Department of Computer Science
National Tsing Hua University

CS 5244: Introduction to Cyber Physical Systems

Unit 9: Composition of State Machines

(Ch. 5)

Instructor: Cheng-Hsin Hsu

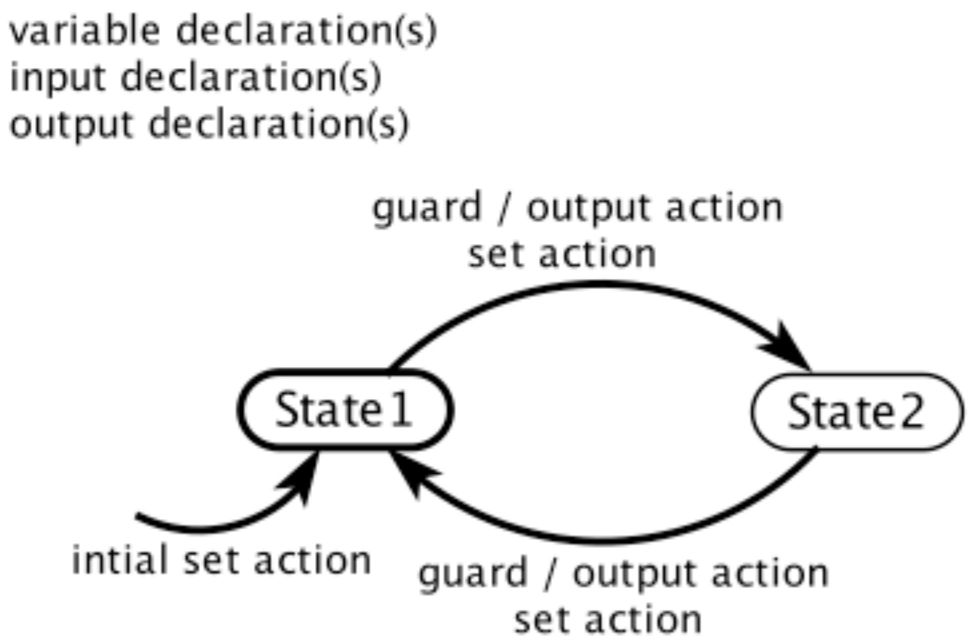
Acknowledgement: The instructor thanks Profs. Edward A. Lee & Sanjit A. Seshia at UC Berkeley for sharing their course materials

Modeling with State Machines

Questions:

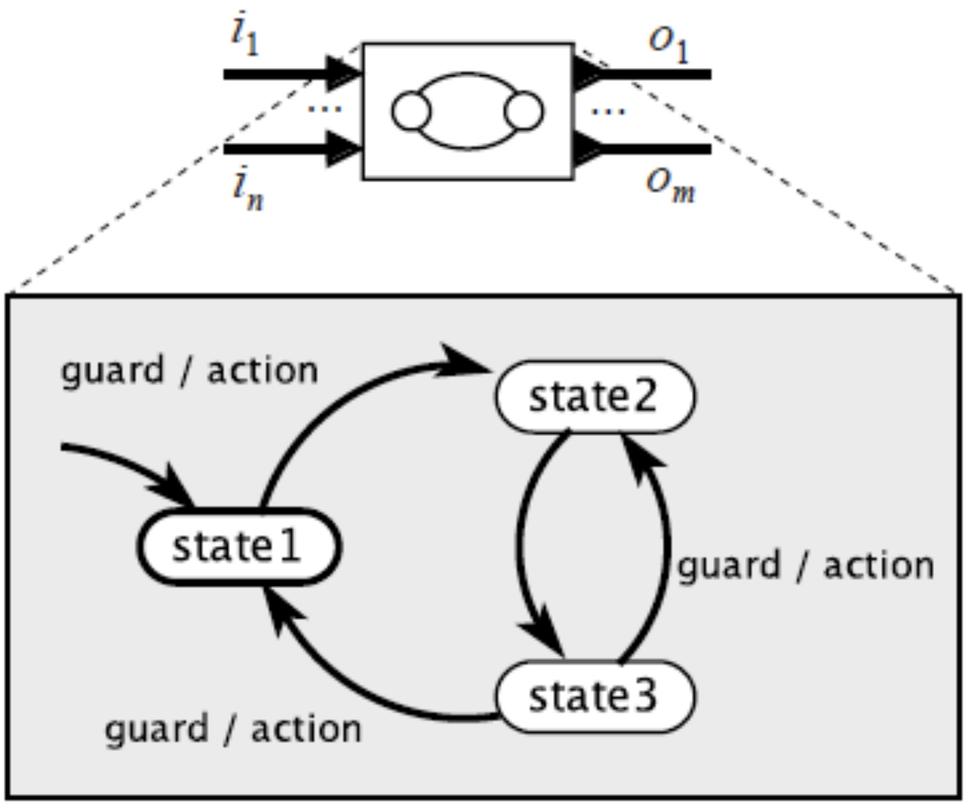
- How to represent the system for:
  - Systematic analysis
  - So that a computer program can manipulate it
- How to model its environment
- How to compose subsystems to make bigger systems
- How to check whether the system satisfies its specification in its operating environment

Notation for Extended State Machines



Actor Model for State Machines

Expose inputs and outputs, enabling composition:



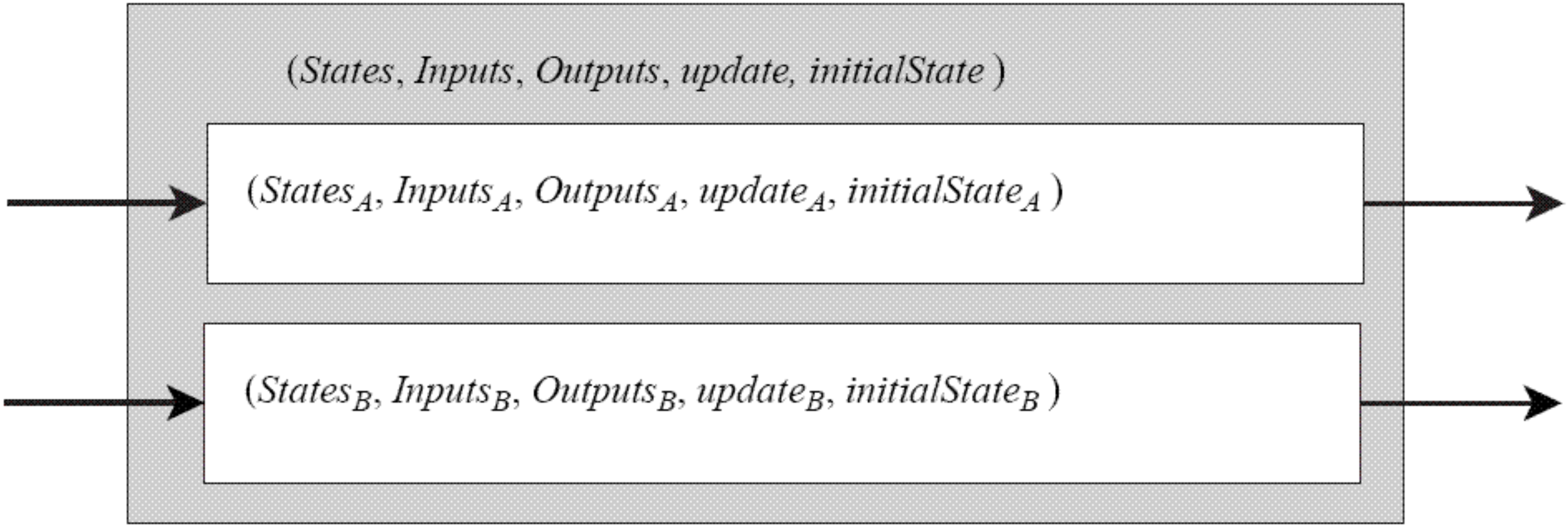
Composition of State Machines

Side-by-side composition

Cascade composition

Feedback composition

Side-by-Side Composition



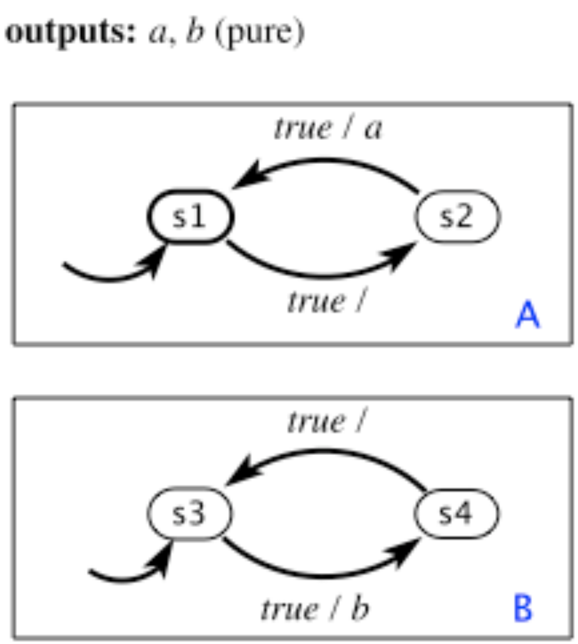
A key question: When do these machines react?

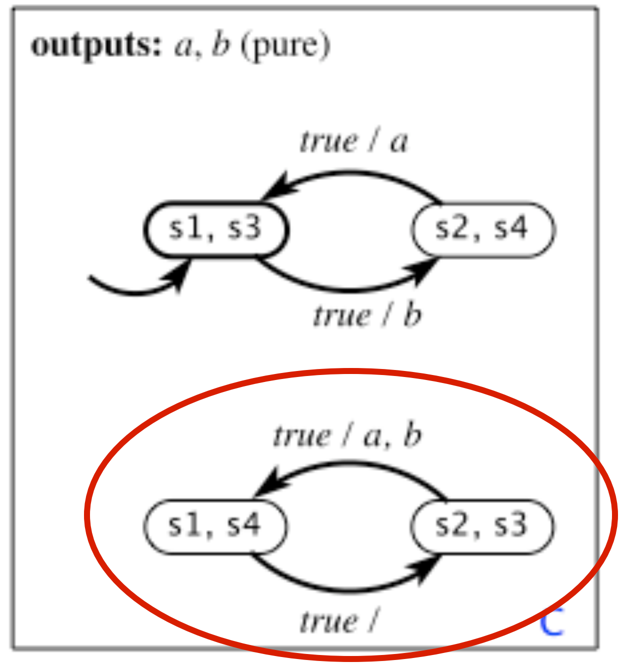
Two possibilities:

- Together (synchronous composition)
- Independently (asynchronous composition)

Synchronous Composition

$$S_C = S_A \times S_B$$



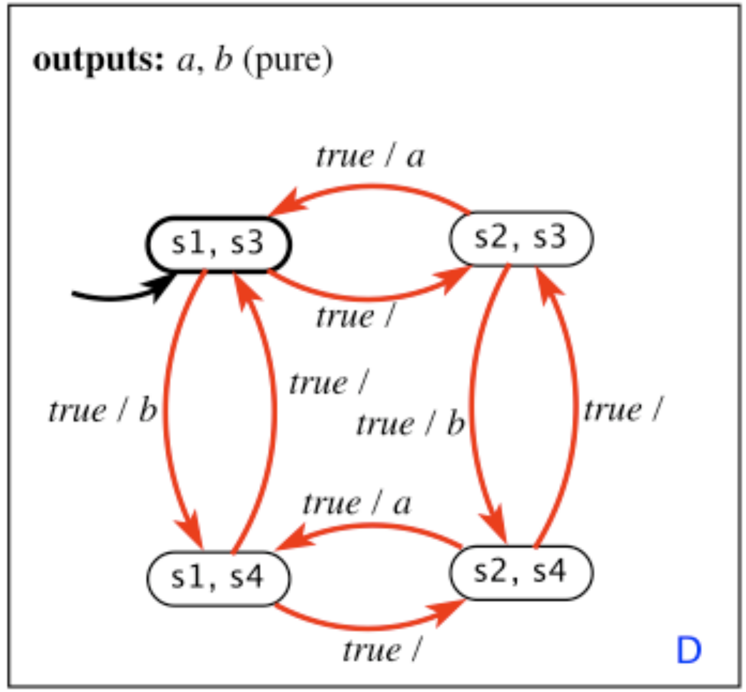


Synchronous composition

Note that these two states are not *reachable*.

