Lecture 14 - March 7

Reactive System: Bridge Controller

Announcements

- Slides updated to include First Refinement
- Released: Lab2 solution video, PracticeTest1 solution
- To be completed by the final exam:

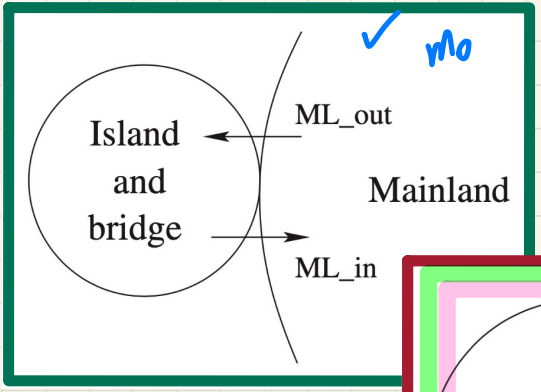
  Makeup lectures for WT1, WT2, ProgTest1, ProgTest2

Lecture

Reactive System: Bridge Controller

First Refinement: State and Events

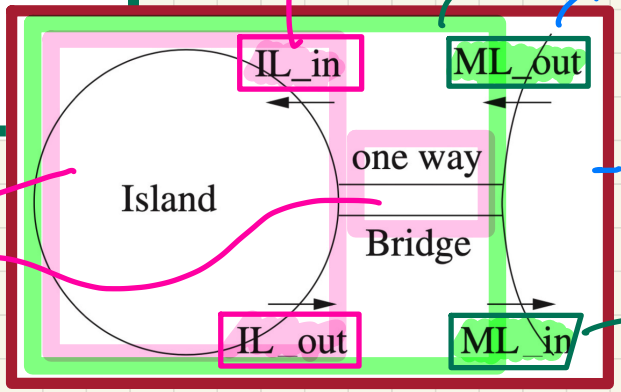
Bridge Controller: Abstraction in the 1st Refinement



m0

m0:

initial, most abstract



2 new events

the initial abstraction between m1 (no distinction IL and bridge)

m1:

second, more concrete

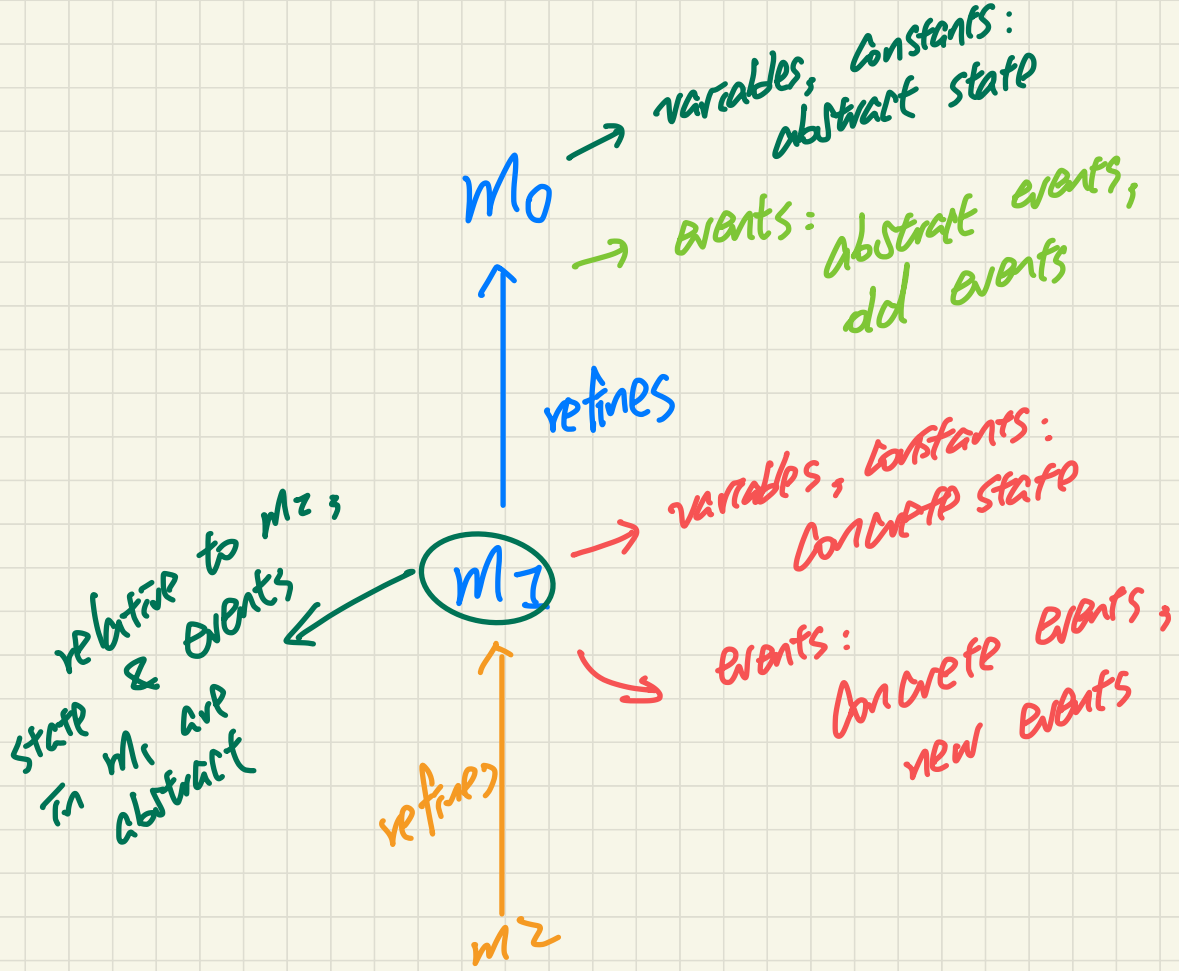
- Both m0 and m1 model the same system.

