

# Model Checking 4

Lecture 20

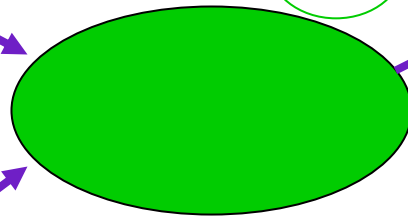
# Model Checking

```
void add(Object o) {  
    buffer[head] = o;  
    head = (head+1)%size;  
}  
  
Object take() {  
    ..  
    tail=(tail+1)%size;  
    return buffer[tail];  
}
```

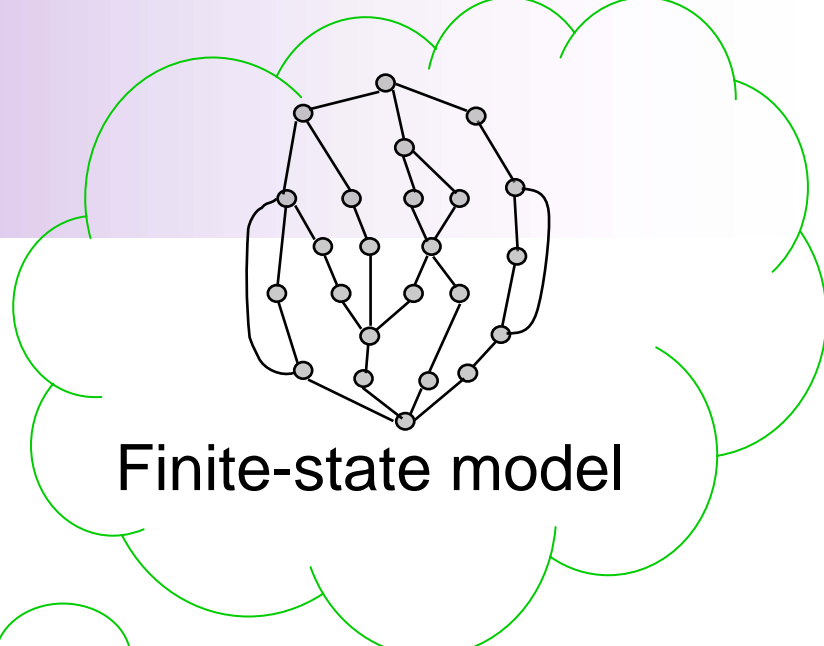
Finite-state Program

```
Property 1: ...  
Property 2: ...  
...
```