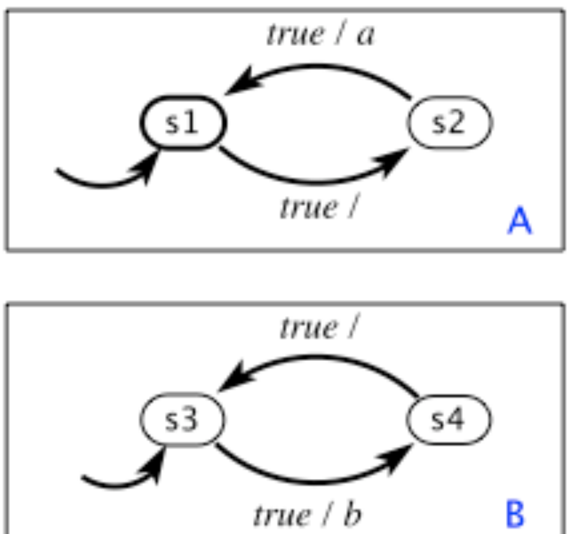
Asynchronous Composition

$$S_C = S_A \times S_B$$



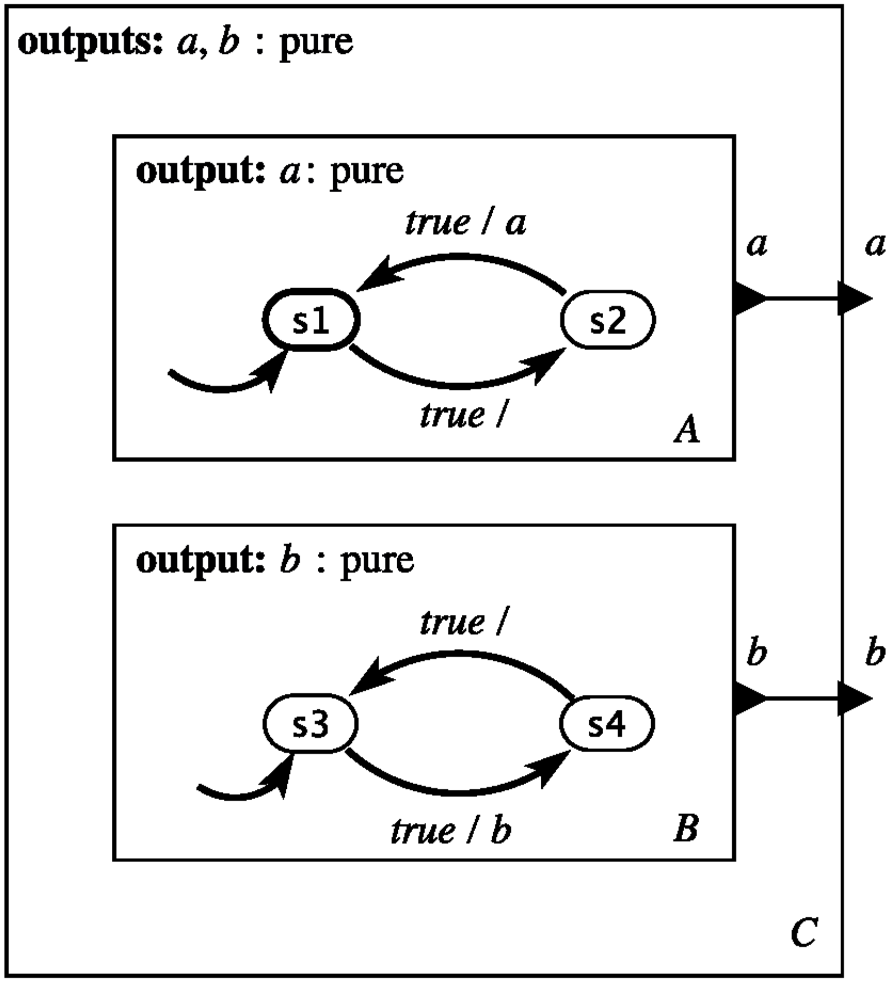
Asynchronous composition using interleaving semantics

outputs: a, b (pure)



Note that now all states are reachable.

Syntax vs. Semantics



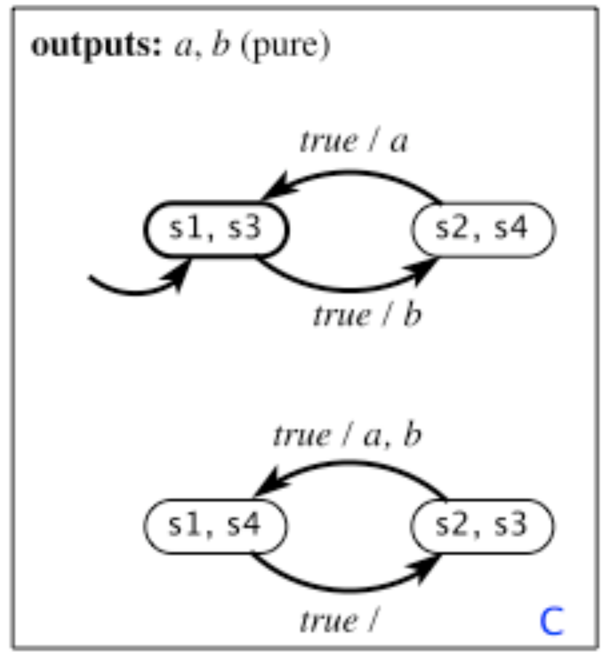
Synchronous or Asynchronous composition?

If asynchronous, does it allow simultaneous transitions in A & B?

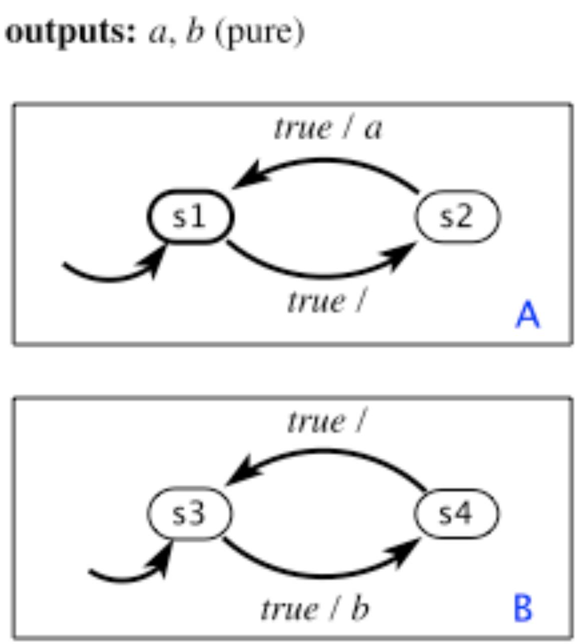
Asynchronous composition is *not* synchronous composition with stuttering transitions.

