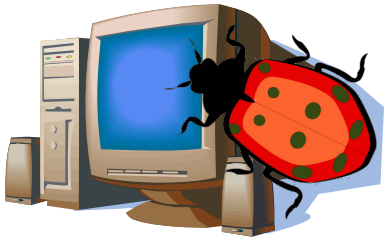
Model Checking a Networked System Without the Network

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Rachid Guerraoui

Maysam Yabandeh

Bugs



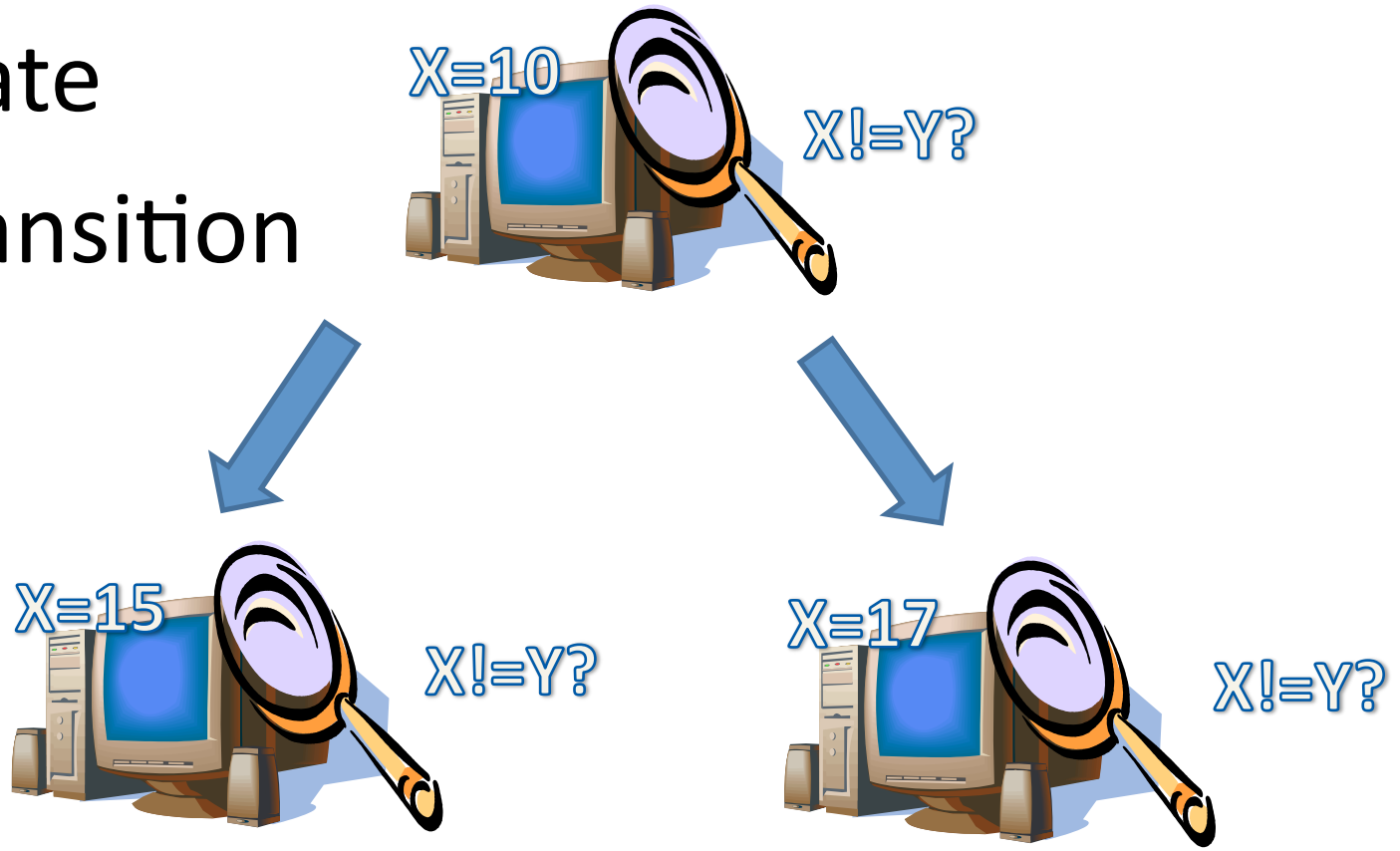
- Testing
  - Looking for the bugs

- Debugging
  - Fixing the found bugs

www.soydemac.com

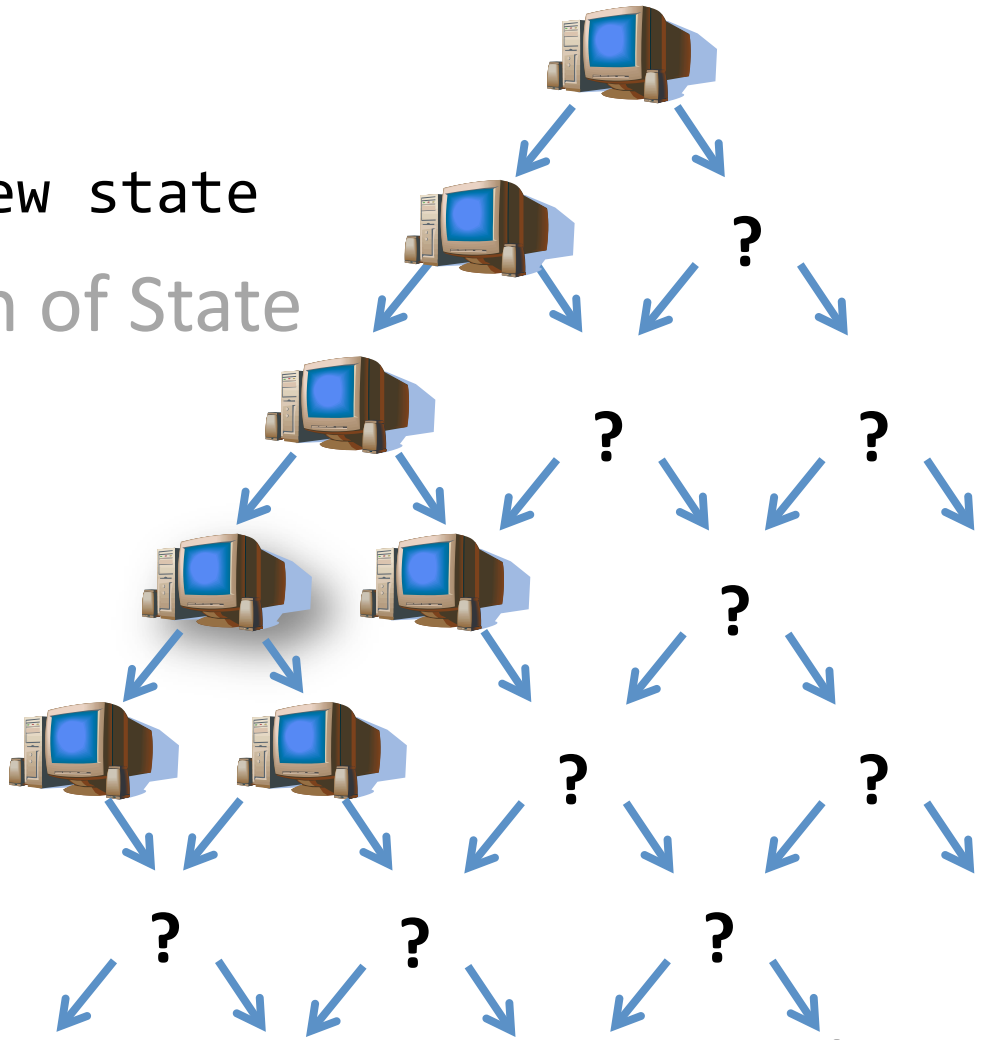
Testing

1. State
2. Transition



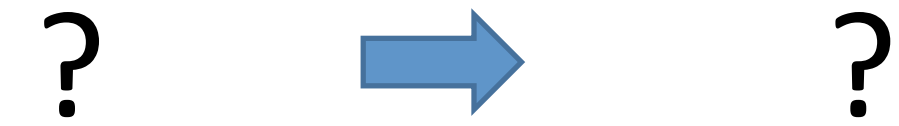
Model Checking (MC)

- `While (…)`
  - `An old state → A new state`
- Exhaustive exploration of State Space



