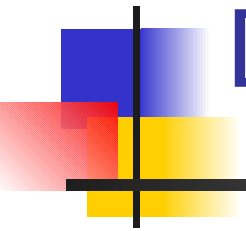
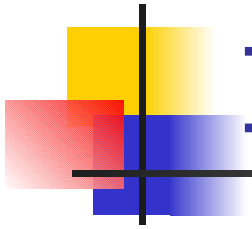


Probabilistic Verification of Discrete Event Systems

Håkan L. S. Younes

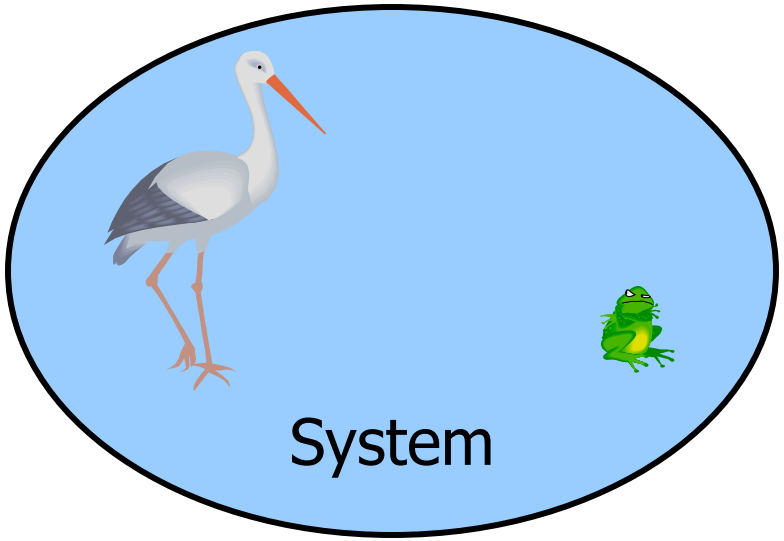


Introduction

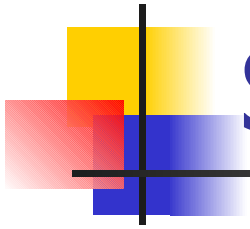
- Verify properties of discrete event systems
- Probabilistic and real-time properties
- Properties expressed using CSL
- Acceptance sampling
- Guaranteed error bounds

“The Hungry Stork”

“The Hungry Stork”

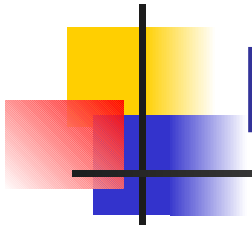


“The probability is at least 0.7 that the stork satisfies its hunger within 180 seconds”



Systems

- A stork hunting for frogs
- The CMU post office
- The Swedish telephone system
- The solar system