Stuttering is a reaction where no output is produced and the state is not changed.

$$S_C = S_A \times S_B$$



Synchronous composition



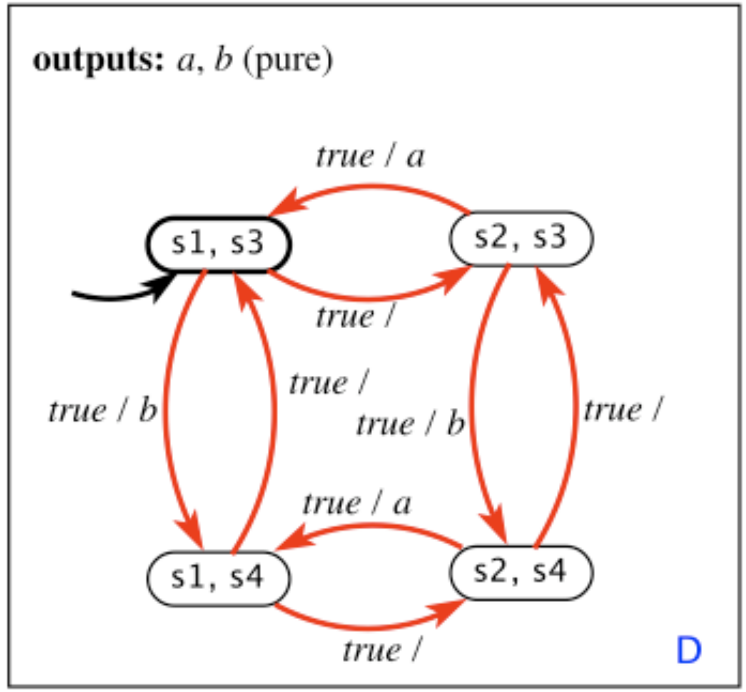
These two FSMs cannot stutter.

If they react, they change state.

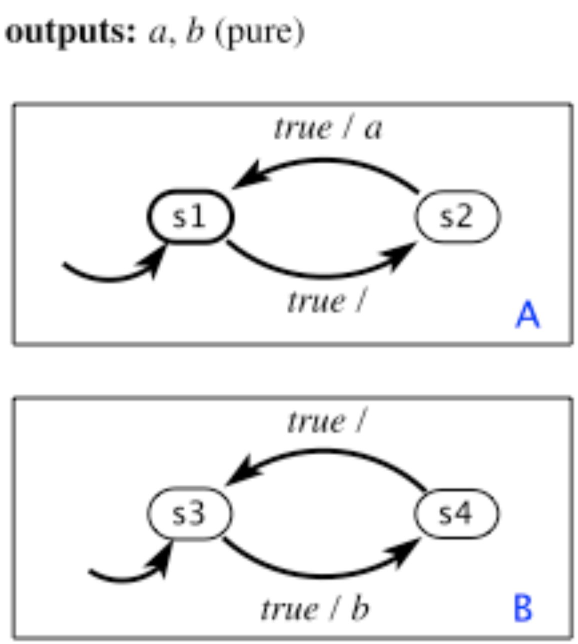
Asynchronous composition is *not* synchronous composition with stuttering transitions.

Stuttering is a reaction where no output is produced and the state is not changed.

$$S_C = S_A \times S_B$$



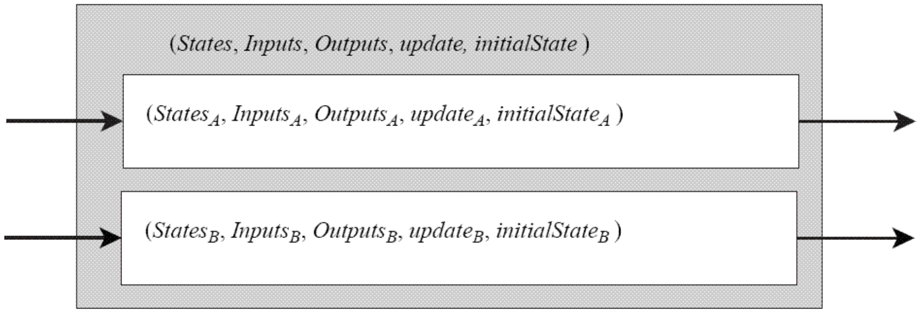
Asynchronous composition with interleaving semantics



These two FSMs cannot stutter.

If they react, they change state.

Side-by-Side Synchronous Composition



Definition of the side-by-side composite machine:

$$States = States_A \times States_B$$
$$Inputs = Inputs_A \times Inputs_B$$
$$Outputs = Outputs_A \times Outputs_B$$
$$initialState = (initialState_A, initialState_B)$$
$$((s_A(n+1), s_B(n+1)), (y_A(n), y_B(n)))$$
$$= update((s_A(n), s_B(n)), (x_A(n), x_B(n))),$$

where

$$(s_A(n+1), y_A(n)) = update_A(s_A(n), x_A(n)) \text{ and}$$
$$(s_B(n+1), y_B(n)) = update_B(s_B(n), x_B(n))$$