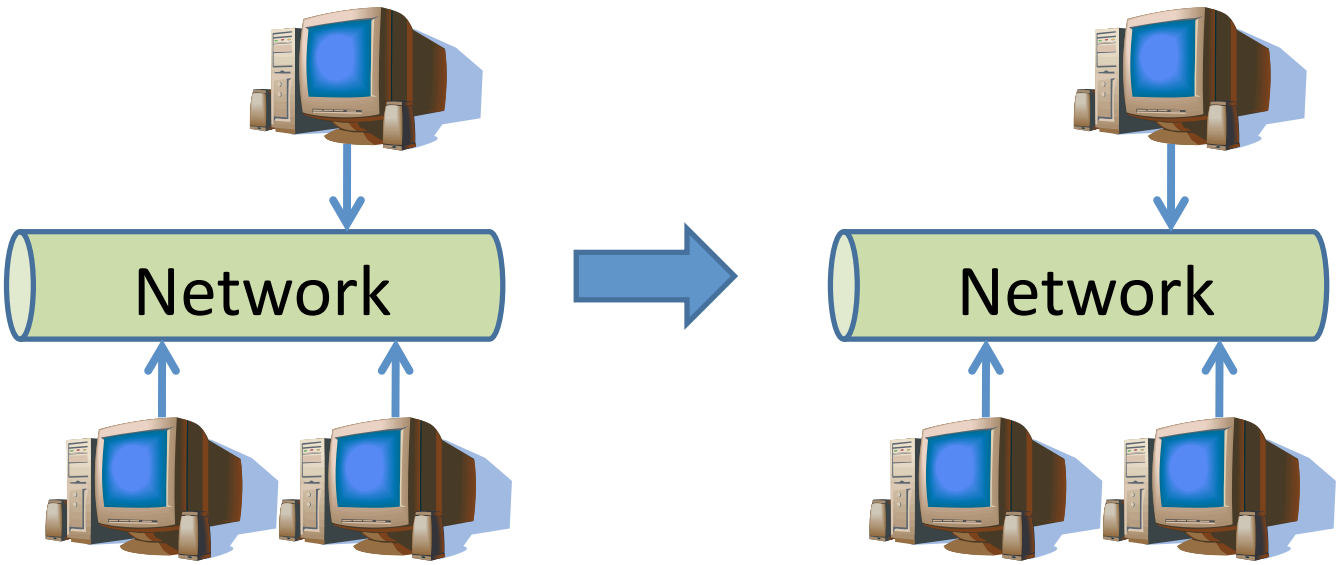
[Classic] MC Distributed Systems

- While (…)
  - An old state → A new state



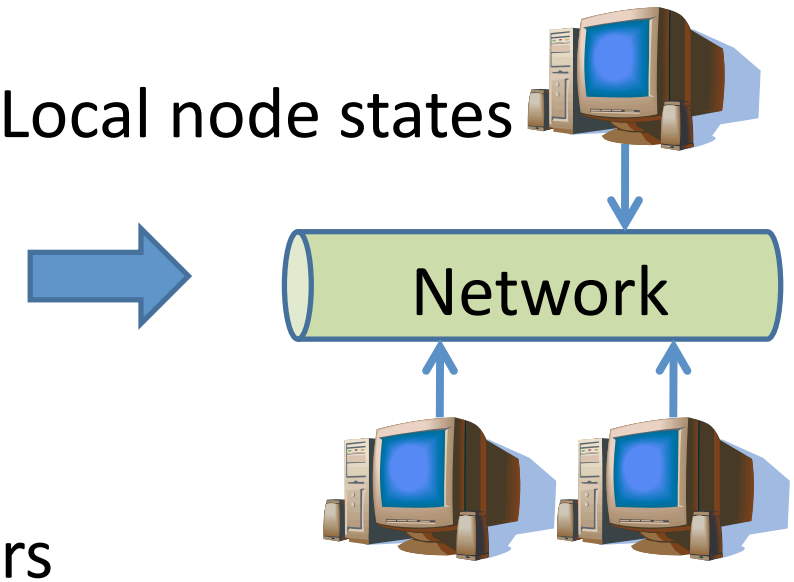
[Classic] MC Distributed Systems

- `While (…)`
  - `An old state → A new state`



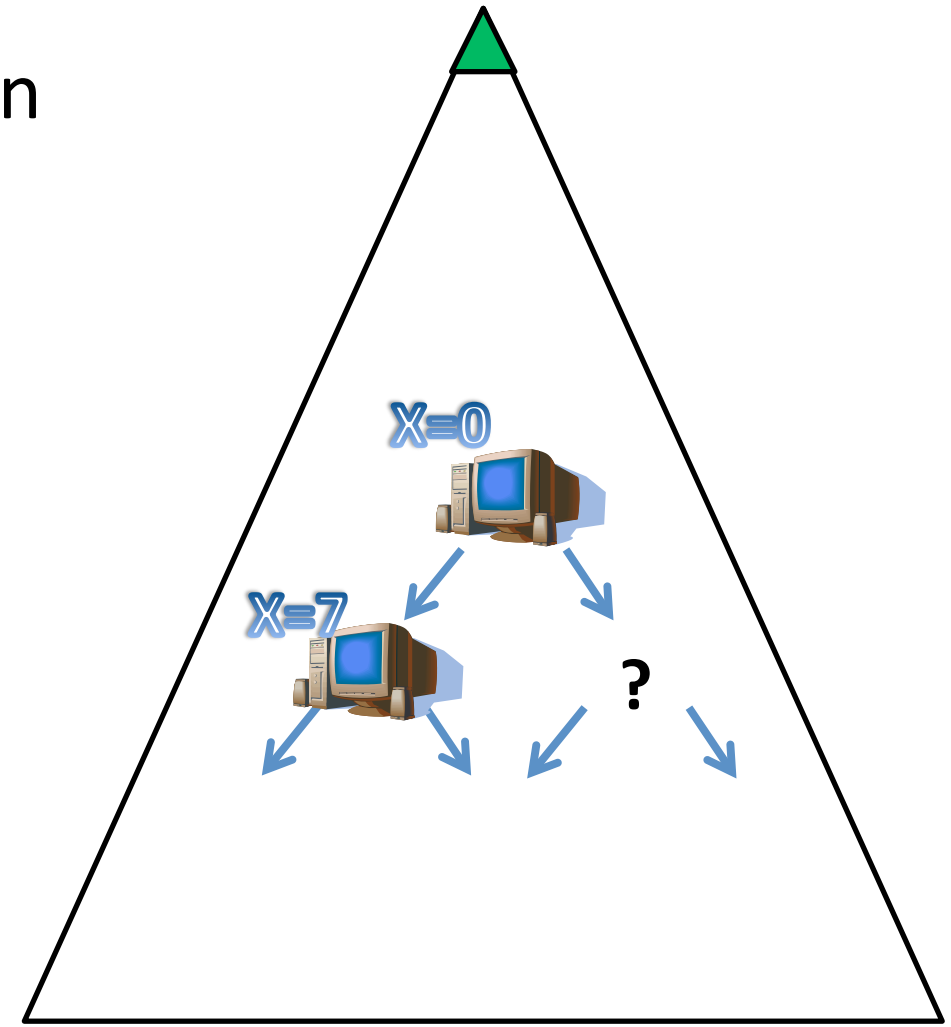
[Classic] MC Distributed Systems

- `While (…)`
  - `An old state → A new state`
- (Global) State:
  - System state, i.e., Local node states
  - + Network state
- Transition:
  - Message handler
  - Local events: timers

