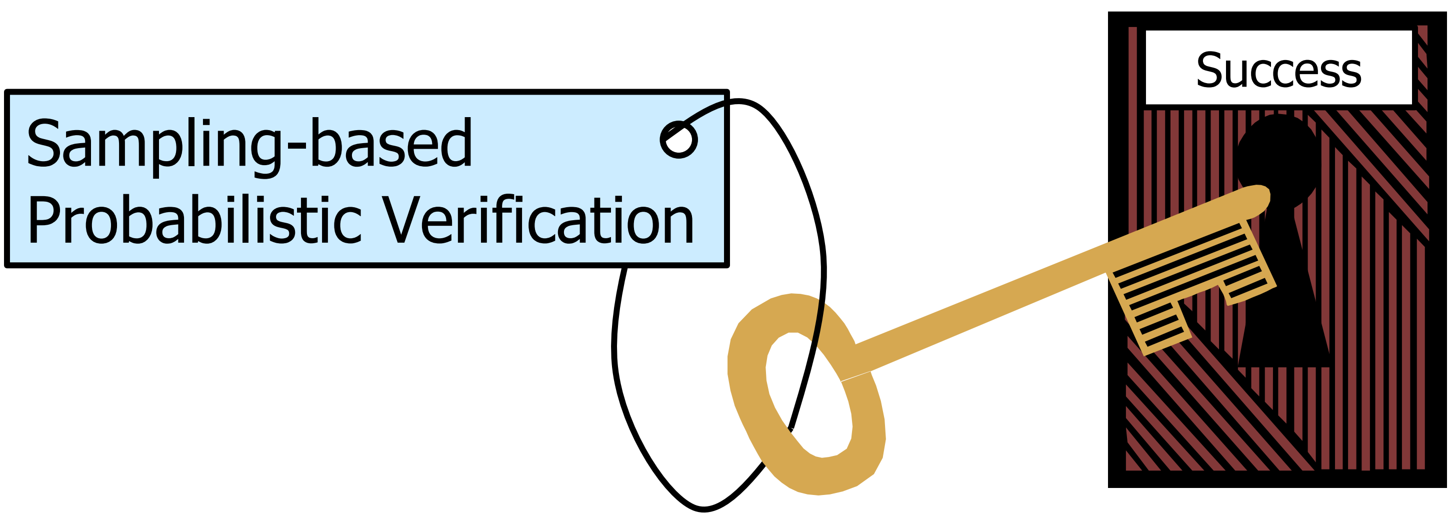
Planning with Concurrency in Continuous-time Stochastic Domains (thesis proposal)

Håkan L. S. Younes

Thesis Committee:
Reid G. Simmons (chair)
Geoffrey J. Gordon
Jeff Schneider
David J. Musliner, Honeywell

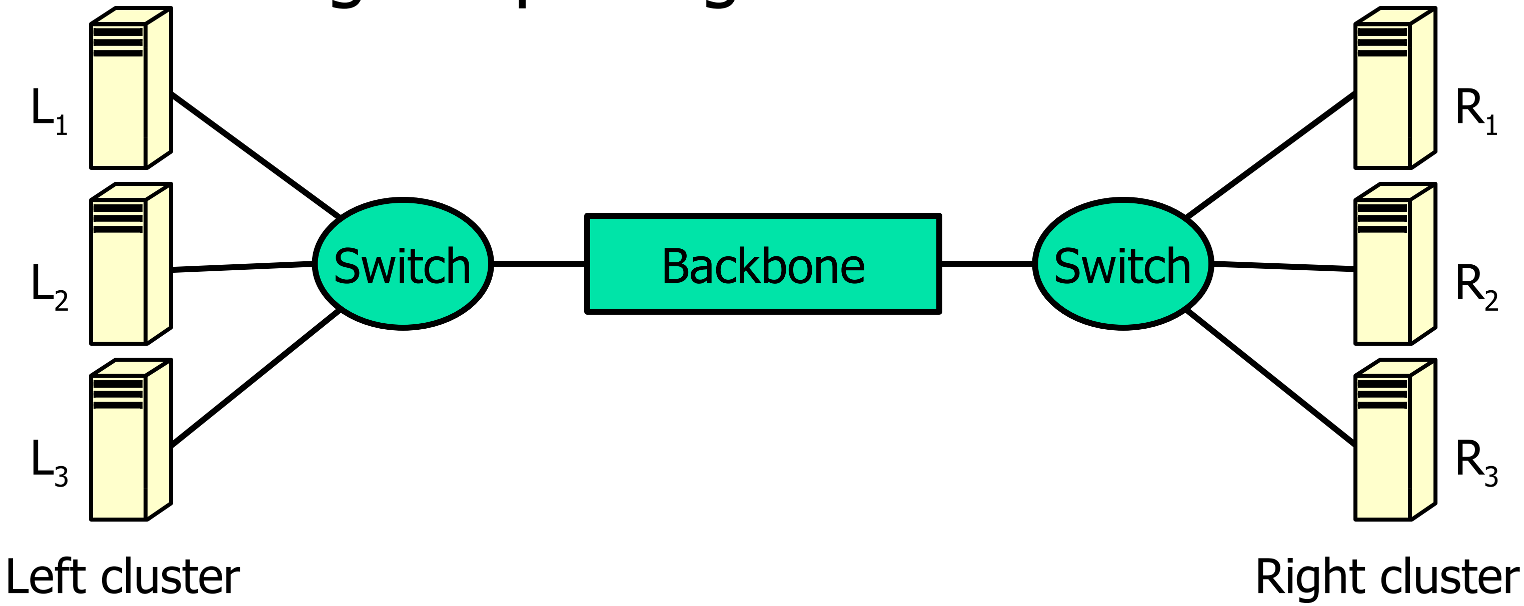
Aim of Thesis Research

- Policy generation for continuous-time stochastic systems with concurrency



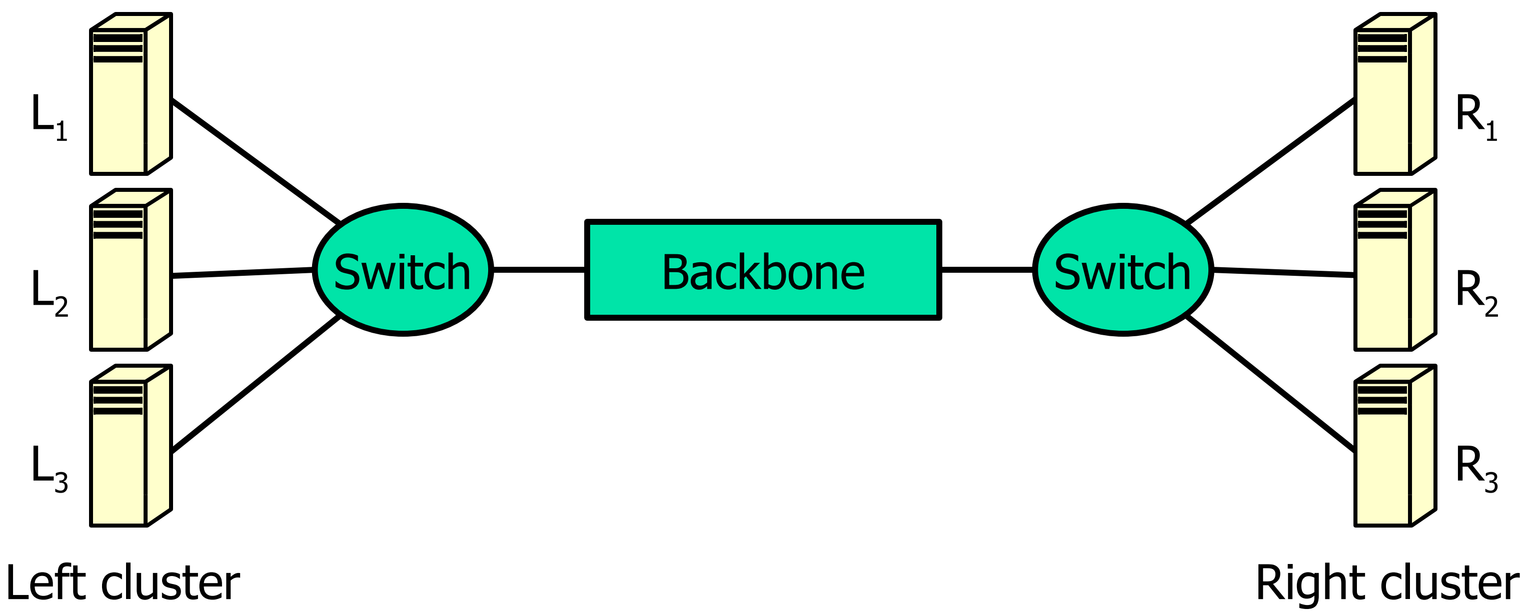
Motivating Example

- Dependable workstation cluster
  - Components may fail at any point in time
  - Single repair agent