Requirement



Model Checker



Finite-state model

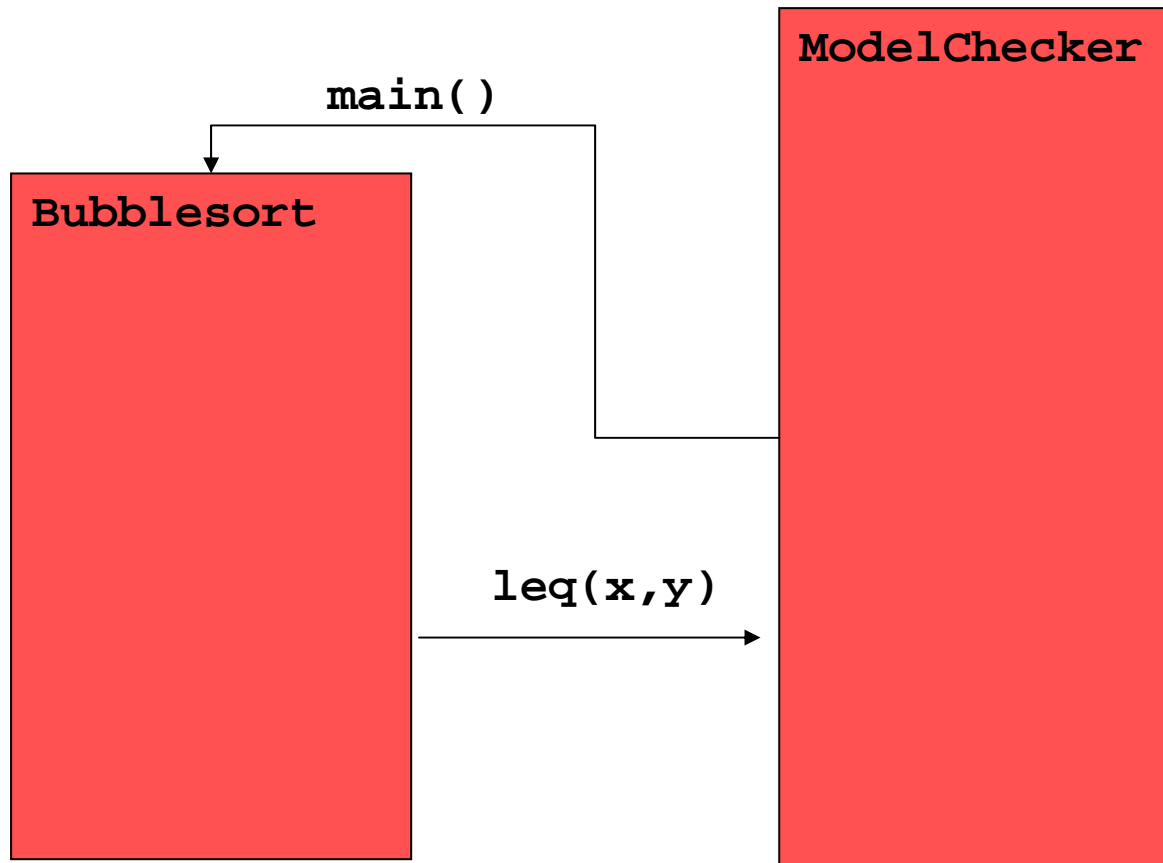
OK

or

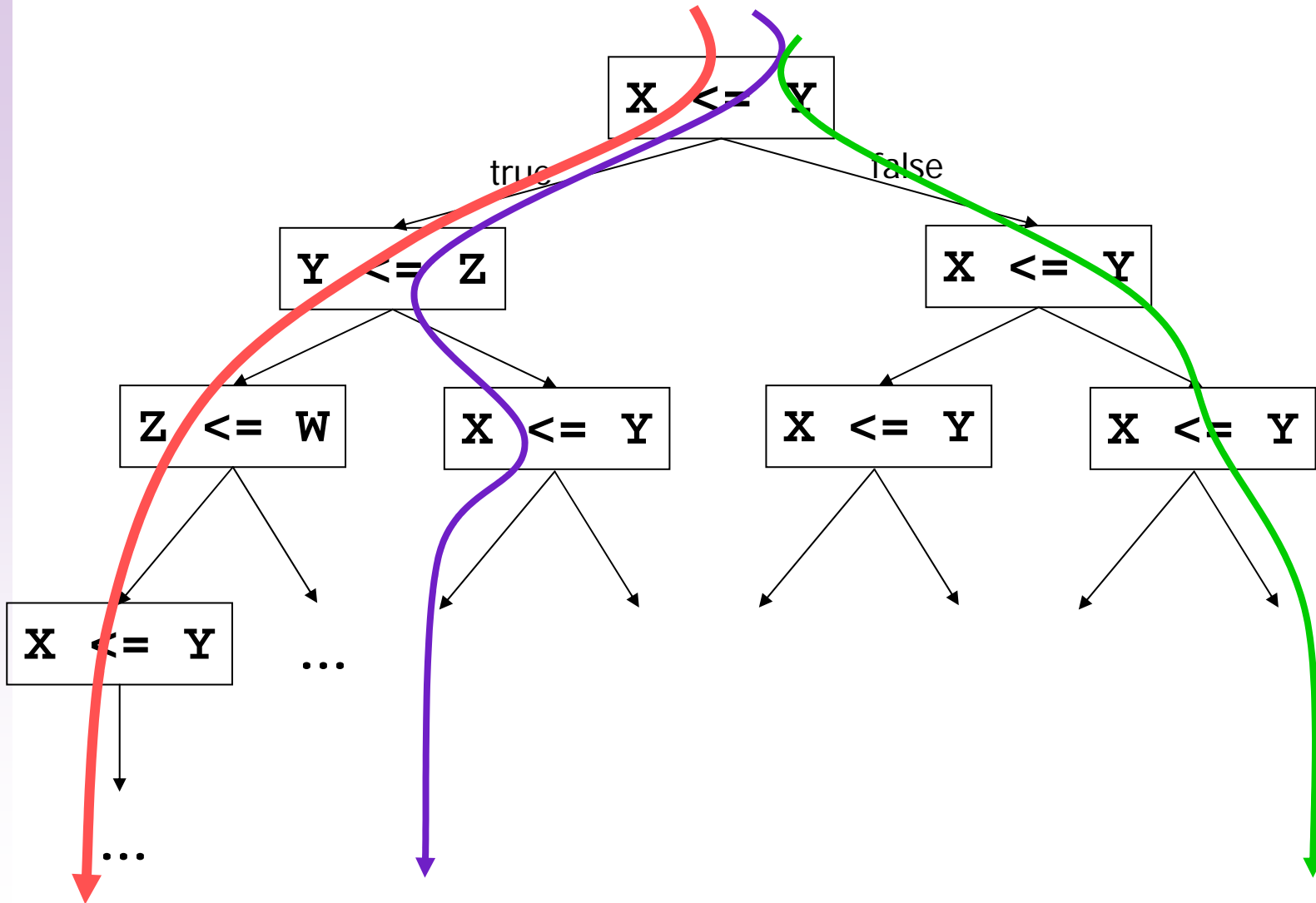
Error trace

```
Line 5: ...  
Line 12: ...  
Line 15: ...  
Line 21: ...  
Line 25: ...  
Line 27: ...  
...  
Line 41: ...  
Line 47: ...
```

# Your Model Checker



# Your Model Checker



# Homework 5

- ModelChecker calls Bubblesort multiple times
- Bubblesort calls ModelChecker.leq() to compare MyInts
