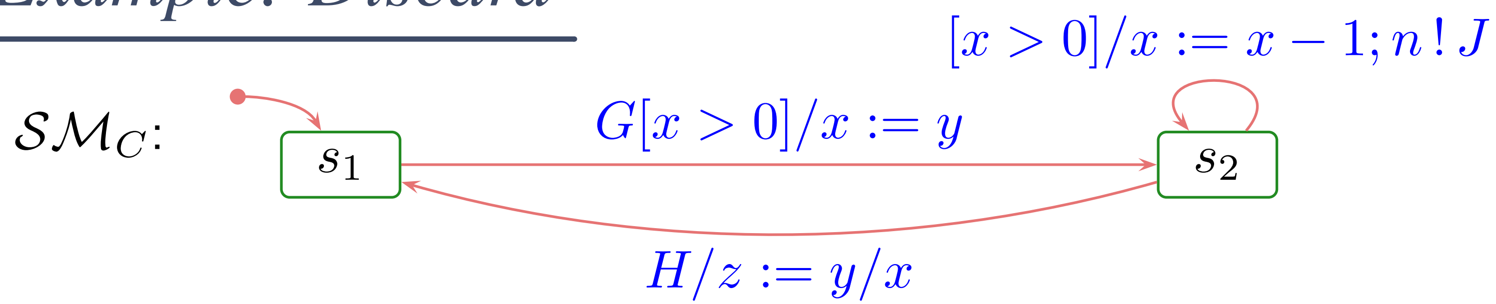
- the event u_E is removed from the ether, i.e.

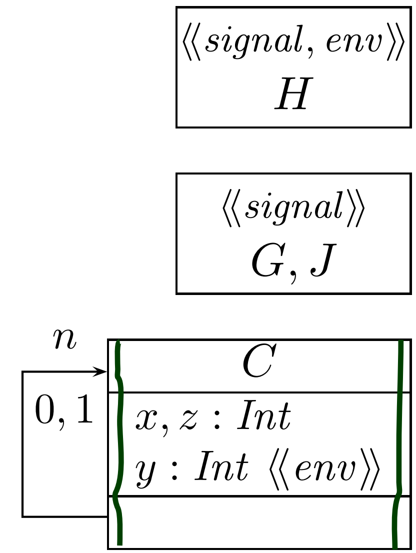
$$\varepsilon' = \varepsilon \ominus u_E,$$

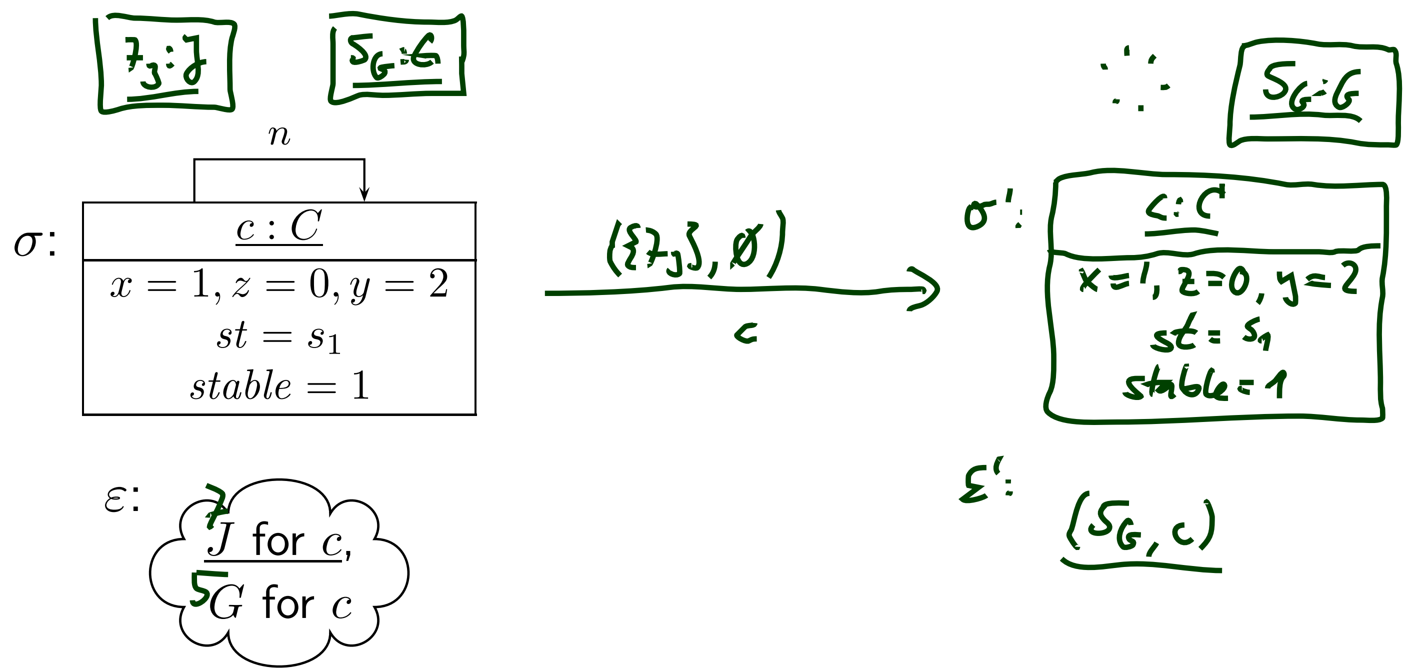
- consumption of u_E is observed, i.e.

