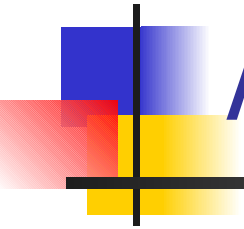
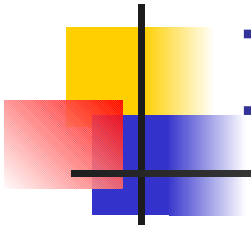


Probabilistic Verification of Discrete Event Systems using Acceptance Sampling

Håkan L. S. Younes

Carnegie Mellon University

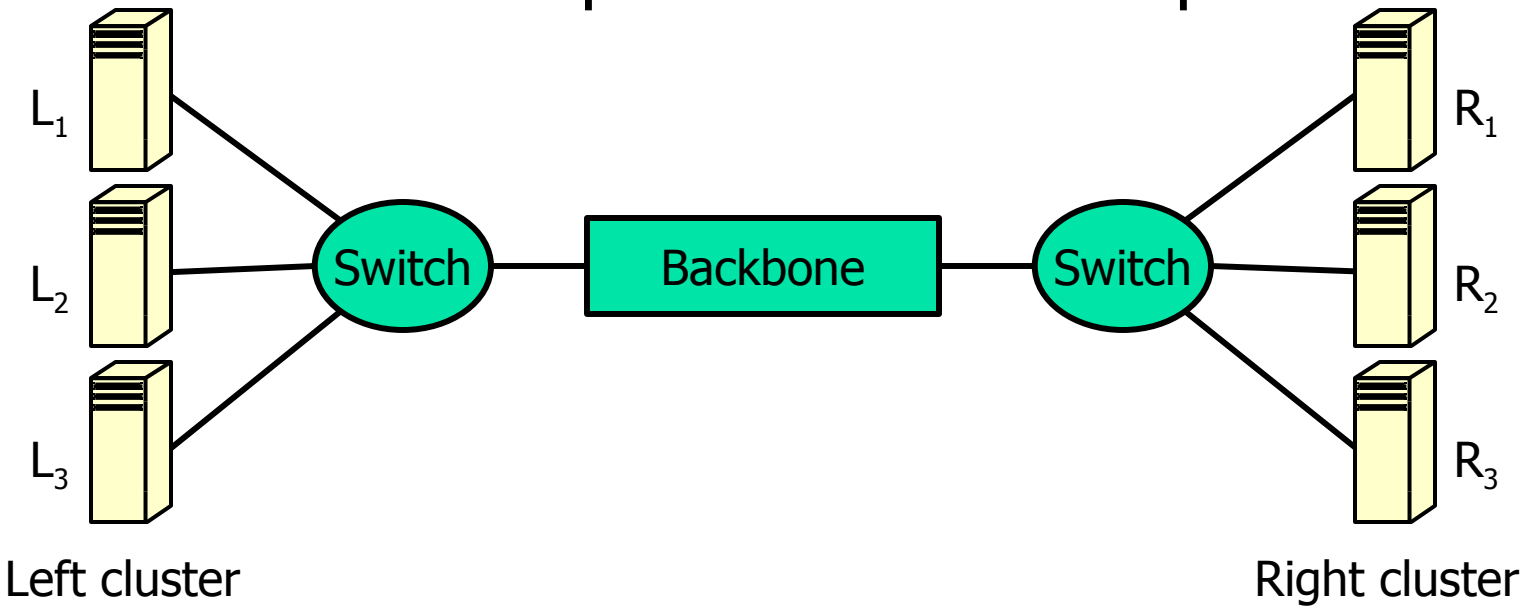


Introduction

- Verify properties of continuous-time discrete event systems
- Model independent approach
- Time-bounded probabilistic properties expressed using CSL
- Acceptance sampling
- Guaranteed error bounds

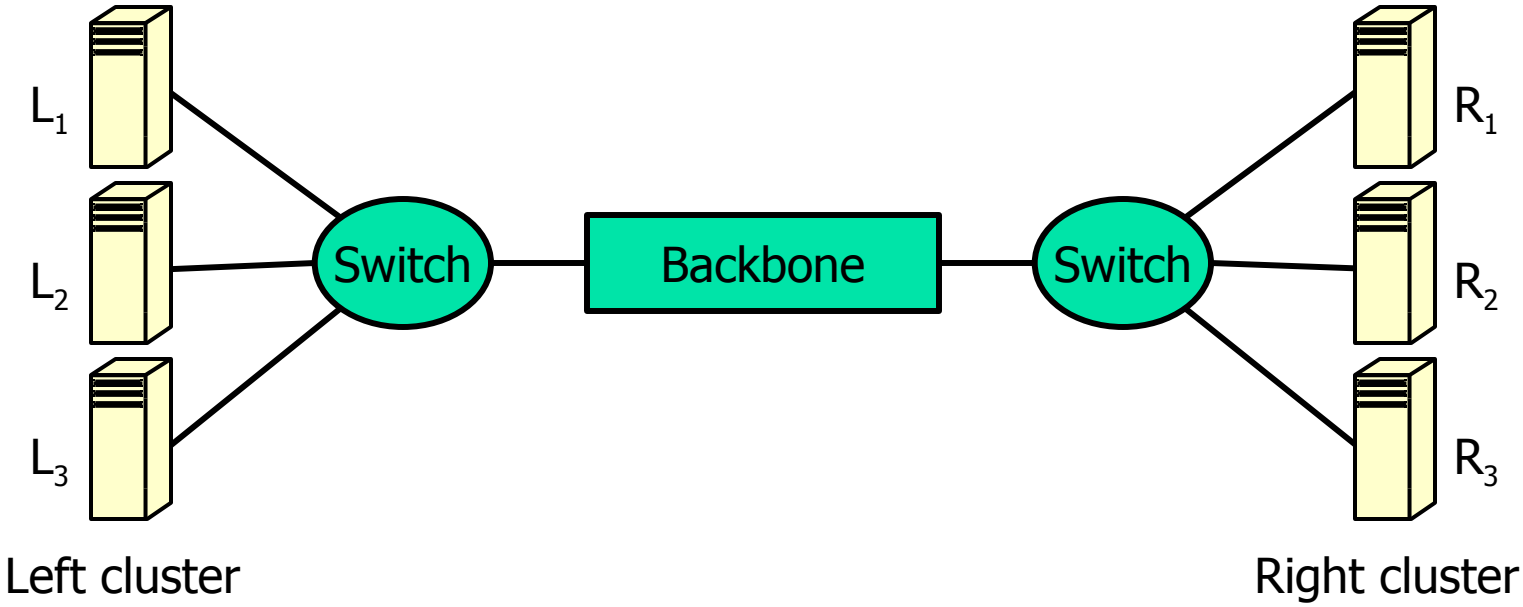
Motivating Example

- Dependable workstation cluster
  - Components may fail at any point in time
  - Failed components can be repaired

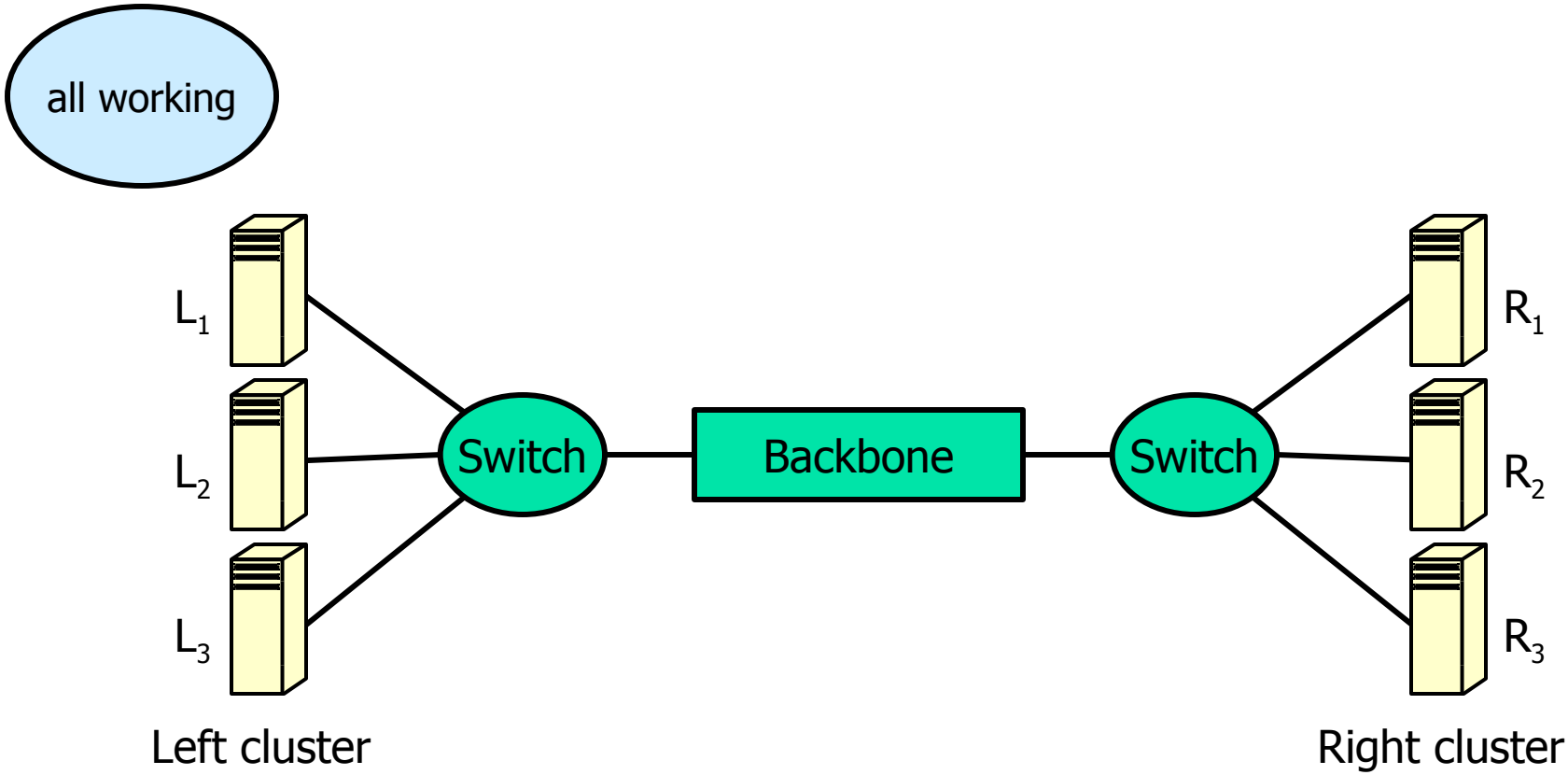


Motivating Example

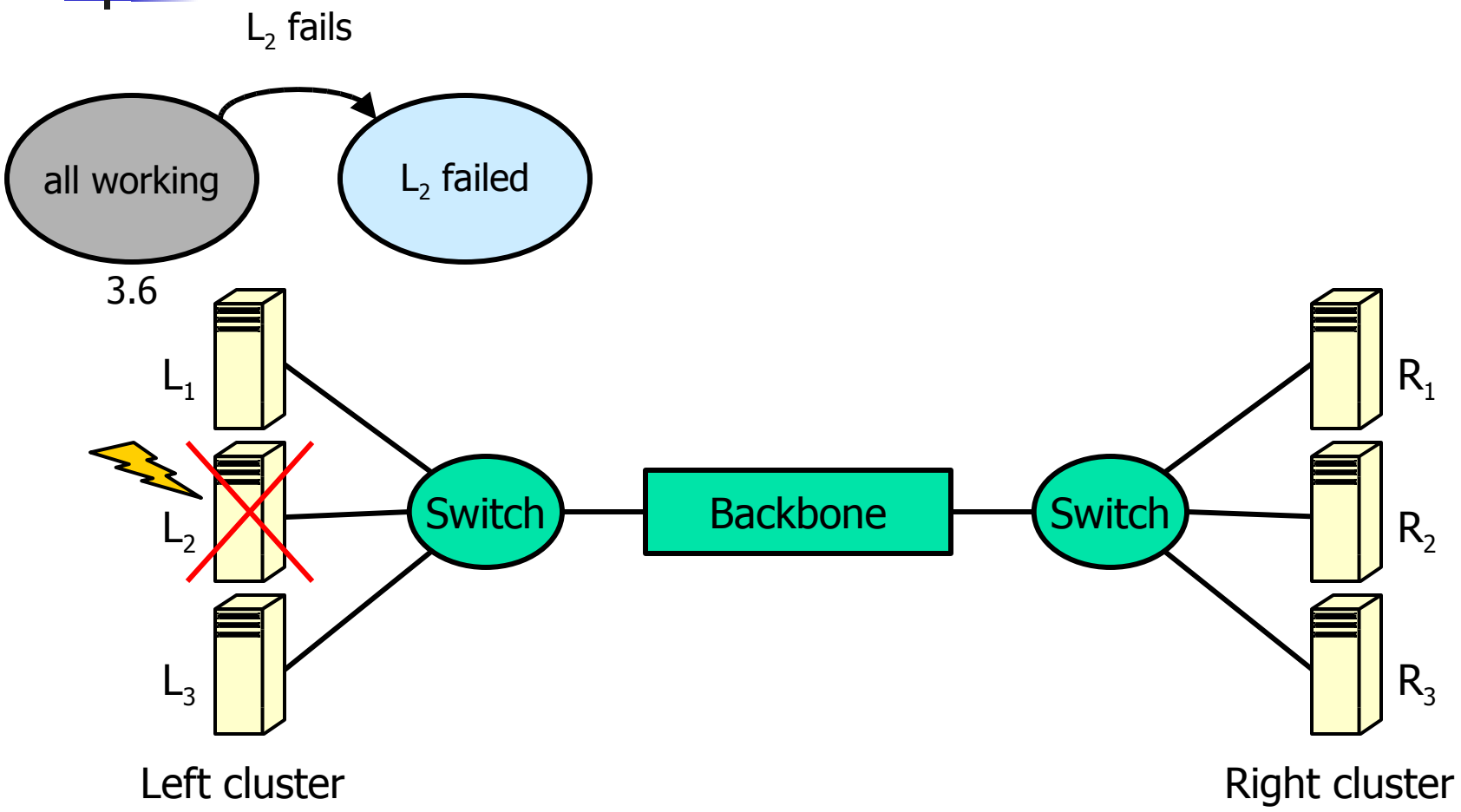
- Property of interest:
  - Probability at most 0.1 that QoS drops below minimum within 50 time units