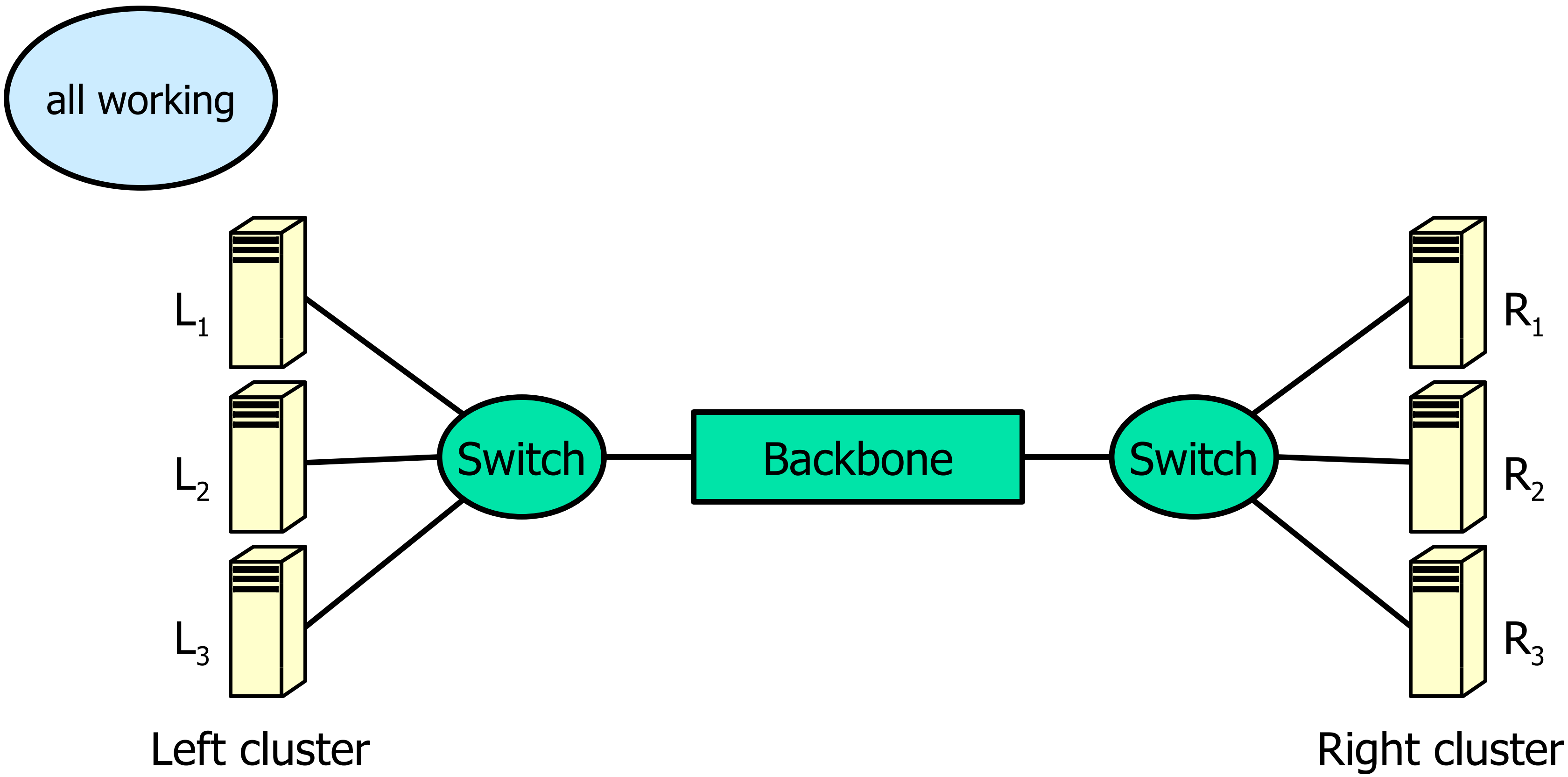


Motivating Example

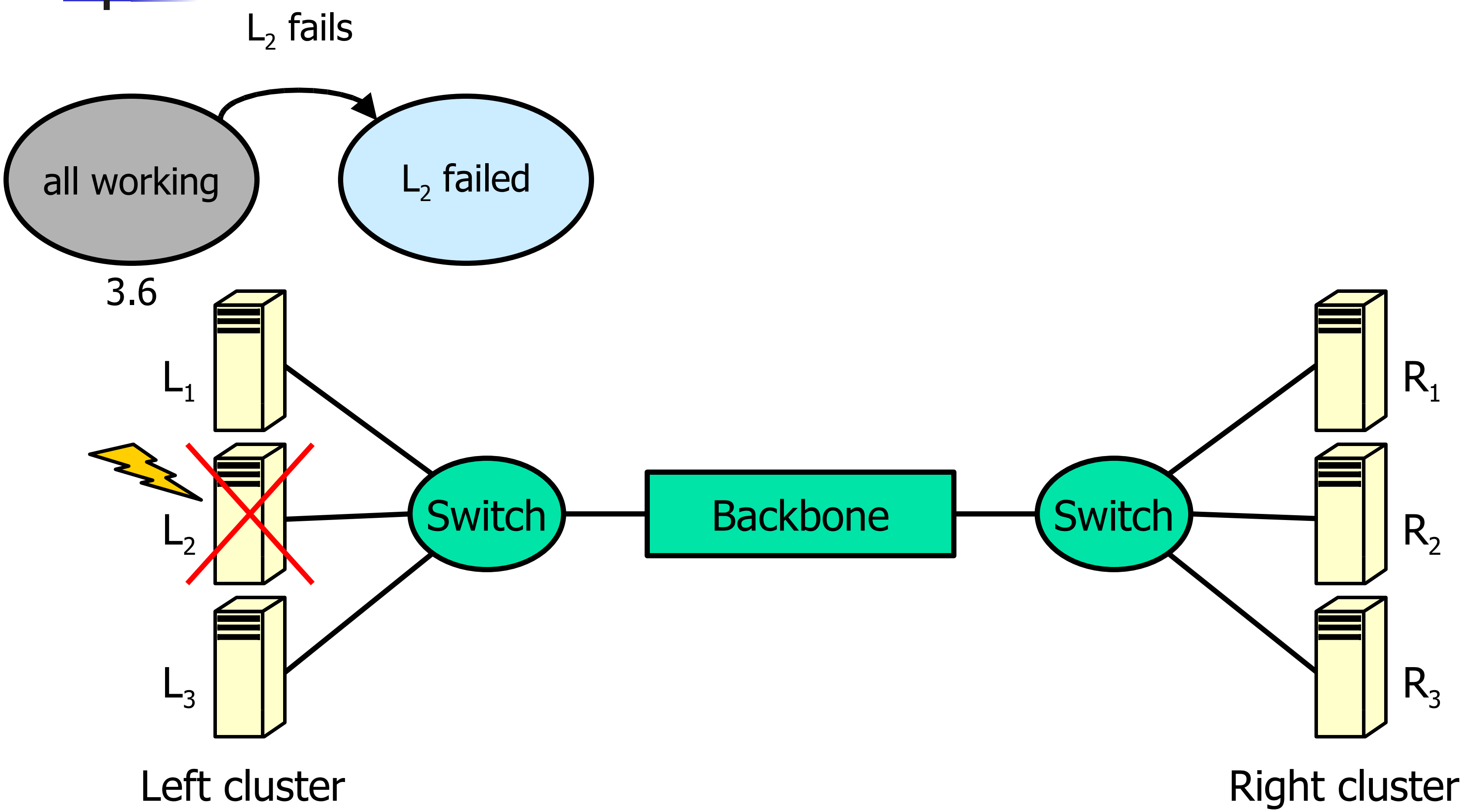
- Planning problem:
  - Find repair policy that maintains satisfactory quality of service