$$\mathit{cons} = \{u_E\}, \quad \mathit{Snd} = \emptyset.$$

Example: Discard







- $u \in \text{dom}(\sigma) \cap \mathscr{D}(C)$ ✓
  $u_E \in \mathscr{D}(E), u_E \in ready(\varepsilon, u)$ ✓
- $\forall (\boxed{s}, F, expr, act, s') \in \rightarrow (\mathcal{SM}_C)$:
  $F \neq E \vee I[\![expr]\!](\sigma, u) = 0$ ✓
- $\sigma(u)(stable) = 1$ ✓, $\sigma(u)(st) = s$, ✓
- $\sigma' = \sigma[u.stable \mapsto b]$ ✓ $\setminus \{u_E \mapsto \sigma(u_E)\}$ ✓
- $\varepsilon' = \varepsilon \ominus u_E$
- $cons = \{u_E\}, \quad Snd = \emptyset$

(ii) Dispatch

$$(\sigma, \varepsilon) \xrightarrow[u]{(cons, Snd)} (\sigma', \varepsilon')$$

if

- $u \in \mathrm{dom}(\sigma) \cap \mathscr{D}(C) \wedge \exists\, u_E \in \mathscr{D}(E) : u_E \in \mathit{ready}(\varepsilon, u)$
- u is stable and in state machine state s, i.e. $\sigma(u)(\mathit{stable}) = 1$ and $\sigma(u)(st) = s$,
- a transition is **enabled**, i.e.

$$\exists\,(s, F, \mathit{expr}, \mathit{act}, s') \in \rightarrow(\mathcal{SM}_C) : F = E \wedge I[\![\mathit{expr}]\!](\tilde{\sigma}, u) = 1$$

  where $\tilde{\sigma} = \sigma[u.\mathit{params}_E \mapsto u_E]$.

and

- (σ', ε') results from applying t_{act} to (σ, ε) and removing u_E from the ether, i.e.