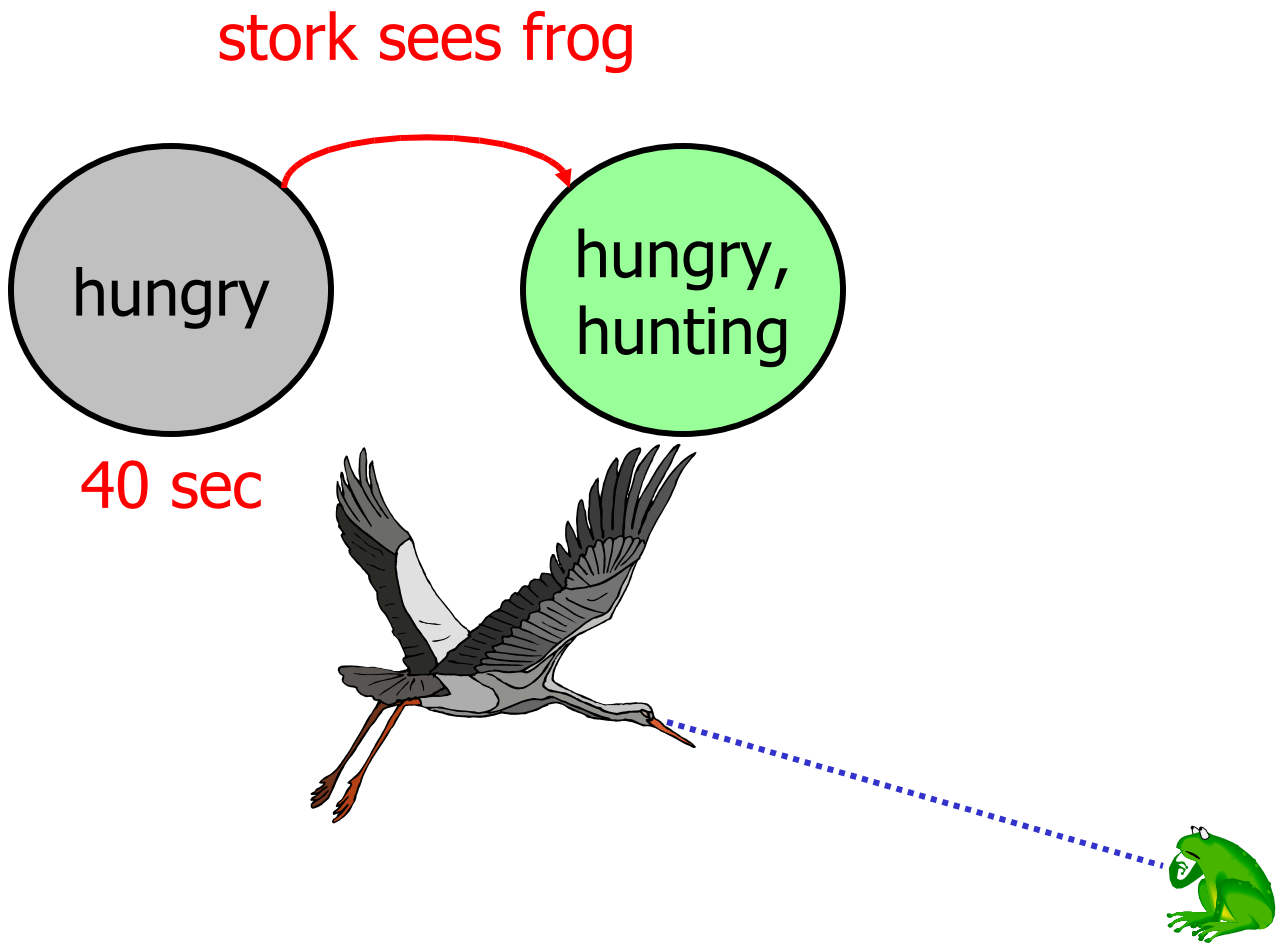


Discrete Event Systems

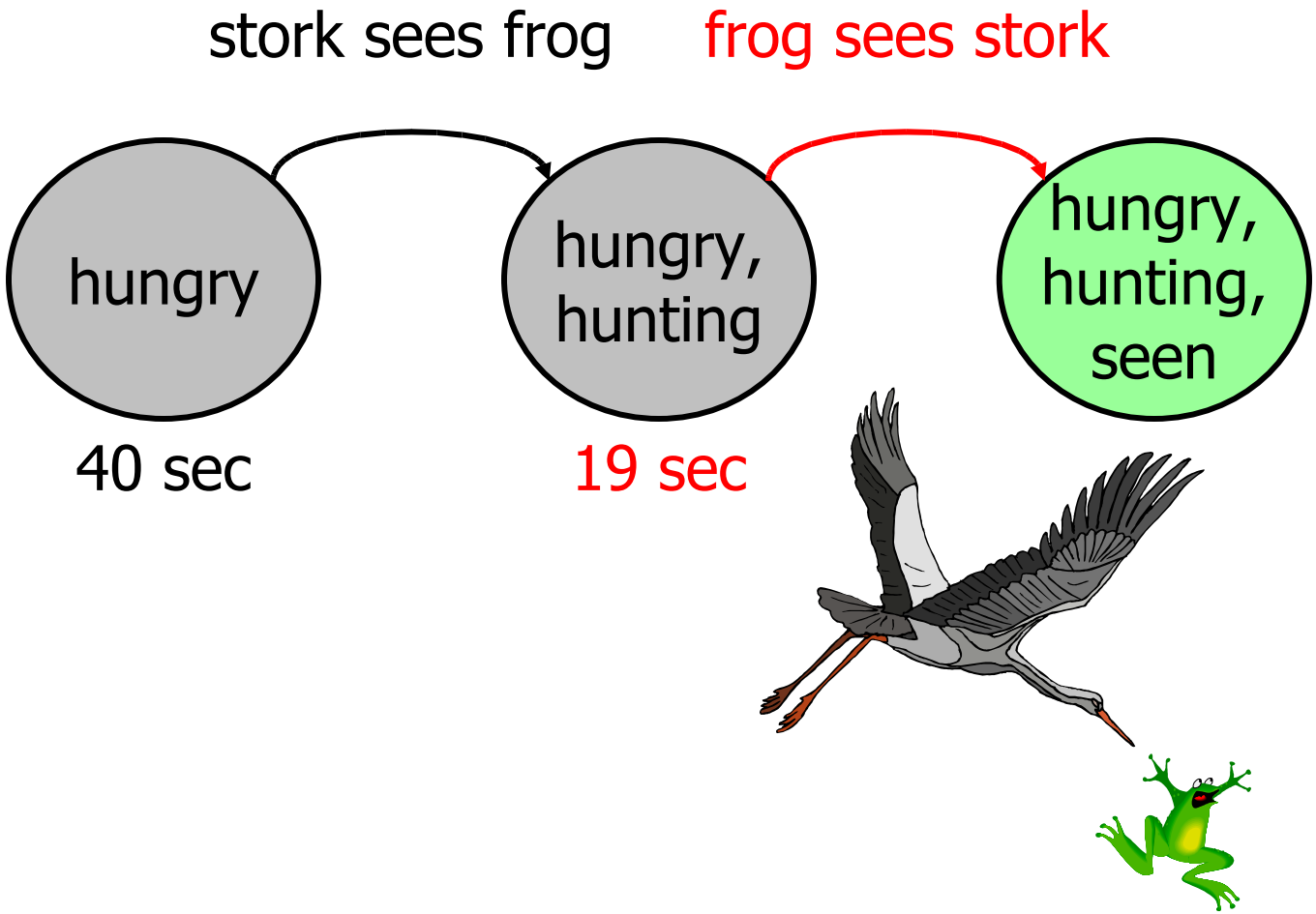
- Discrete state changes at the occurrence of events
  - A stork hunting for frogs
  - The CMU post office
  - The Swedish telephone system
  - ~~The solar system~~