- ModelChecker.leq() must provide consistent answers on each test of Bubblesort
- ModelChecker must do enough tests of Bubblesort to verify correctness of Bubblesort
  - with different behaviors of ModelChecker.leq on each run

# Verisoft

Developed by Patrice Godefroid at Lucent

Model checker for communications software

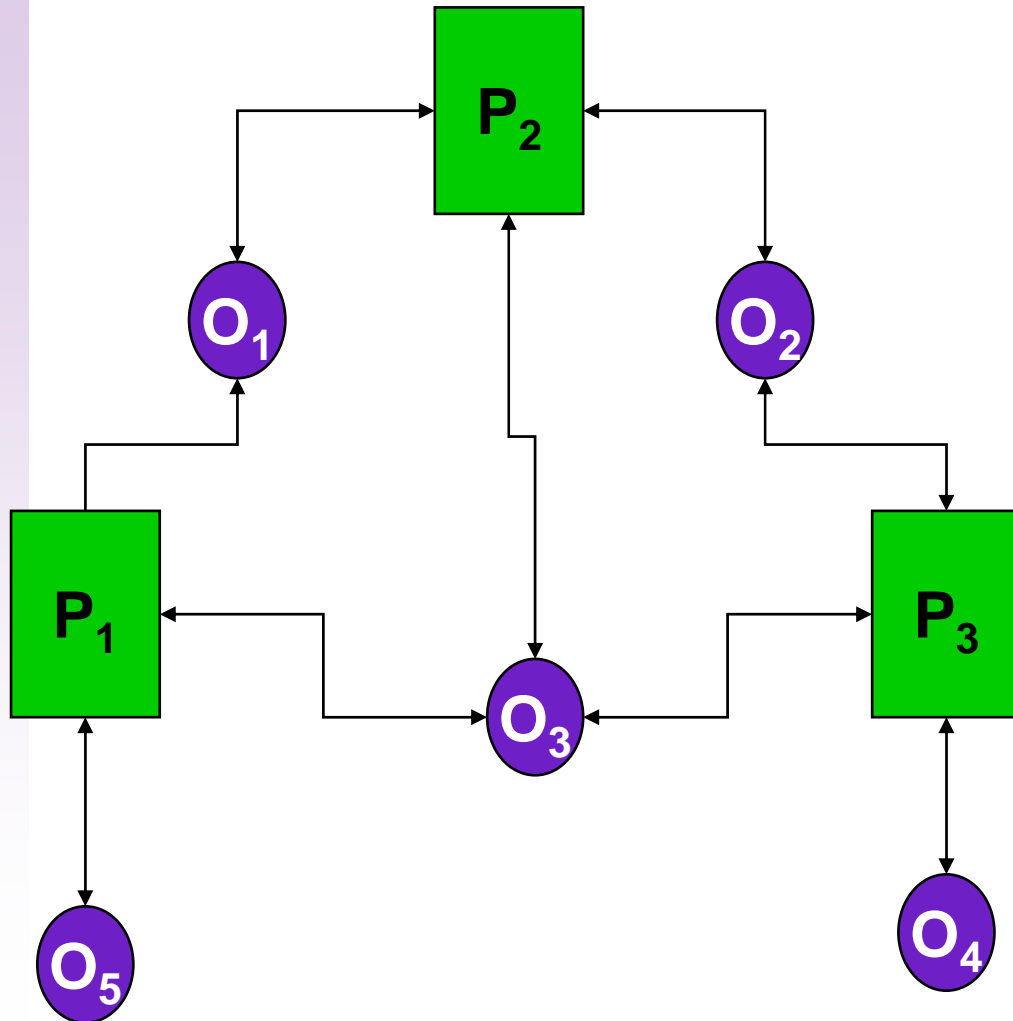
Operates on **real code** (C, C++)