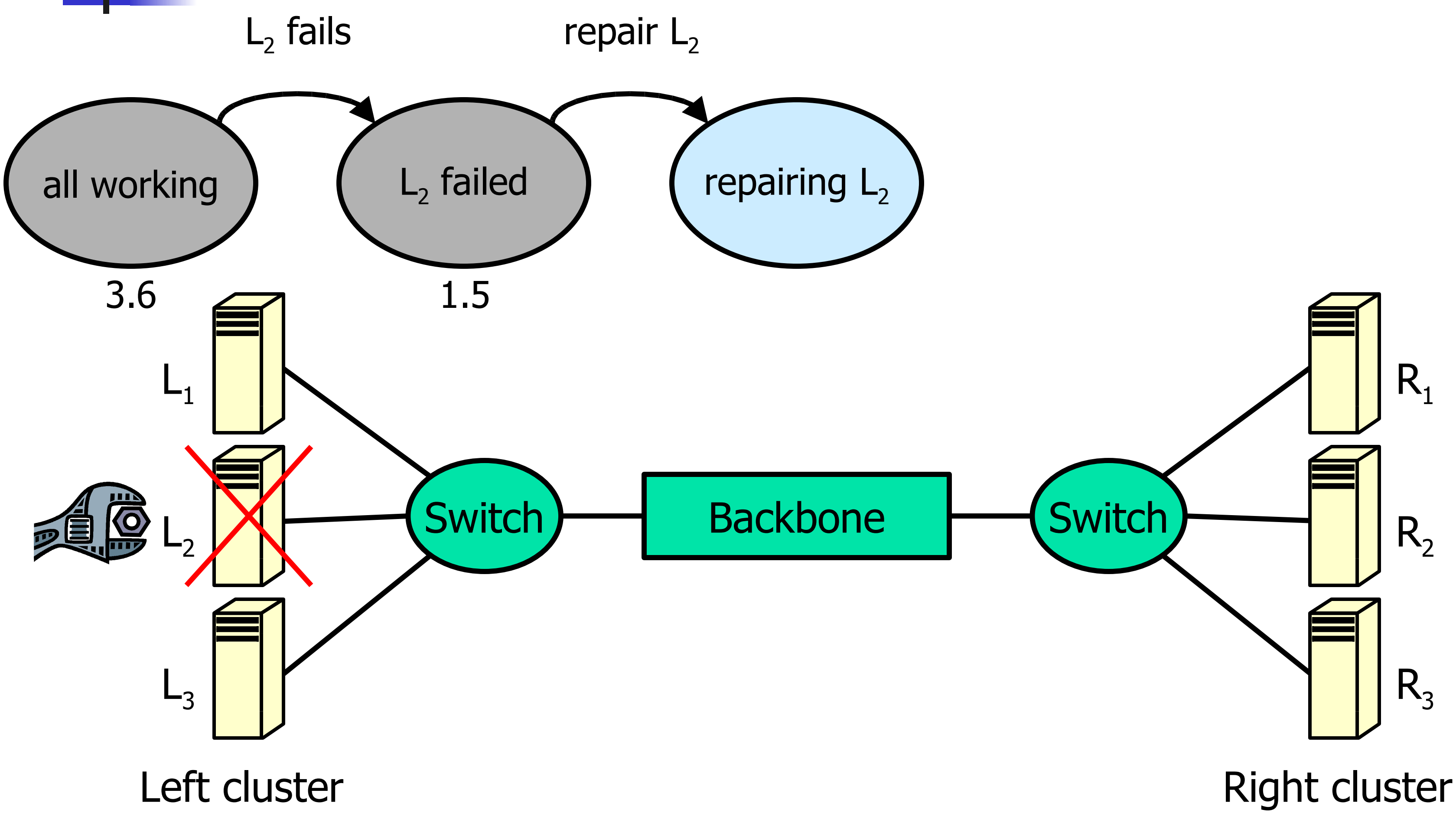
Motivating Example



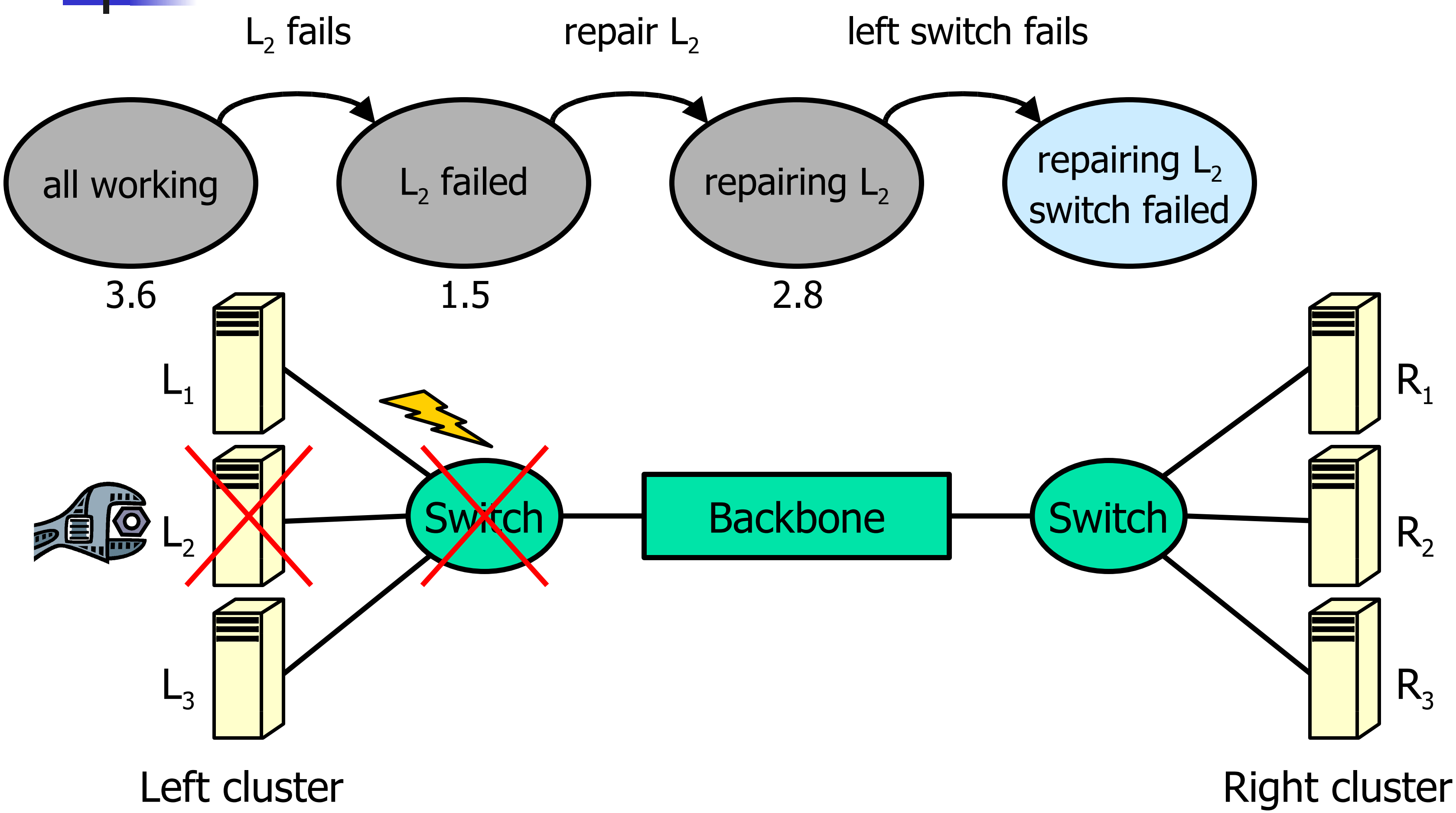
Motivating Example



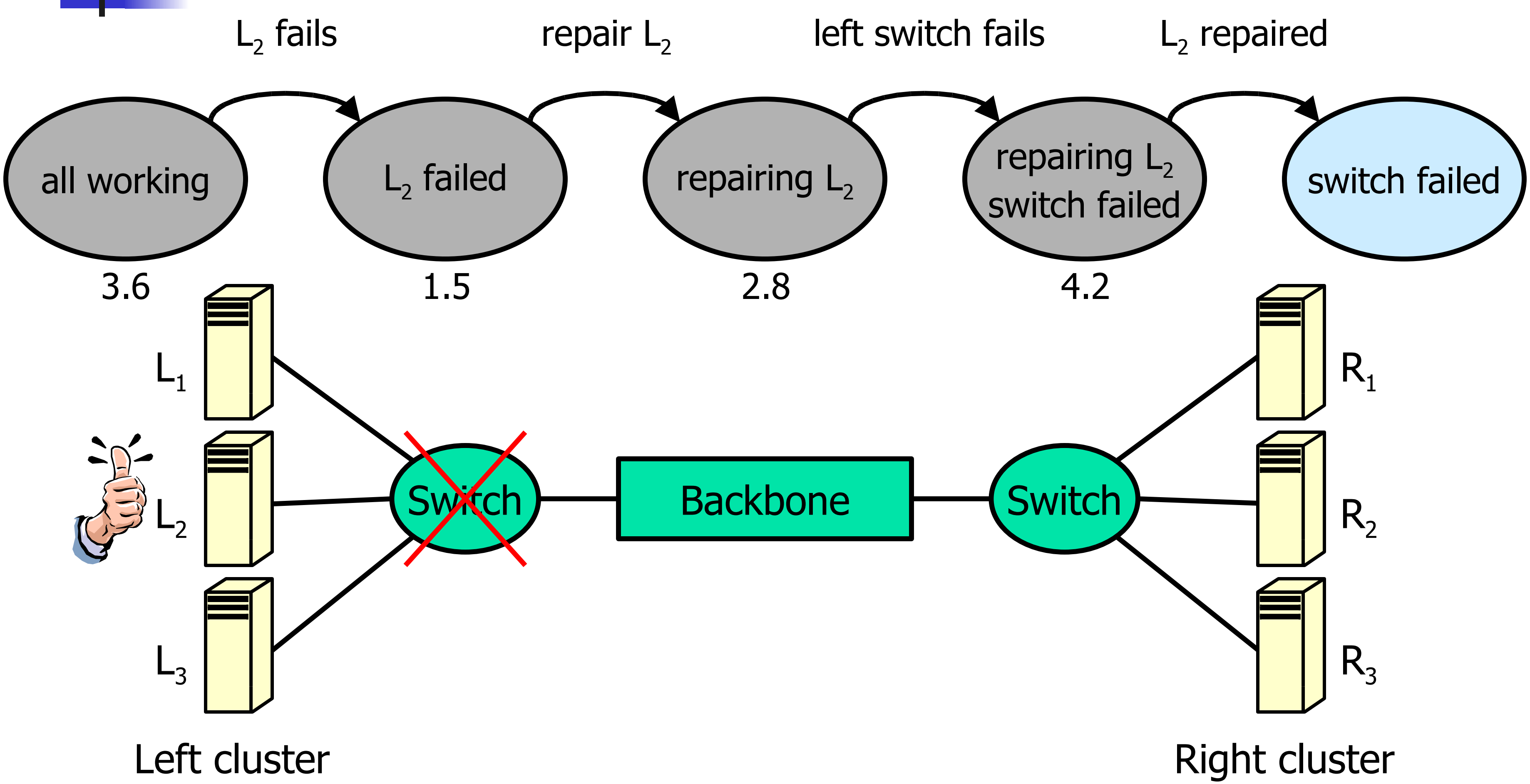
Motivating Example



Motivating Example



Motivating Example

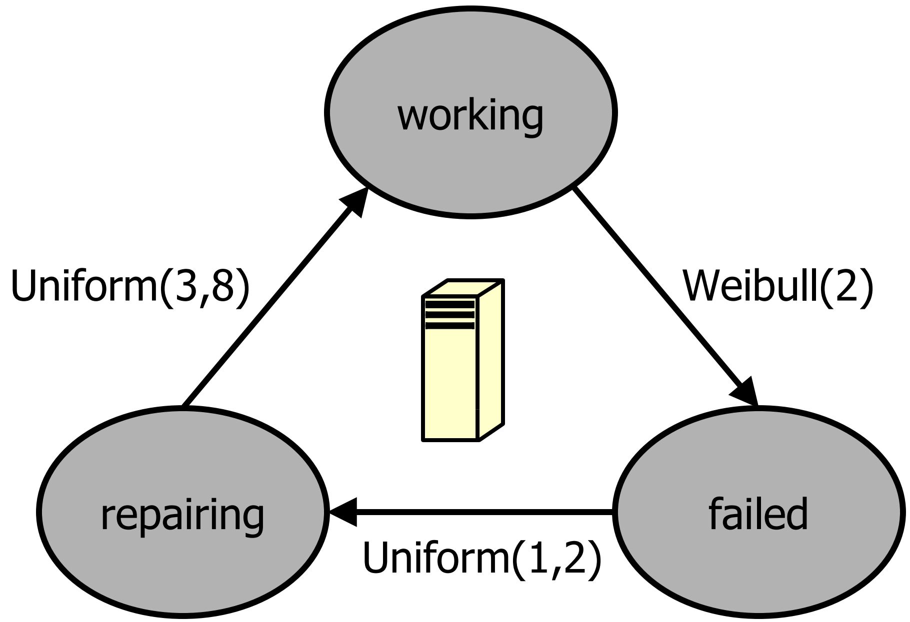


Why Challenging?