“The Hungry Stork” as a Discrete Event System

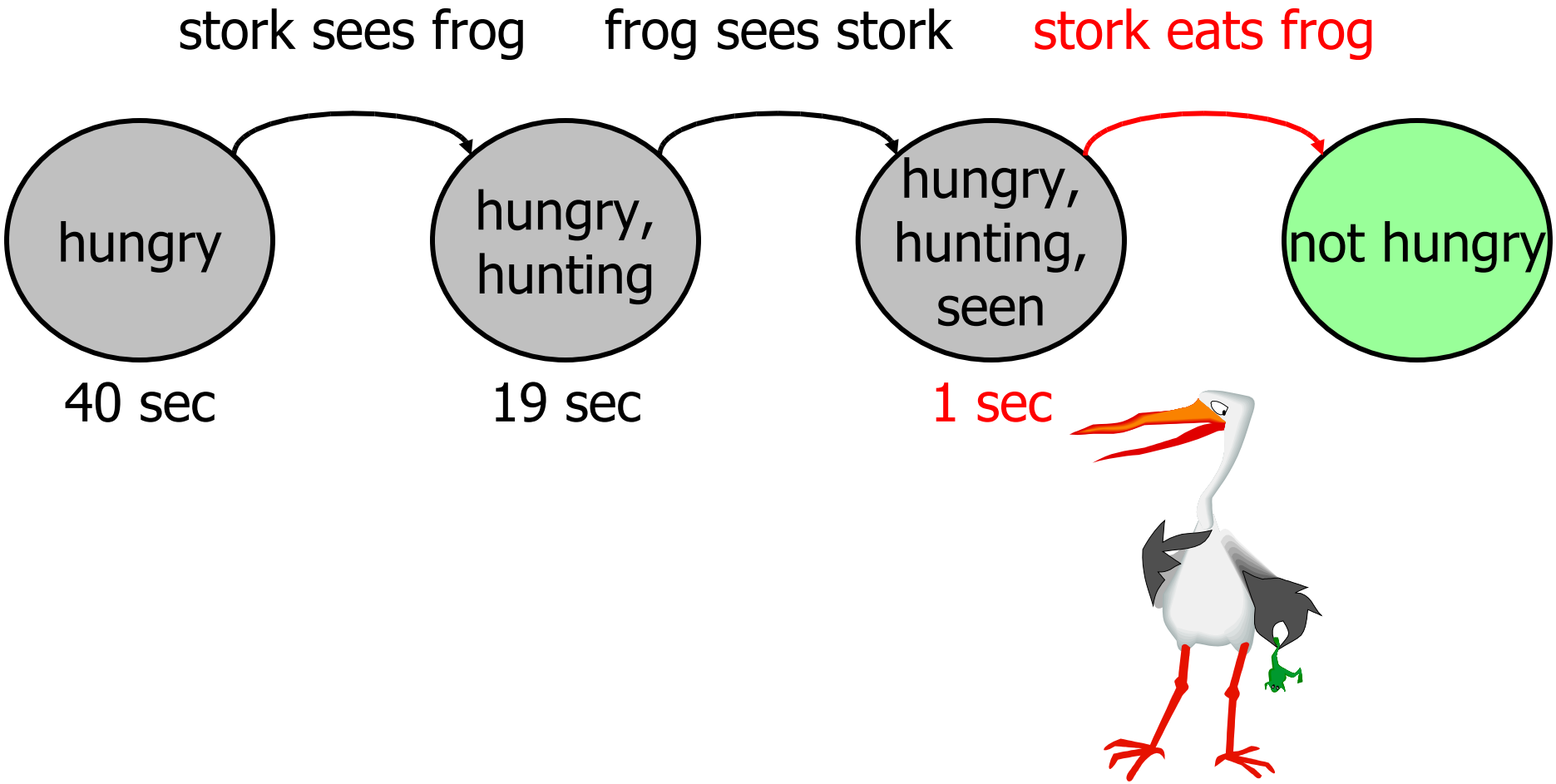
"The Hungry Stork" as a Discrete Event System