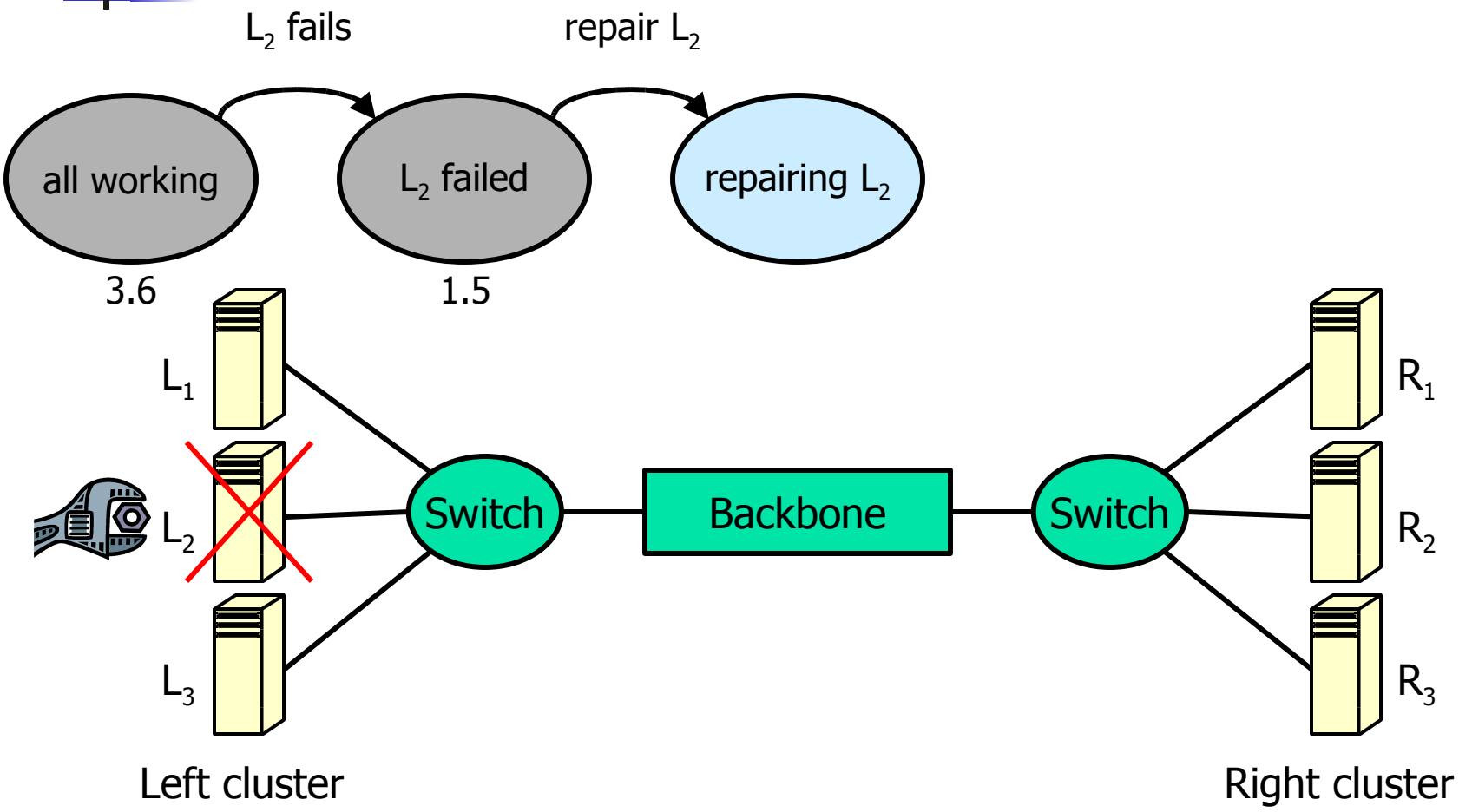
Motivating Example



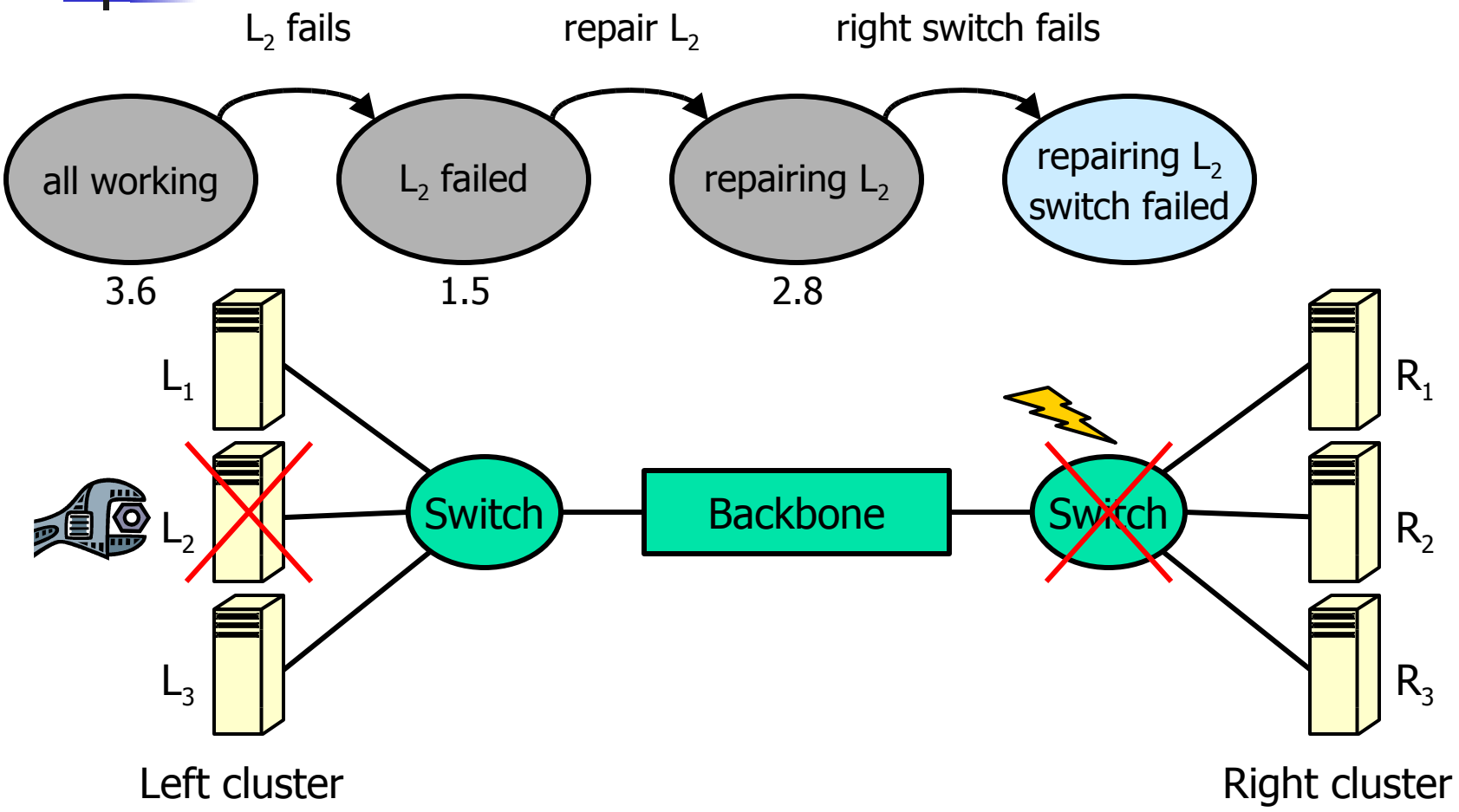
Motivating Example



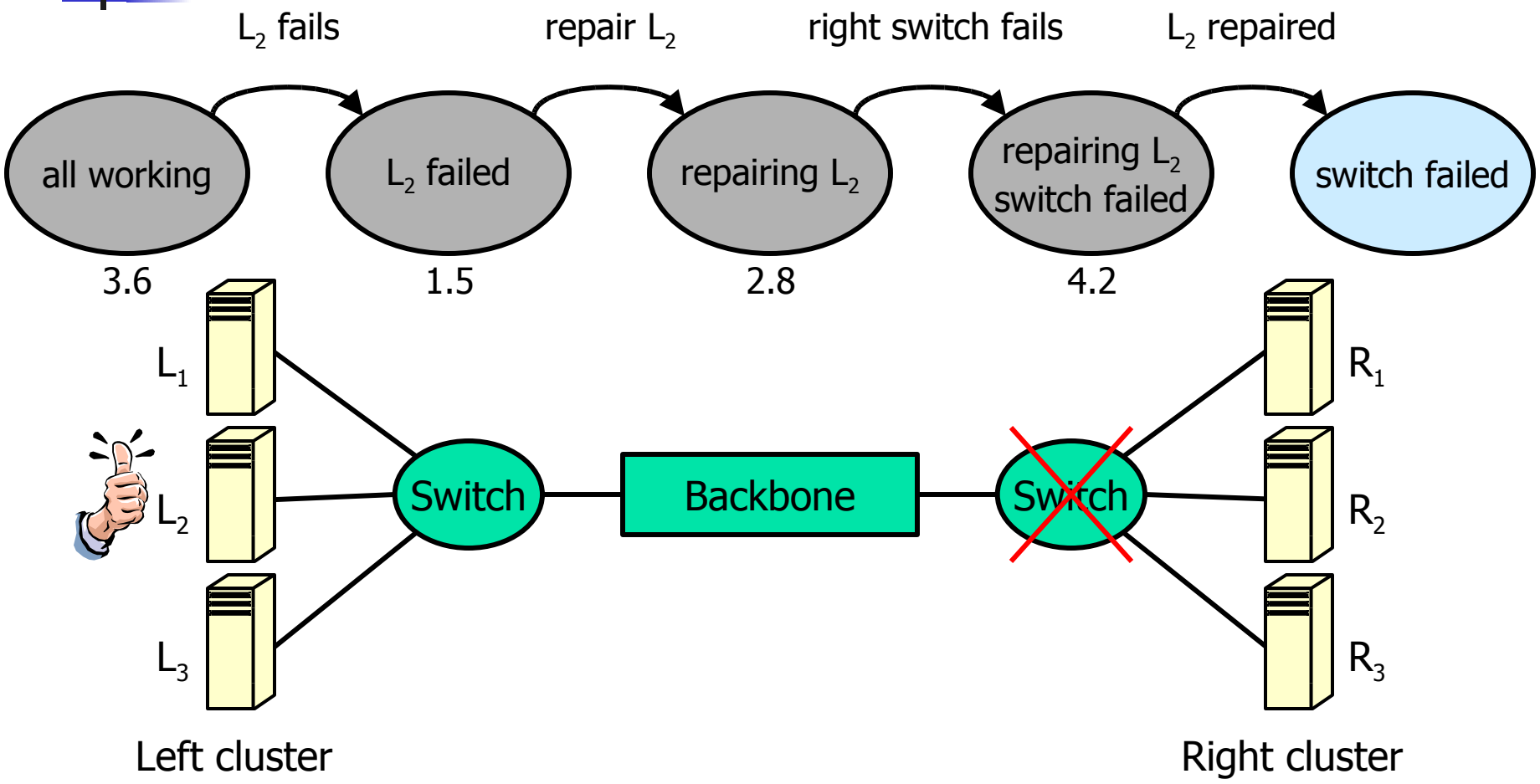
Motivating Example



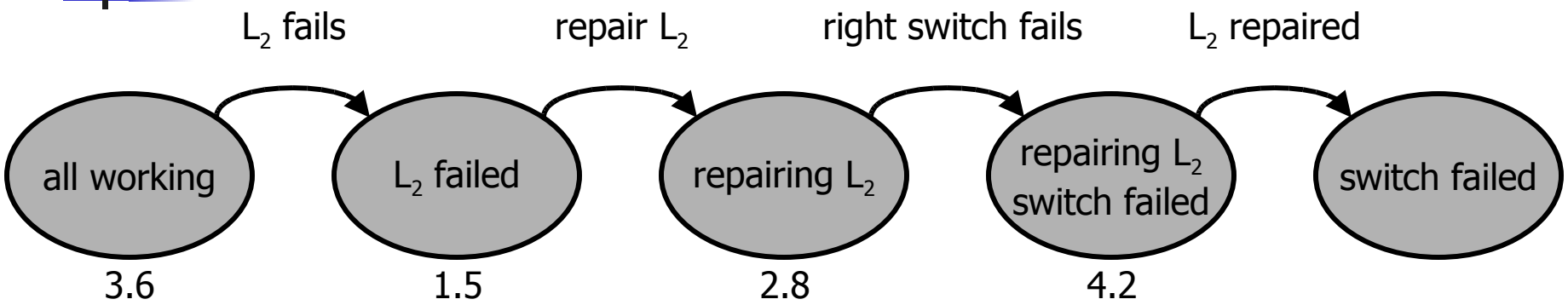
Motivating Example

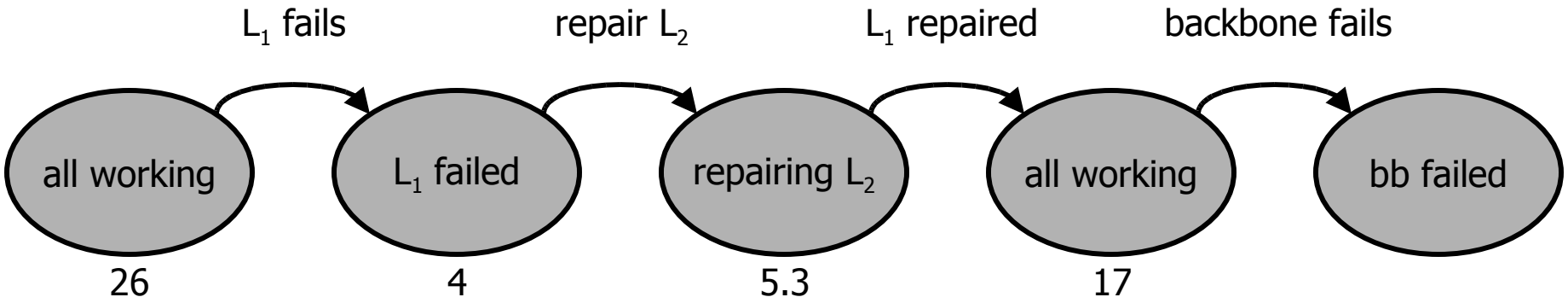


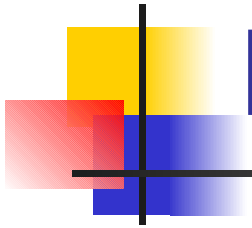
Motivating Example



Sample Execution Paths

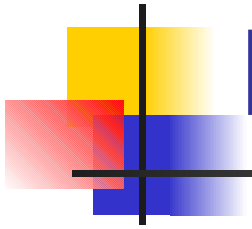






Properties of Interest

- Time-bounded probabilistic properties
  - "The probability is at most 0.1 that QoS drops below minimum within 50 time units"