Same idea as your model checker

Repeatedly run the program

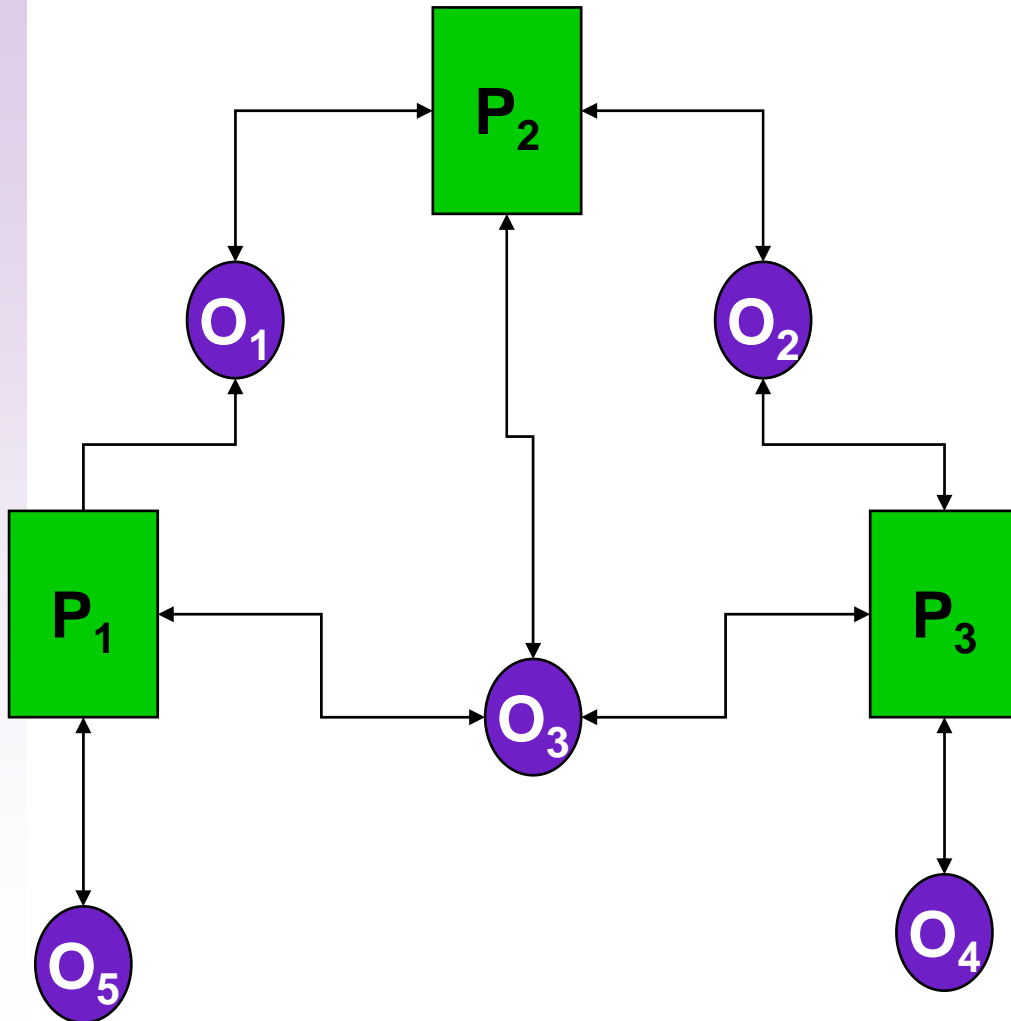
inspecting and controlling its behavior

# Verisoft



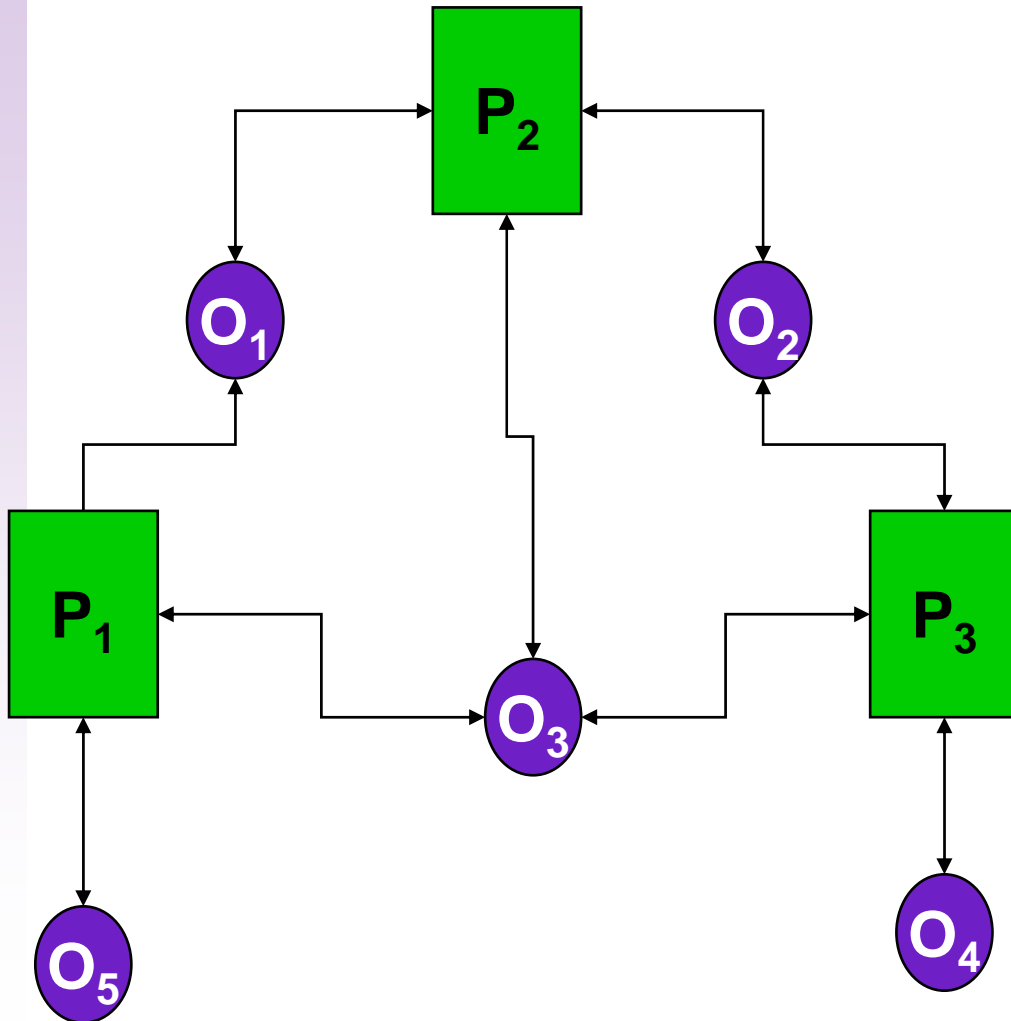
**Processes** execute code written in C, C++, Java, Tcl, etc.

Processes interact through **communication objects** like message queues, semaphores, shared memory, TCP connection, etc.



Local operations of each process is **invisible**

Operations that access communication objects are **visible**



When the next operation to be executed by all processes is visible, system is in a **global state**