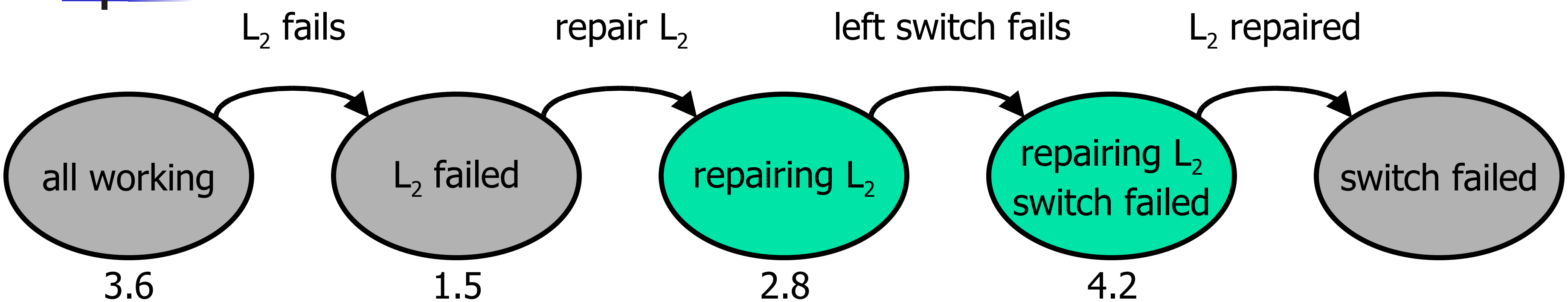
- Continuous-time domain:
  - State transitions can occur at any point in time along a continuous time-axis
- Concurrent actions and events:
  - Left switch fails while repairing L_2
- Non-Markovian:
  - Weibull distribution for time to hardware failure

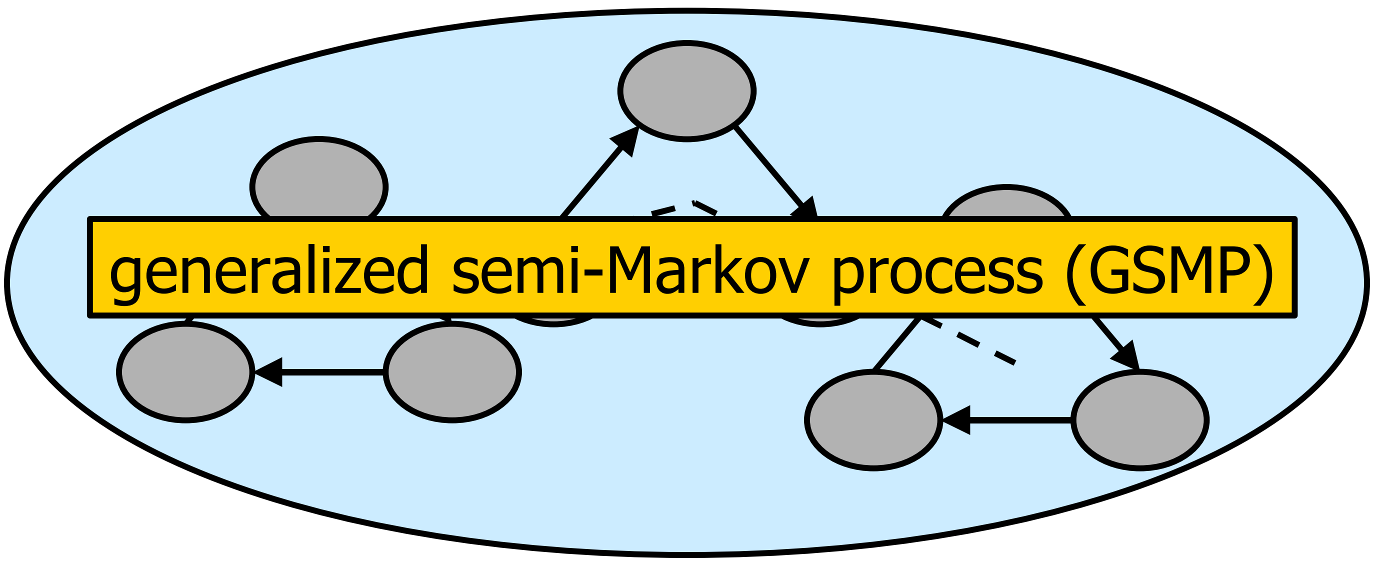
Semi-Markovian?

- Each component can be viewed as a semi-Markov process



Concurrent Semi-Markov Processes





Potential Application Domains

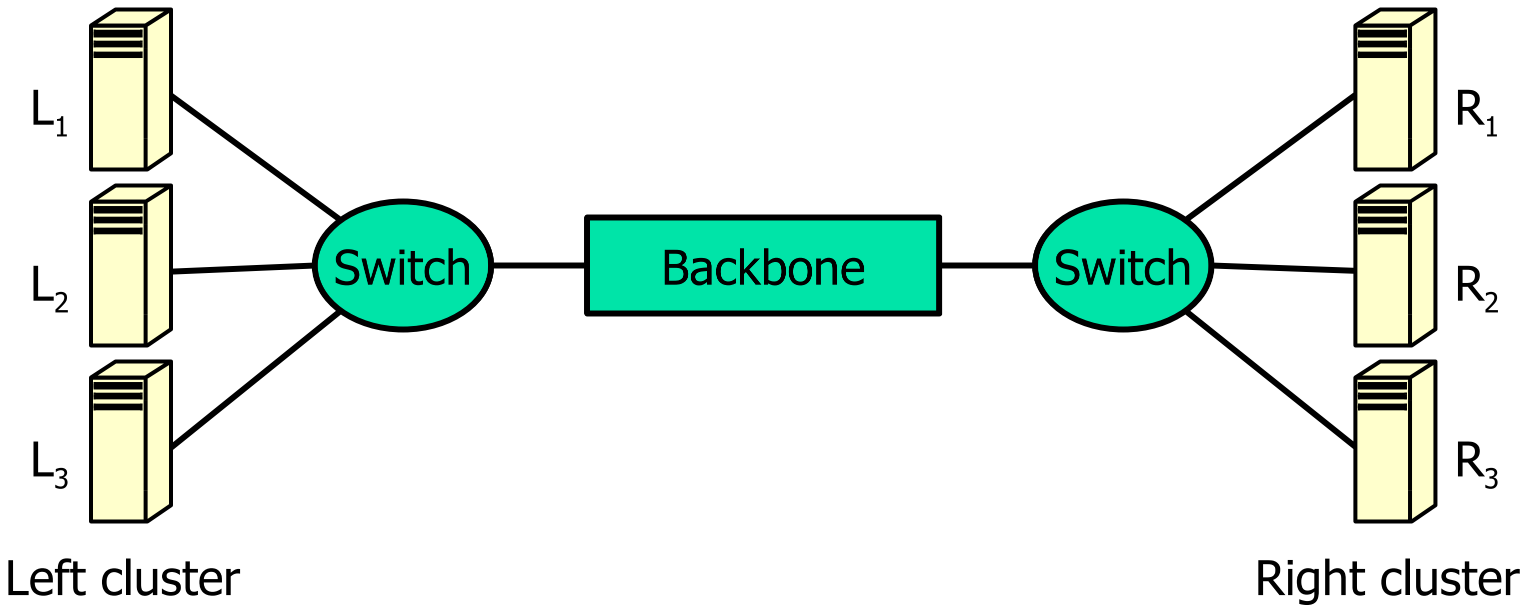
- Multi-agent systems
- Process plant control
  - Startup/shutdown procedures
- Robotic control
  - Mars rover
  - Large degree-of-freedom humanoid

Planning Problem

- Input:
  - Domain model (GSMP)
  - Initial state
  - Goal condition
- Output:
  - Policy mapping states to actions

Motivating Example

- Maintain three interconnected workstations for 24 hours with probability at least 0.9



Approach

- Generate, Test, and Debug (GTD) (Simmons 1988)