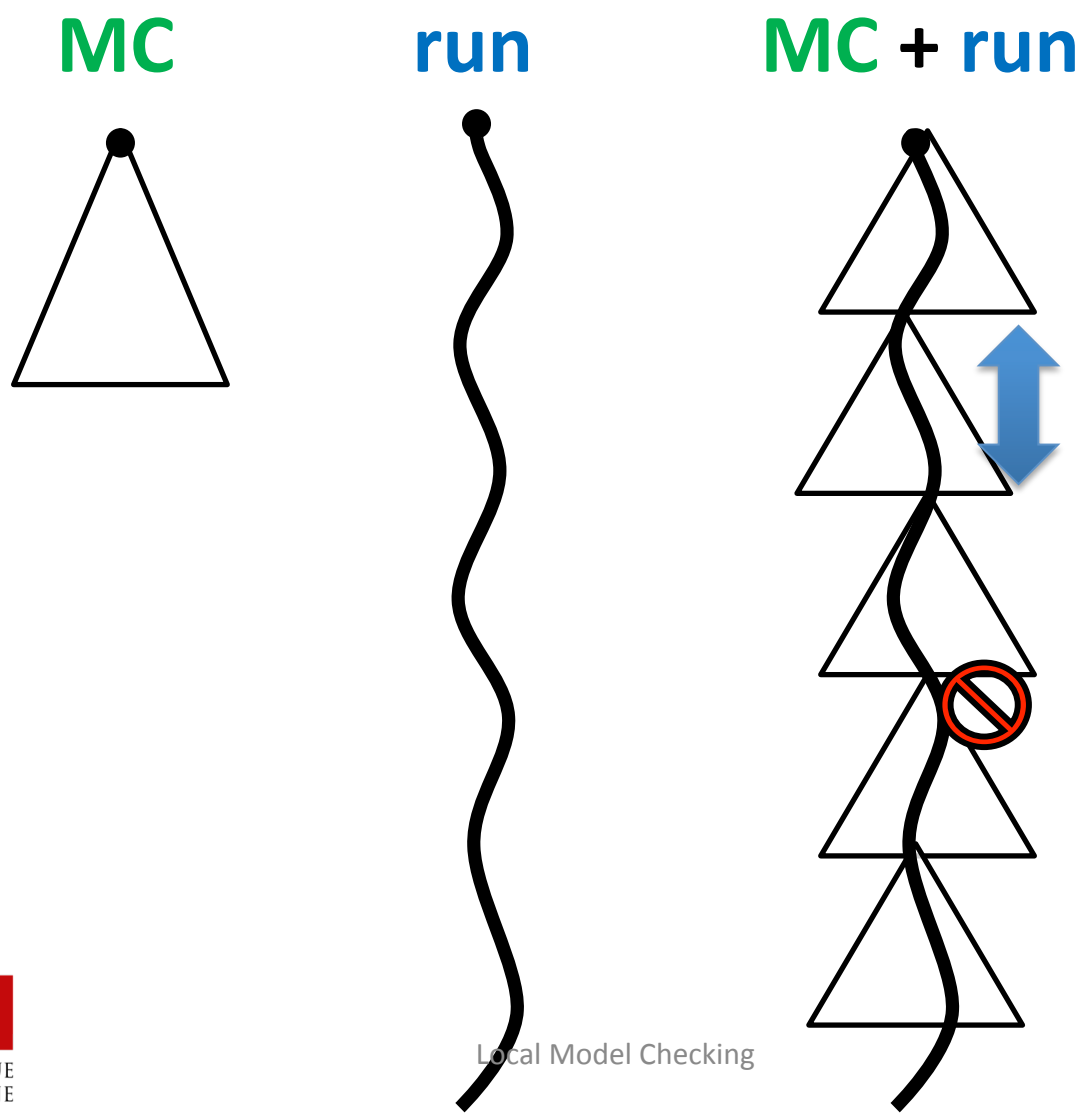


State Space Coverage

- Exponential Explosion
- ~~Exhaustive~~

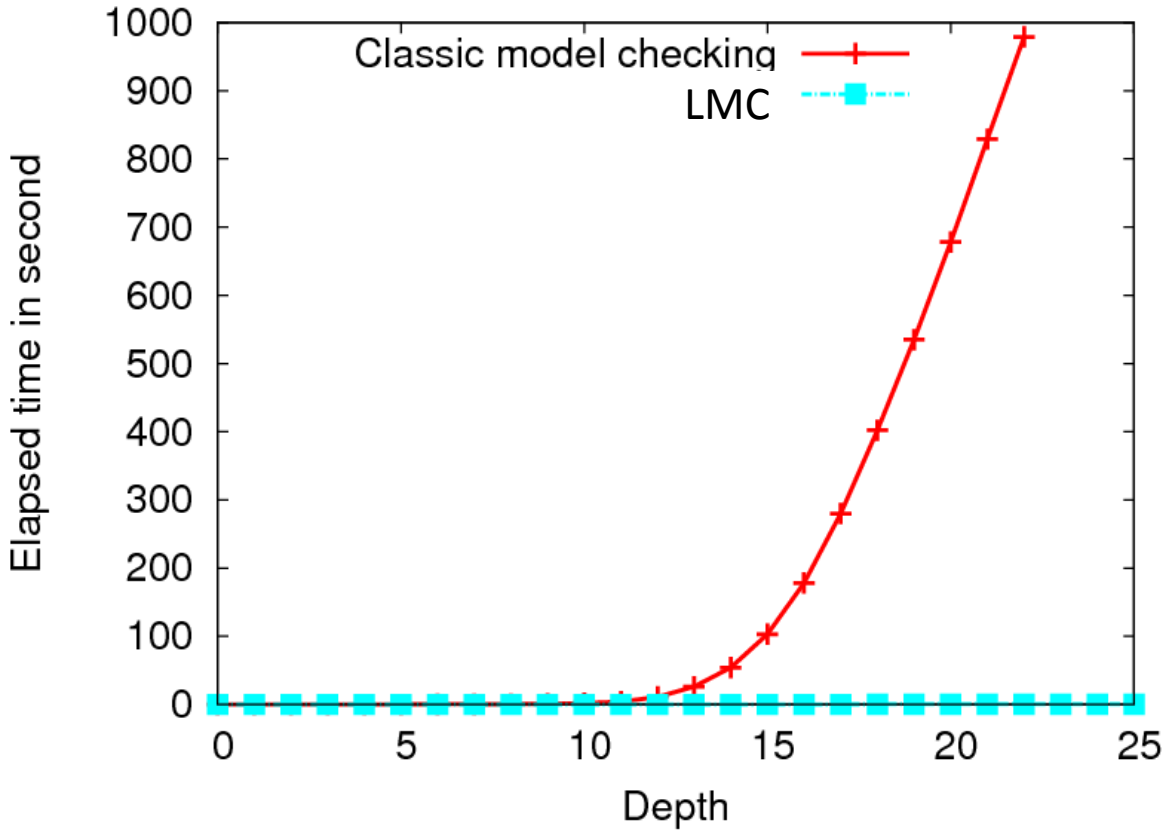


Online Model Checking [NSDI'09]



How about Paxos?

- Simple state space: **one** local event (propose)

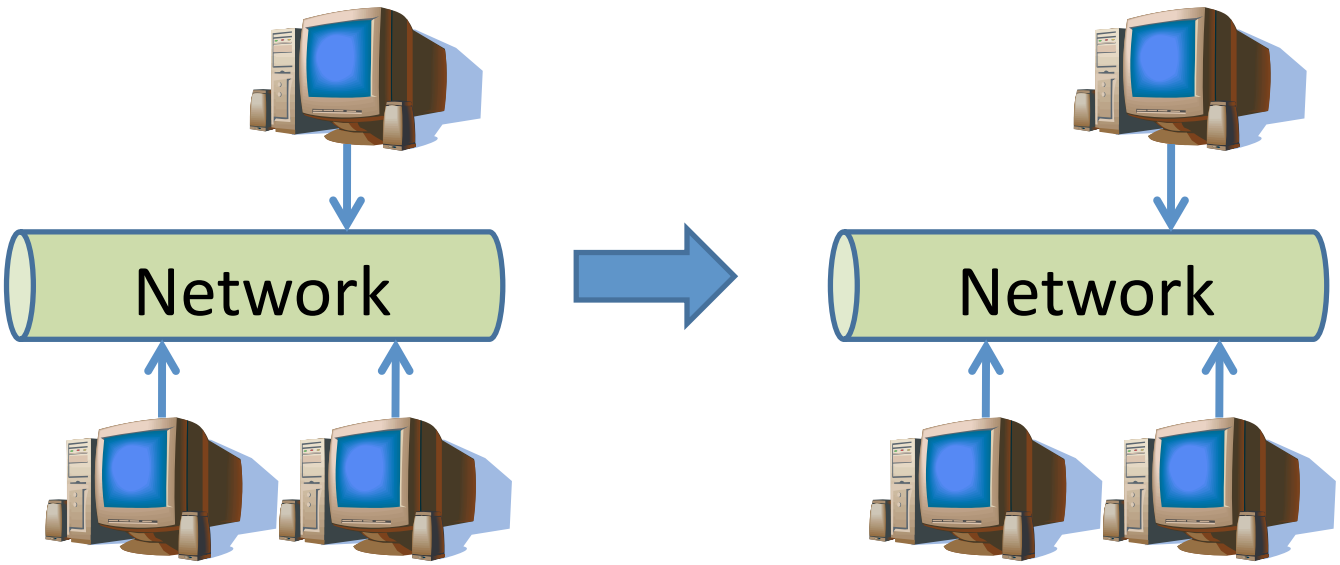


Agenda

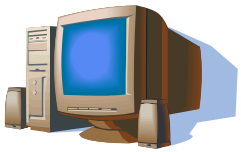
1. Classic [Global] Model Checking
2. What is the problem?
3. Local Model Checking (LMC)

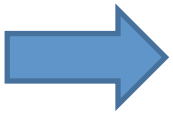
Global Model Checking

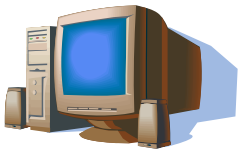
- Huge global state size



Local ~~Global~~ Model Checking (LMC)