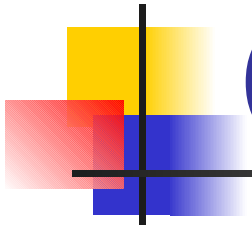
Properties of Interest

- Time-bounded probabilistic properties
  - "The probability is at most 0.1 that QoS drops below minimum within 50 time units"

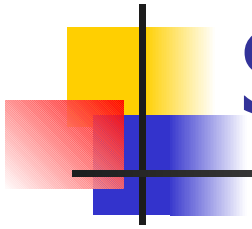
CSL formula:

$$\neg Pr_{\geq 0.1}(\text{true } U^{\leq 50} \neg \text{Minimum QoS})$$

Continuous Stochastic Logic (CSL)



- State formulas
  - Truth value is determined in a single state
- Path formulas
  - Truth value is determined over an execution path



State Formulas

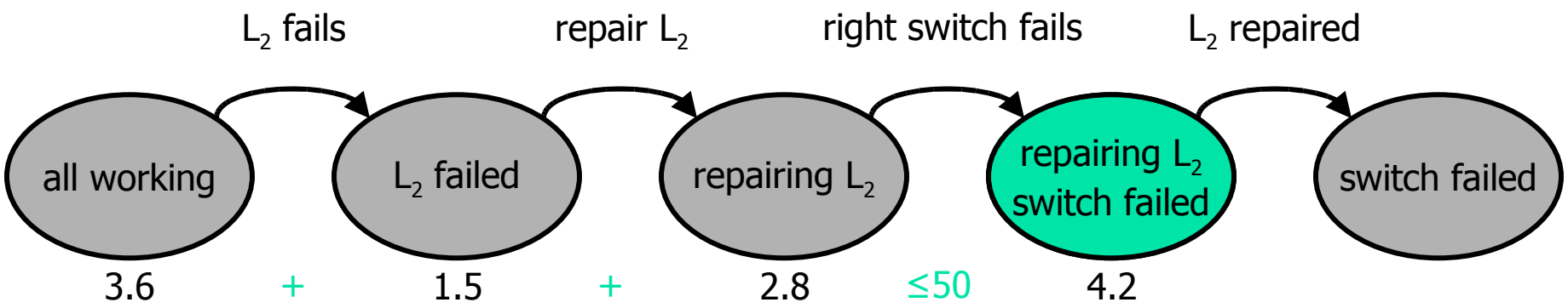
- Standard logic operators: $\neg\varphi$, $\varphi_1 \wedge \varphi_2$ ...
- Probabilistic operator: $Pr_{\geq\theta}(\rho)$
  - True iff probability is at least θ that ρ holds

Path Formulas

- Until: $\varphi_1 \; U^{\leq t} \; \varphi_2$
  - Holds iff φ_2 becomes true in some state along the execution path before time t, and φ_1 is true in all prior states

Verifying Path Formulas

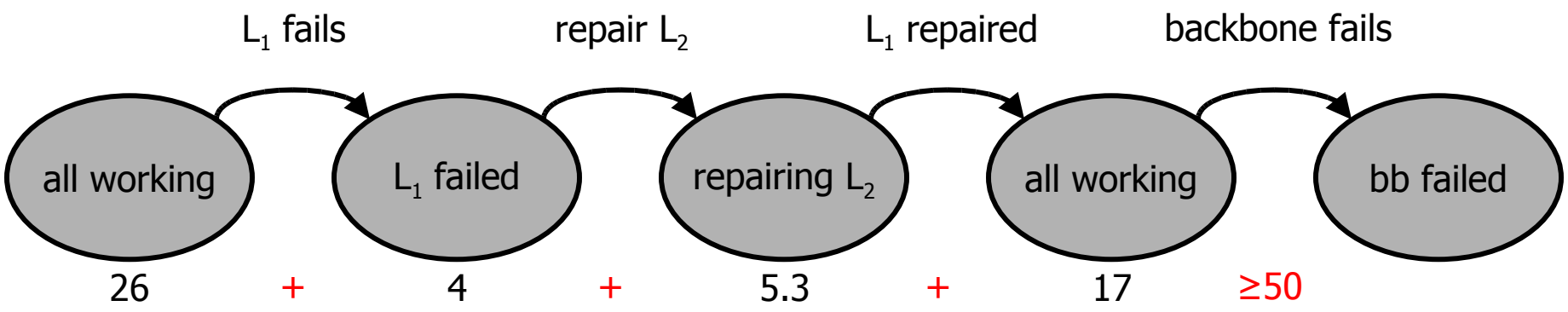
- true $U^{\leq 50}$ ¬Minimum QoS



True!

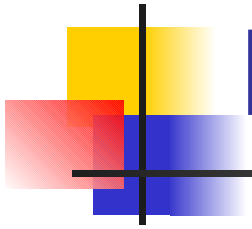
Verifying Path Formulas

- true $U^{\leq 50}$ ¬Minimum QoS

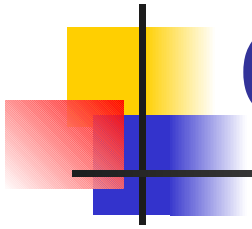


False!

Verifying Probabilistic Properties

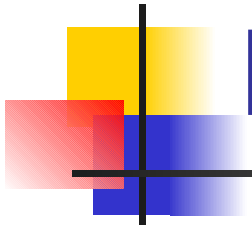


- "The probability is at least θ that ρ"
  - Symbolic Methods
    - Pro: Exact solution
    - Con: Works only for restricted class of systems
  - Sampling
    - Pro: Works for any system that can be simulated
    - Con: Uncertainty in correctness of solution



Our Approach

- Use simulation to generate sample execution paths
- Use sequential acceptance sampling to verify probabilistic properties



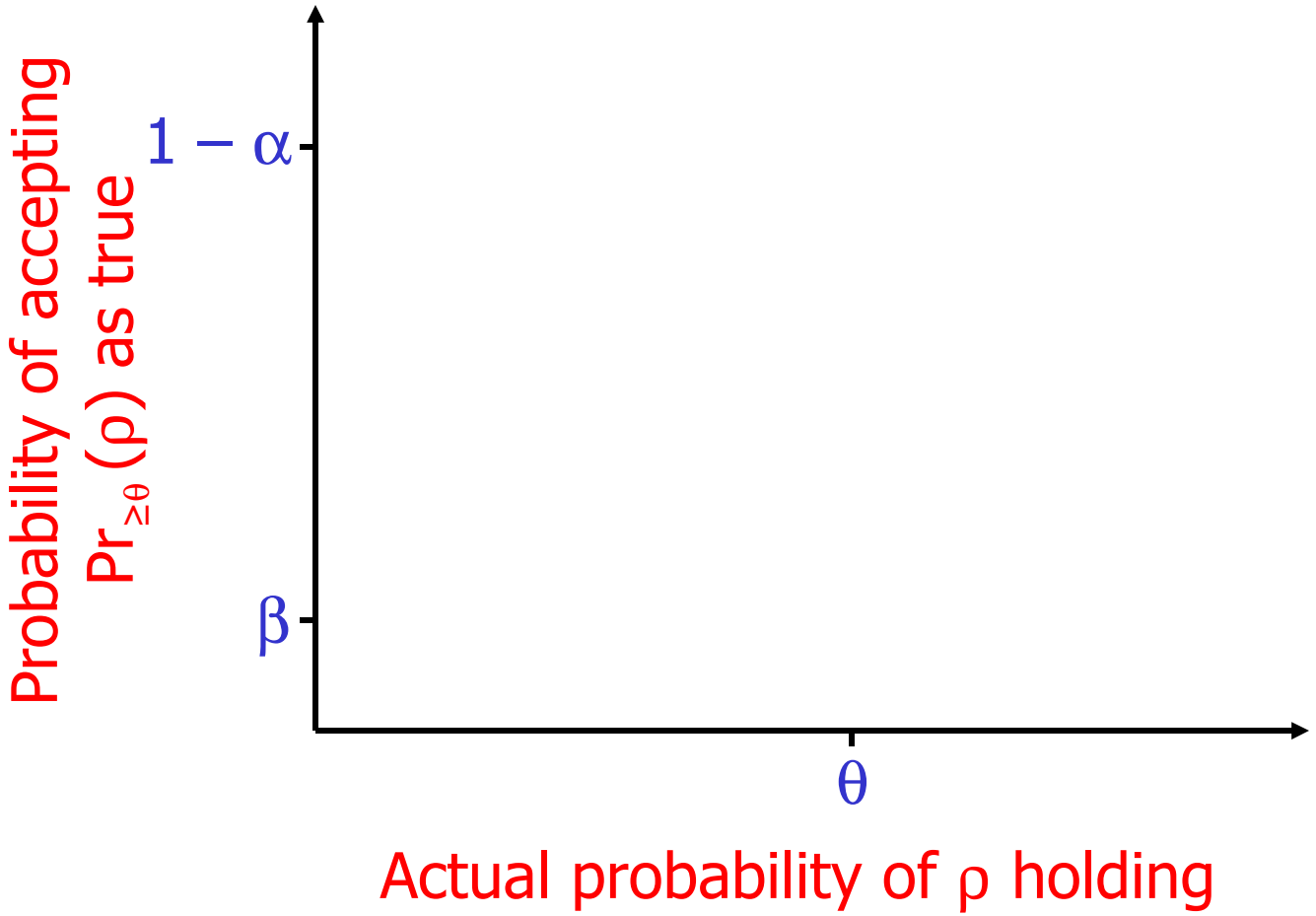
Error Bounds

- Probability of false negative: $\leq \alpha$
  - We say that φ is false when it is true
- Probability of false positive: $\leq \beta$
  - We say that φ is true when it is false