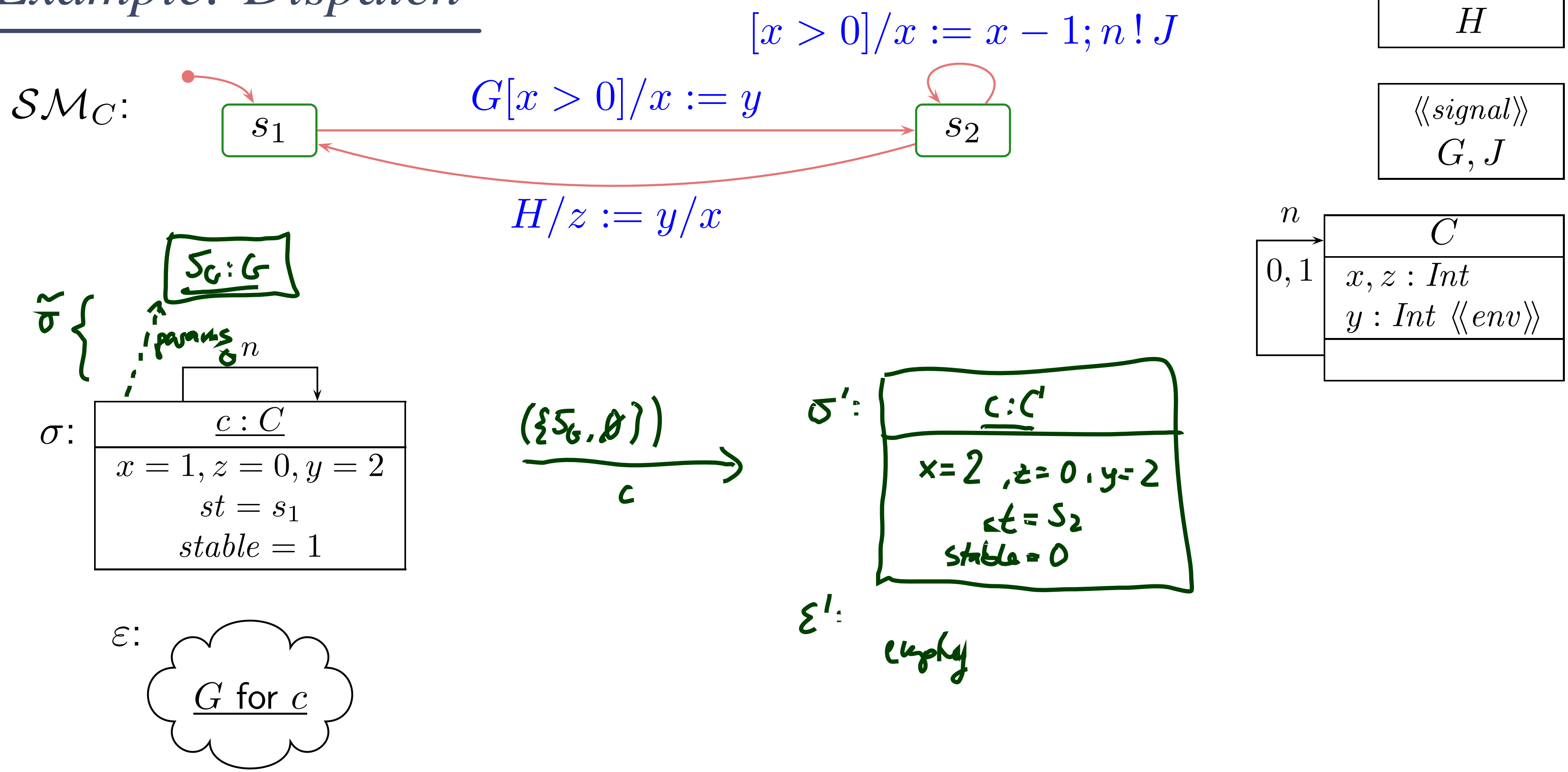
$$(\sigma'', \varepsilon') \in t_{act}[u](\tilde{\sigma}, \varepsilon \ominus u_E),$$
$$\sigma' = (\sigma''[u.st \mapsto s', u.\mathit{stable} \mapsto b, u.\mathit{params}_E \mapsto \emptyset])\overbrace{|_{\mathscr{D}(\mathscr{C}) \setminus \{u_E\}}}^{\text{remove } u_E}$$

  where b **depends** (see (i))

- Consumption of u_E and the side effects of the action are observed, i.e.

$$\mathit{cons} = \{u_E\}, \quad \mathit{Snd} = \mathit{Obs}_{t_{act}}[u](\tilde{\sigma}, \varepsilon \ominus u_E).$$

Example: Dispatch



- $u \in \mathrm{dom}(\sigma) \cap \mathscr{D}(C)$
  $u_E \in \mathscr{D}(E)$, $u_E \in ready(\varepsilon, u)$
- $\exists\,(s, F, expr, act, s') \in\to (\mathcal{SM}_C)$:
  $F = E \wedge I[\![expr]\!](\tilde{\sigma}, u) = 1$
- $\tilde{\sigma} = \sigma[u.params_E \mapsto u_E]$.
- $\sigma(u)(stable) = 1,\ \sigma(u)(st) = s,$
- $(\sigma'', \varepsilon') = t_{act}(\tilde{\sigma}, \varepsilon \ominus u_E)$
- $\sigma' = (\sigma''[u.st \mapsto s', u.stable \mapsto b, u.params_E \mapsto \emptyset])|_{\mathscr{D}(\mathscr{C})\setminus\{u_E\}}$
- $cons = \{u_E\}, \quad Snd = Obs_{t_{act}}[u](\tilde{\sigma}, \varepsilon \ominus u_E)$

(iii) Continue Run-to-Completion

$$(\sigma, \varepsilon) \xrightarrow[u]{(cons, Snd)} (\sigma', \varepsilon')$$

if

- there is an unstable object u of a class $\mathscr{C}$, i.e.

$$u \in \mathrm{dom}(\sigma) \cap \mathscr{D}(C) \wedge \sigma(u)(stable) = 0$$

- there is a transition without trigger enabled from the current state $s = \sigma(u)(st)$, i.e.

$$\exists\, (s, \_, expr, act, s') \in \to (\mathcal{SM}_C) : I[\![expr]\!](\sigma, u) = 1$$

and

- (σ', ε') results from applying t_{act} to (σ, ε), i.e.