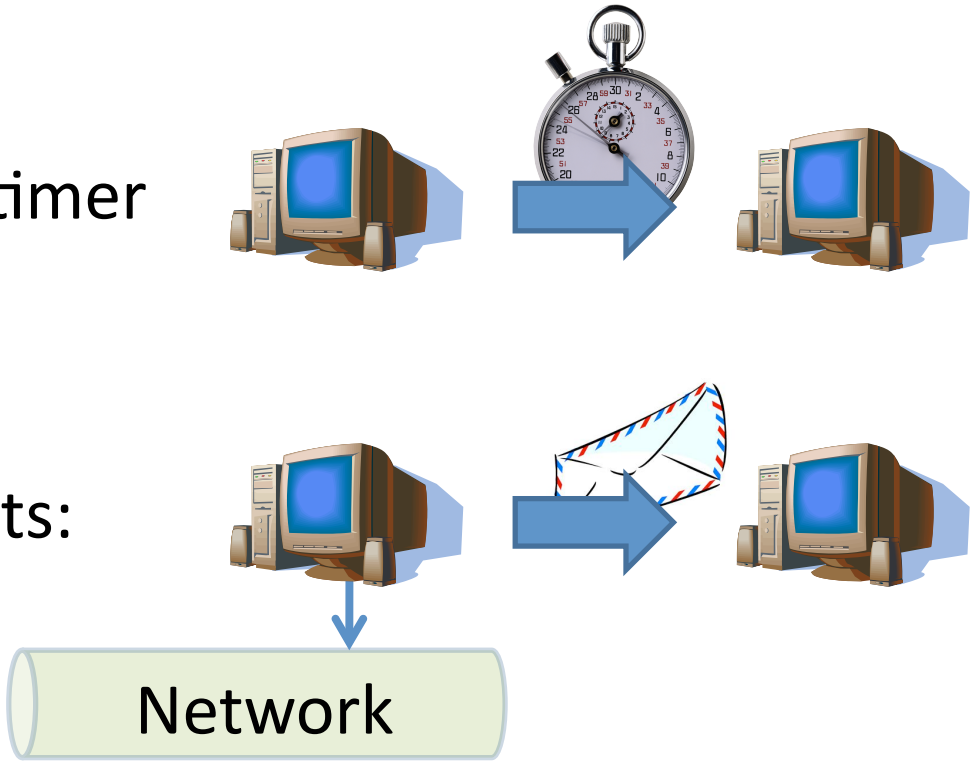




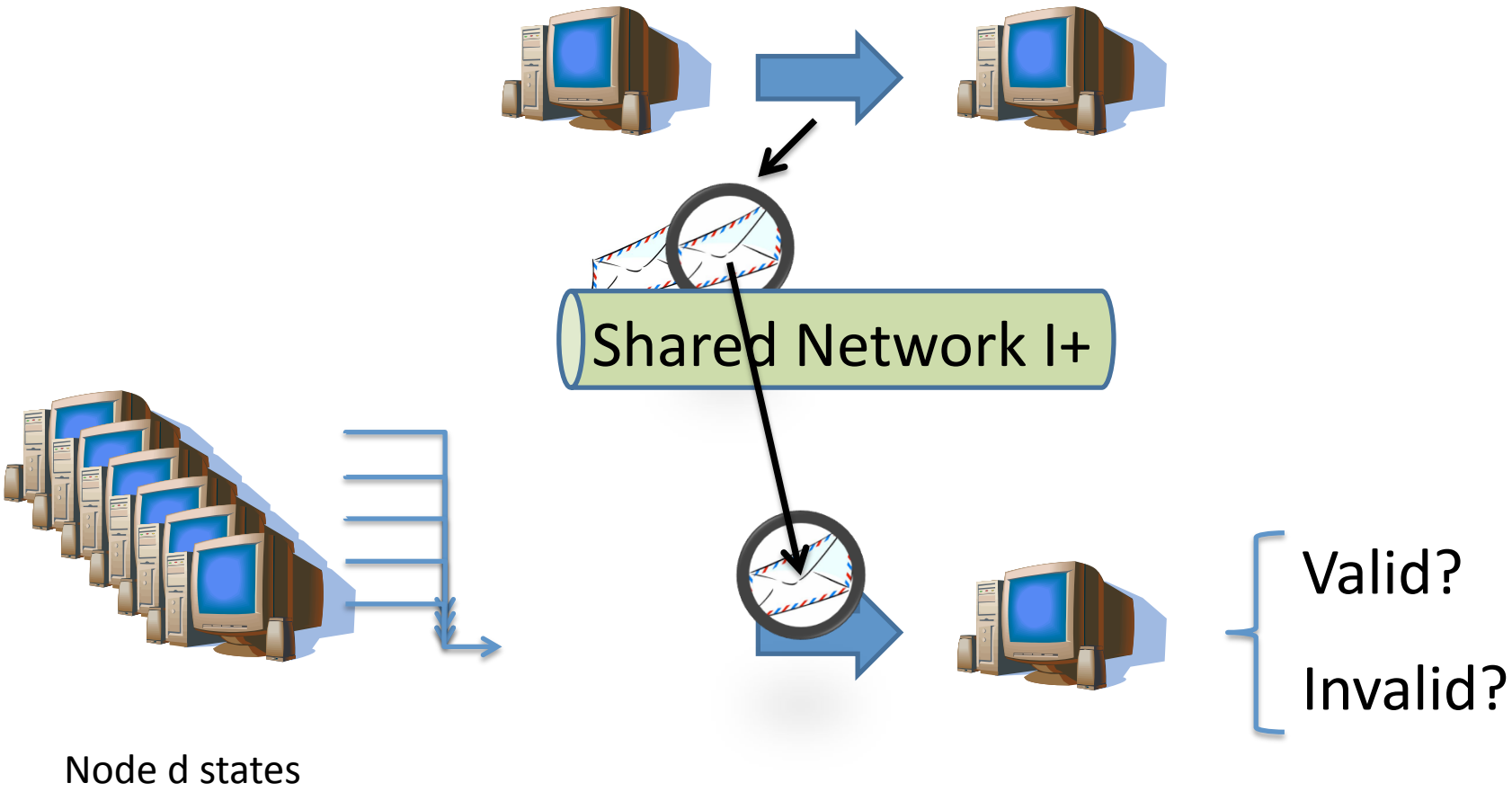


Local Model Checking (LMC)

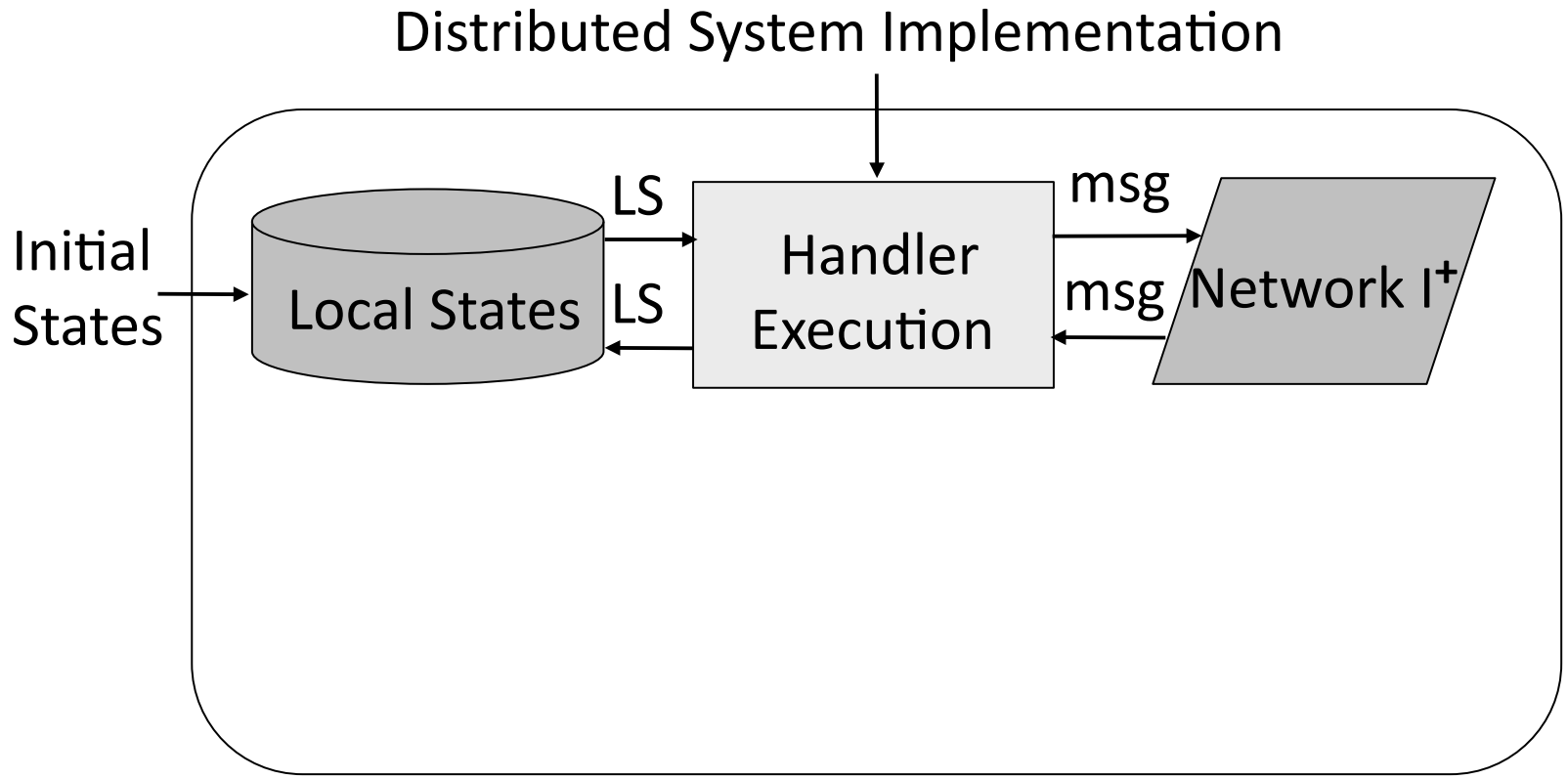
- Transition?
  - Local events: timer
  - Network events:



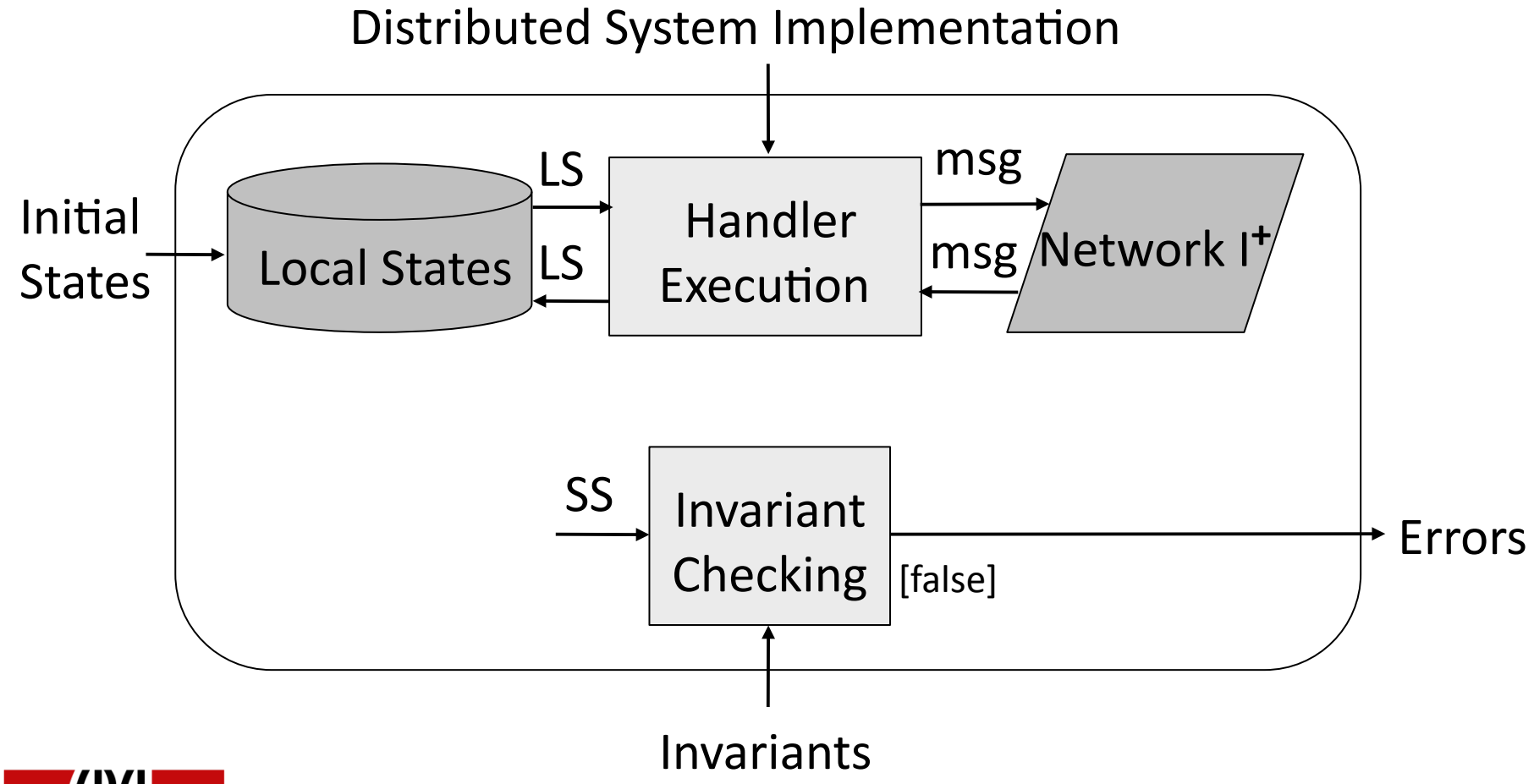
LMC: Shared Network



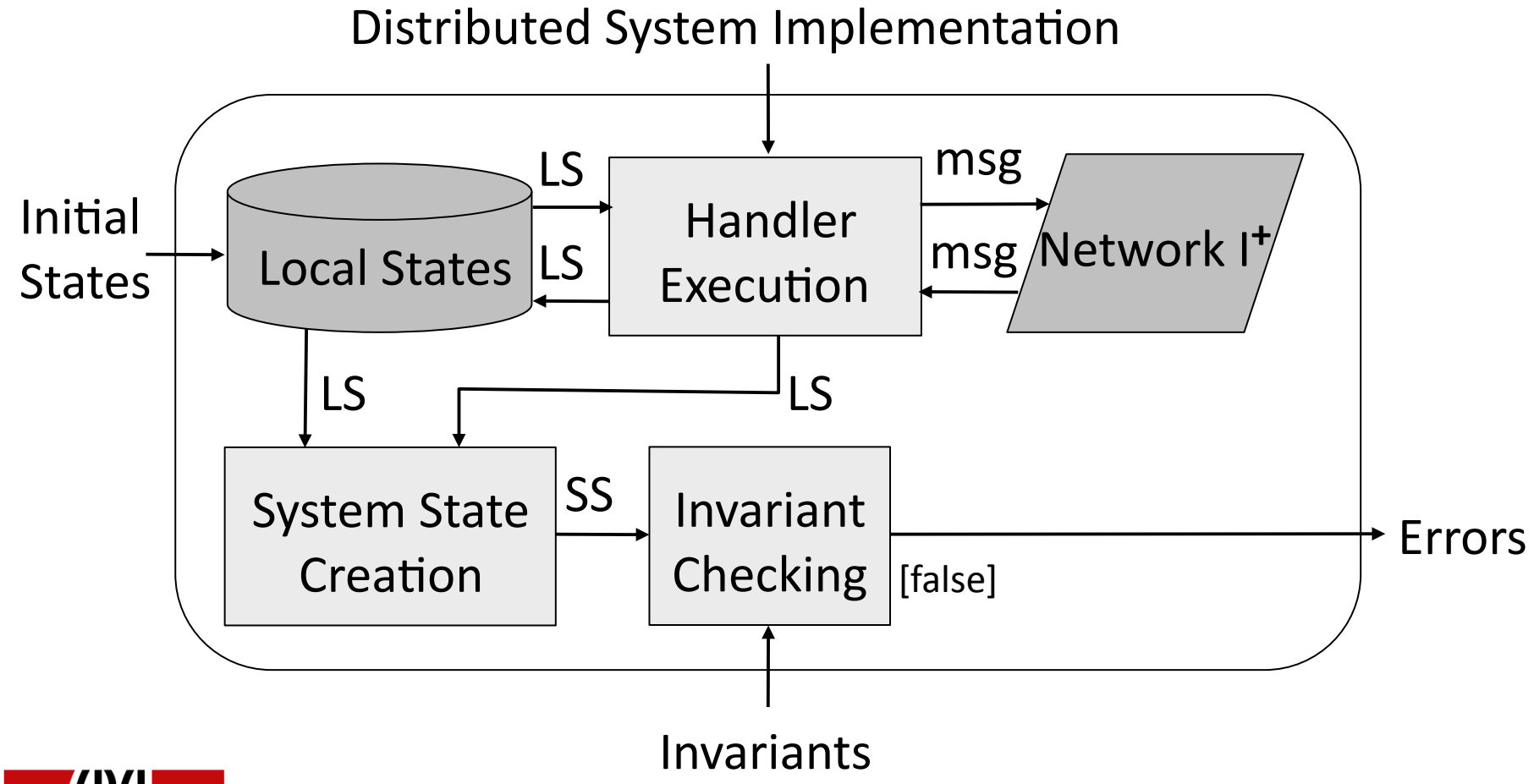
LMC: Architecture



LMC: Architecture

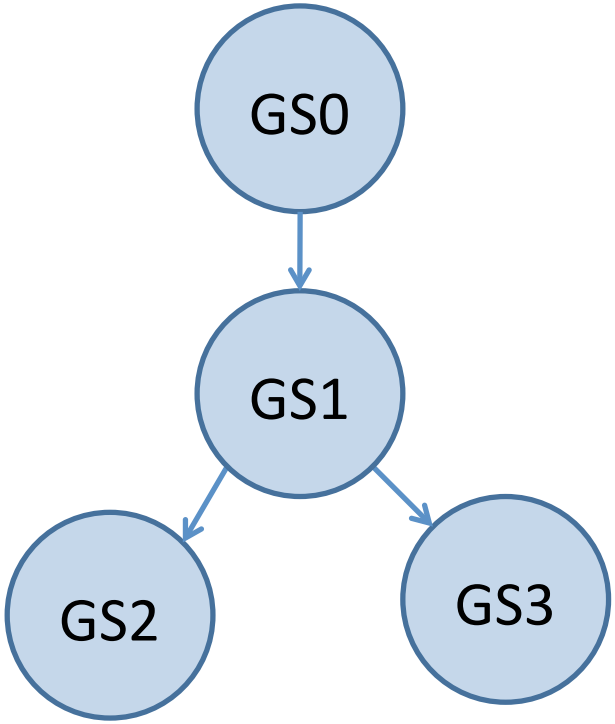


LMC: Architecture

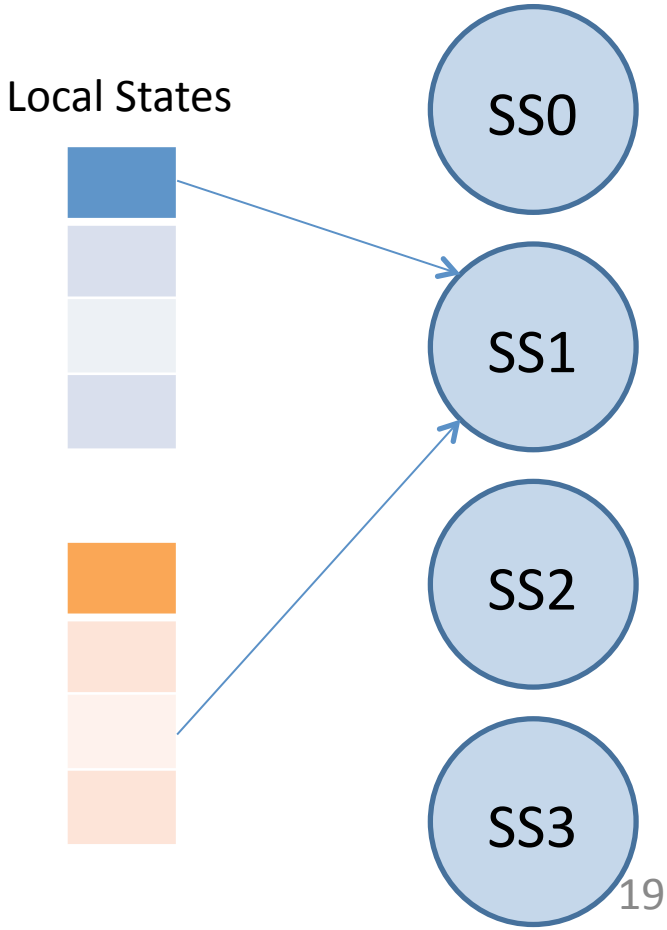


LMC: System State Creation

Global Model Checking