**Transition** = one visible op + sequence of invisible ops by some process (moves from one global state to another)

# Properties

- Deadlocks
- Assertion violations
- Livelocks (no enabled transition for a process in  $x$  transitions)
- Divergences (process does not communicate with the rest of the system in  $x$  seconds)

# Technique

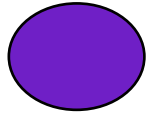
Systematically explore the global state space of the system!

# Difficulty

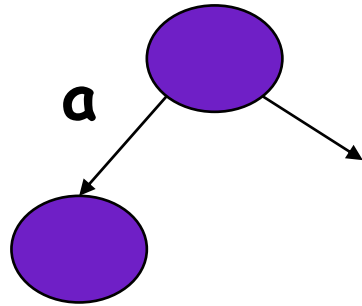
Each global state is **HUGE** (includes stack and heap of each process)

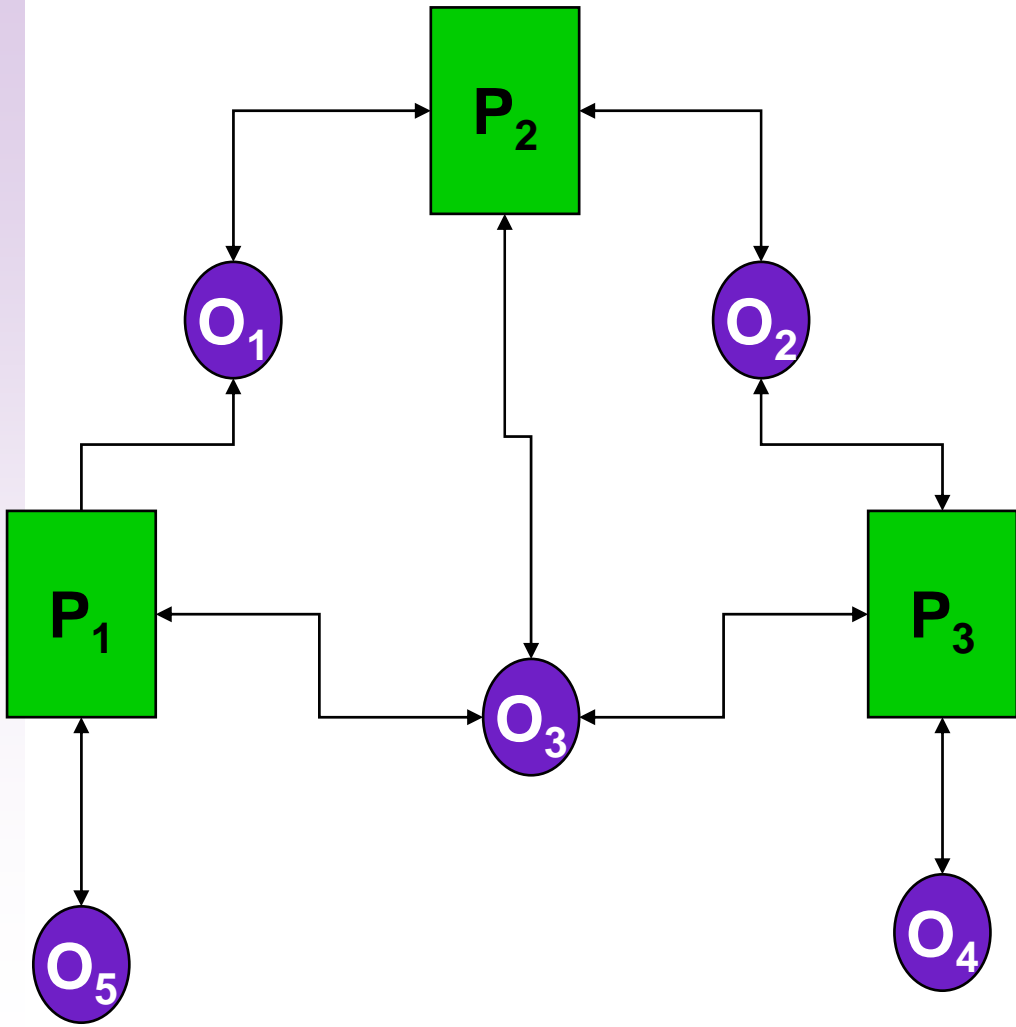
**Idea:** Don't store states (stateless search!)