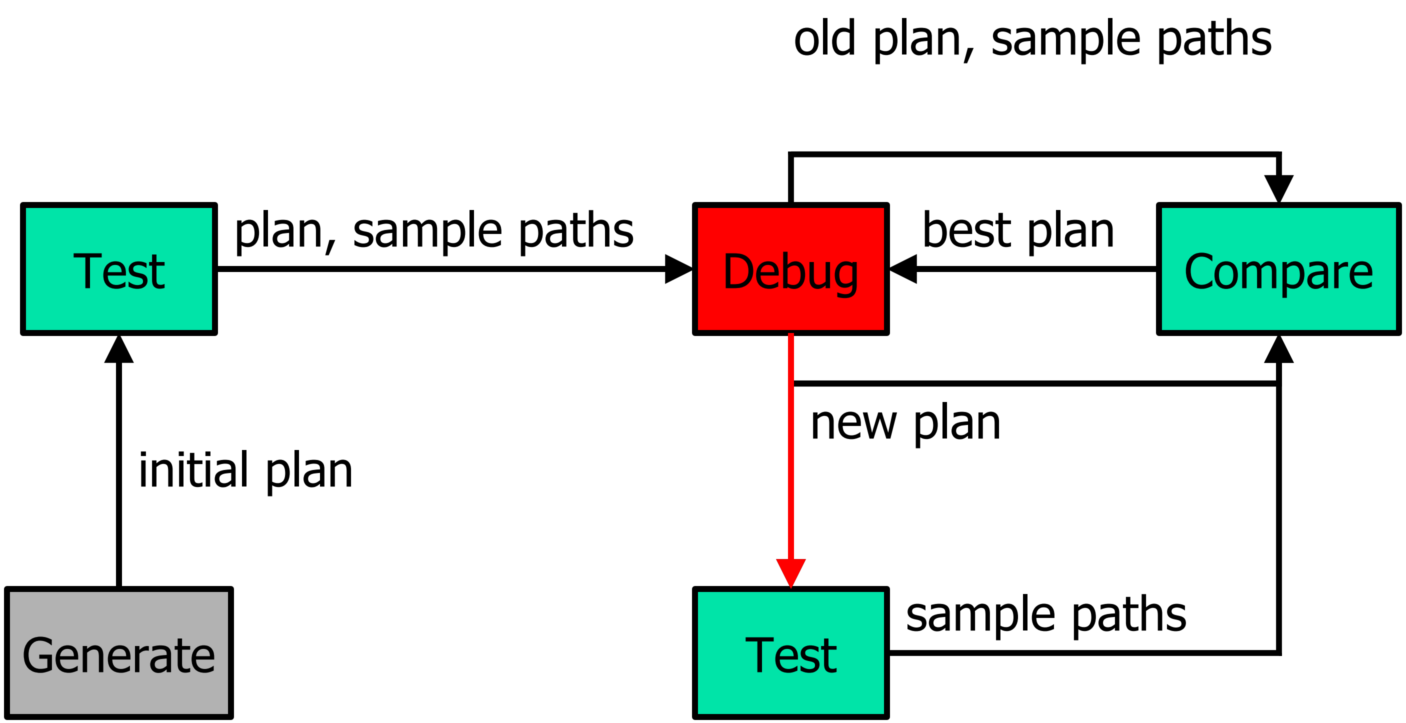
The GTD Paradigm

- Generate initial plan making simplifying assumptions
- Test if plan satisfies goal condition
- Debug plan if not satisfactory

My Adaptation of GTD

- Generate
  - assume we can quickly generate plan for simplified problem, or start with null-plan
- Test
  - Sampling-based probabilistic verification
- Debug
  - Analysis of sample execution paths

Approach: Schematic View



Test Step

- Verify goal condition in initial state
  - Use simulation to generate sample execution paths
  - Use sequential acceptance sampling (Wald 1945) to verify probabilistic properties

Debug Step

- Analyze sample execution paths to find reasons for failure
  - Negative sample paths provide information on how a plan can fail

Generic Repair Procedure

1. Select some state along some negative sample path
2. Change the action planned for the selected state

Develop heuristics to make informed state/action choices

Hill-climbing Search for Satisfactory Plan

1. Compare repaired plan to original plan
   - Fast sampling-based comparison
2. Select the better of the two plans

Approach Summary

- Plan verification using discrete event simulation and acceptance sampling
- Plan repair using sample path analysis
  - Heuristics to guide repair selection
- Hill-climbing search for satisfactory plan
  - Fast sampling-based plan comparison

Related Work

- Probabilistic planning
- Planning with concurrency
- Policy search for POMDPs
- Probabilistic verification

Probabilistic Planning

- Anytime synthetic projection (Dummond & Bresina 1990)
  - Modal temporal logic for goal specification
  - Generate initial solution path
  - “Robustify”: Find solution paths for deviations to ideal path

Planning with Concurrency

- CIRCA (Musliner et al. 1995)
  - Policy generation for timed automata
  - Non-deterministic model
- Concurrent temporally extended actions (Rohanimanesh & Mahadevan 2001)
  - Decision-theoretic approach
  - SMDP Q-learning

Policy Search for POMDPs

- Pegasus (Ng & Jordan 2000)
  - Estimate value function by generating sample execution paths
- GPOMDP (Baxter & Bartlett 2001)
  - Simulation-based method for estimating gradient of average reward

Probabilistic Verification

- PRISM (Kwiatkowska et al. 2002)
  - Symbolic probabilistic model checking
  - Requires Markov model

Previous Work

- Heuristic deterministic planning
- Sampling-based probabilistic verification
- Initial work on sample path analysis

Heuristic Deterministic Planning

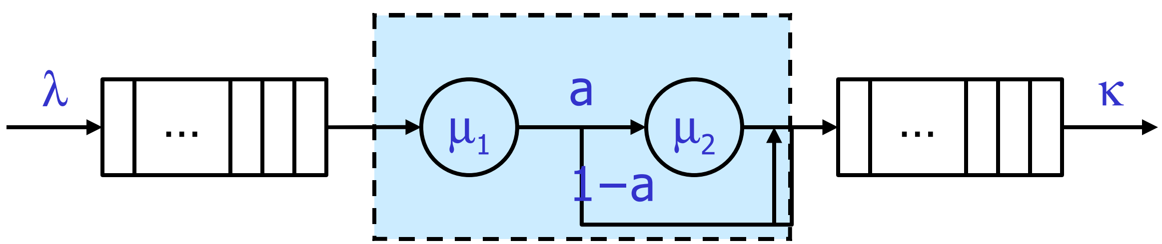
- VHPOP: A heuristic partial order planner
  - Distance-based heuristics for plan ranking
  - Novel flaw selection strategies
  - “Best Newcomer” at 3rd International Planning Competition

Sampling-based Probabilistic Verification

- Verifying CSL properties of discrete event systems
- Sampling-based vs. symbolic methods:
  - Sampling-based approach…
    - uses less memory
    - scales well with size of state space
    - adapts to difficulty of problem (sequential)
  - Symbolic methods…
    - give exact results
    - handle time-unbounded and steady-state properties

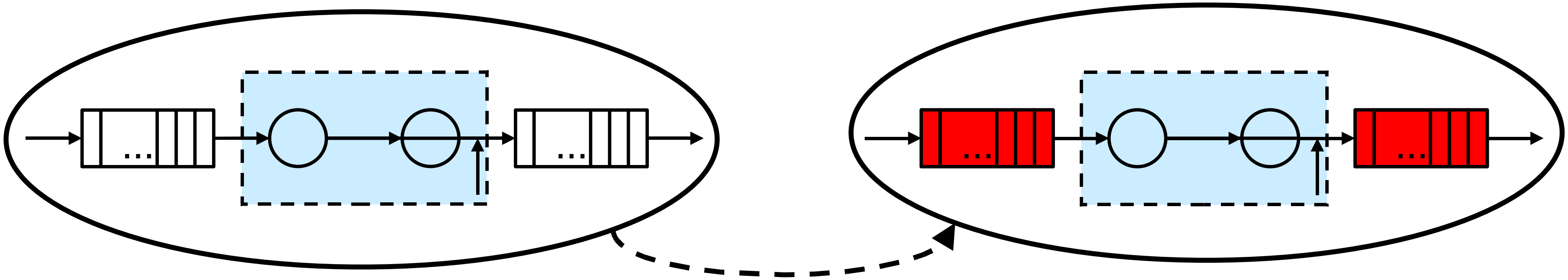
Tandem Queuing Network

- $M/Cox_2/1$ queue sequentially composed with $M/M/1$ queue
- Each queue has capacity n
- State space of size $O(n^2)$