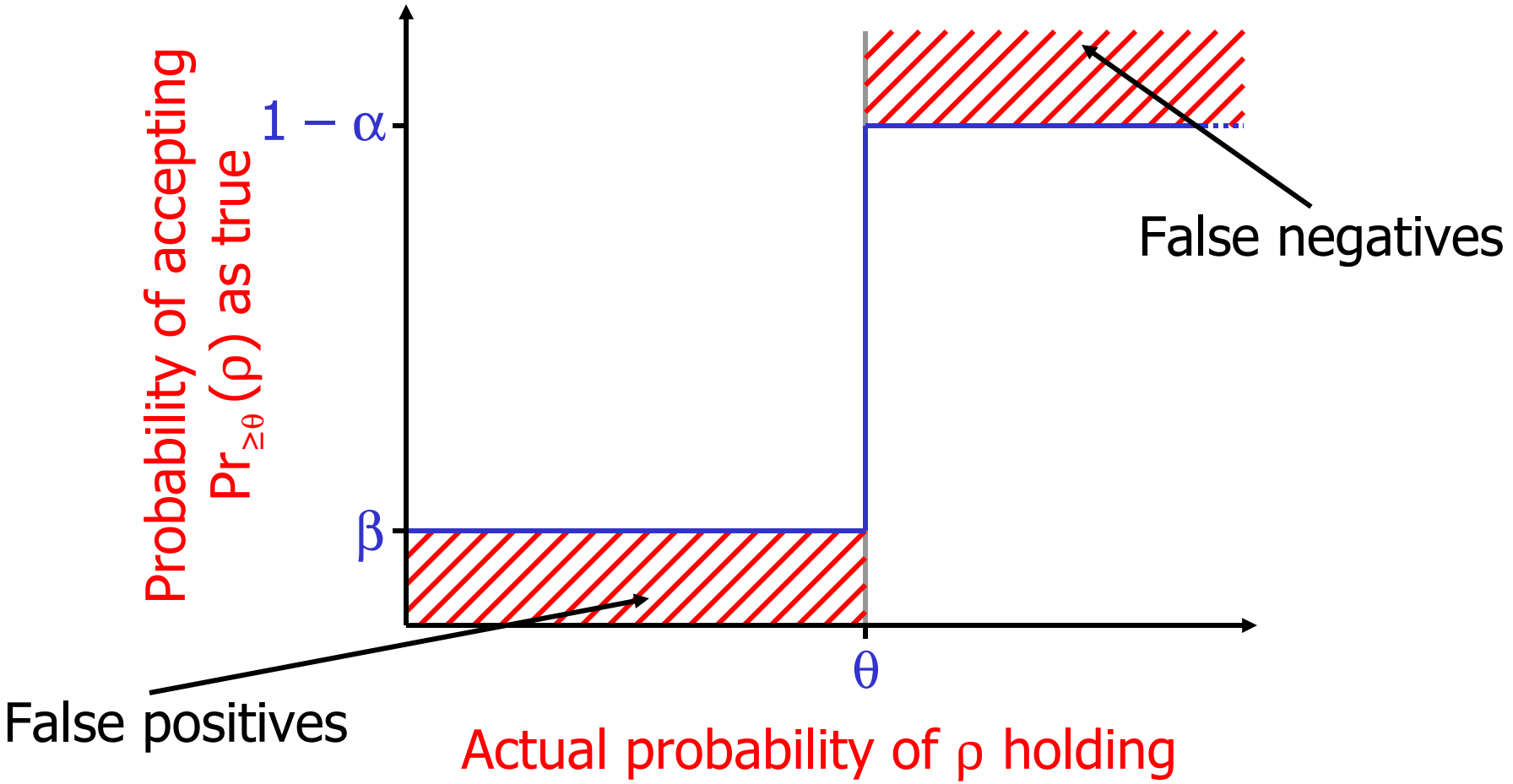
Acceptance Sampling

- Hypothesis: $Pr_{\geq\theta}(\rho)$

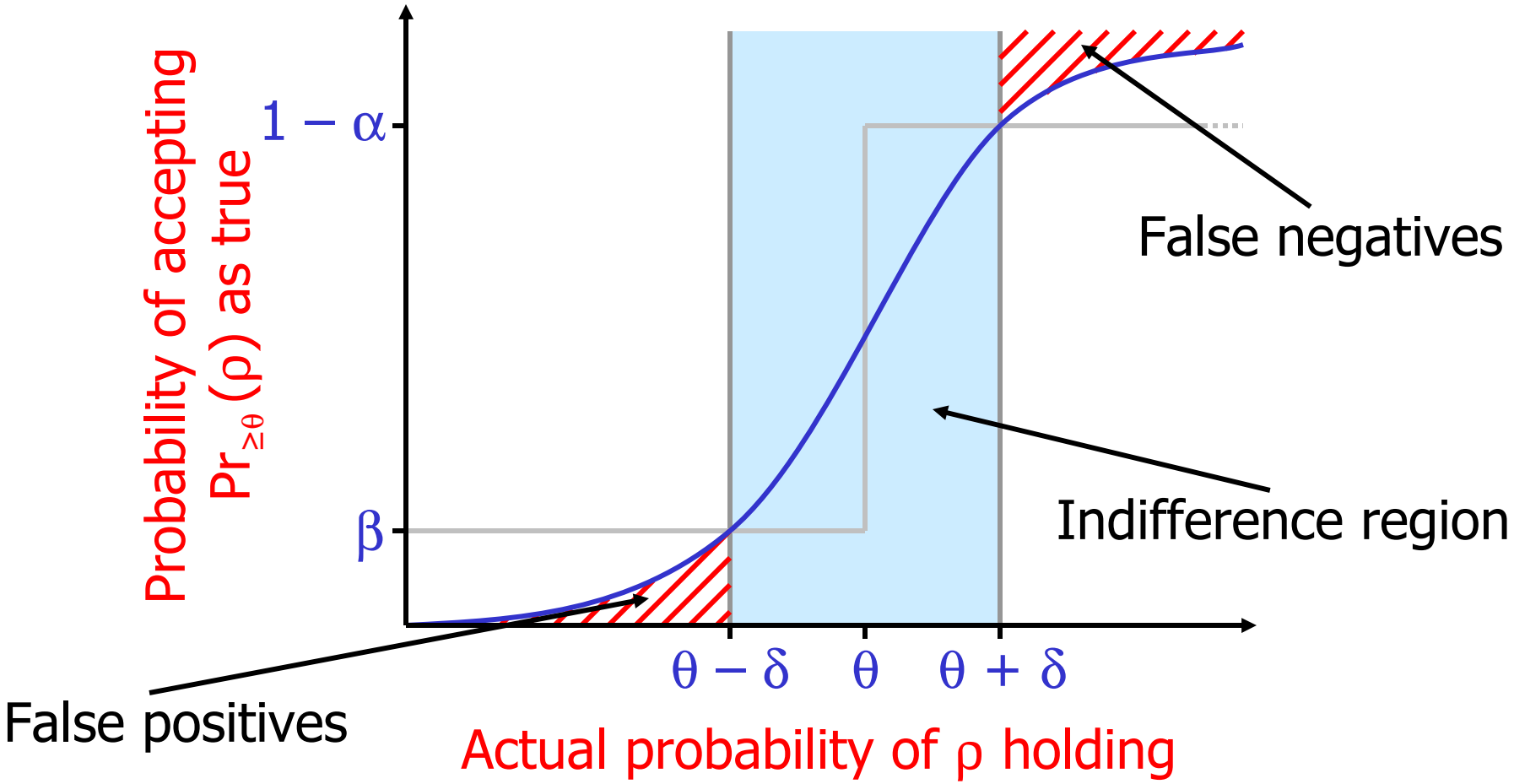
Performance of Test



Ideal Performance

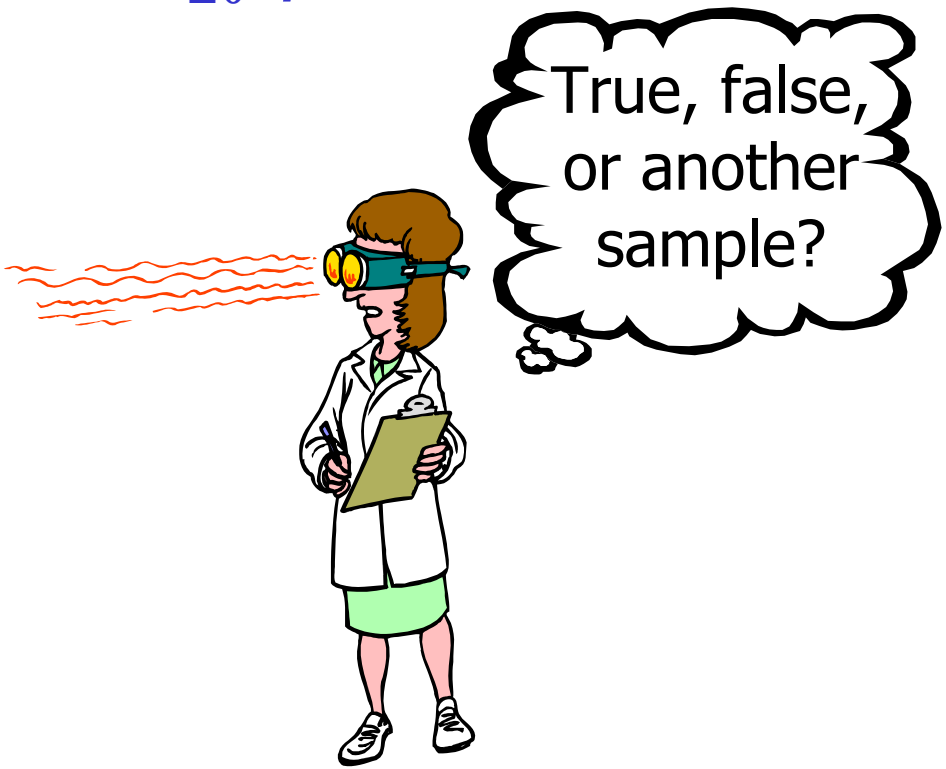


Actual Performance

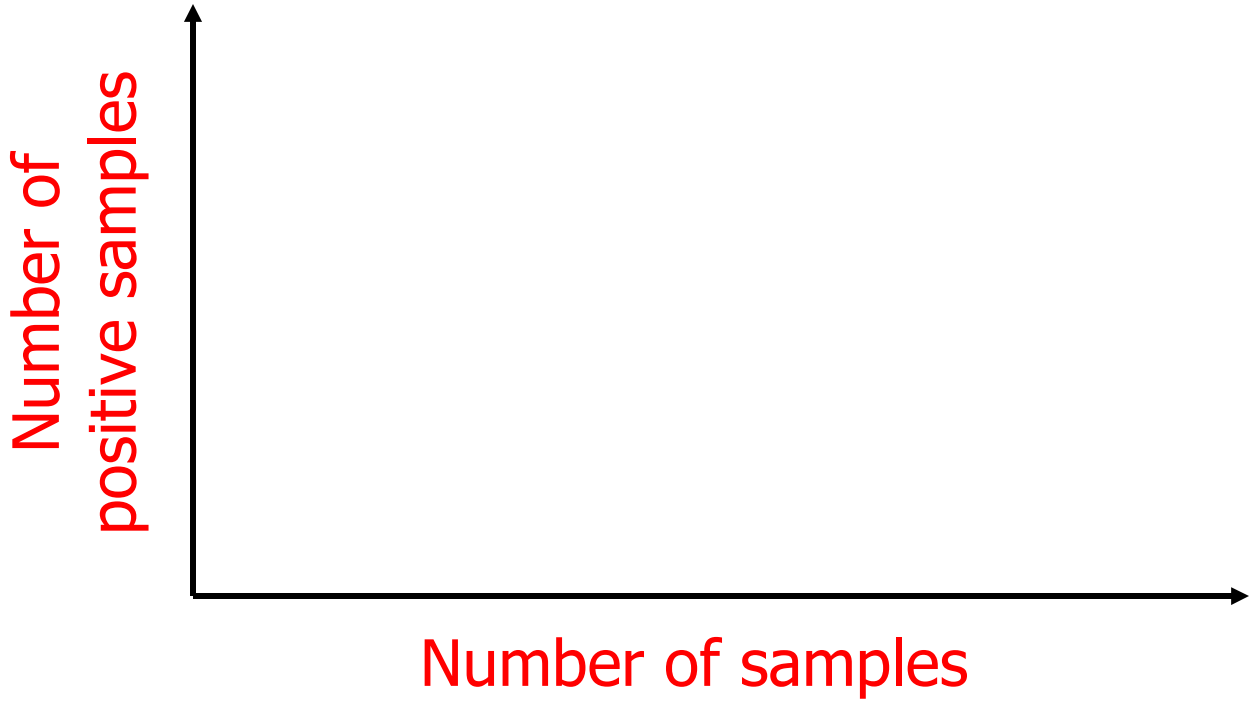


Sequential Acceptance Sampling

- Hypothesis: $Pr_{\geq\theta}(\rho)$

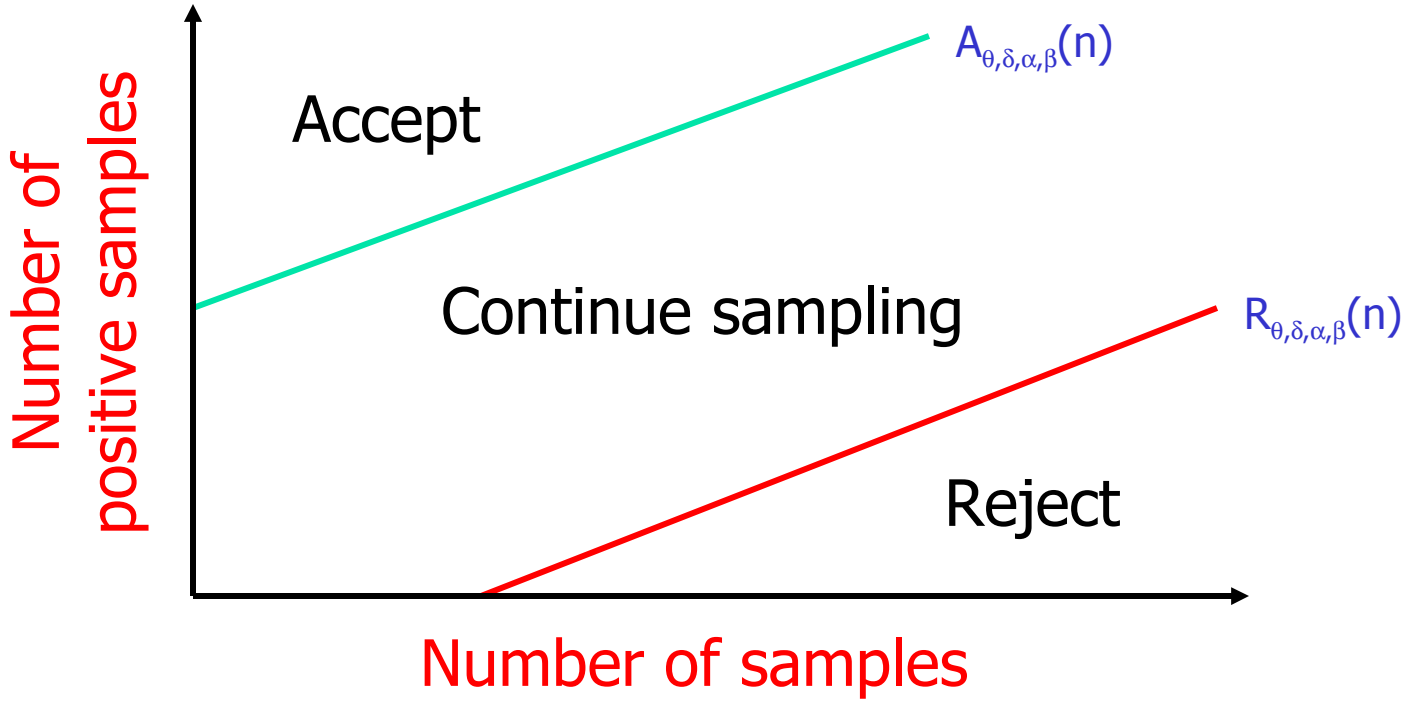


Graphical Representation of Sequential Test



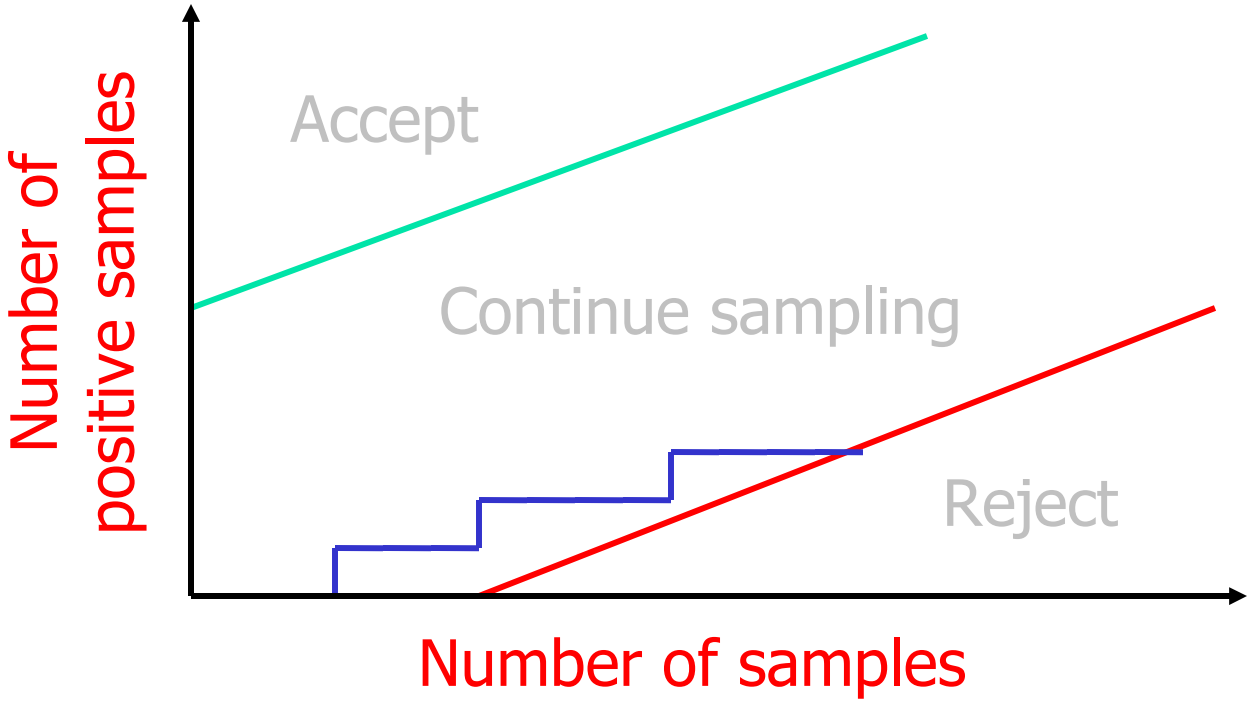
Graphical Representation of Sequential Test

- We can find an acceptance line and a rejection line given θ, δ, α, and β



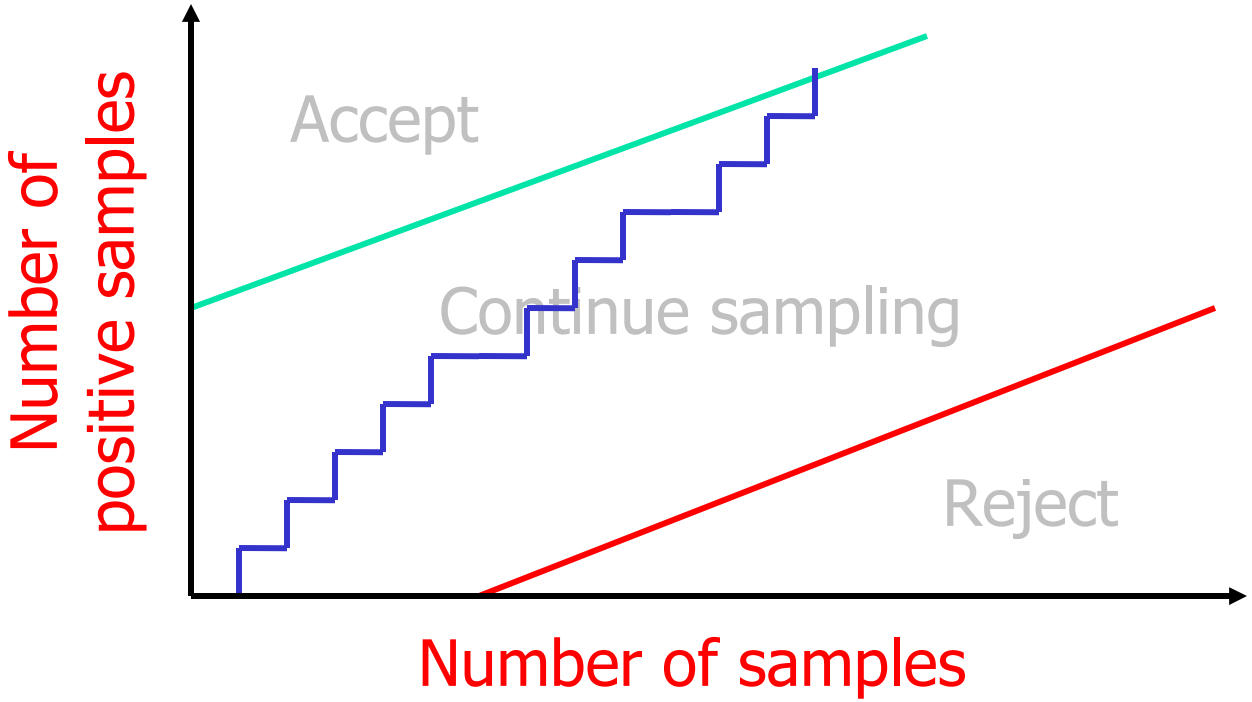
Graphical Representation of Sequential Test

- Reject hypothesis

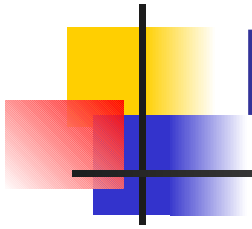


Graphical Representation of Sequential Test

- Accept hypothesis

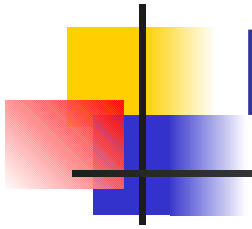