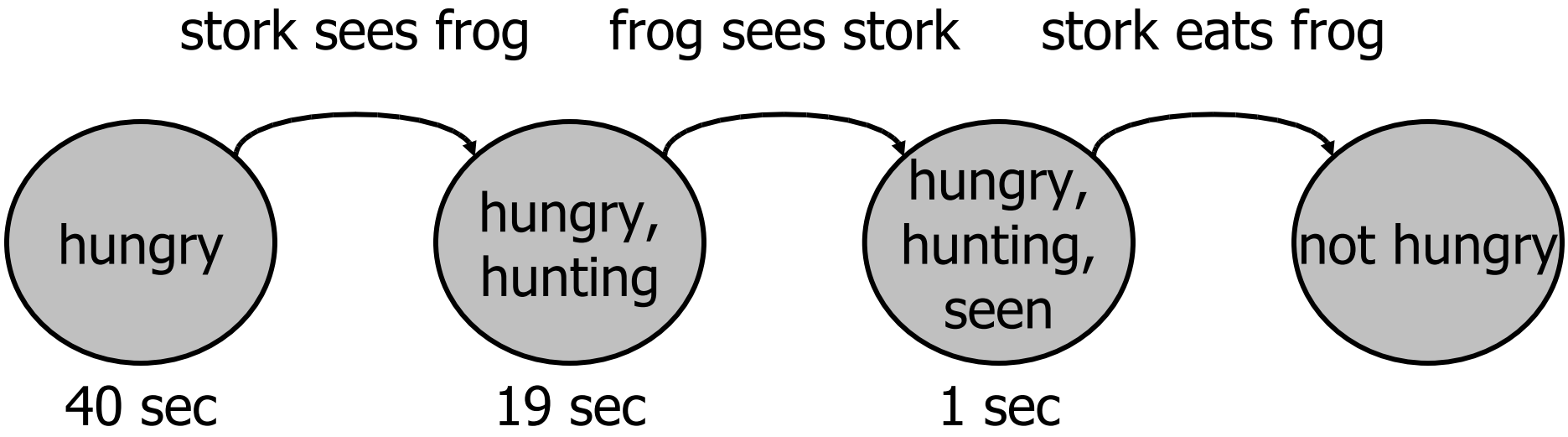
"The Hungry Stork" as a Discrete Event System

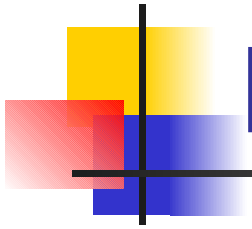


"The Hungry Stork" as a Discrete Event System



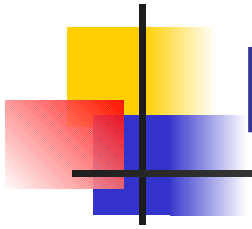
Sample Execution Paths





Properties of Interest

- Probabilistic real-time properties
  - “The probability is at least 0.7 that the stork satisfies its hunger within 180 seconds”

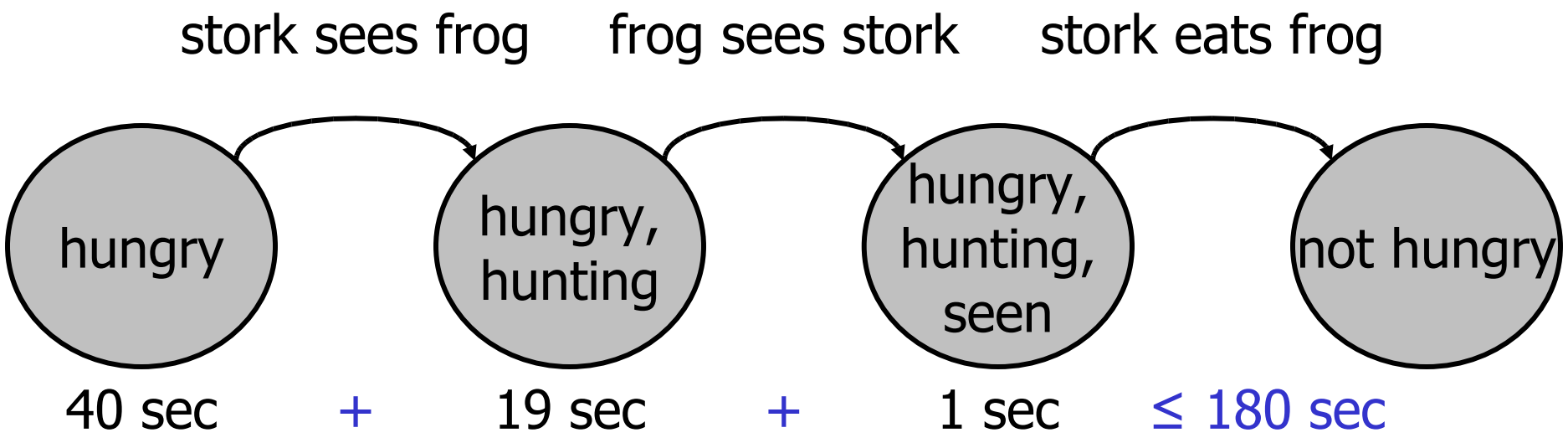


Properties of Interest

- Probabilistic real-time properties
  - "The probability is at least 0.7 that the stork satisfies its hunger within 180 seconds"