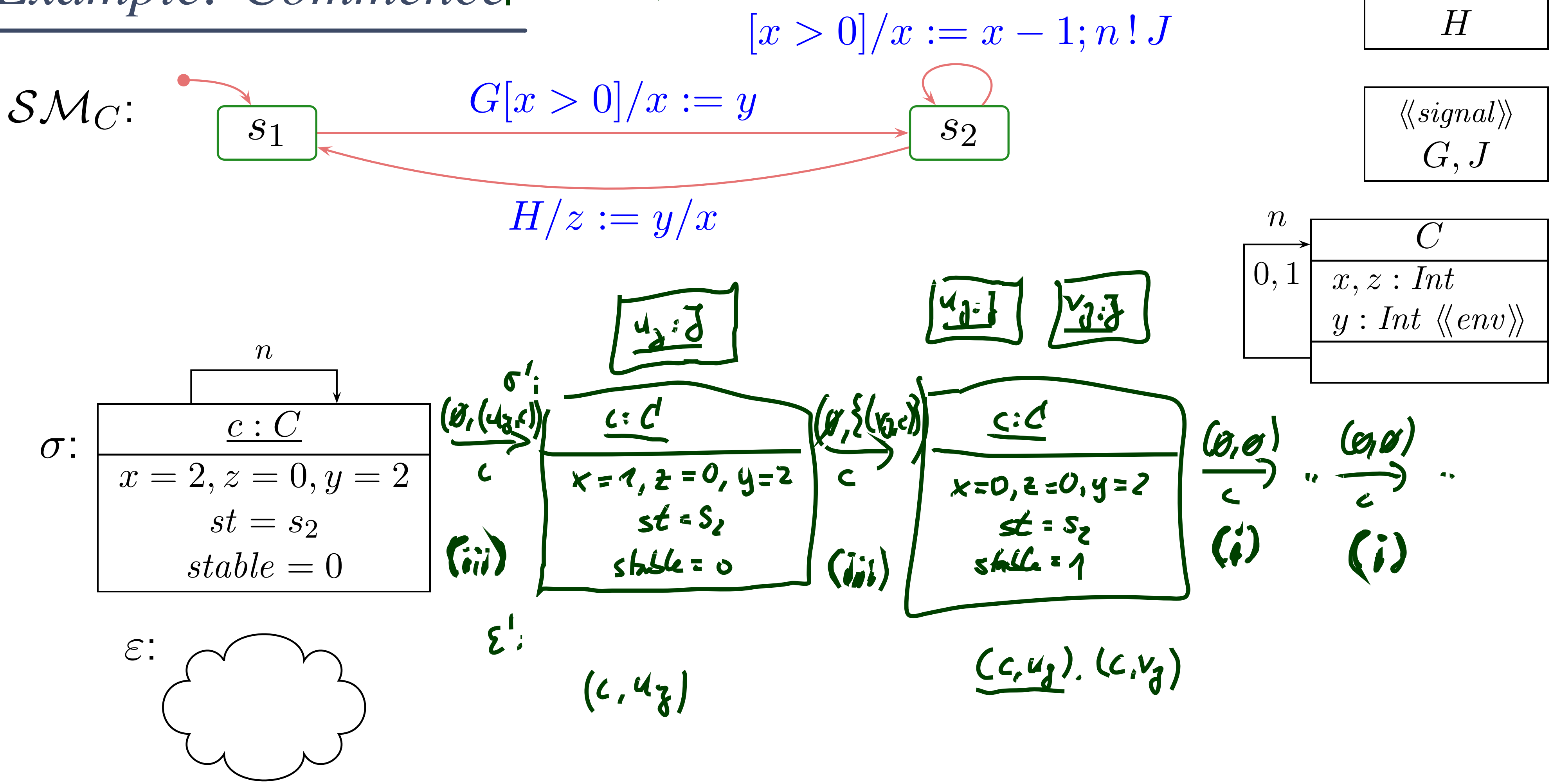
$$(\sigma'', \varepsilon') \in t_{act}[u](\sigma, \varepsilon), \quad \sigma' = \sigma''[u.st \mapsto s', u.stable \mapsto b]$$

  where b **depends** as before.