Verifying Probabilistic Properties



- Verify $Pr_{\geq\theta}(\rho)$ with error bounds α and β
  - Generate sample execution paths using simulation
  - Verify ρ over each sample execution path
    - If ρ is true, then we have a positive sample
    - If ρ is false, then we have a negative sample
  - Use sequential acceptance sampling to test the hypothesis $Pr_{\geq\theta}(\rho)$

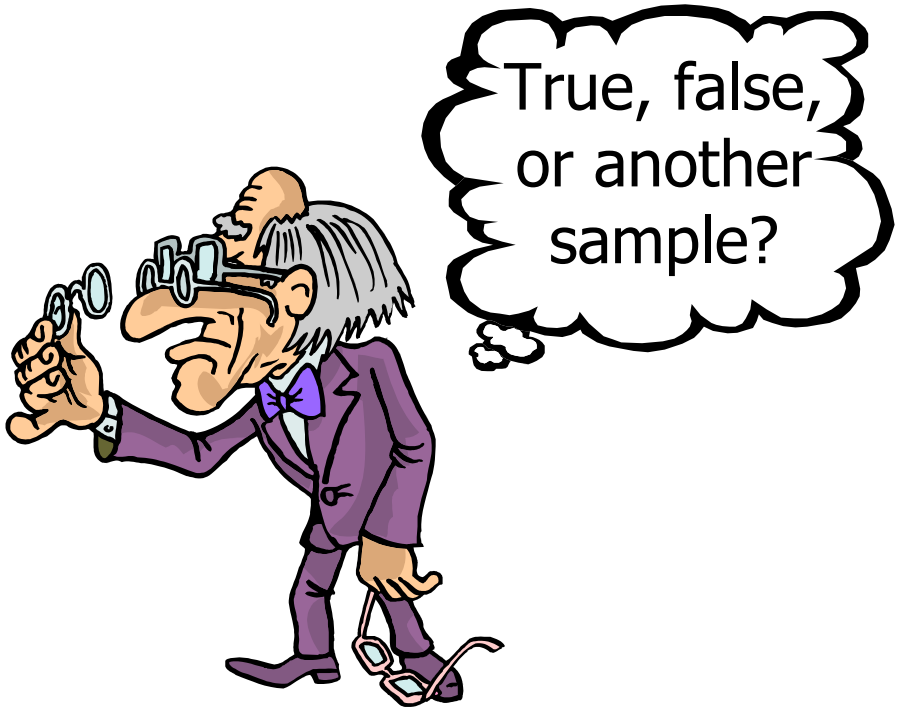
Verification of Nested Probabilistic Statements



- Suppose ρ, in $Pr_{\geq\theta}(\rho)$, contains probabilistic statements
  - Error bounds α' and β' when verifying ρ

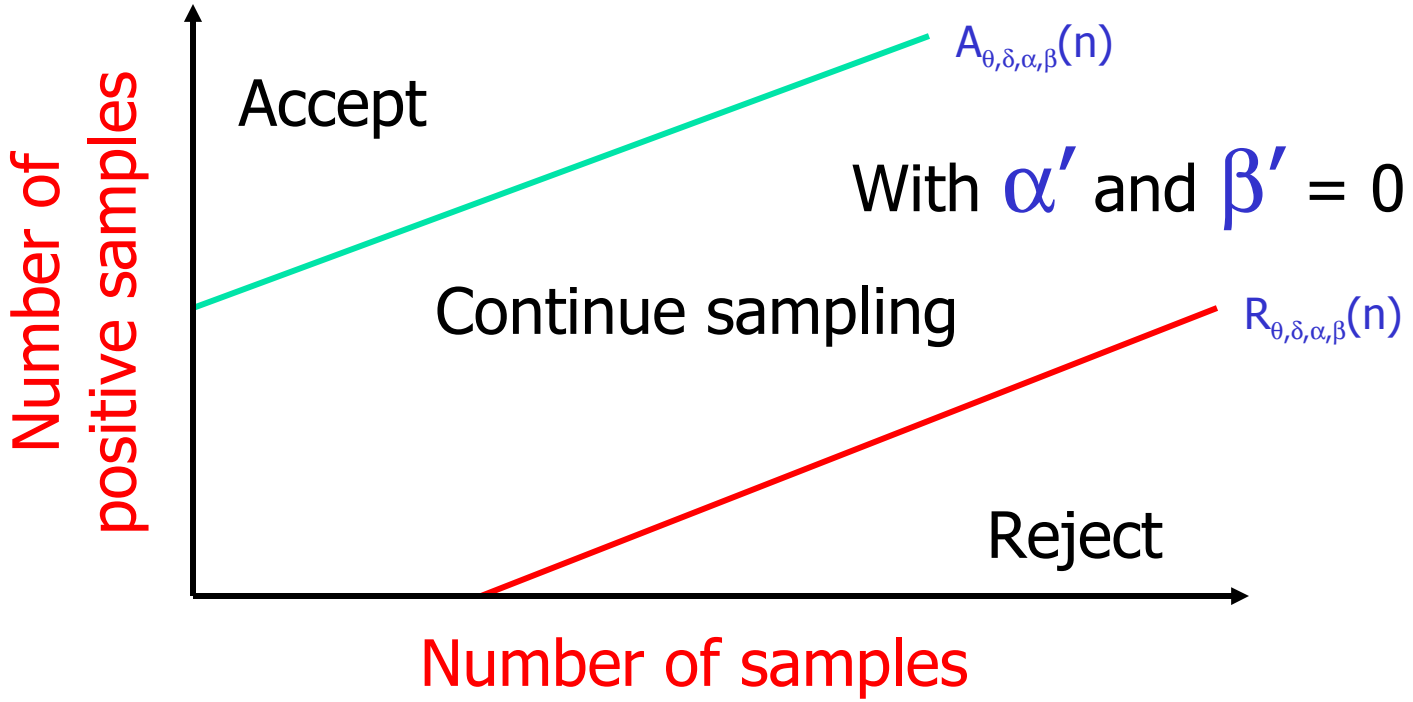
Verification of Nested Probabilistic Statements

- Suppose ρ, in $Pr_{\geq\theta}(\rho)$, contains probabilistic statements



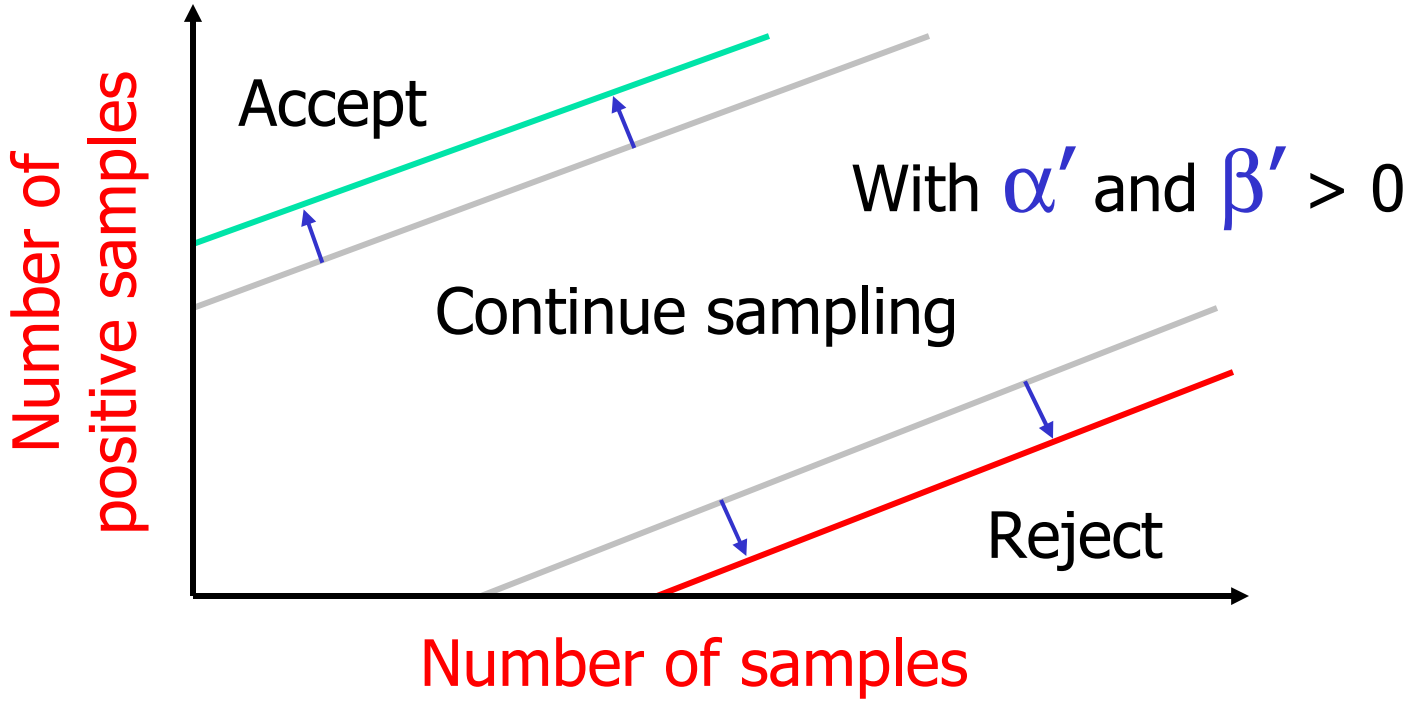
Modified Test

- Find an acceptance line and a rejection line given θ, δ, α, β, α', and β':