# Stateless search

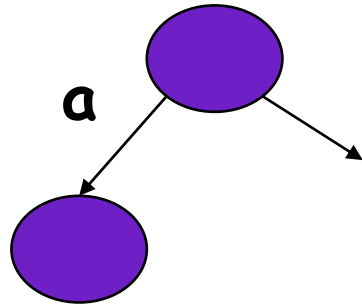


# Stateless search

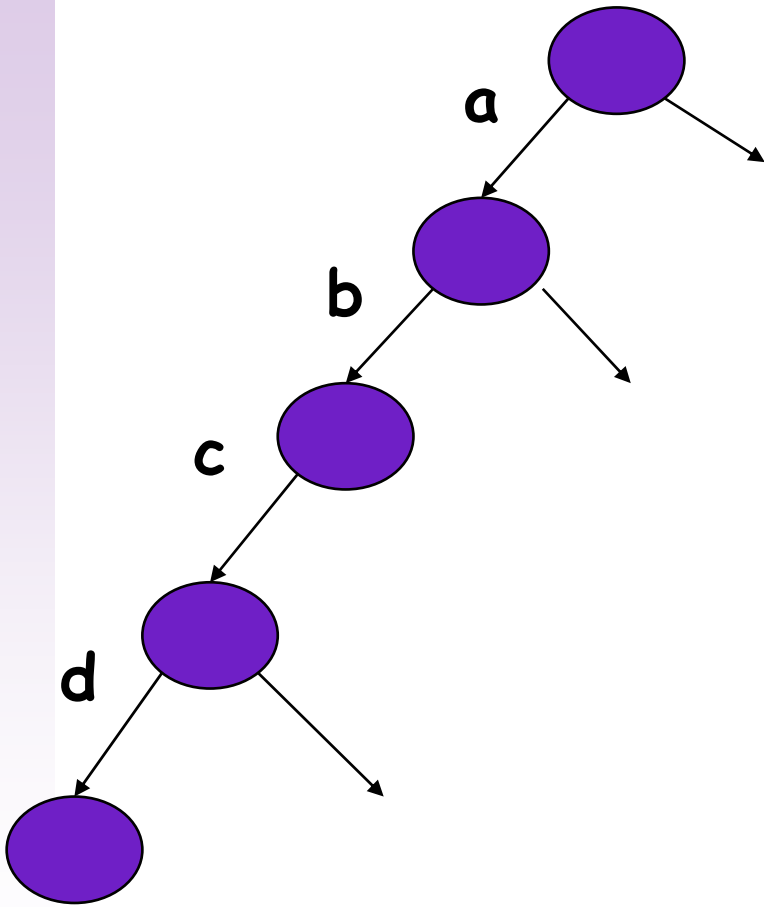




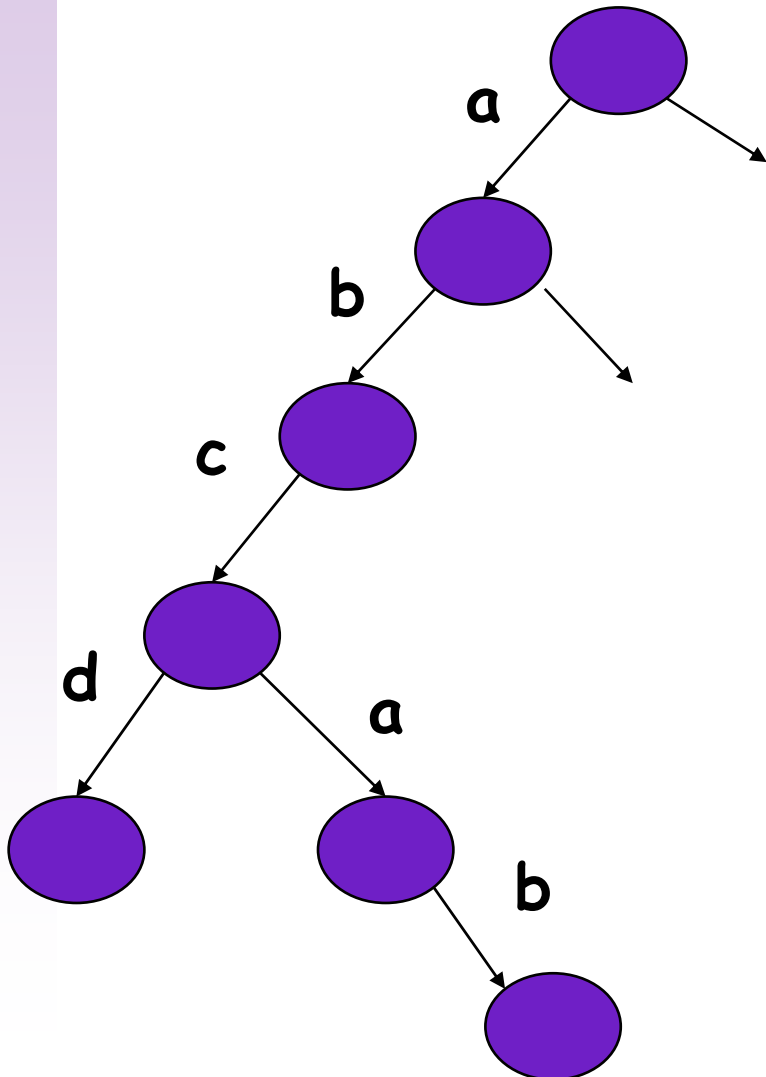
# Stateless search



# Stateless search

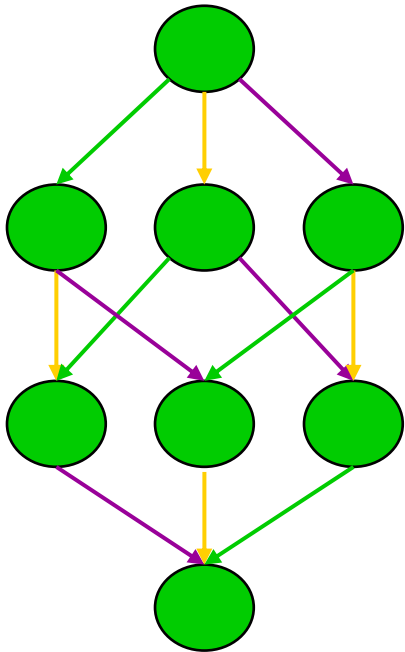


# Stateless search



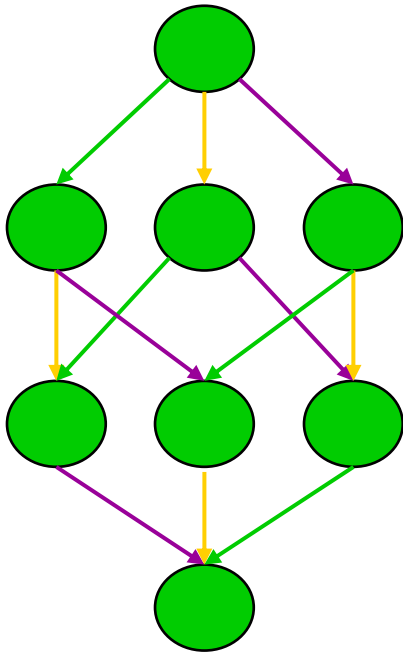