Verifying Real-time Properties

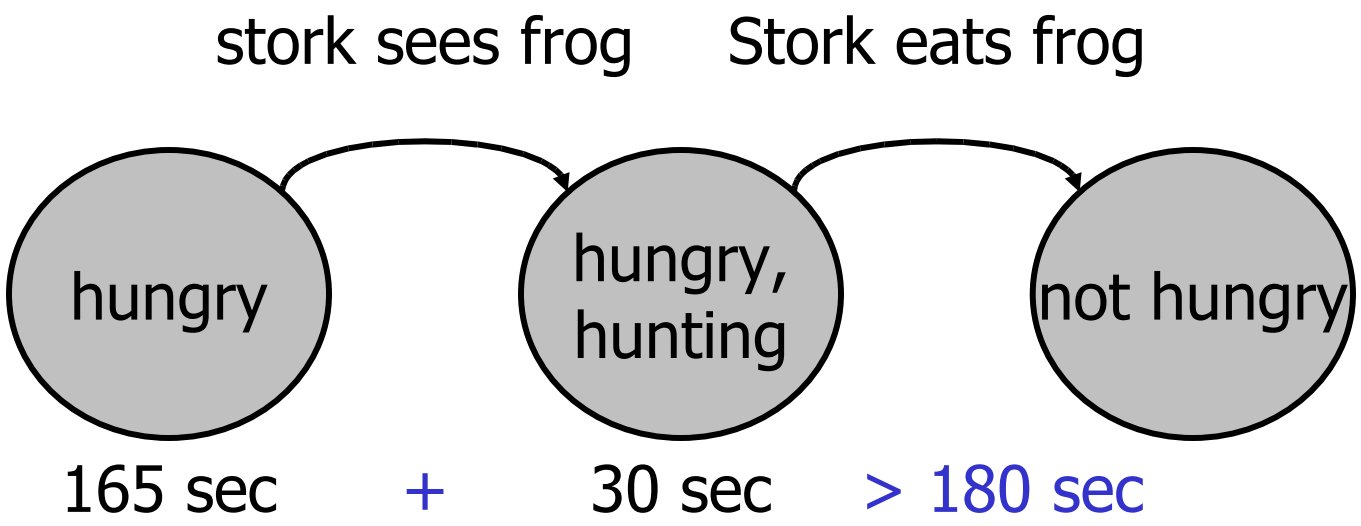
- "The stork satisfies its hunger within 180 seconds"



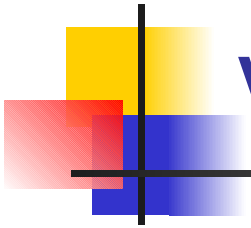
True!

Verifying Real-time Properties

- "The stork satisfies its hunger within 180 seconds"

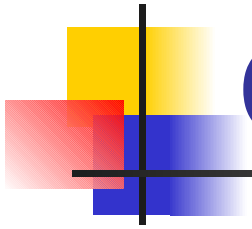


False!



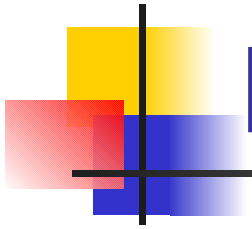
Verifying Probabilistic Properties

- "The probability is at least 0.7 that X"
  - Symbolic Methods
    - Pro: Exact solution
    - Con: Works for a restricted class of systems
  - Sampling
    - Pro: Works for all systems that can be simulated
    - Con: Uncertainty in correctness of solution



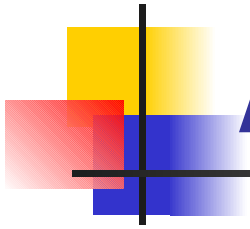
Our Approach

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



Acceptance Sampling

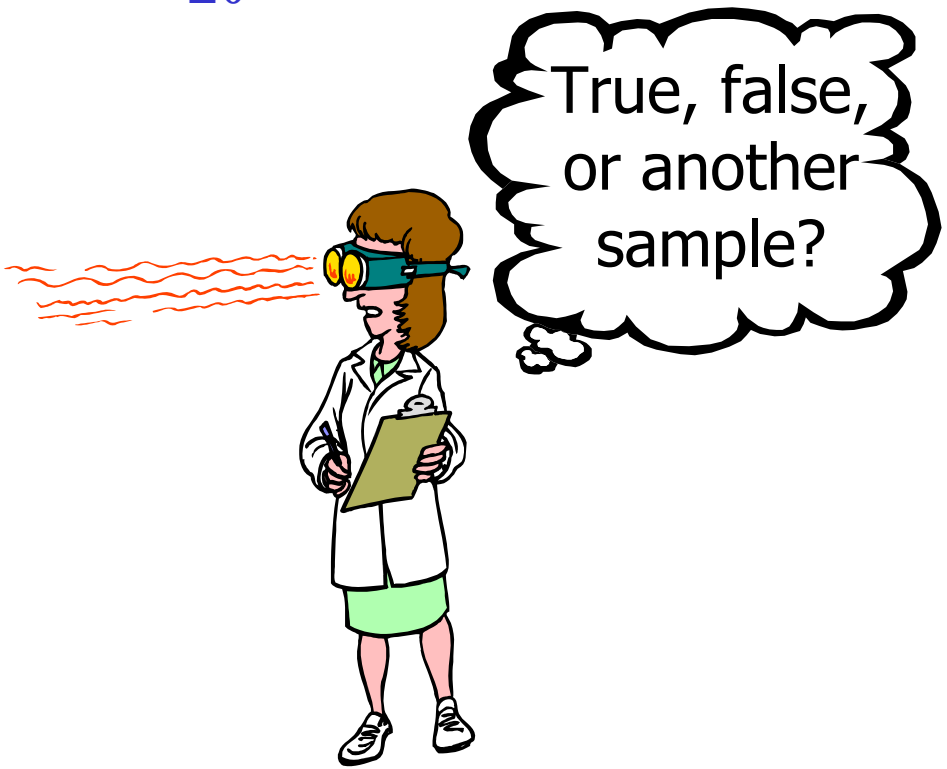
- Hypothesis: "The probability is at least θ that X"