Asynchronous Composition

$M_1 = (S_1, I_1, O_1, U_1, s_{10})$ and $M_2 = (S_2, I_2, O_2, U_2, s_{20})$

M is the asynchronous composition of M_1 and M_2

$= (S_1 \times S_2,\ I_1 \times I_2, O_1 \times O_2, U, (s_{10}, s_{20}))$

where

$U((s_1, s_2), (i_1, i_2)) = ((s_1', s_2'), (o_1, o_2))$

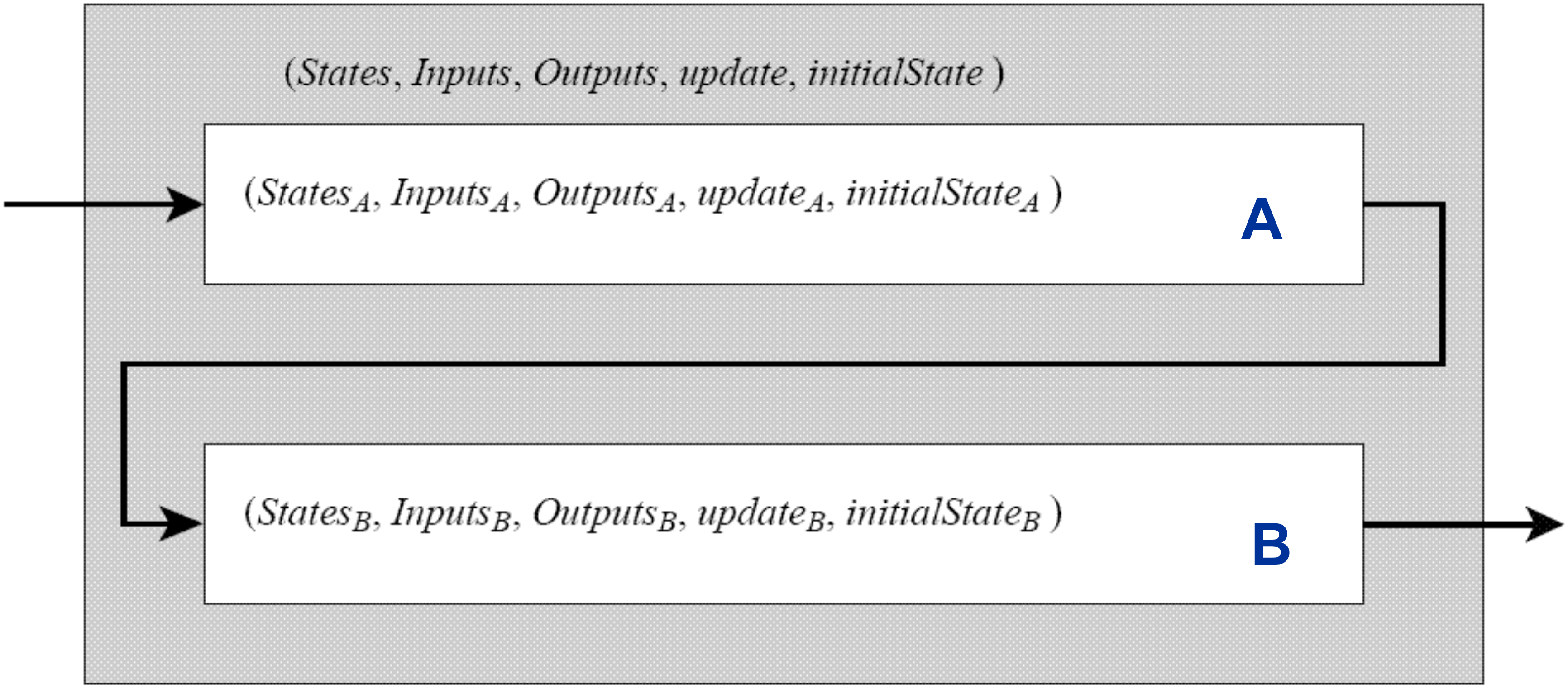
and

$(s_1', o_1) = U_1(s_1, i_1)$ AND $s_2' = s_2$ & o_2 is absent

OR $(s_2', o_2) = U_2(s_2, i_2)$ AND $s_1' = s_1$ & o_1 is absent

(note interleaving semantics)

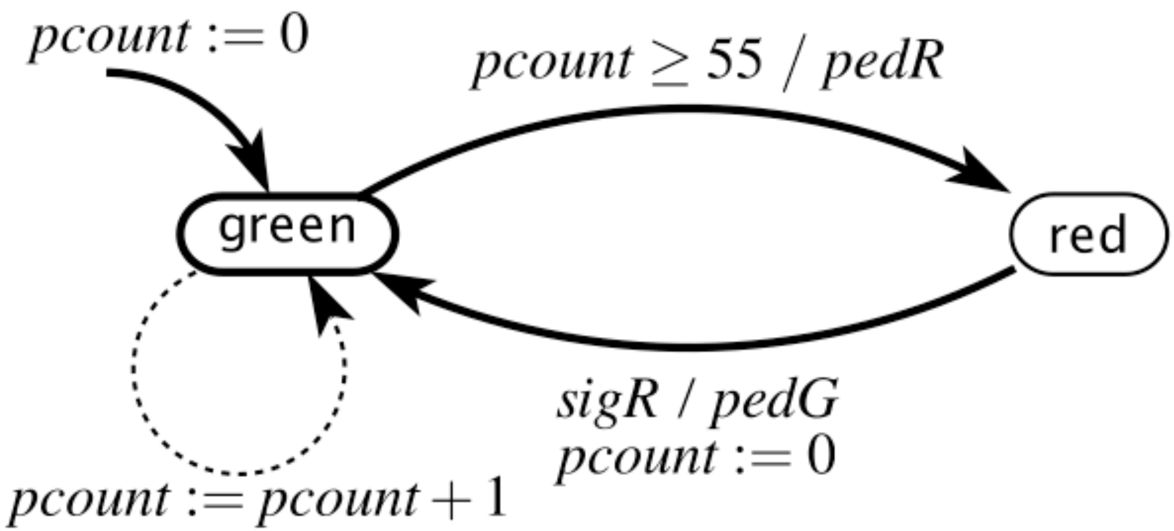
Cascade Composition



Output port(s) of A connected to input port(s) of B

Example: Pedestrian Light

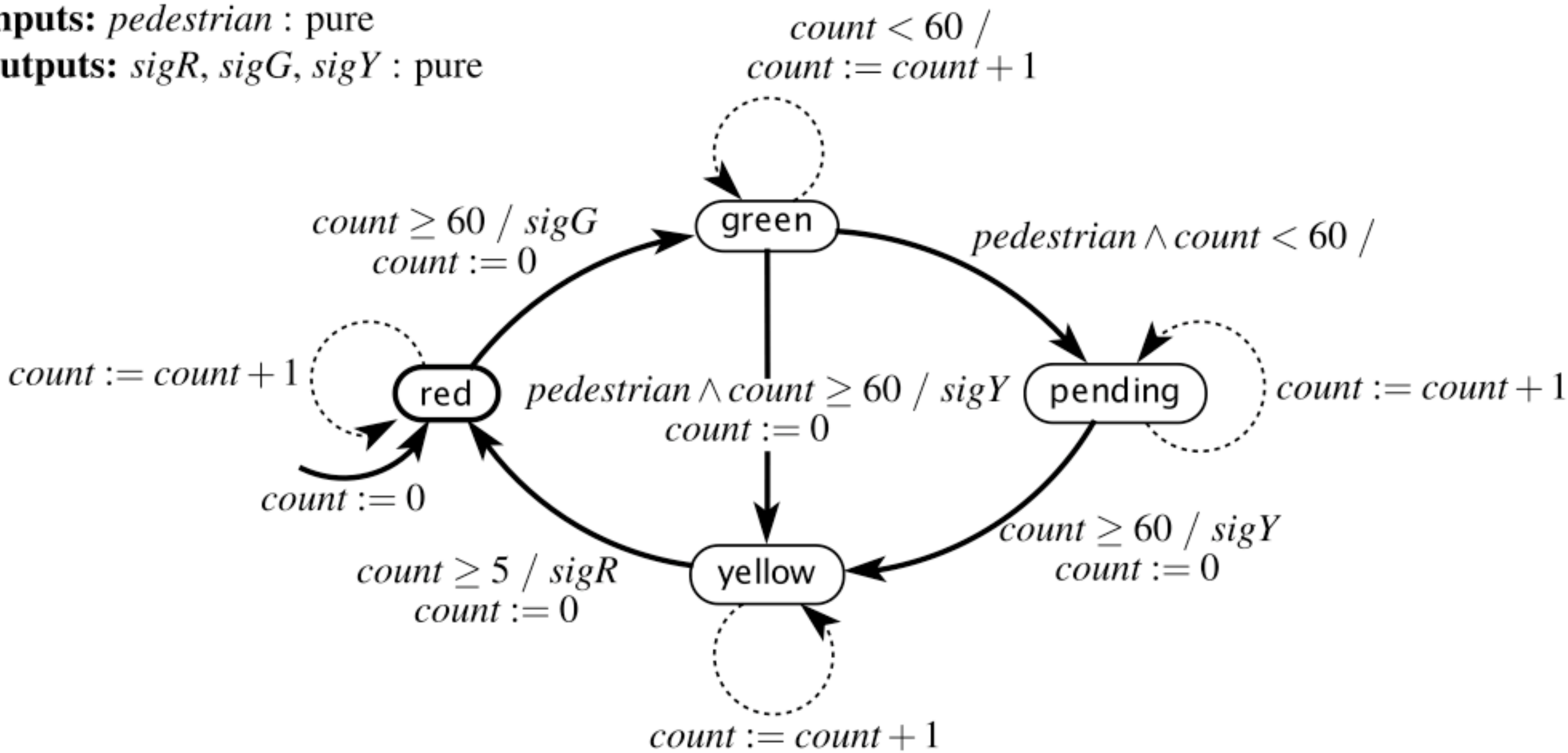
variable: *pcount*: $\{0, \cdots, 55\}$
input: *sigR*: pure
outputs: *pedG*, *pedR*: pure