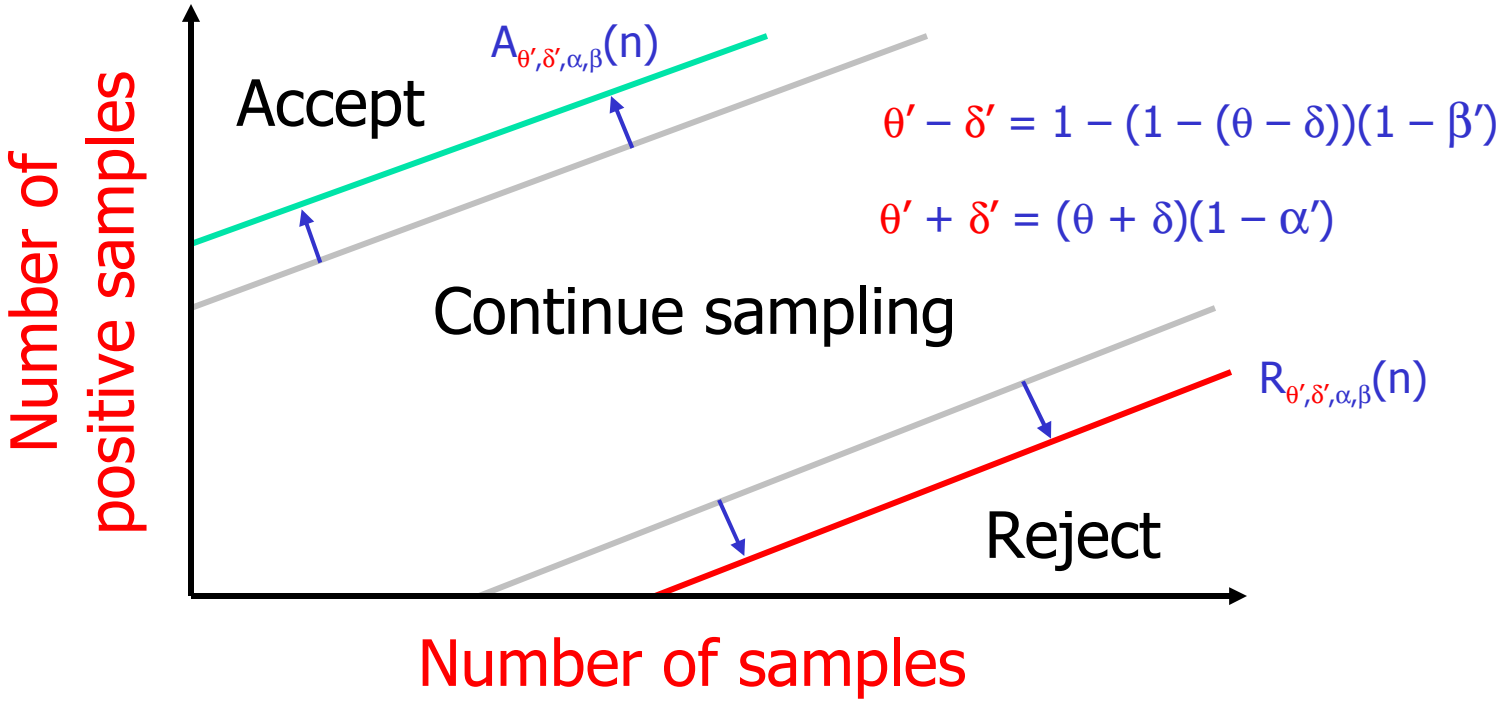
Modified Test

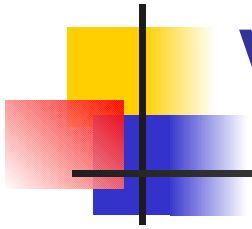
- Find an acceptance line and a rejection line given θ, δ, α, β, α', and β':



Modified Test

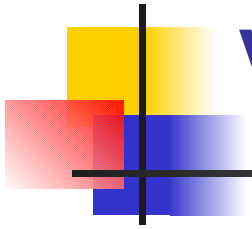
- Find an acceptance line and a rejection line given θ, δ, α, β, α', and β':





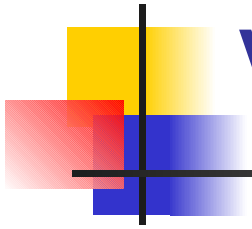
Verification of Negation

- To verify $\neg\varphi$ with error bounds α and β
  - Verify φ with error bounds β and α



Verification of Conjunction

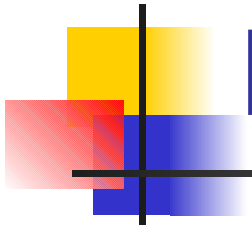
- Verify $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$ with error bounds α and β
  - Verify each φ_i with error bounds α and β/n



Verification of Path Formulas

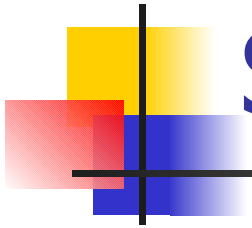
- To verify $\varphi_1 \, U^{\leq t} \, \varphi_2$ with error bounds α and β
  - Convert to disjunction
    - $\varphi_1 \, U^{\leq t} \, \varphi_2$ holds if φ_2 holds in the first state, or if φ_2 holds in the second state and φ_1 holds in all prior states, or …

More on Verifying Until



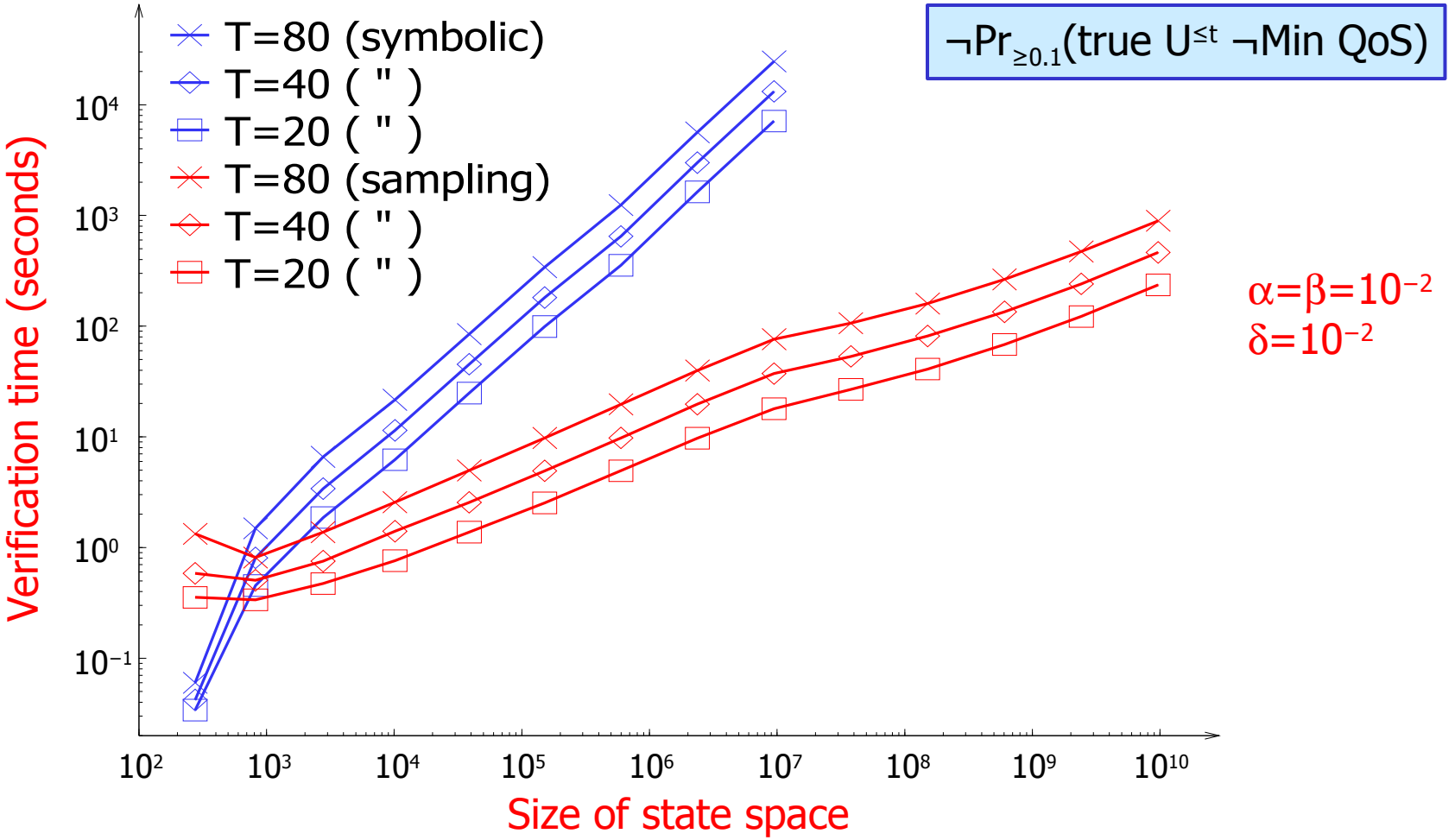
- Given $\varphi_1 \, U^{\leq t} \, \varphi_2$, let n be the index of the first state more than t time units away from the current state
- Disjunction of n conjunctions c_1 through c_n, each of size i
- Simplifies if φ_1 or φ_2, or both, do not contain any probabilistic statements

Sampling-Based vs. Symbolic Methods

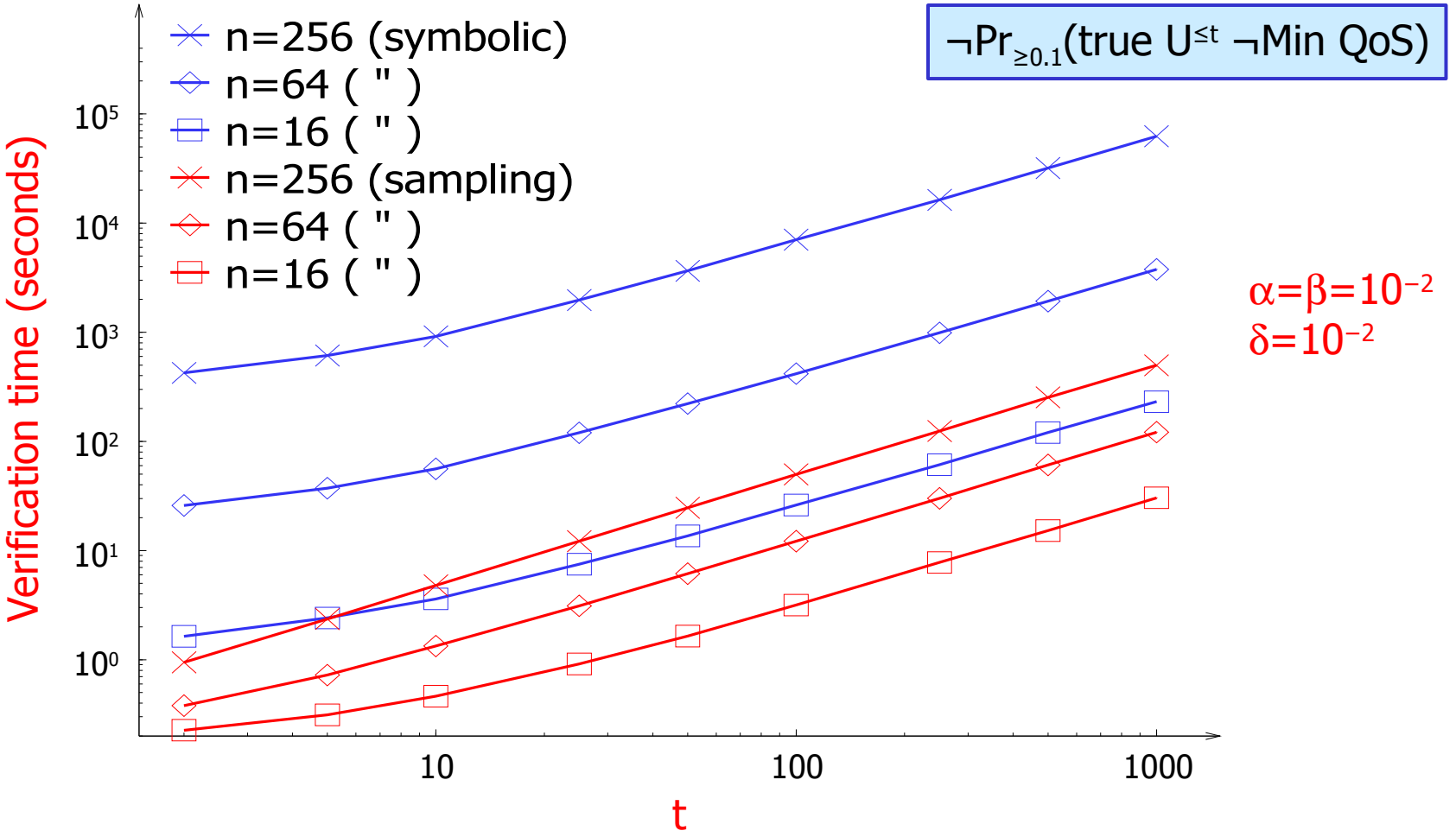


- Sampling-based methods…
  - use less memory
  - scale better with size of state space
  - adapt to difficulty of problem (sequential)
- Symbolic methods…
  - give exact results
  - handle time-unbounded and steady-state properties