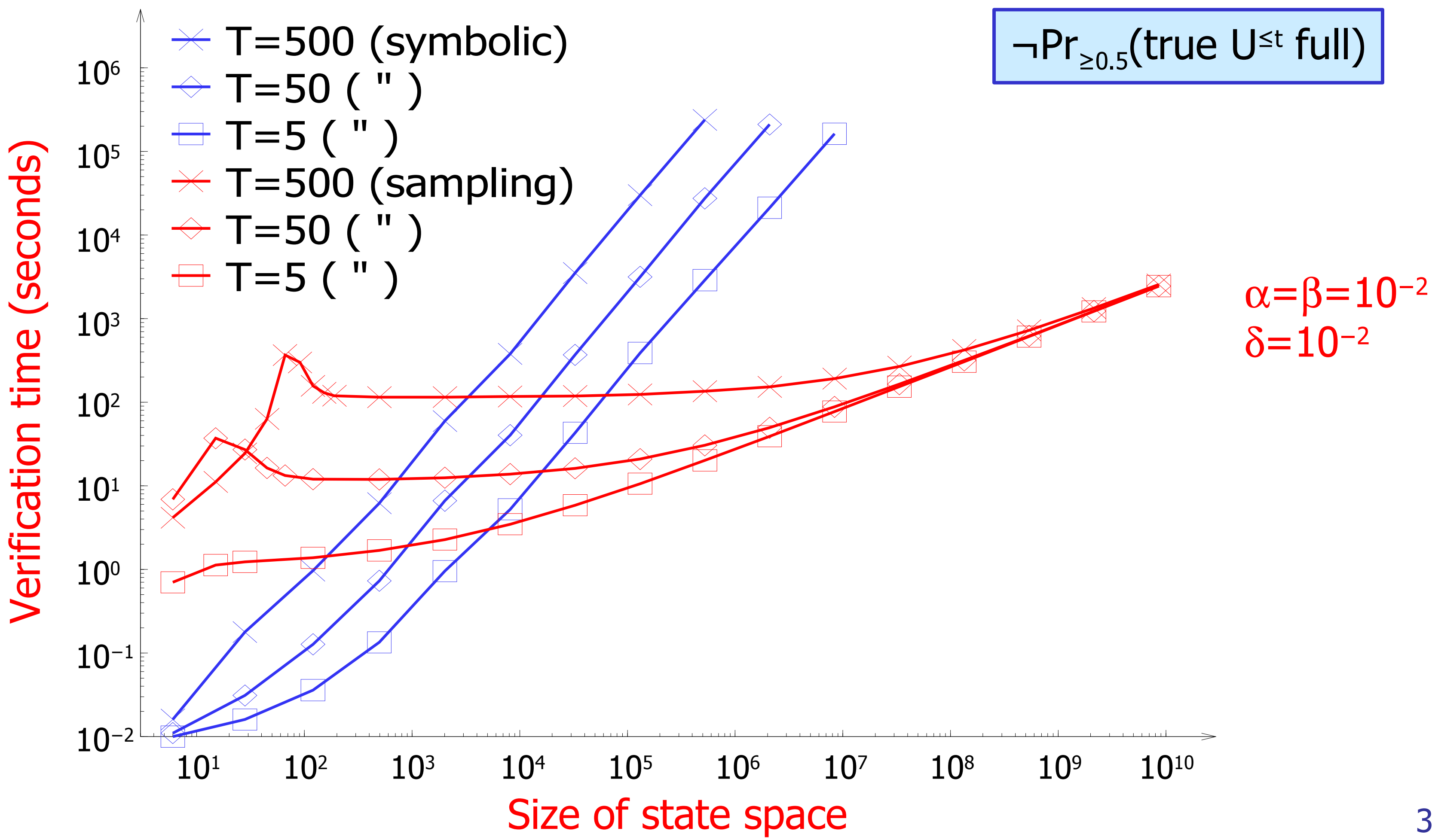


Tandem Queuing Network (property)

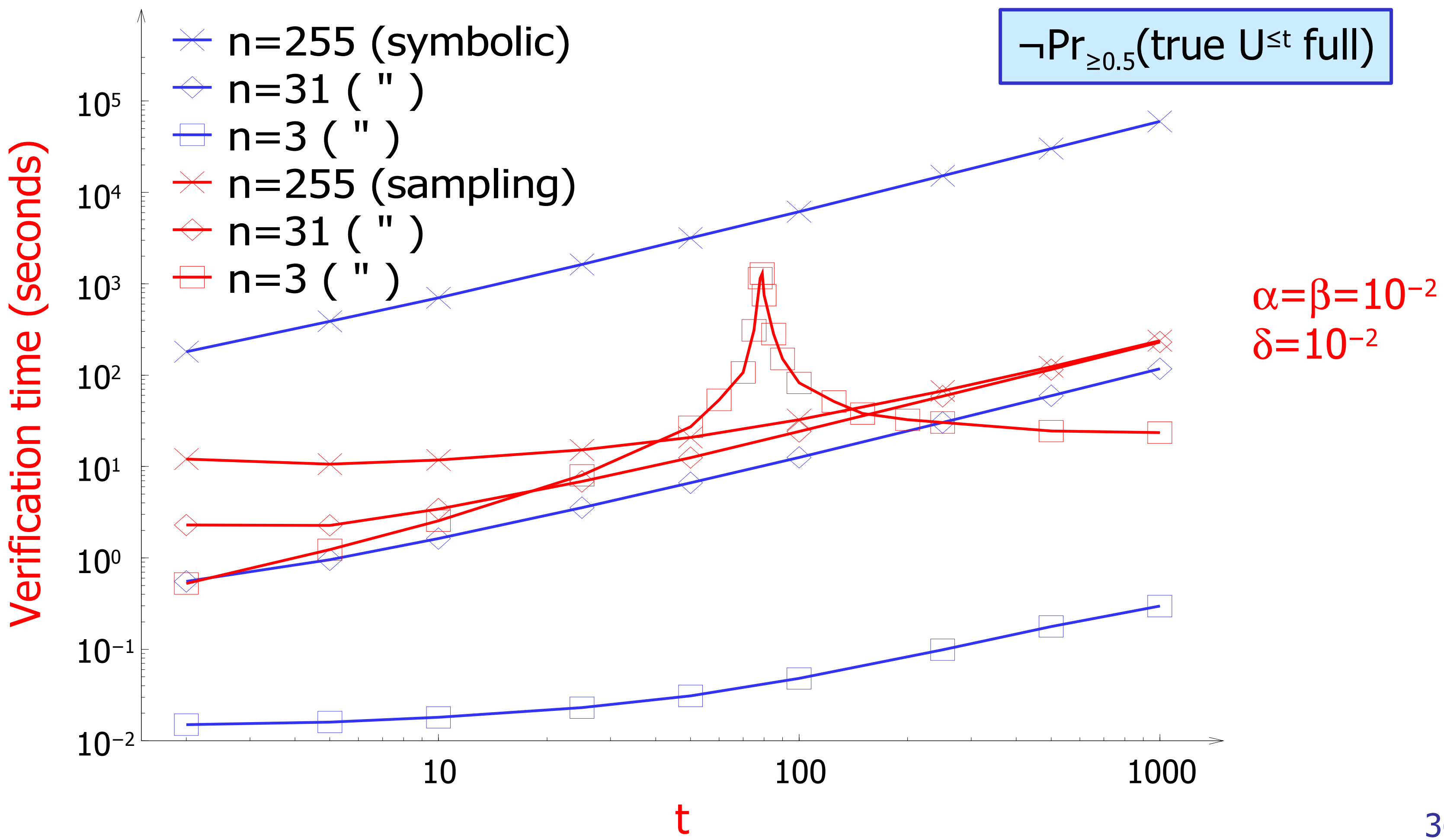
- When empty, probability is less than 0.5 that system becomes full within t time units:
  - $\neg Pr_{\geq 0.5}(\text{true } U^{\leq t} \text{ full})$ in state "empty"



Tandem Queuing Network (results)

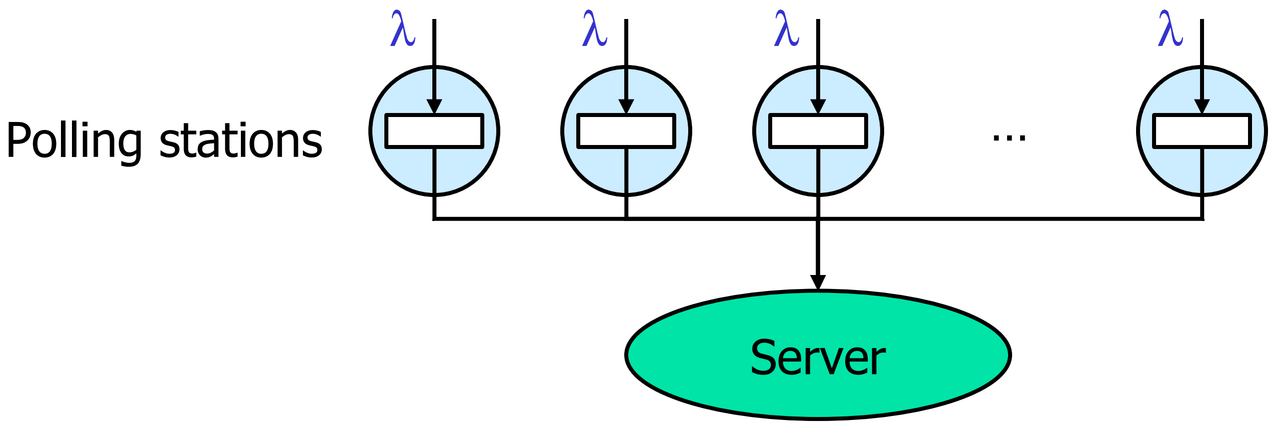


Tandem Queuing Network (results)



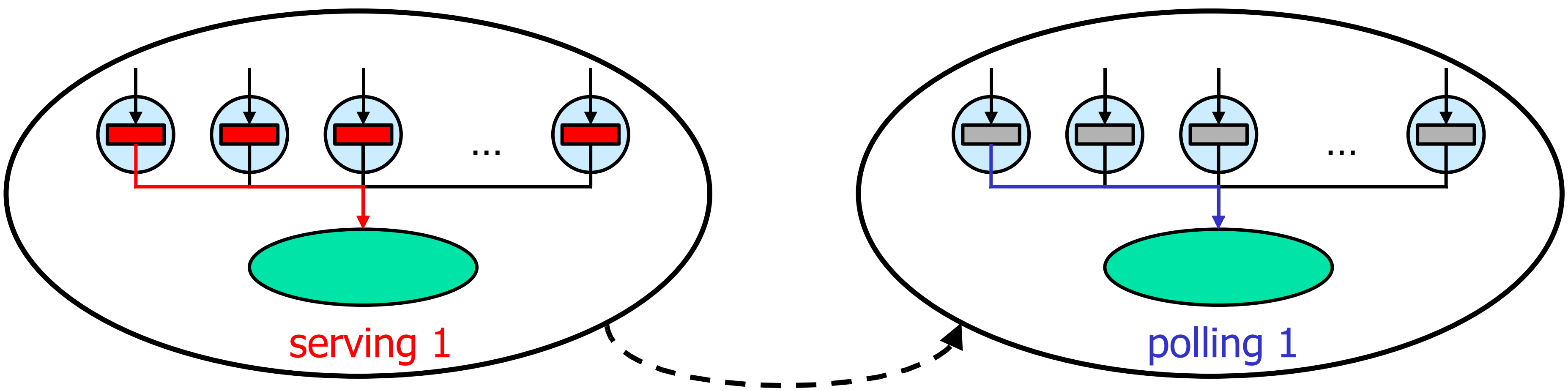
Symmetric Polling System

- Single server, n polling stations
- Stations are attended in cyclic order
- Each station can hold one message
- State space of size $O(n \cdot 2^n)$