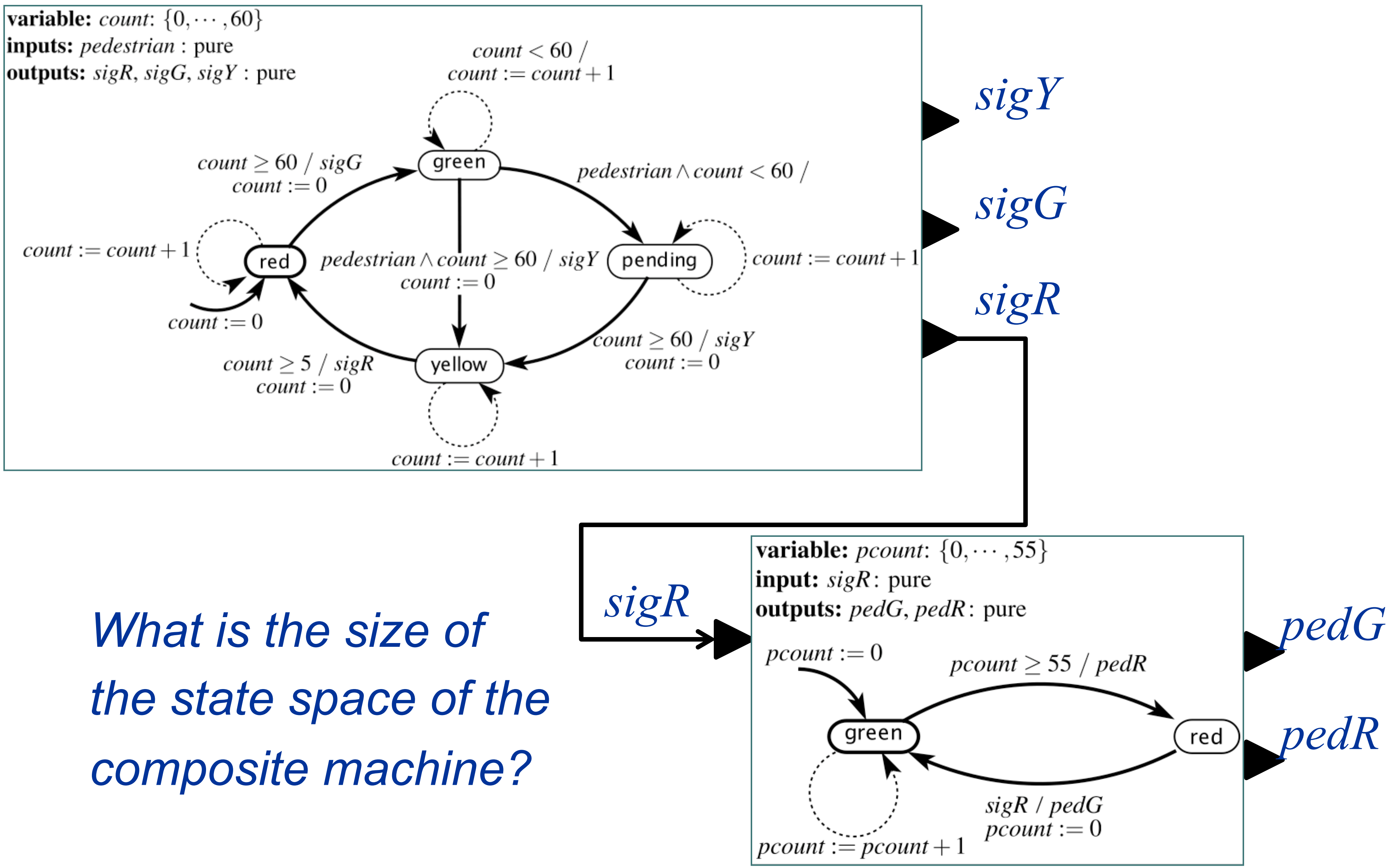
This light stays green for 55 seconds, then goes red. Upon receiving a sigR input, it repeats the cycle.

Example: Car Light

variable: *count*: $\{0, \cdots, 60\}$
inputs: *pedestrian* : pure
outputs: *sigR*, *sigG*, *sigY* : pure