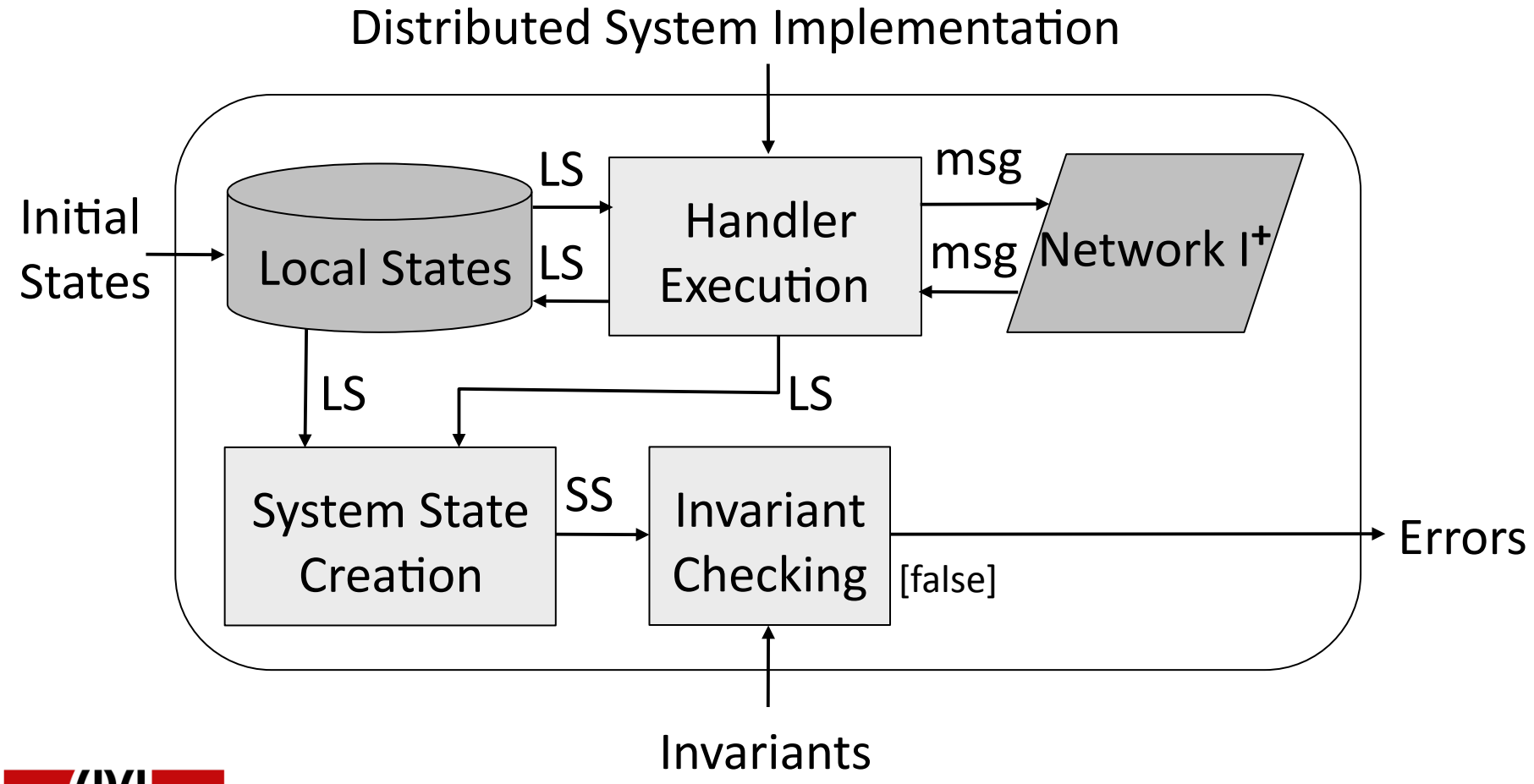
Local Model Checking



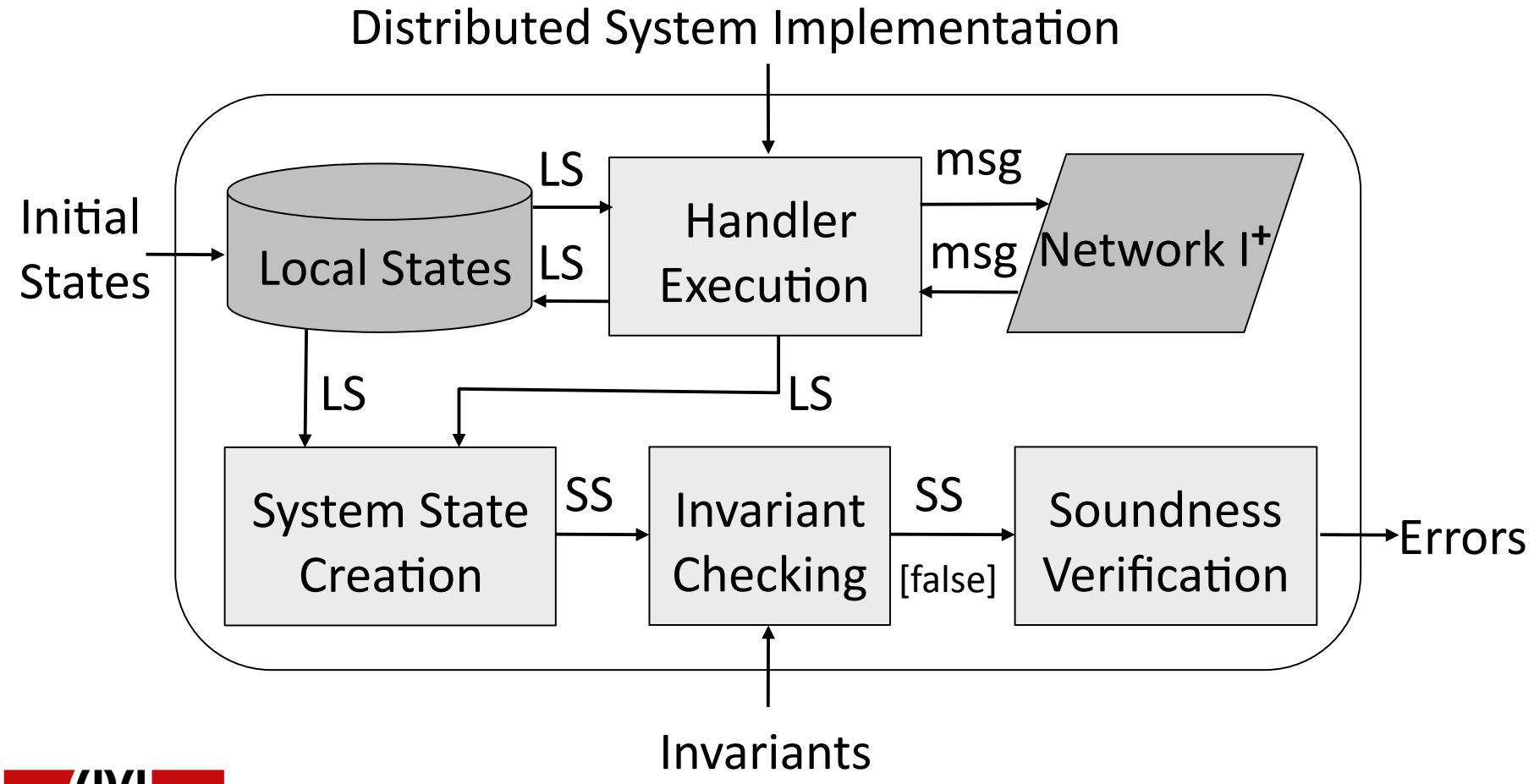
LMC: System State Creation

- System States =
  - Combination of all local states: LMC-GEN
  - Only when there is a chance of bug
    - Paxos: different chosen values: LMC-OPT
- Complete
- Invalid states → Unsound bug report