- Only the side effects of the action are observed, i.e.

$$cons = \emptyset, \quad Snd = Obs_{t_{act}}[u](\sigma, \varepsilon).$$

Example: ~~Commence~~ Continue



- $u \in \mathrm{dom}(\sigma) \cap \mathscr{D}(C), \sigma(u)(stable) = 0$ ✓
- $\exists\, (s, \_, expr, act, s') \in \rightarrow (\mathcal{SM}_C)$:
  $I[\![expr]\!](\sigma, u) = 1$ ✓
- $\sigma(u)(st) = s$ ✓
- $(\sigma'', \varepsilon') = t_{act}(\sigma, \varepsilon)$,
- $\sigma' = \sigma''[u.st \mapsto s', u.stable \mapsto b]$
- $cons = \emptyset$, $\quad Snd = Obs_{t_{act}}(\sigma, \varepsilon)$

(iv) Environment Interaction

Assume that a set $\mathscr{E}_{env} \subseteq \mathscr{E}$ is designated as **environment events** and a set of attributes $V_{env} \subseteq V$ is designated as **input attributes**.

Then

$$(\sigma, \varepsilon) \xrightarrow[env]{(cons, Snd)} (\sigma', \varepsilon')$$

if either (!)

- an environment event $E \in \mathscr{E}_{env}$ is spontaneously sent to an alive object $u \in \mathrm{dom}(\sigma)$, i.e.

$$\sigma' = \sigma \,\dot{\cup}\, \{u_E \mapsto \{v_i \mapsto d_i \mid 1 \leq i \leq n\}, \quad \varepsilon' = \varepsilon \oplus (u, u_E)$$

  where $u_E \notin \mathrm{dom}(\sigma)$ and $atr(E) = \{v_1, \dots, v_n\}$.
- Sending of the event is observed, i.e. $cons = \emptyset$, $Snd = \{u_E,)\}$.

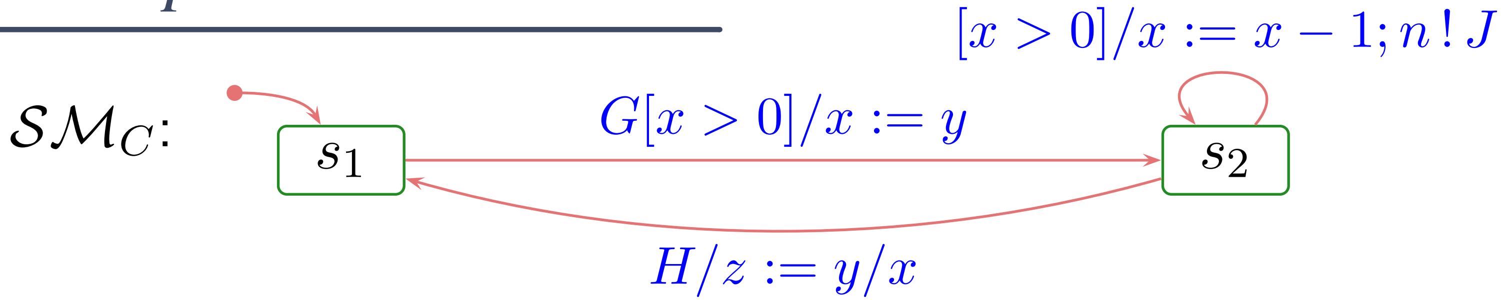
or

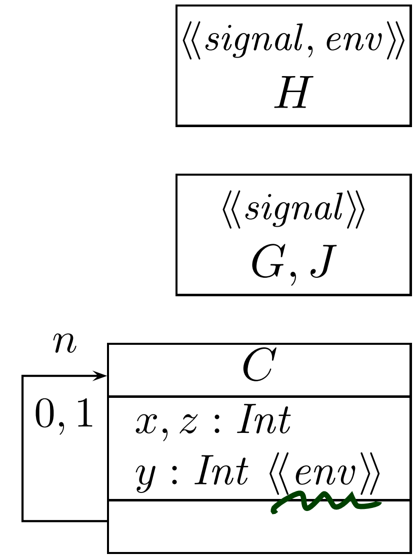
- Values of input attributes change freely in alive objects, i.e.