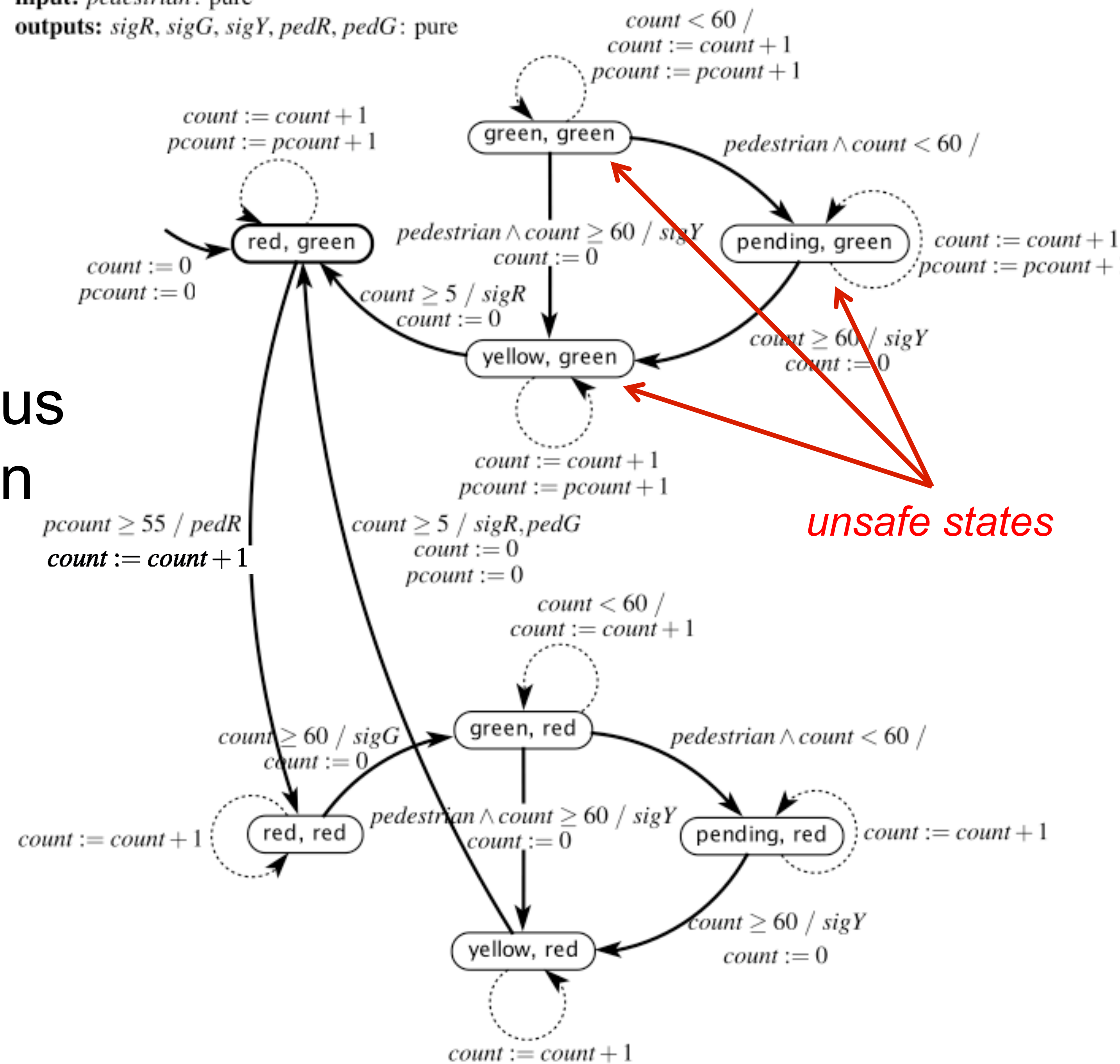
Pedestrian Light with Car Light



What is the size of the state space of the composite machine?

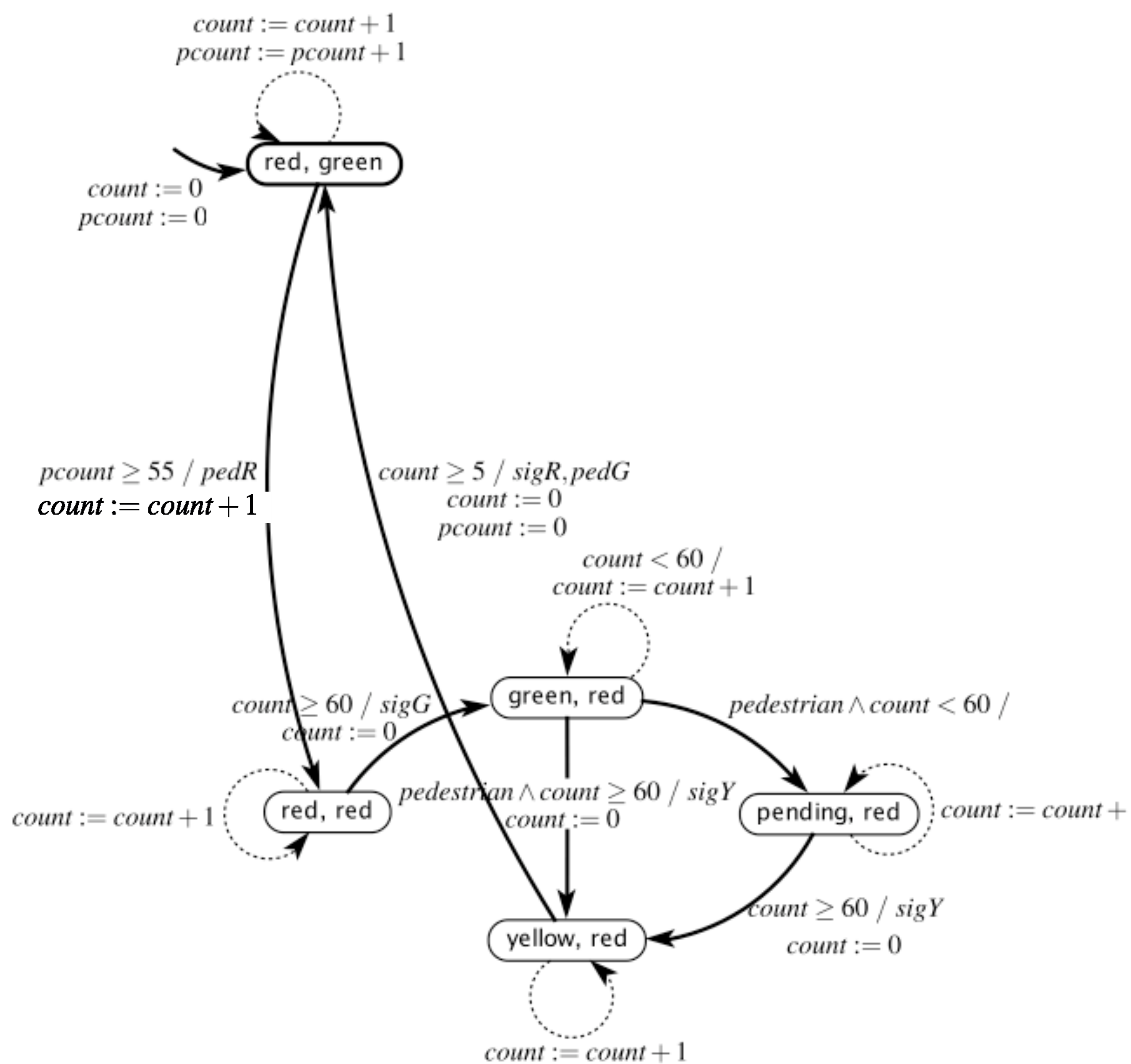
variables: *count*: $\{0, \cdots, 60\}$, *pcount*: $\{0, \cdots, 55\}$
input: *pedestrian*: pure
outputs: *sigR*, *sigG*, *sigY*, *pedR*, *pedG*: pure