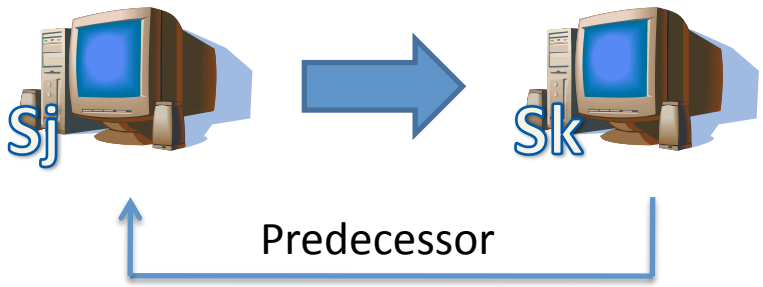
LMC: Architecture

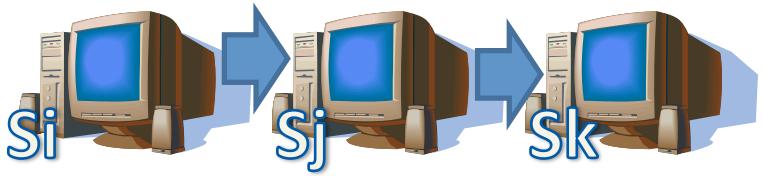


LMC: Architecture

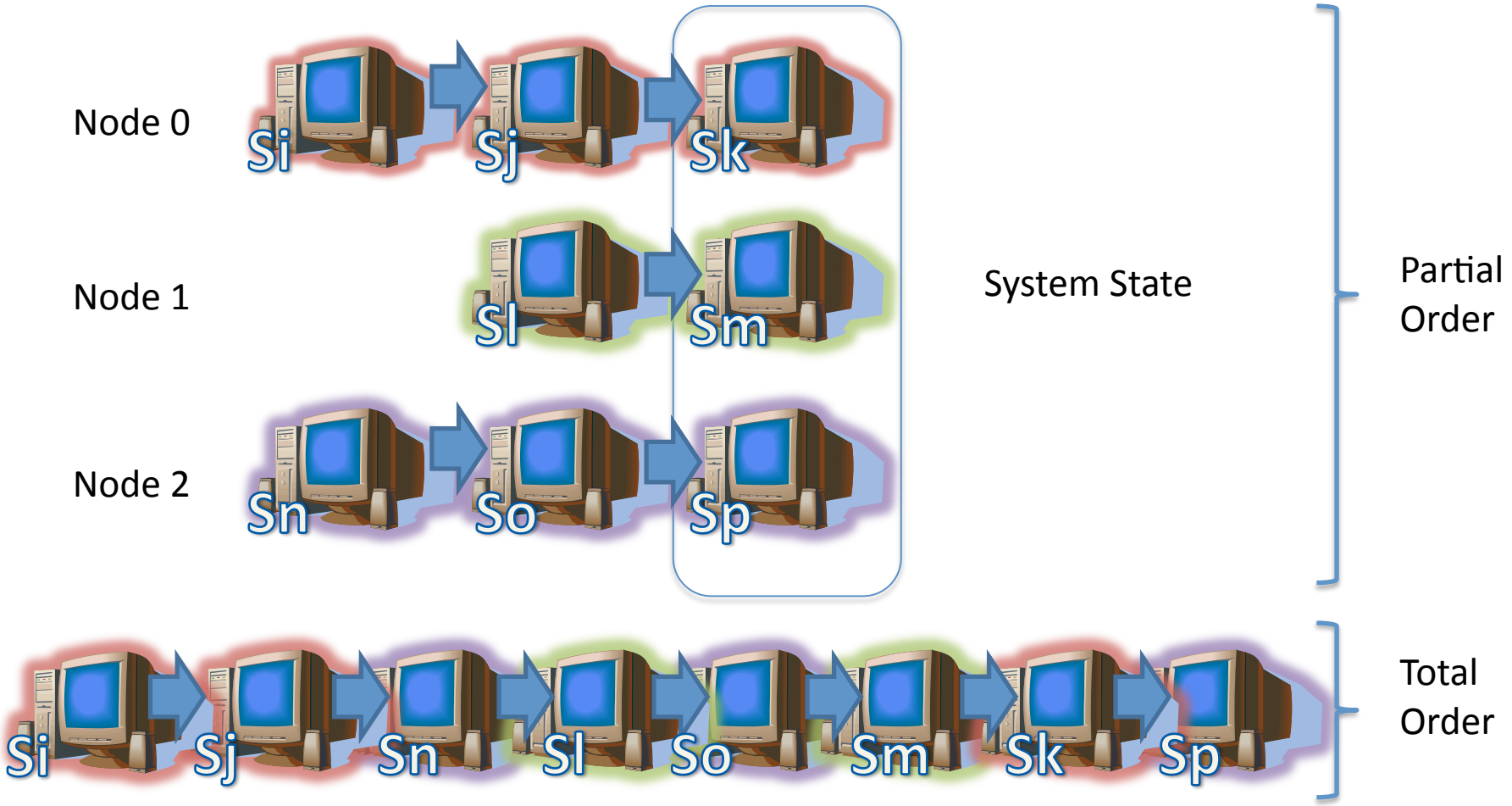


Soundness Verification





Soundness Verification



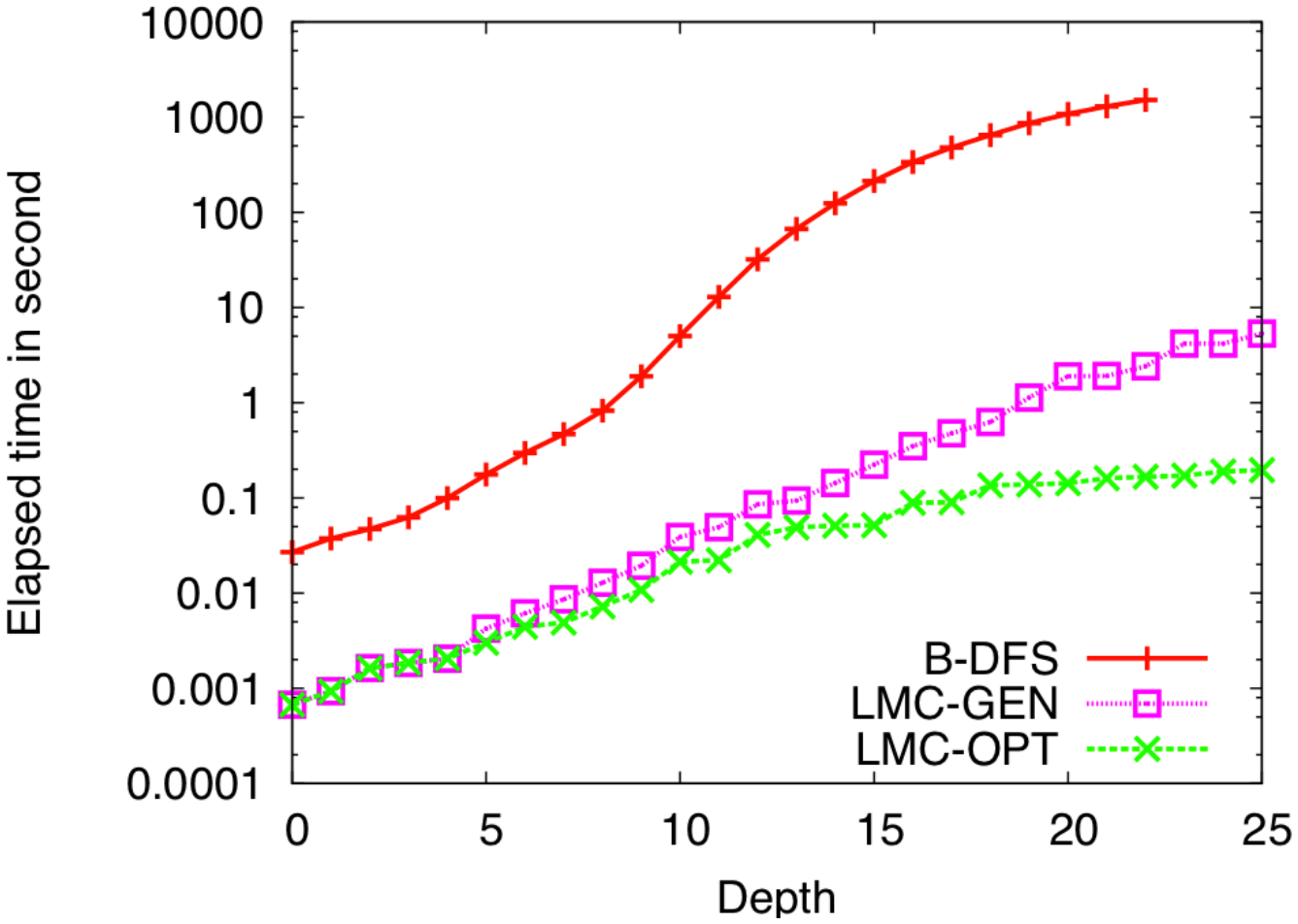
Soundness Verification

- Input: set of event stacks
- Output: valid or invalid
- Greedy Algorithm:
  - `While exist enabled event e = stack`$_i$`.top()`
    - `Run e`
    - `stack`$_i$`.pop()`
  - `Return valid if there is no event left`

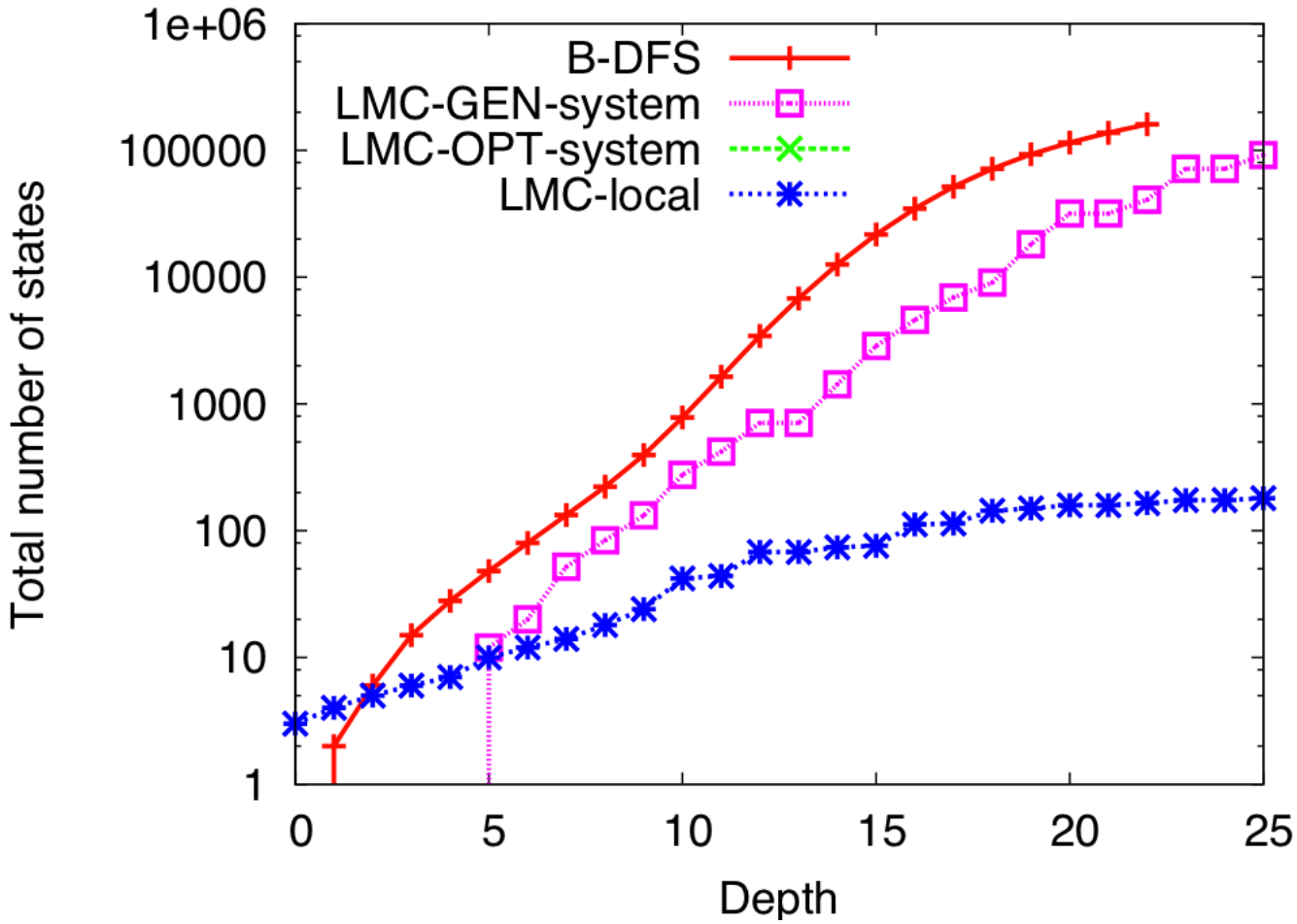
Evaluation

- Prototype using MaceMC [NSDI '07]
  - Mace: structured C++ programs
  - (De)Serialization of states
  - Handler boundary → events
- Paxos: Nodes agree on the same values
  - A very complex distributed algorithm
  - Lots of communication at each round
- State space: 3 Nodes, 1 Proposes
- B-DFS: memory efficient MC [NSDI '07]
  - Bounded Depth First Search
  - Maintain a DB of state hashes to avoid loops