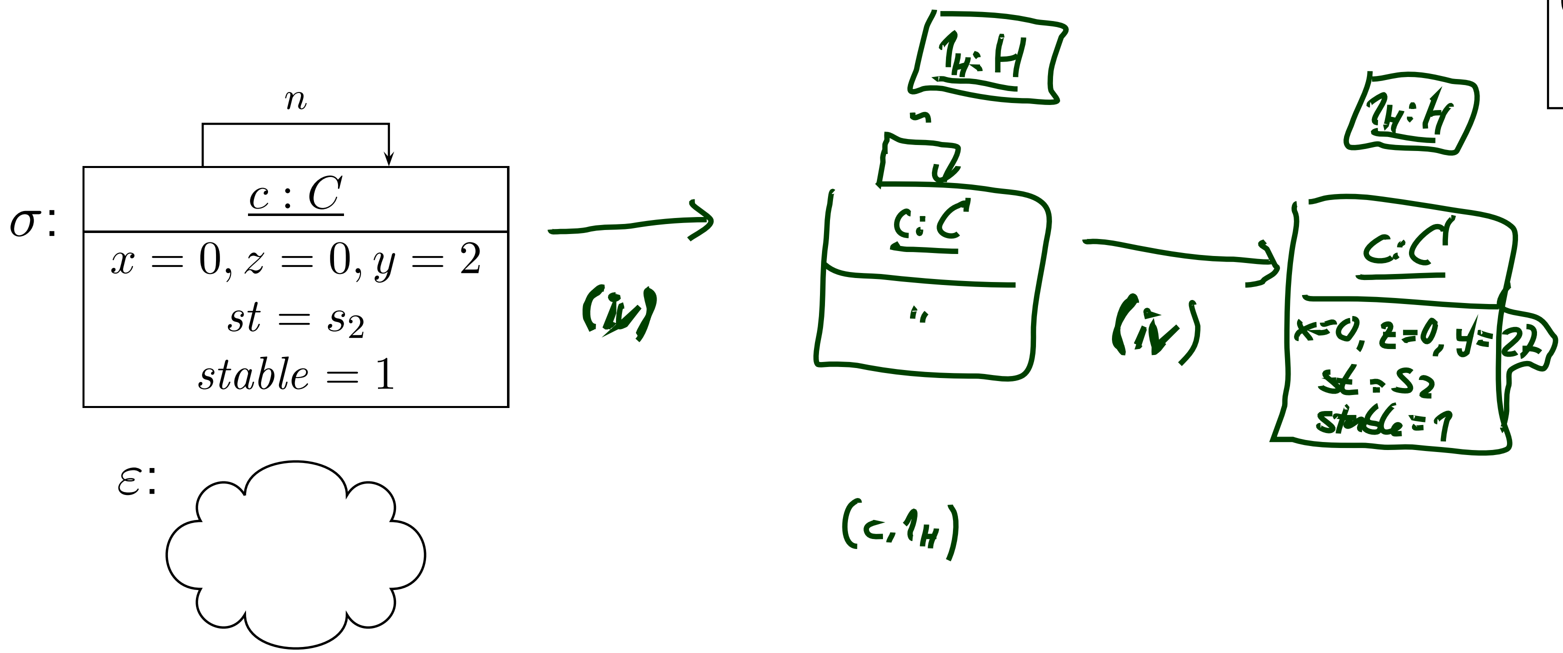
$$\forall\, v \in V \;\forall\, u \in \mathrm{dom}(\sigma) : \sigma'(u)(v) \neq \sigma(u)(v) \implies v \in V_{env}.$$

  and no objects appear or disappear, i.e. $\mathrm{dom}(\sigma') = \mathrm{dom}(\sigma)$.
- $\varepsilon' = \varepsilon$.

Example: Environment







- $\sigma' = \sigma \dot\cup \{u_E \mapsto \{v_i \mapsto d_i \mid 1 \leq i \leq n\}$
- $\varepsilon' = \varepsilon \oplus u_E$ where $u_E \notin \mathrm{dom}(\sigma)$ and $atr(E) = \{v_1, \ldots, v_n\}$.
- $u \in \mathrm{dom}(\sigma)$
- $cons = \emptyset$, $Snd = \{(env, E(\vec{d}))\}$.