Acceptance Sampling

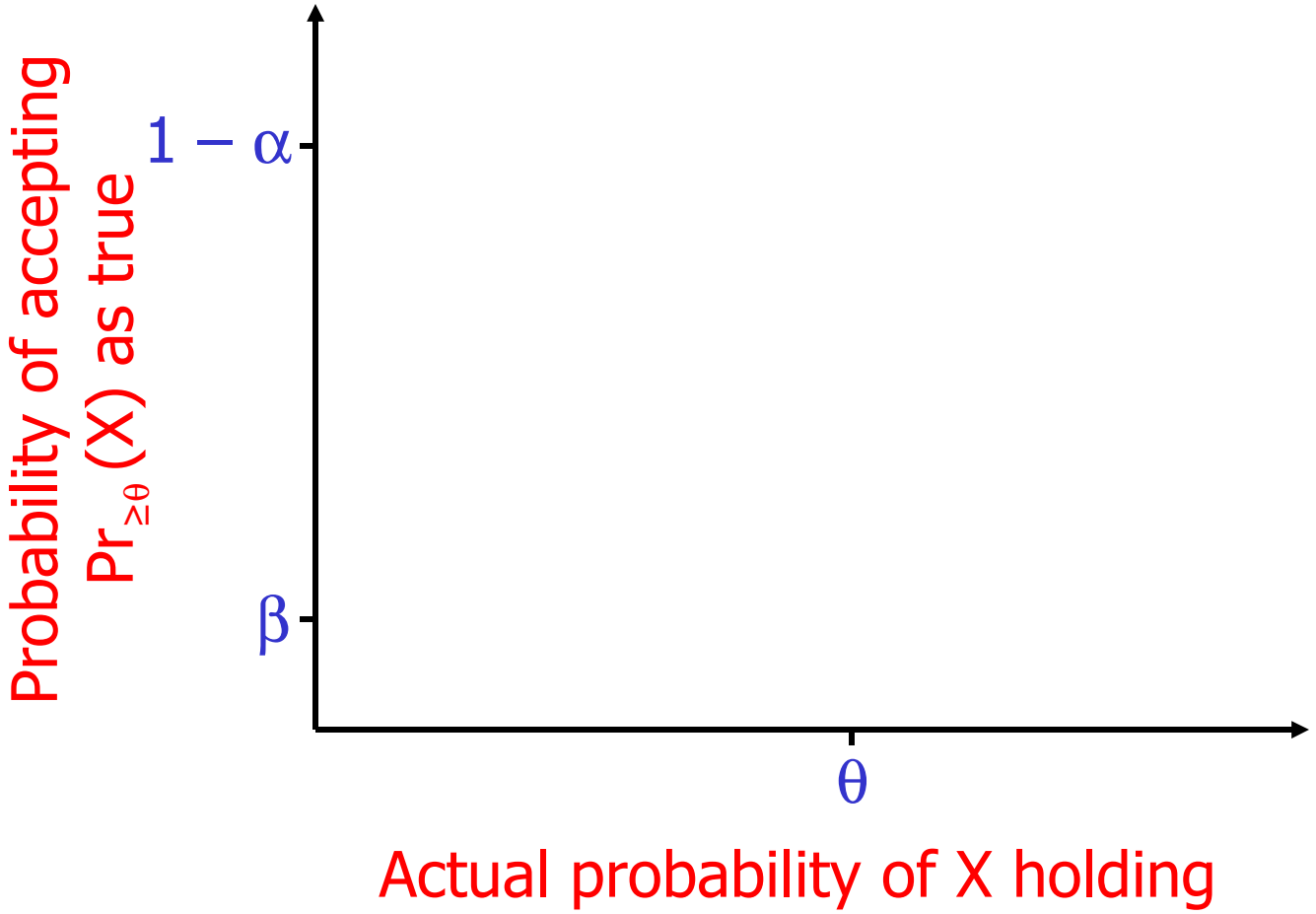
- Hypothesis: $Pr_{\geq\theta}(X)$

Sequential Acceptance Sampling