- 3 GHz Intel® Pentium® 4 CPU, 1 MB L2 cache

Evaluation: Elapsed Time

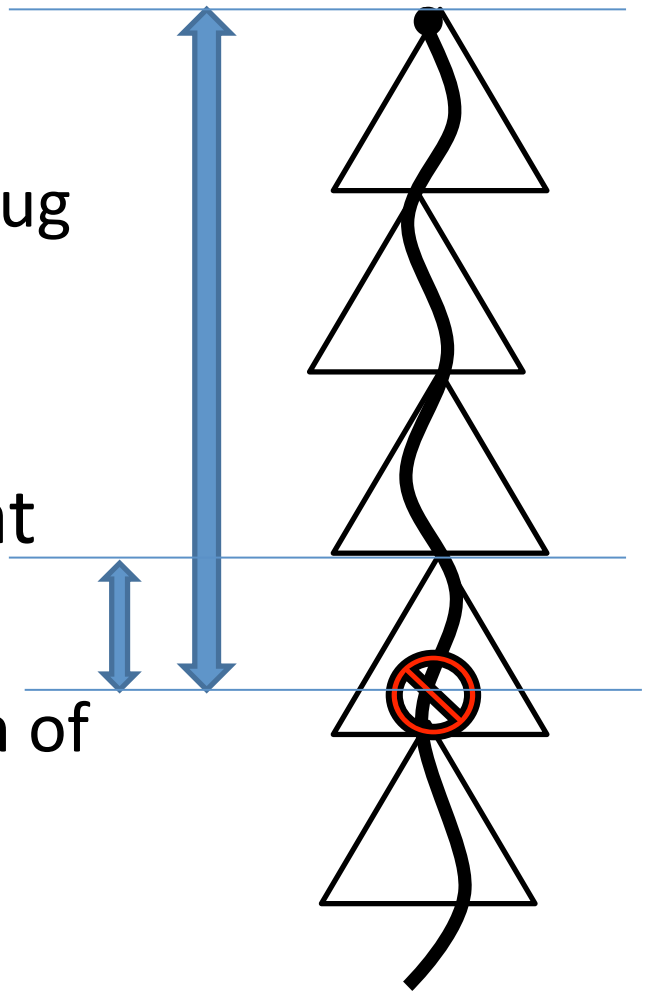


Evaluation: # of states



Testing

- Paxos:
  - Injected a previously reported bug [Liu et al., NSDI'07]
  - Rediscovers the bug
- 1Paxos: an efficient variant that uses one acceptor
  - Found a new bug in initialization of acceptor



Related Work

LMC = CA + MA

- CA: Cartesian Abstraction [Ball, et. al, **2001**] [Flanagan and Qadeer, **2003**][Henzinger et. al, **2003**] [Malkis et. al, **2006**] [Malkis et. al, **2010**]
- Thread-modular MC
  - Local asserts → no system state creation
  - Complete, Unsound → Theorem proving vs. Testing
- MA: Monotonic Abstraction [Mitchell, **2002**]
  - Shared Network

Summary

Local ~~Global~~ Model Checking

- Local states, Shared network state
- Postpones State Explosion Problem
- Optimistic
- Decouples Exploration, System State Creation, and Soundness Verification
  - In Parallel
  - LMC-OPT (generalize?)

Thank you …