Dependable Workstation Cluster (results)

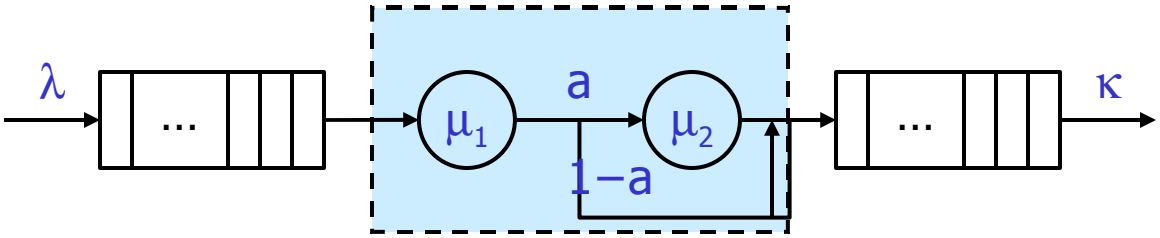


Dependable Workstation Cluster (results)



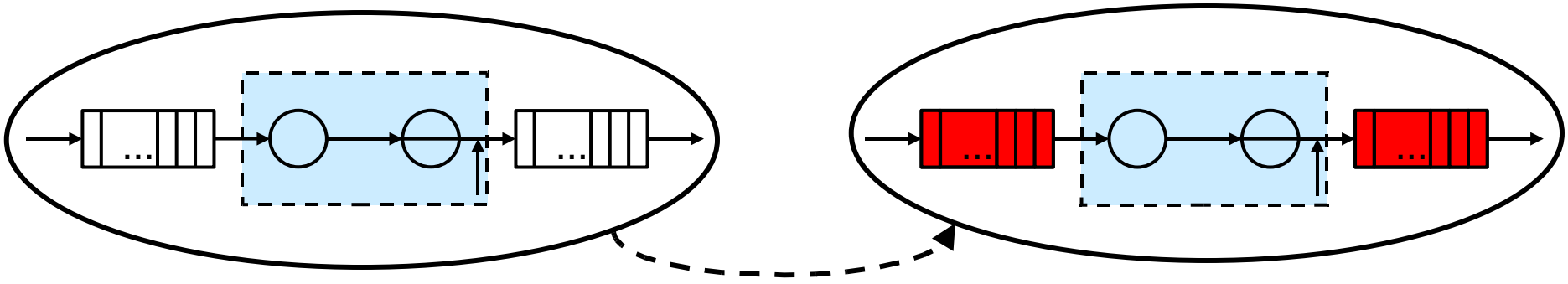
Tandem Queuing Network

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

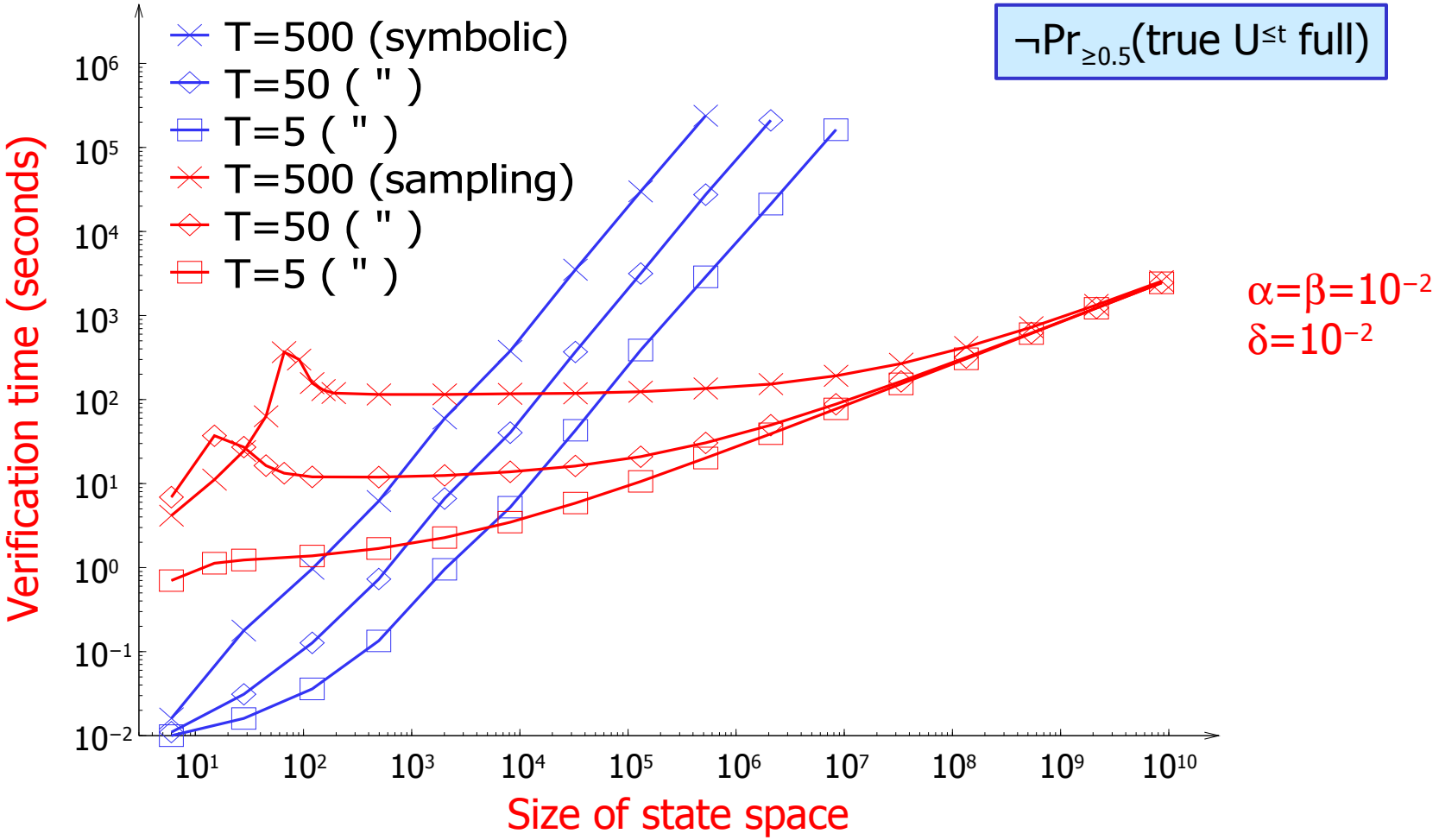


Tandem Queuing Network (property)

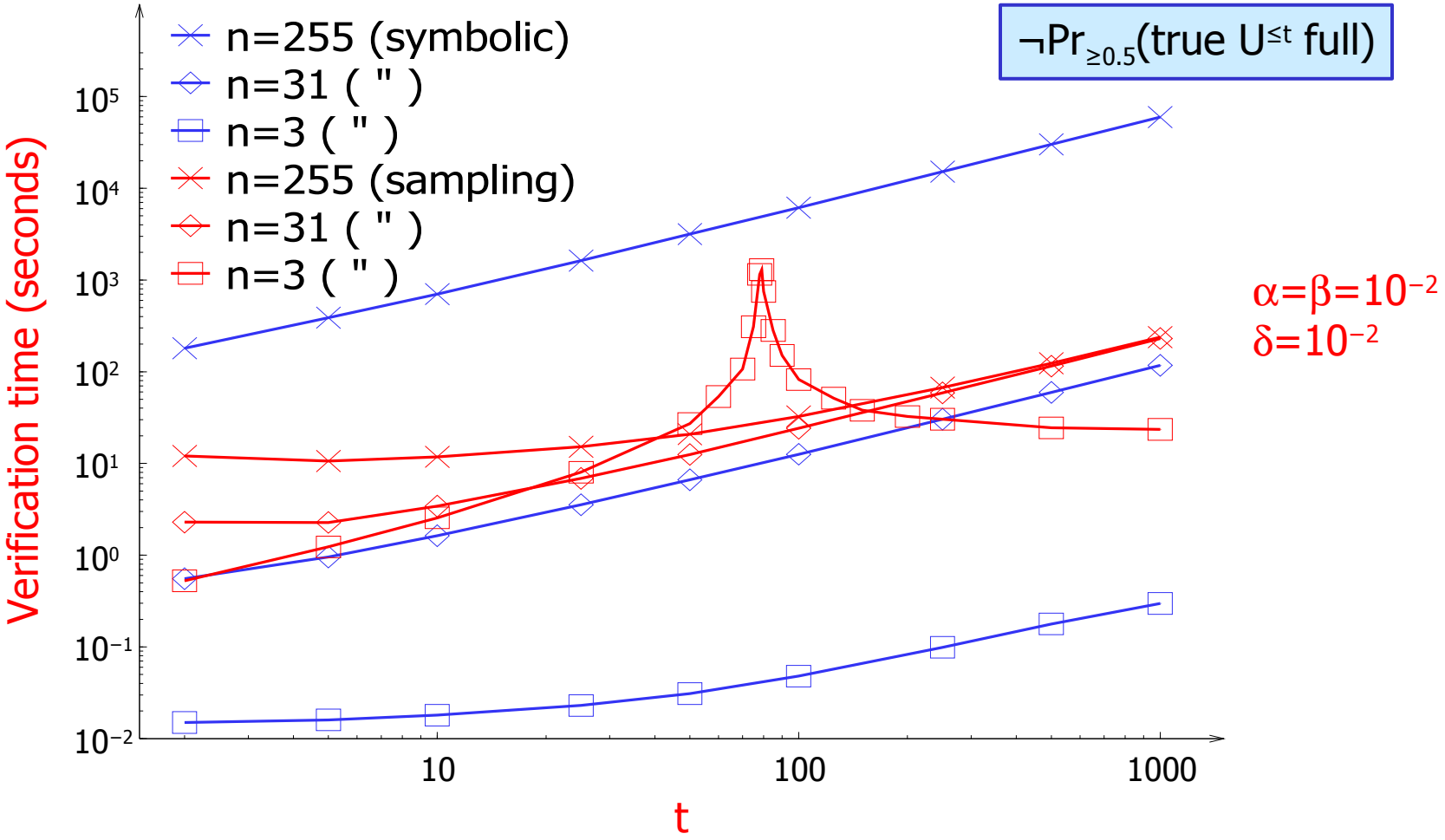
- When empty, probability is less than 0.5 that system becomes full within t time units:
  - $\neg Pr_{\geq 0.5}(true\ U^{\leq t}\ full)$ in state "empty"



Tandem Queuing Network (results)

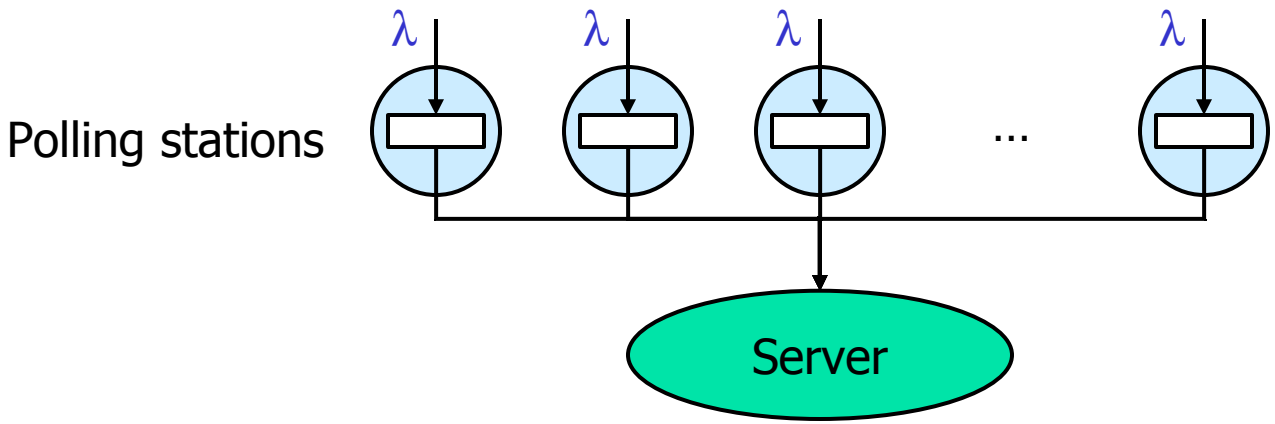


Tandem Queuing Network (results)