2 old events

- m1 refines m0 by adding more state vars. & events

separate components in the more concrete 1st refinement

REQ1	The system is controlling cars on a bridge connecting the mainland to an island.
REQ3	The bridge is one-way or the other, not both at the same time.

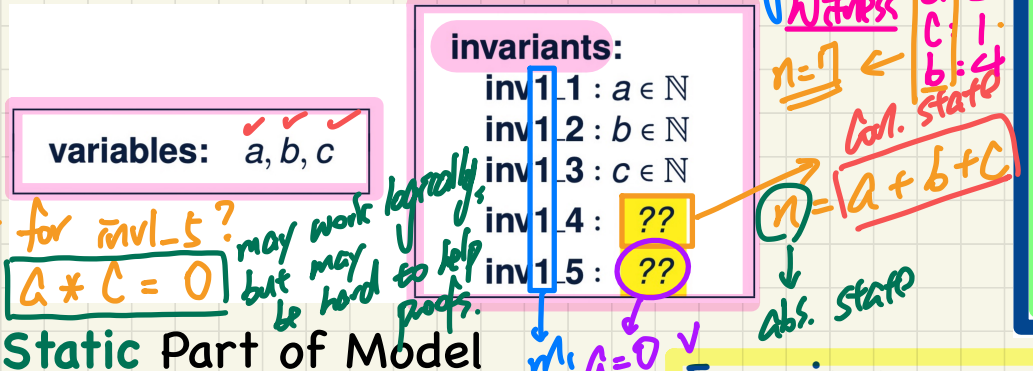

M0
variables, constants:
abstract state
events:
abstract events,
dd events
refines
M1
variables, constants:
concrete state
events:
concrete events,
new events
relative to M2;
state & events
in M1 are
abstract
refines
M2

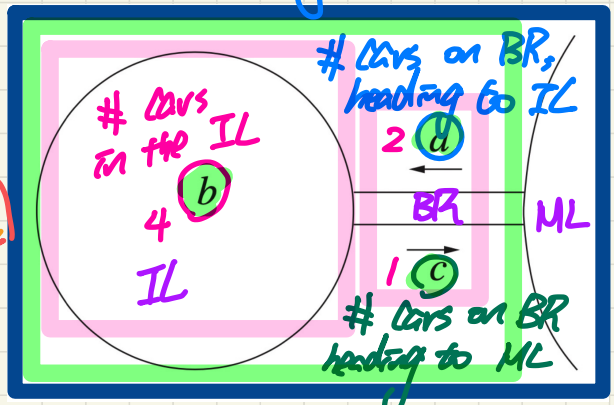
Bridge Controller: State Space of the 1st Refinement

REQ1	The system is controlling cars on a bridge connecting the mainland to an island.
REQ3	The bridge is one-way or the other, not both at the same time.

Dynamic Part of Model

What's wrong if M1 is constructed by inv1_1 to inv1_4 only?





Static Part of Model

constants: d

axioms:
axm0_1 : $d \in \mathbb{N}$
axm0_2 : $d > 0$

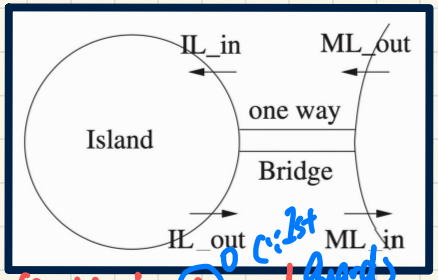
Exercises

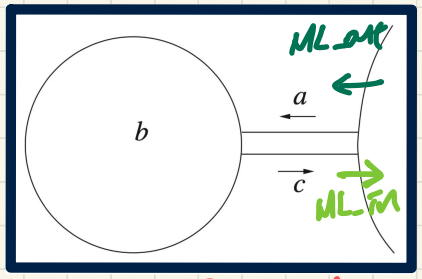
n a, b, c

inv1_4: linking abstract & concrete states

inv1_5: bridge is one-way

Bridge Controller: Guards of "old" Events 1st Refinement





ML_out: A car exits **mainland** (getting on the **bridge**).

ML_out
when
??
then
$a := a + 1$
end

① $c = 0$
② $a + b < d$
1 effect
BAP: $a' = a + 1 \wedge b' = b \wedge c' = c$

ML_in: A car enters **mainland** (getting off the **bridge**).

ML_in
when
??
then
$c := c - 1$
end

BAP: $c' = c - 1 \wedge a' = a \wedge b' = b$
$c > 0$
Q. Necessary to add a guard "$a = 0$"? $a = 0$
No. $\because c > 0 \wedge (a = 0 \vee c = 0)$

③ $(a+1) + b + c \le d$ (guard)
$\rightarrow (a+1) + b \le d$
$a + b \le d - 1$
$\rightarrow a + b < d$

① From m_0: $n' \quad x \le d$
② From m_1: n'
$x + b + c = x$
$(a+1)$ effect of ML_out

constants: d

axioms:
axm0_1 : $d \in \mathbb{N}$
axm0_2 : $d > 0$

variables: a, b, c

invariants:
inv1_1 : $a \in \mathbb{N}$
inv1_2 : $b \in \mathbb{N}$
inv1_3 : $c \in \mathbb{N}$
inv1_4 : $a + b + c = n$
inv1_5 : $a = 0 \vee c = 0$

$x \le y - 1$
$\equiv x < y$
$x \quad y$

States, Invariants, Events: Abstract vs. Concrete