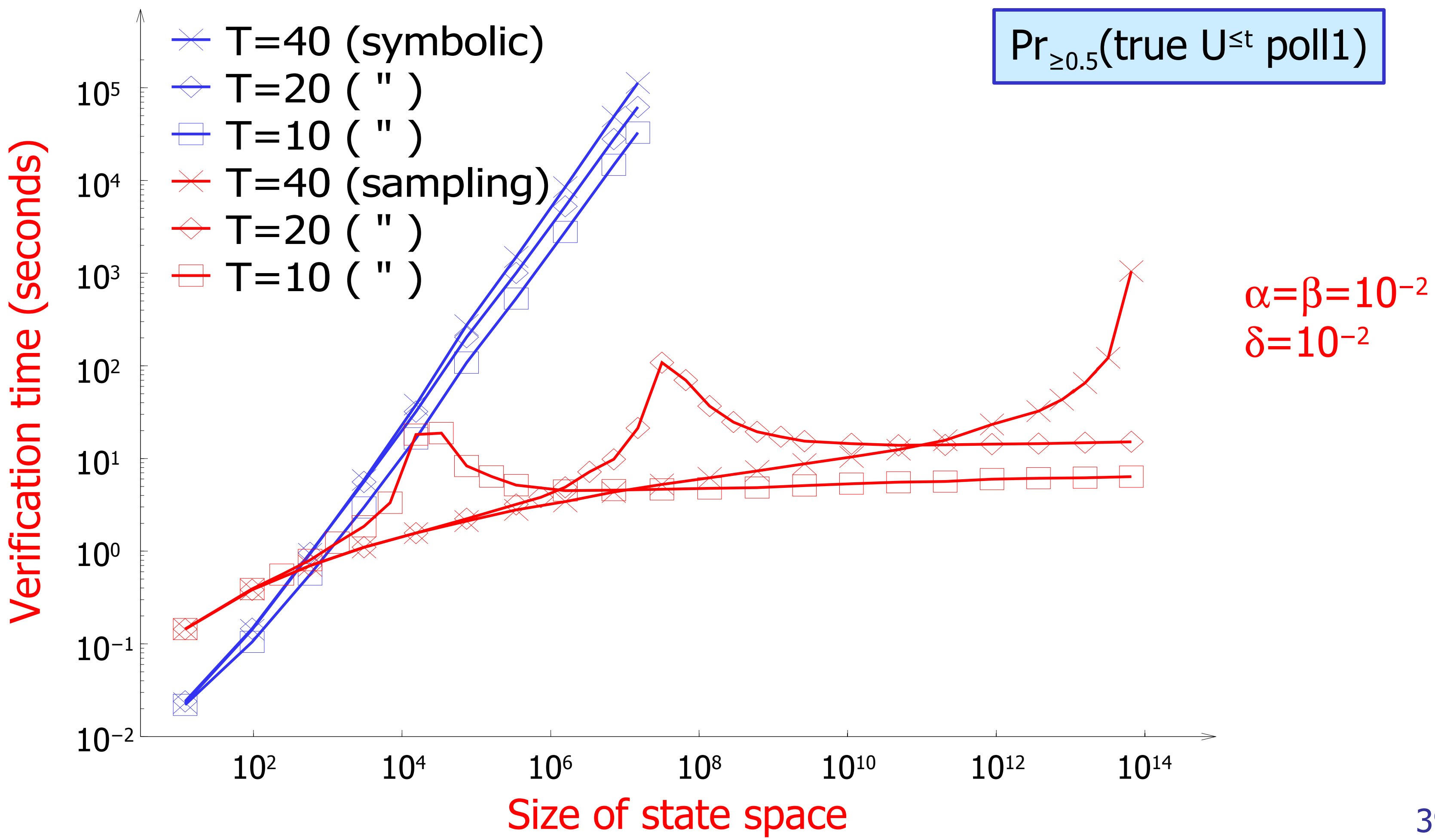


Symmetric Polling System (property)

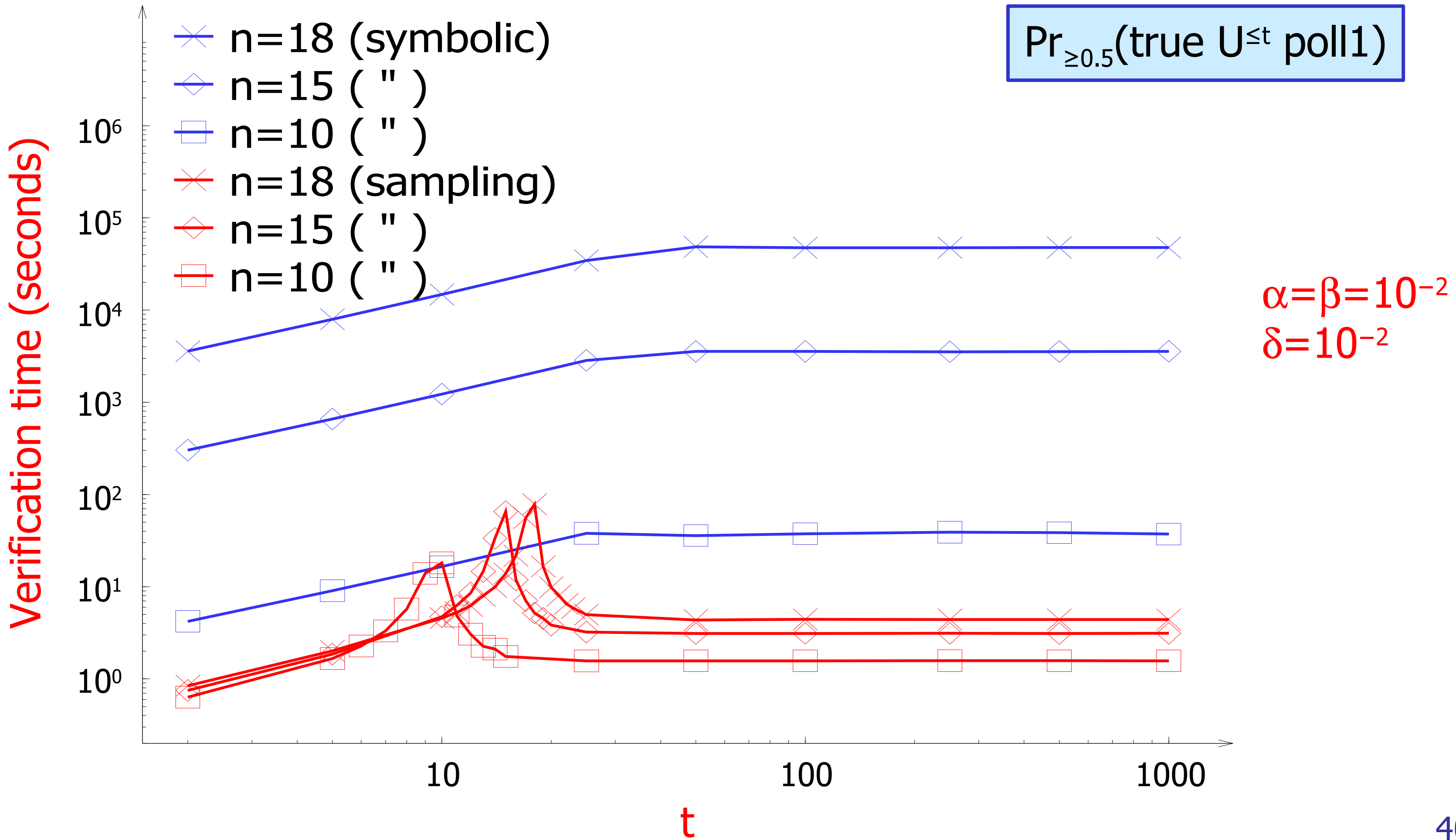
- When full and serving station 1, probability is at least 0.5 that station 1 is polled within t time units:
  - $Pr_{\geq 0.5}(\text{true } U^{\leq t} \text{ poll1})$ in state “full, srv 1”