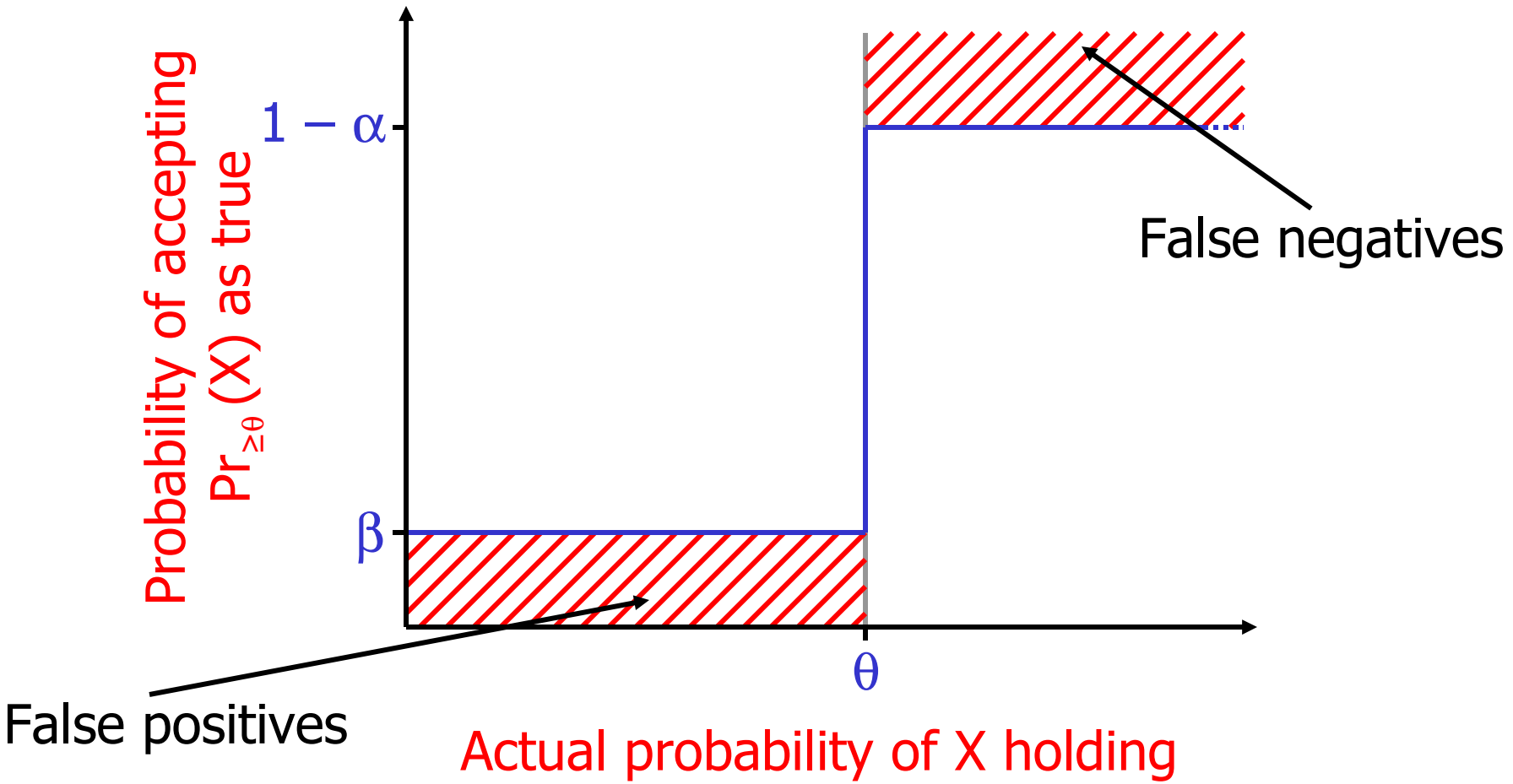
- Hypothesis: $Pr_{\geq\theta}(X)$



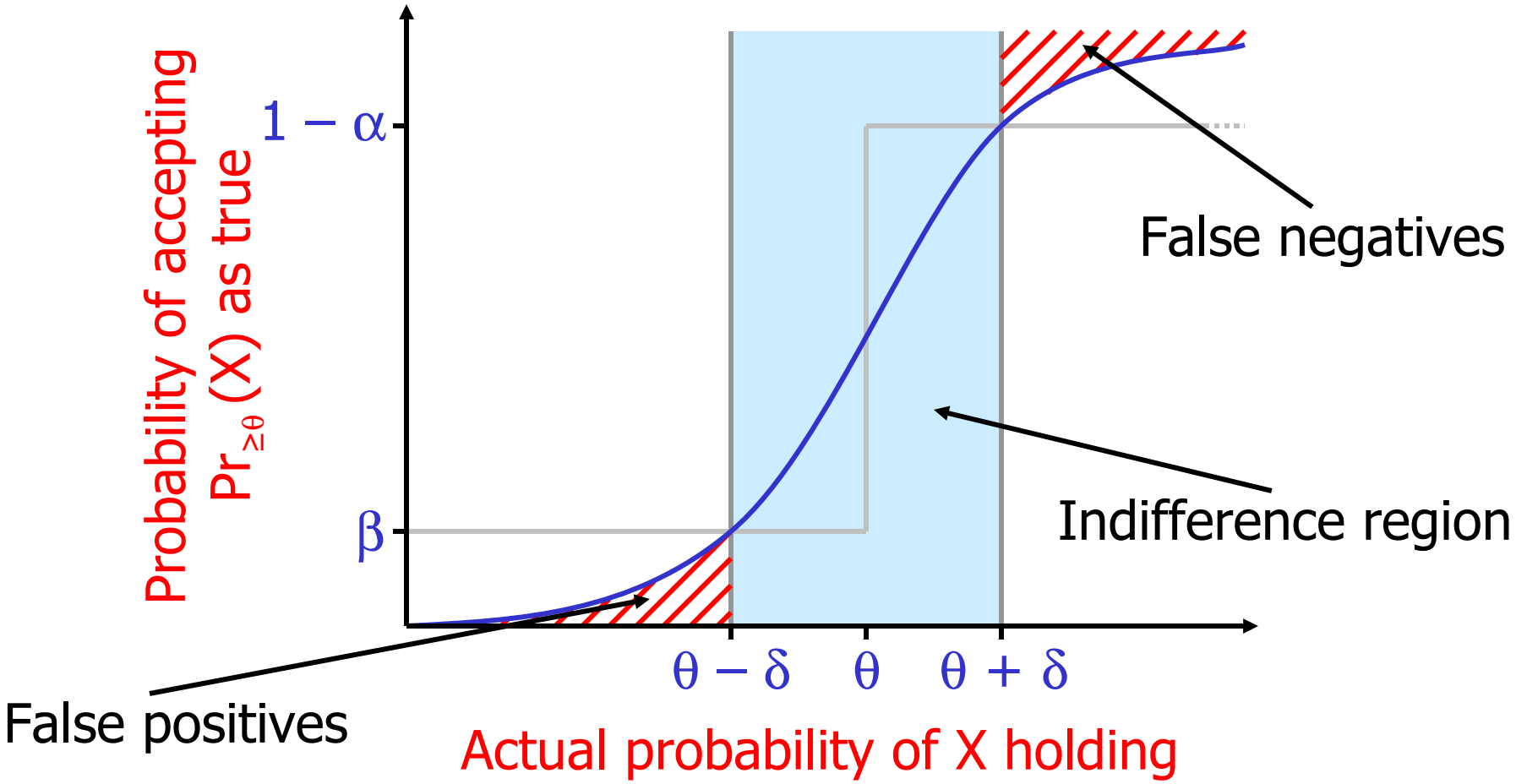