# Problems

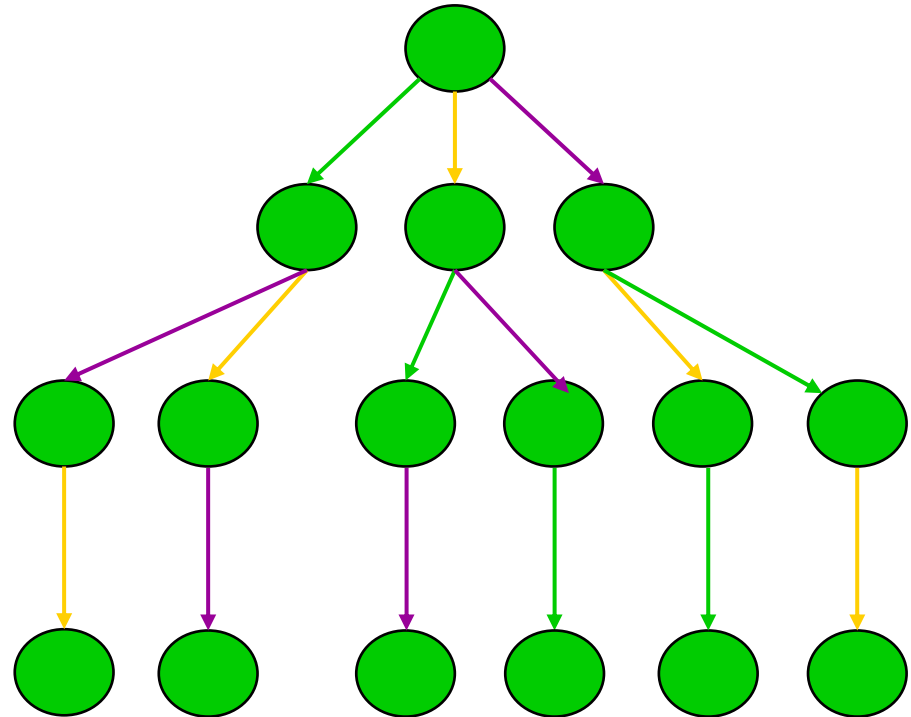


State space

# Problems

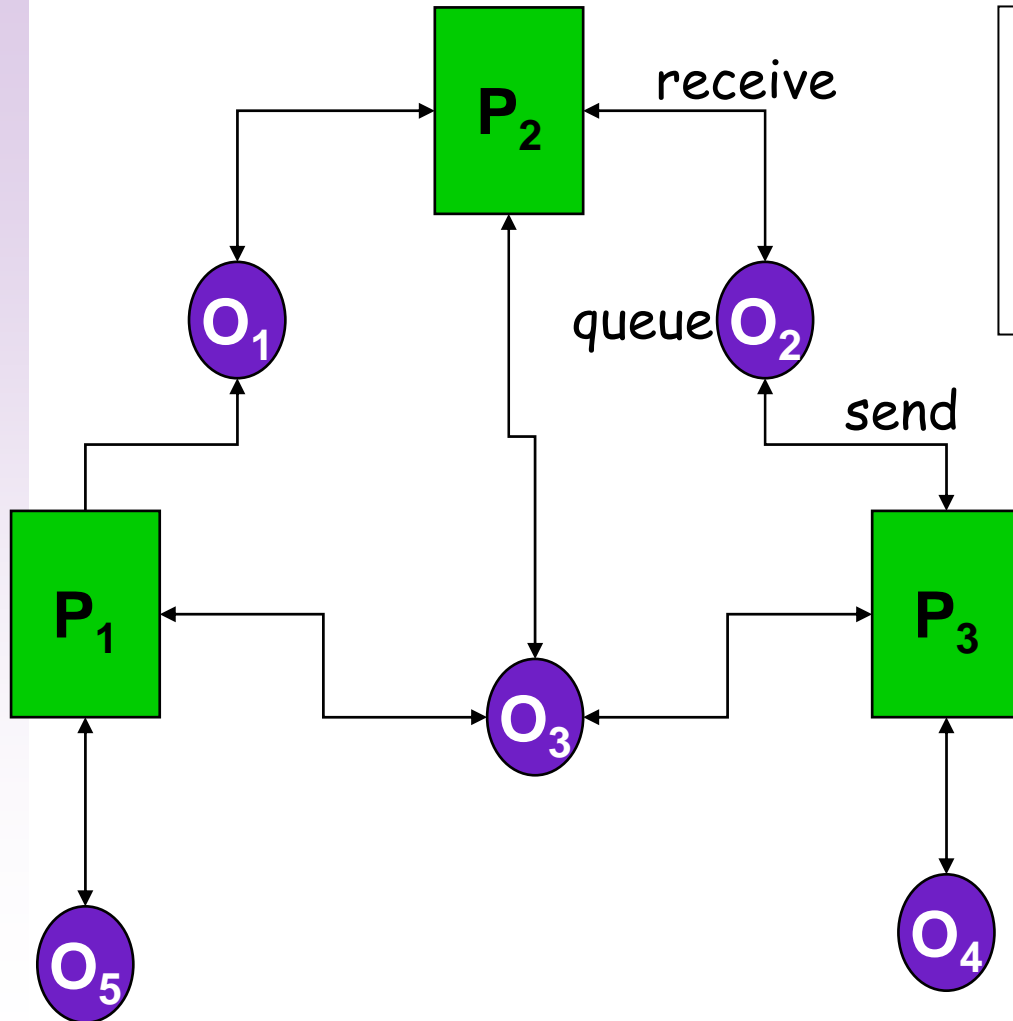


State space



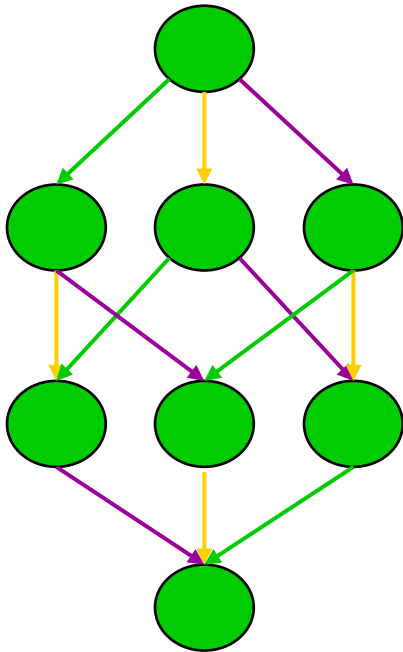
Stateless search

# Partial order reduction



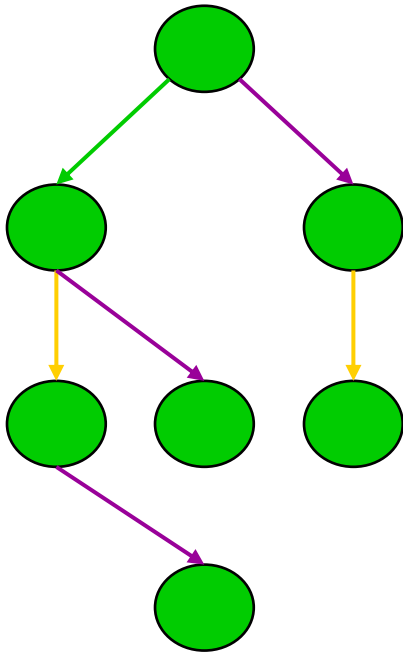
Queue send commutes with queue receive (if queue neither full nor empty)