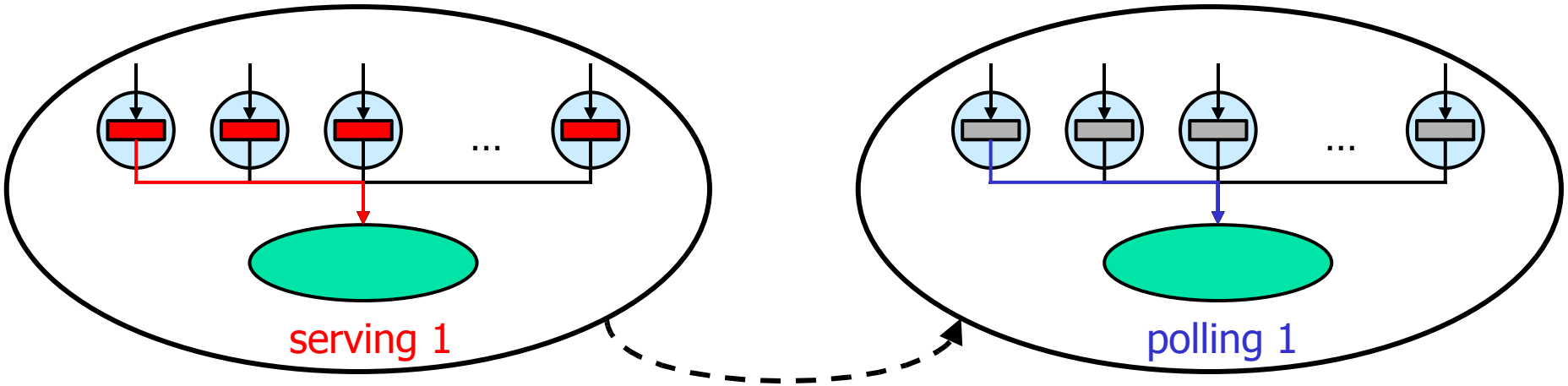
Symmetric Polling System

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

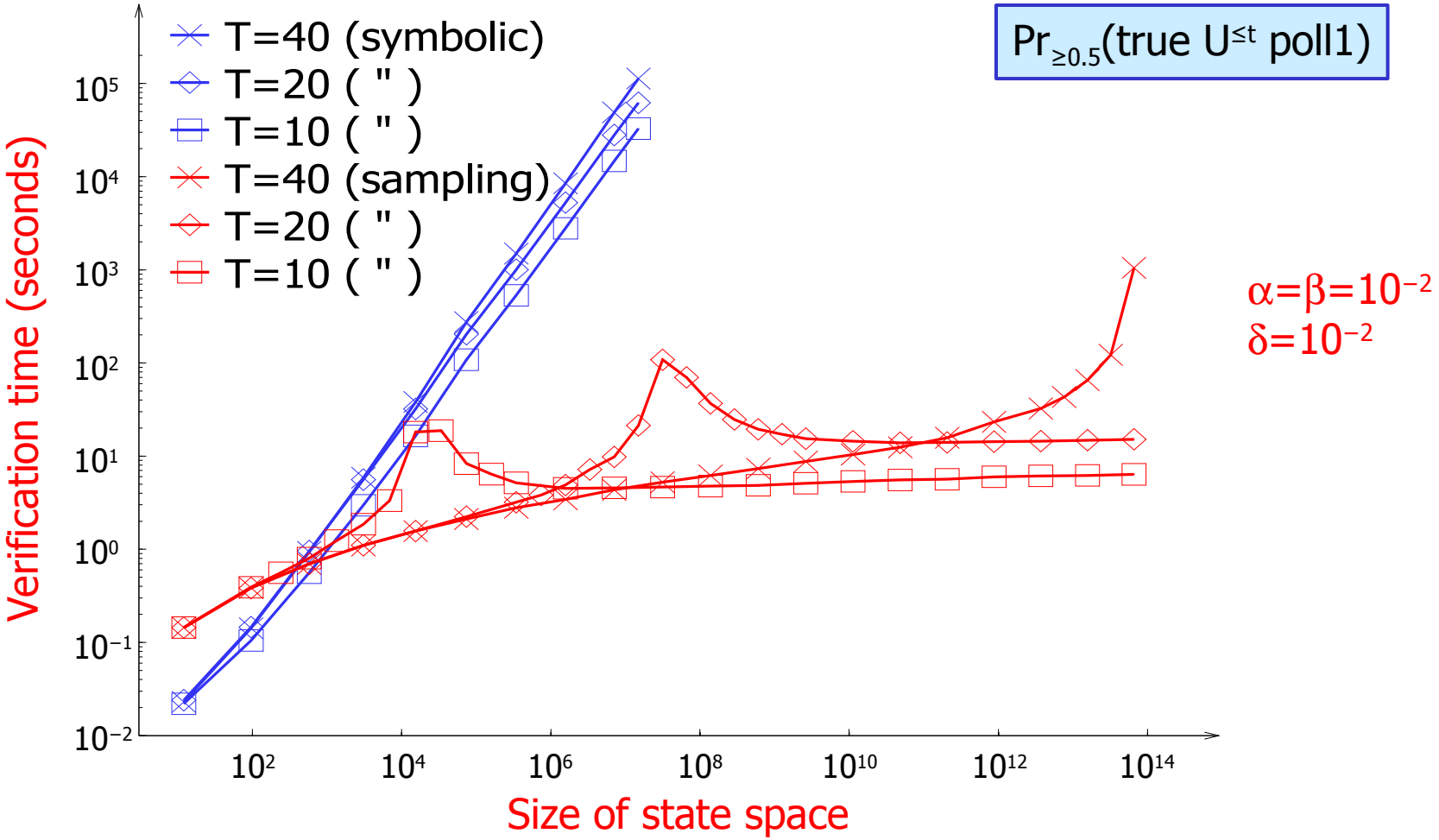


Symmetric Polling System (property)

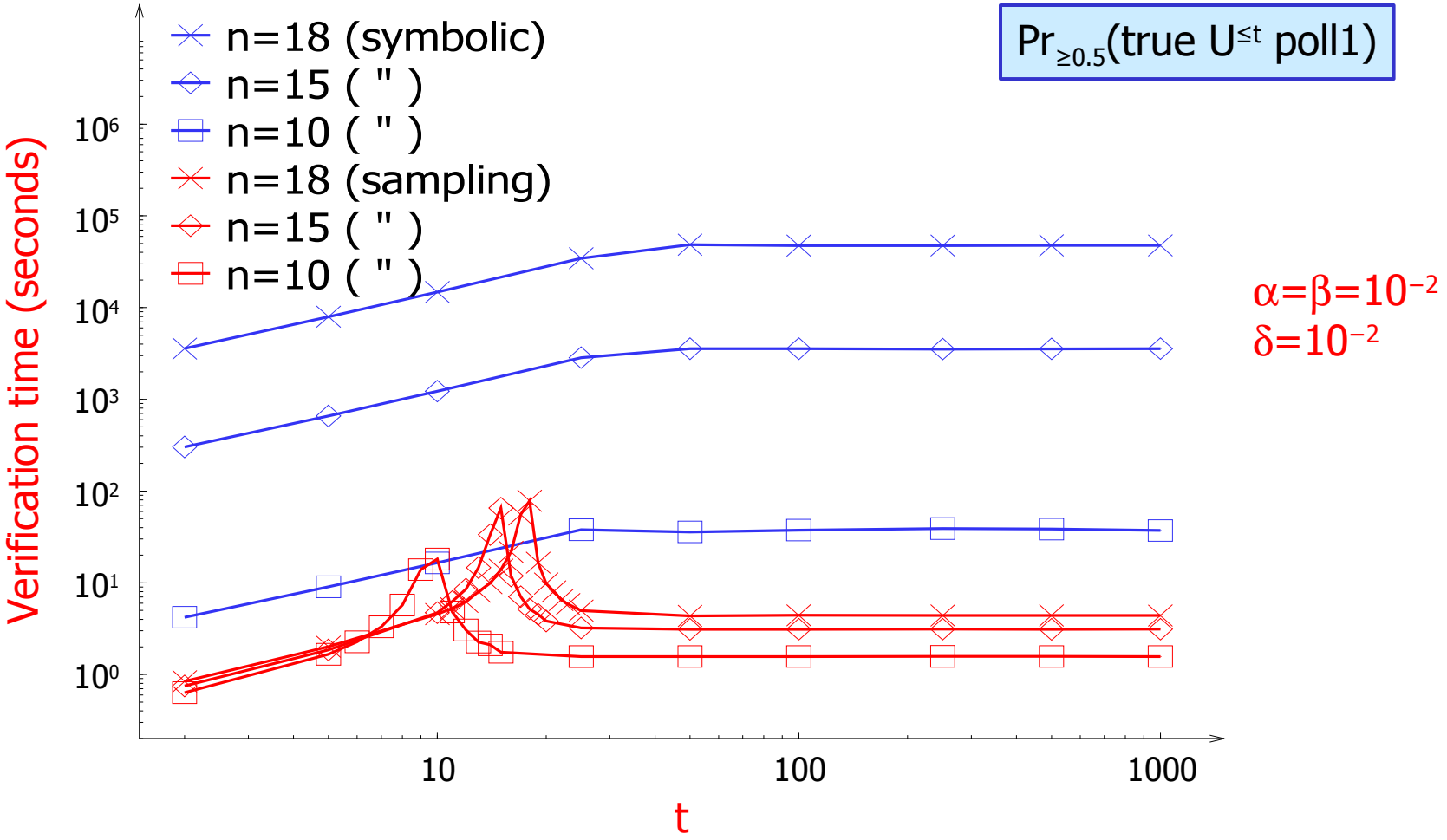
- When full and serving station 1, probability is at least 0.5 that station 1 is polled within t time units:
  - $Pr_{\geq 0.5}(true\ U^{\leq t}\ poll1)$ in state "full, srv 1"



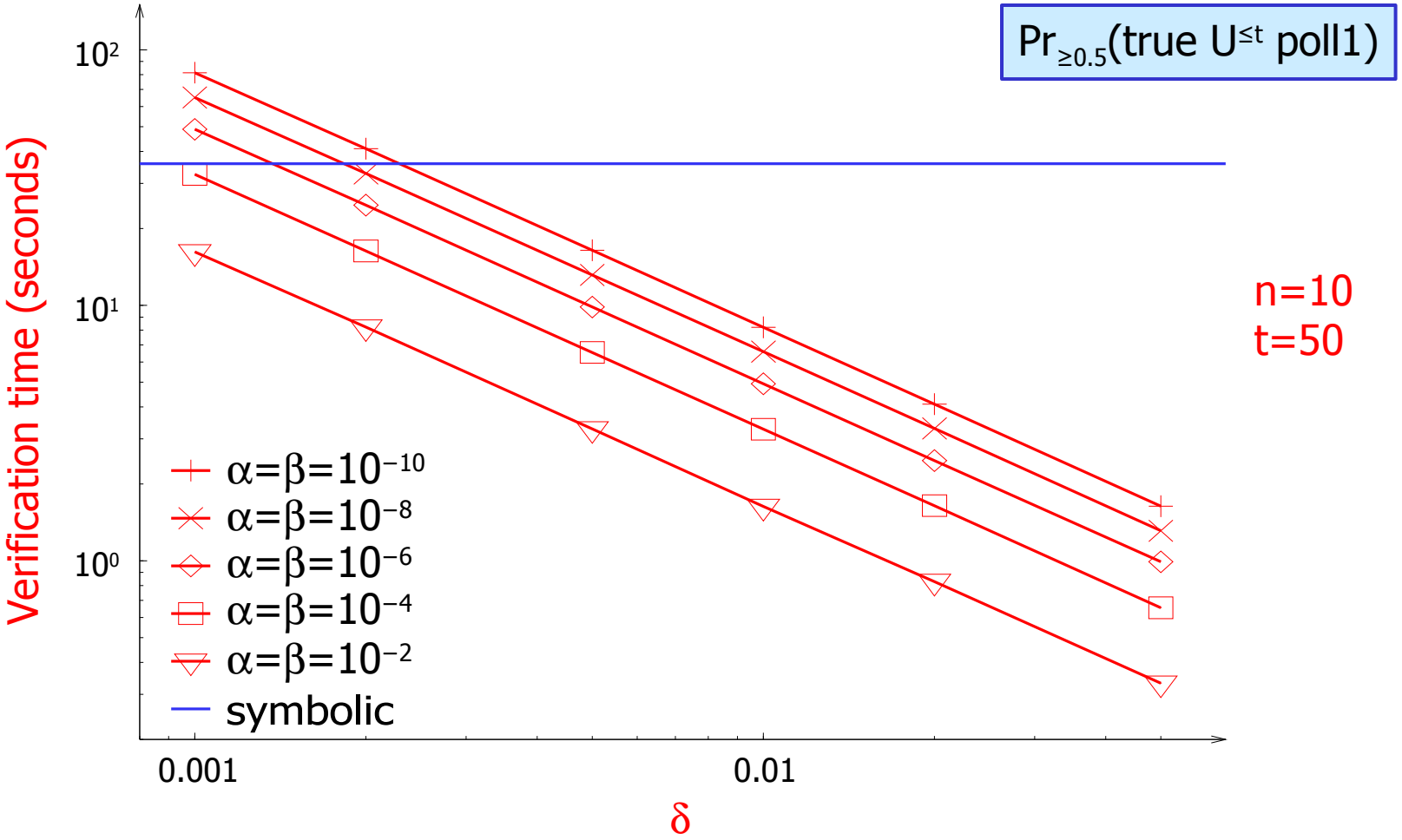
Symmetric Polling System (results)

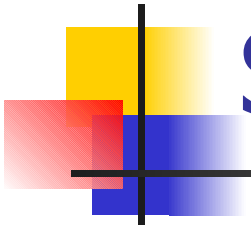


Symmetric Polling System (results)



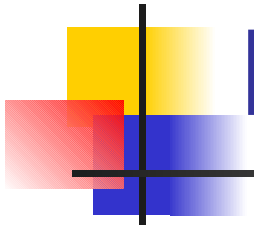
Symmetric Polling System (results)





Summary

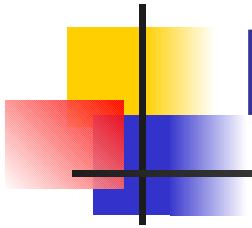
- Model independent probabilistic verification of discrete event systems
- Sample execution paths generated using simulation
- Probabilistic properties verified using sequential acceptance sampling
- Easy to trade accuracy for efficiency



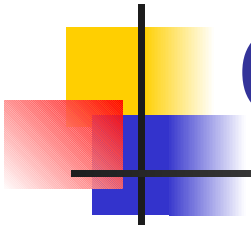
Future Work

- Develop heuristics for formula ordering and parameter selection
- Use in combination with symbolic methods
- Apply to hybrid dynamic systems
- Use verification to aid controller synthesis for discrete event systems

Verification to Aid Probabilistic Planning



- 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



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