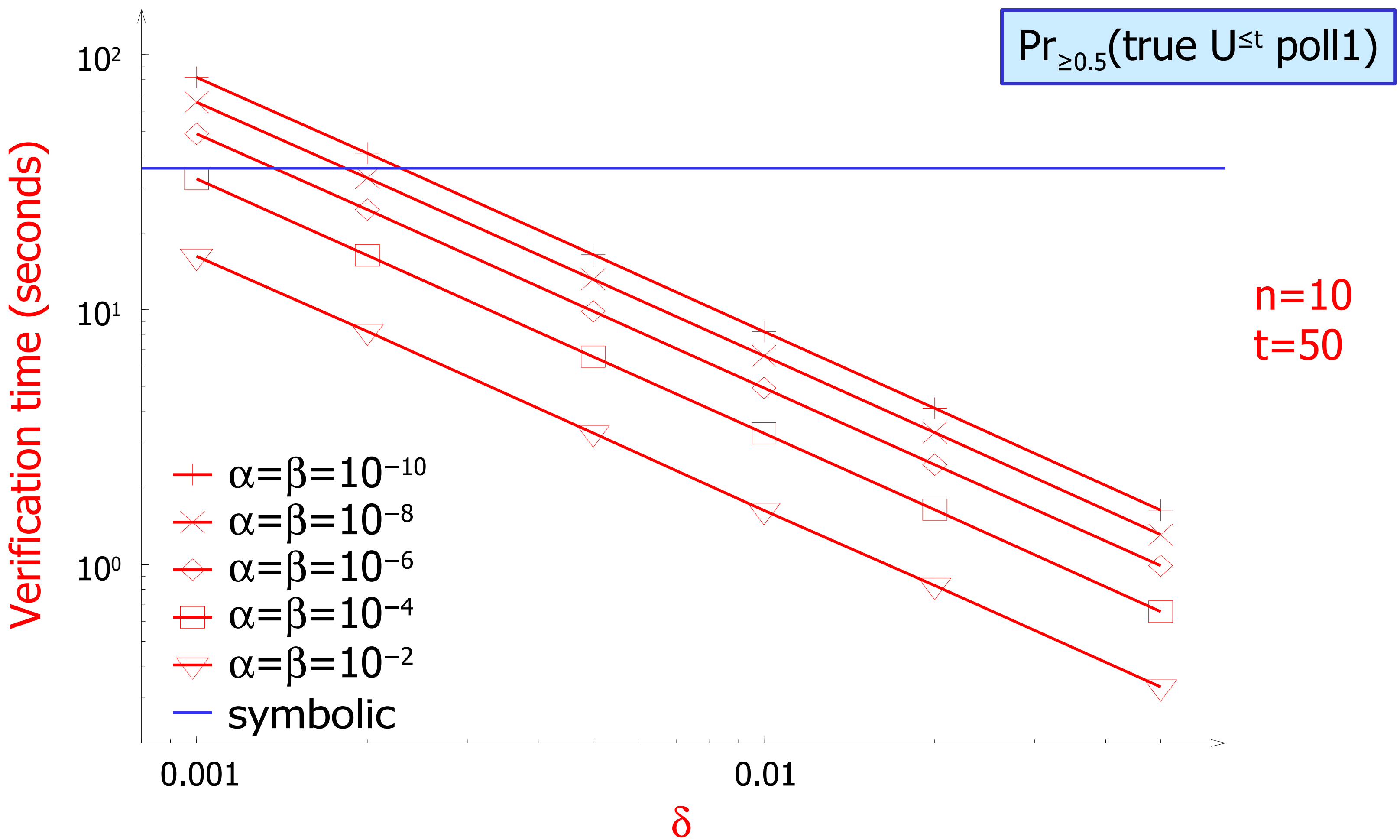
Symmetric Polling System (results)



Symmetric Polling System (results)

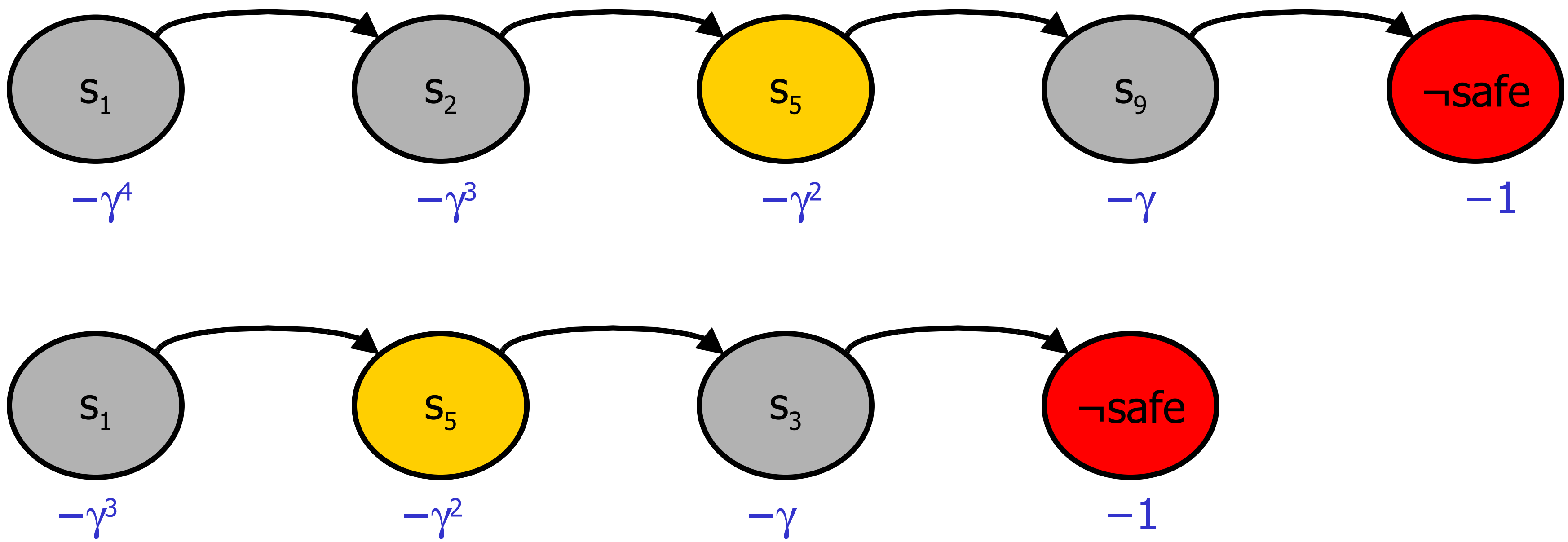


Symmetric Polling System (results)



Initial Work on Sample Path Analysis

$$Pr_{\geq 0.9}(\text{safe } U^{\leq 100} \text{ goal})$$



Proposed Work

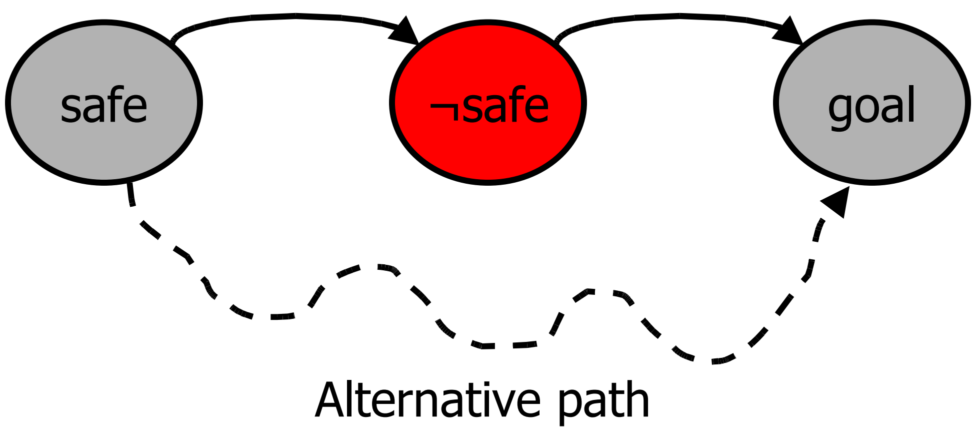
- Heuristic plan repair
- Conjunctive probabilistic goals

Heuristic Plan Repair

- Identify local repair operations for GSMP domain models
- Develop distance-based heuristics for stochastic domains to guide repair selection
- Make use of positive sample execution paths to rank plan repairs
- Evaluate different search strategies, such as hill-climbing and simulated annealing

Heuristics to Guide Action Selection

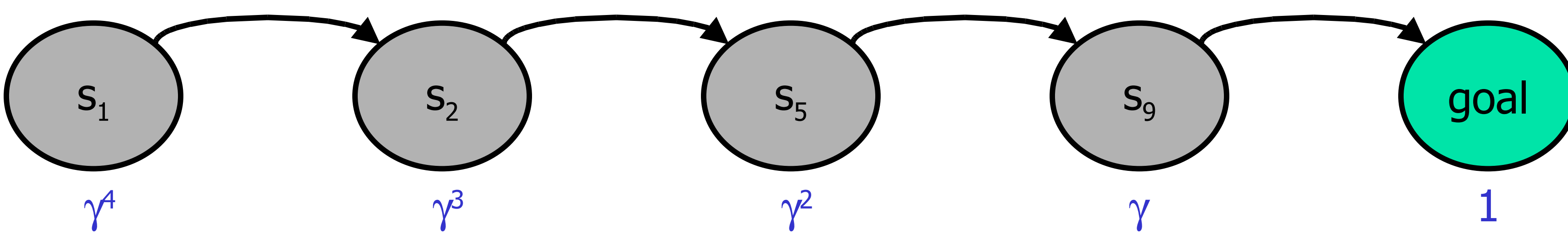
$Pr_{\geq 0.9}(\text{safe } U^{\leq 100} \text{ goal})$



Utilizing Positive Sample Execution Paths

- Positive sample paths show how a plan can succeed
- Analyze positive sample paths to identify positive behavior

$\Pr_{\geq 0.9}(\text{safe } U^{\leq 100} \text{ goal})$



Evaluating Alternative Search Strategies

- Simulated annealing
  - Avoid getting stuck in local optimum
  - Use confidence from plan comparison to probabilistically select a plan

Conjunctive Probabilistic Goals

- Add support for more expressive goal conditions
  - $Pr_{\geq\theta}(\rho) \wedge Pr_{\geq\theta'}(\rho')$
- Can be used to express goal priorities
- Challenges:
  - How to compare two plans
  - Multiple goals to account for when repairing a plan

Thesis

- There is sufficient information in sample execution paths obtained during plan verification to enable efficient and effective repair of plans for continuous-time stochastic domains

Validation

- Construct benchmark domains
  - Develop probabilistic PDDL
  - Extend benchmarks for deterministic temporal planning
- Gain insight into what makes a problem difficult (empirical and theoretical analysis)
- Make sure interesting plans can be generated in reasonable amount of time

Timetable

Date	Activity
Spring 2003	Develop benchmark domains
Summer 2003	Identify repair operations Implement initial version of planning system
Fall 2003	Develop distance-based heuristics for stochastic domains
Winter 2003/04	Experiment with different search strategies
Spring 2004	Add support for conjunctive probabilistic goals
Summer 2004	Theoretical analysis
Fall 2004	Final experimental validation Writing of thesis
Winter 2004/05	Finalizing thesis Thesis defense

Possible Extensions

- Generation of non-stationary policies
- Partial observability
- Continuous-space models
- Decision-theoretic planning