# Partial order reduction



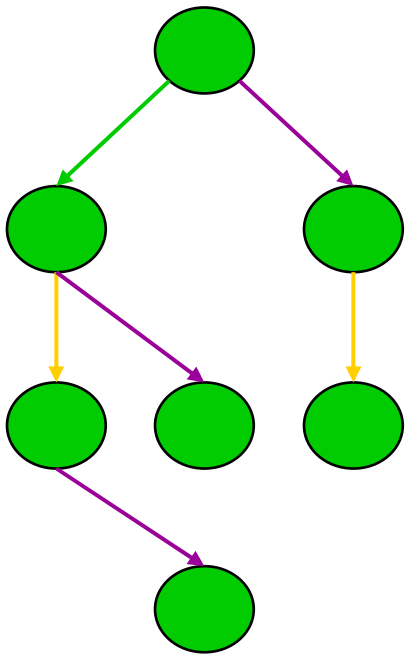
State space

# Partial order reduction

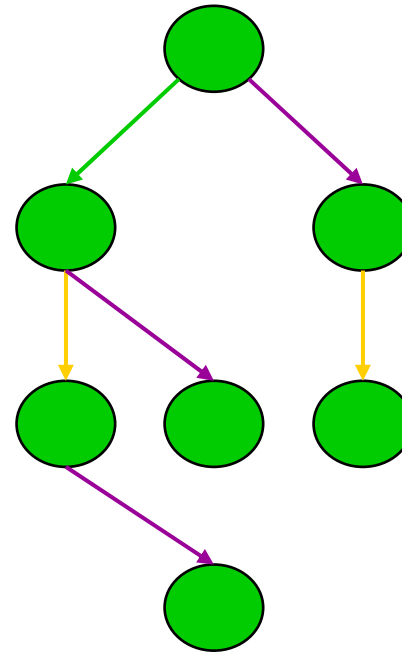


Partial order reduced  
state space

# Partial order reduction



Partial order reduced  
state space



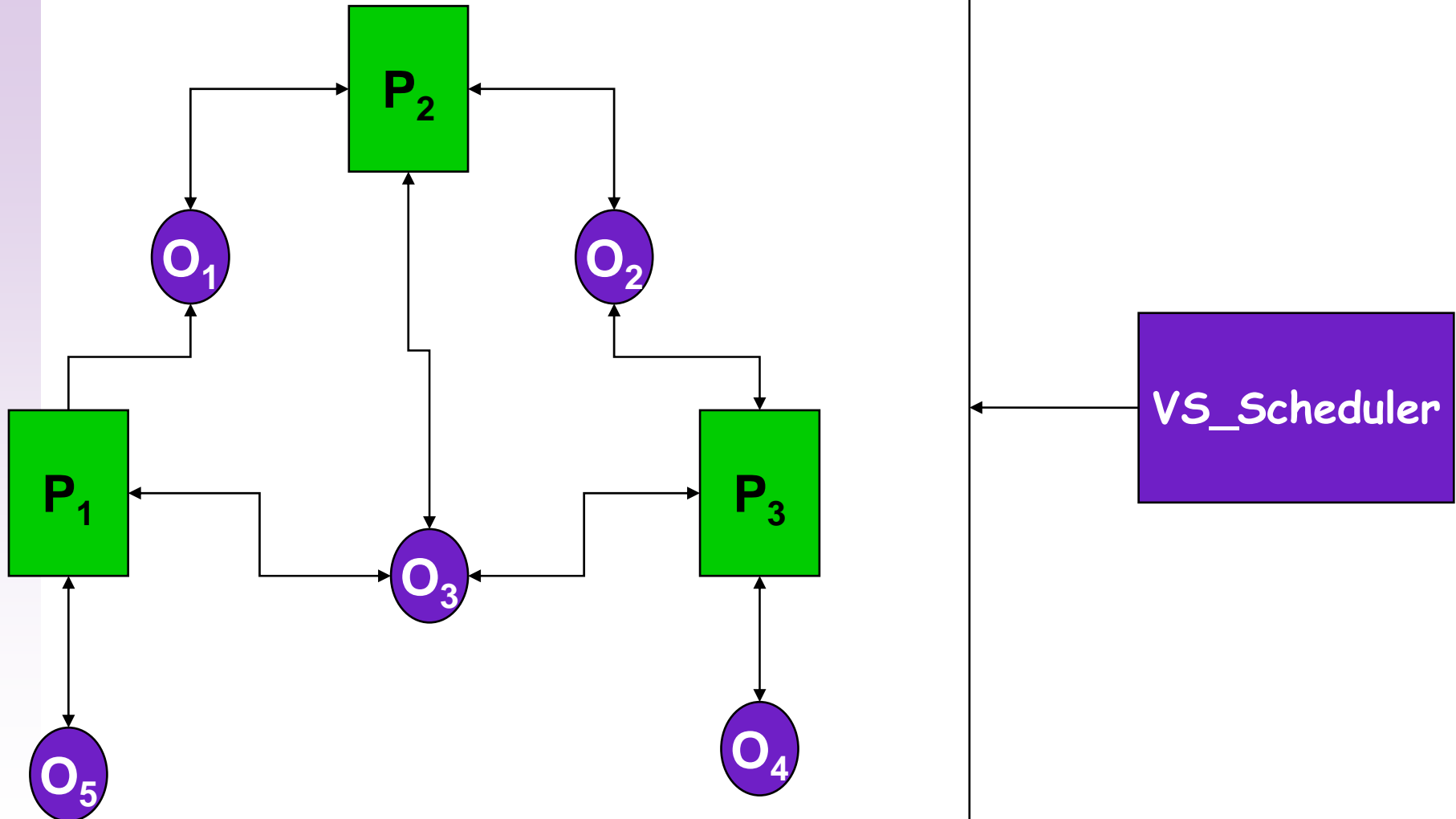
Stateless search

# Insight

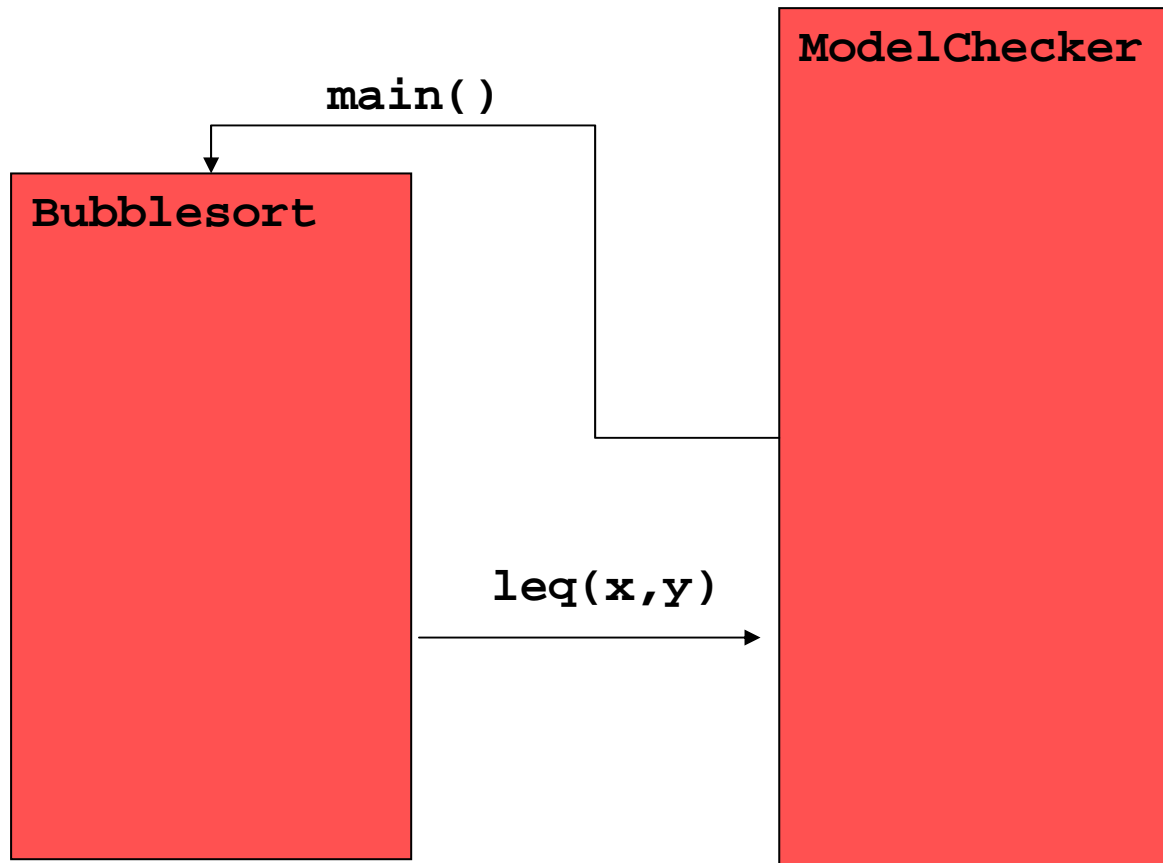
Partial order reduction makes the state space look “mostly” like a tree !

When you traverse a tree, no need to store states (you will visit each state only once!)

# Verisoft Implementation



# Your Implementation



- Your model checker contains key ideas of Verisoft
  - Repeated test runs
  - Controlling (and exhaustively exploring) nondeterministic choices

Mars, July 4, 1997

Lost contact due to real-time priority inversion bug



# Wrapping up ...

- Dawson Engler (Stanford)
  - 2pm Crown 105
  - *Static analysis versus model checking for bug finding*
  - We've seen both - which is better?
- Good luck with homework
  - Email me if you have questions

# Evaluations

- CMPS290G: Topics in Software Engineering
  - Cormac Flanagan
- Volunteer to return forms to BE 227
- Section IV:
  - 1. Pace of course: 1 (too fast) to 5 (too slow).
  - 2. Hours on course per week outside class
    - 1 (0-3hrs)   2 (4-7)   3 (8-11)   4 (12-15)   5 (16-40)
  - 3. Balance of theory and practice
    - 1 (too theoretical)   3 (just right)   5 (too practical)
  - 4. Readings
    - 1 (too much)   3 (okay)   5 (too little)