(v) Error Conditions

$$s \xrightarrow[u]{(cons,Snd)} \#$$

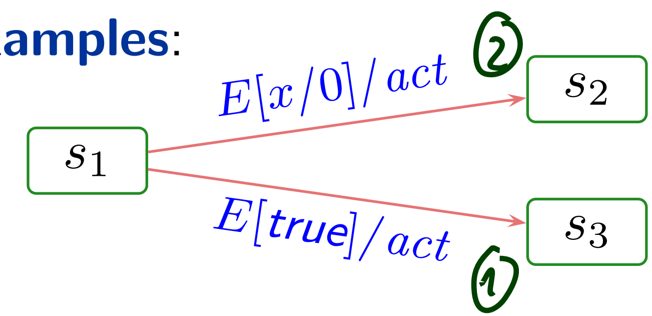
if, in (i), (ii), or (iii),

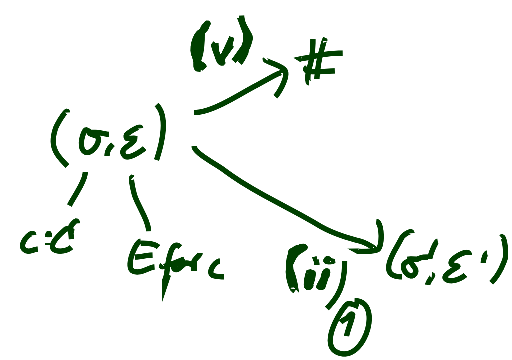
- $I[\![expr]\!]$ is not defined for σ and u, or
- $t_{act}[u]$ is not defined for (σ, ε),

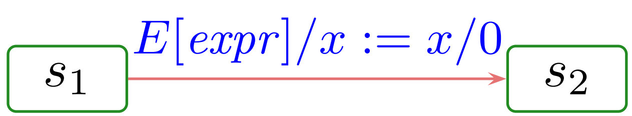
and

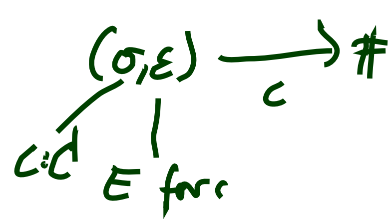
- $cons = \emptyset$, and $Snd = \emptyset$.

Examples:

- 

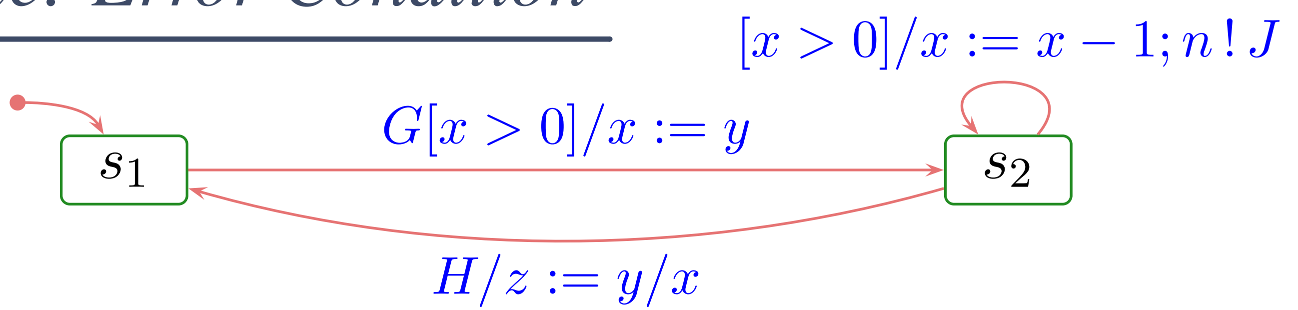


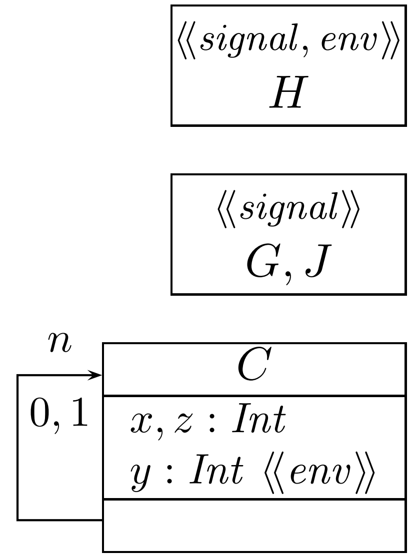
- 



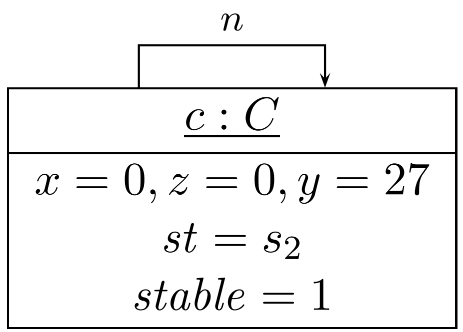
Example: Error Condition

$\mathcal{SM}_C$:

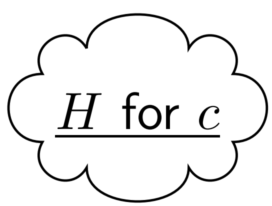




σ:

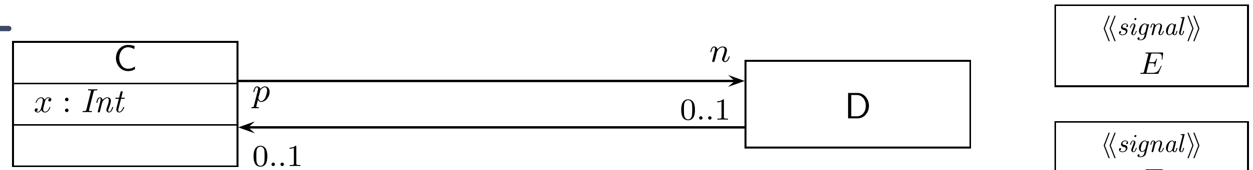


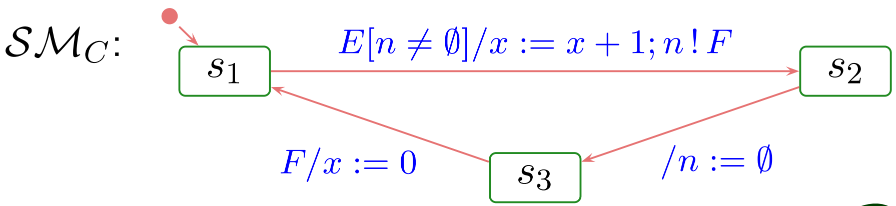
ε:

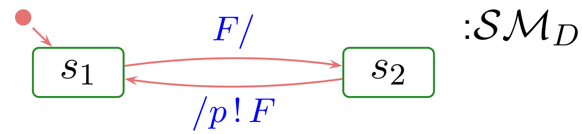


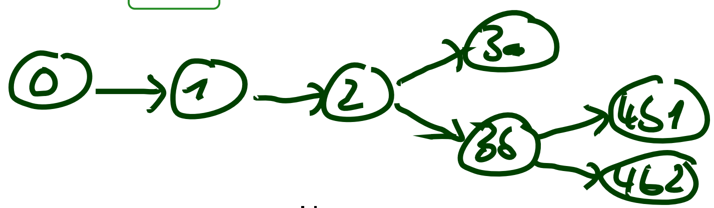
- $I[\![expr]\!]$ not defined for σ and u, or
- $t_{act}[u]$ is not defined for (σ, ε)
- $cons = \emptyset$,
- $Snd = \emptyset$

Example Revisited









	$1_C : C$				$5_D : D$				
Nr.	x	n	st	$stable$	p	st	$stable$	ε	rule
0	27	5_D	s_1	1	1_C	s_1	1	$\underline{(3_F, 1_C)}.(2_E, 1_C)$	
1	27	5_D	s_1	1	1_C	s_1	1	$(2_E, 1_C)$	(i)
2	28	5_D	s_2	0	1_C	s_1	1	$(9_F, 5_D)$	(ii)
3a	28	$\emptyset$	s_3	1	1_C	s_1	1	$(9_F, 5_D)$	(ii)
3b	28	5_D	s_2	0	1_C	s_2	0	–	(ii)
4b1									
4b2									

References

References

Harel, D. and Gery, E. (1997). Executable object modeling with statecharts. *IEEE Computer*, 30(7):31–42.

OMG (2011a). Unified modeling language: Infrastructure, version 2.4.1. Technical Report formal/2011-08-05.

OMG (2011b). Unified modeling language: Superstructure, version 2.4.1. Technical Report formal/2011-08-06.