Synchronous composition

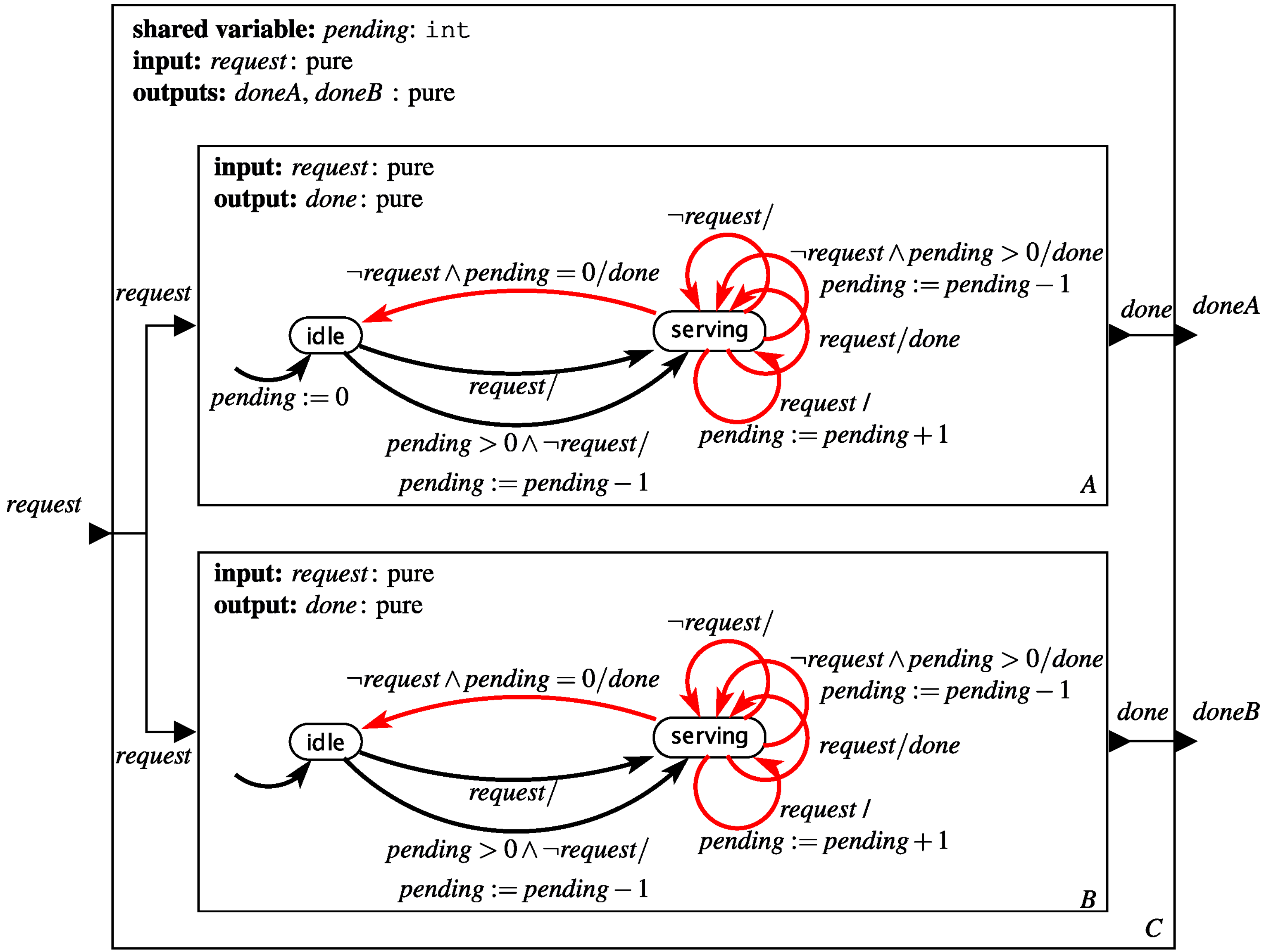


Synchronous composition with unreachable states removed

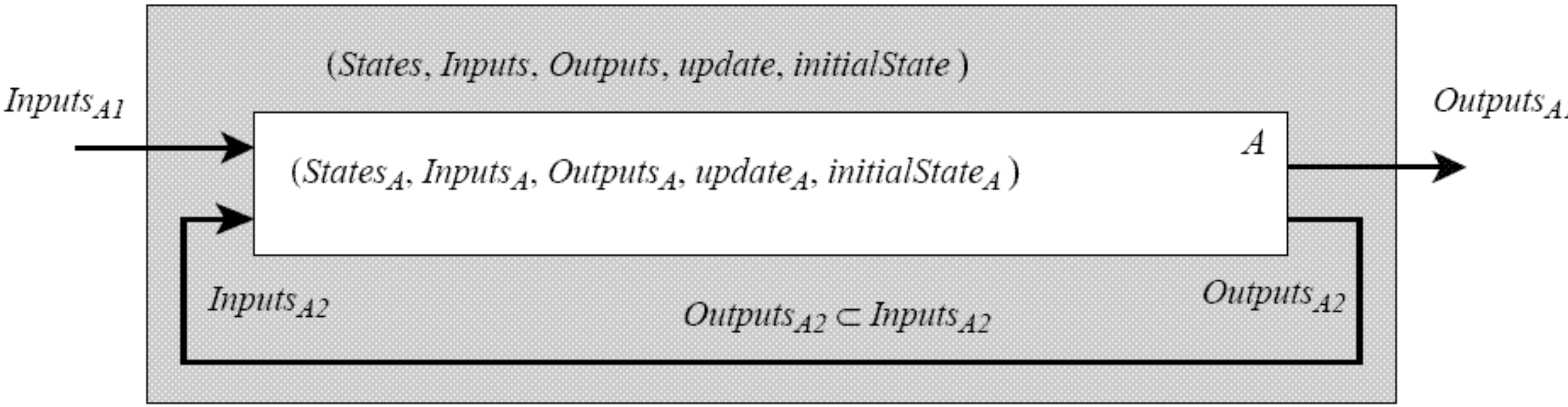
variables: *count*: $\{0, \cdots, 60\}$, *pcount*: $\{0, \cdots, 55\}$
input: *pedestrian*: pure
outputs: *sigR*, *sigG*, *sigY*, *pedR*, *pedG*: pure



Shared Variables: Two Servers



Feedback Composition



More on this later… Very subtle.

Systematic exploration of concurrent behaviors

Use hierarchy (next lecture).

Construct the product automaton using synchronous or asynchronous composition as appropriate.

Specify criteria for correctness (we will use temporal logic).

Reason using the product FSM and the correctness criteria (use systematic, algorithmic techniques).