Performance of Test



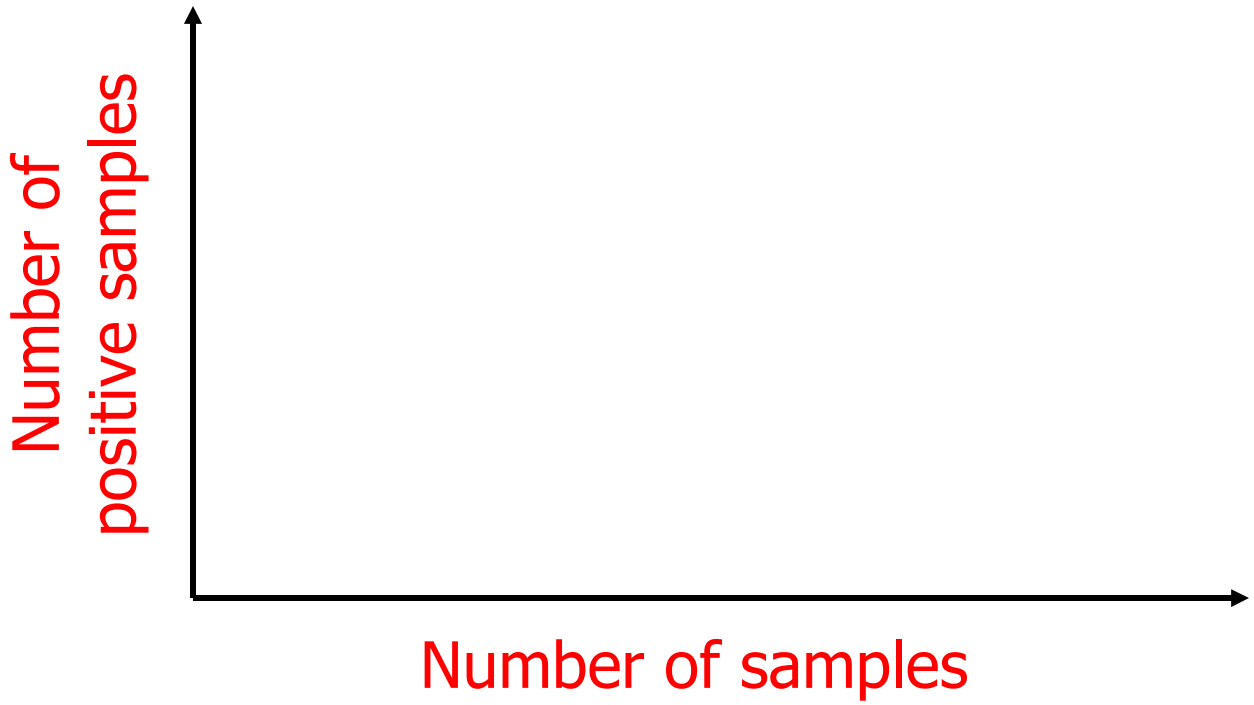
Ideal Performance



Actual Performance

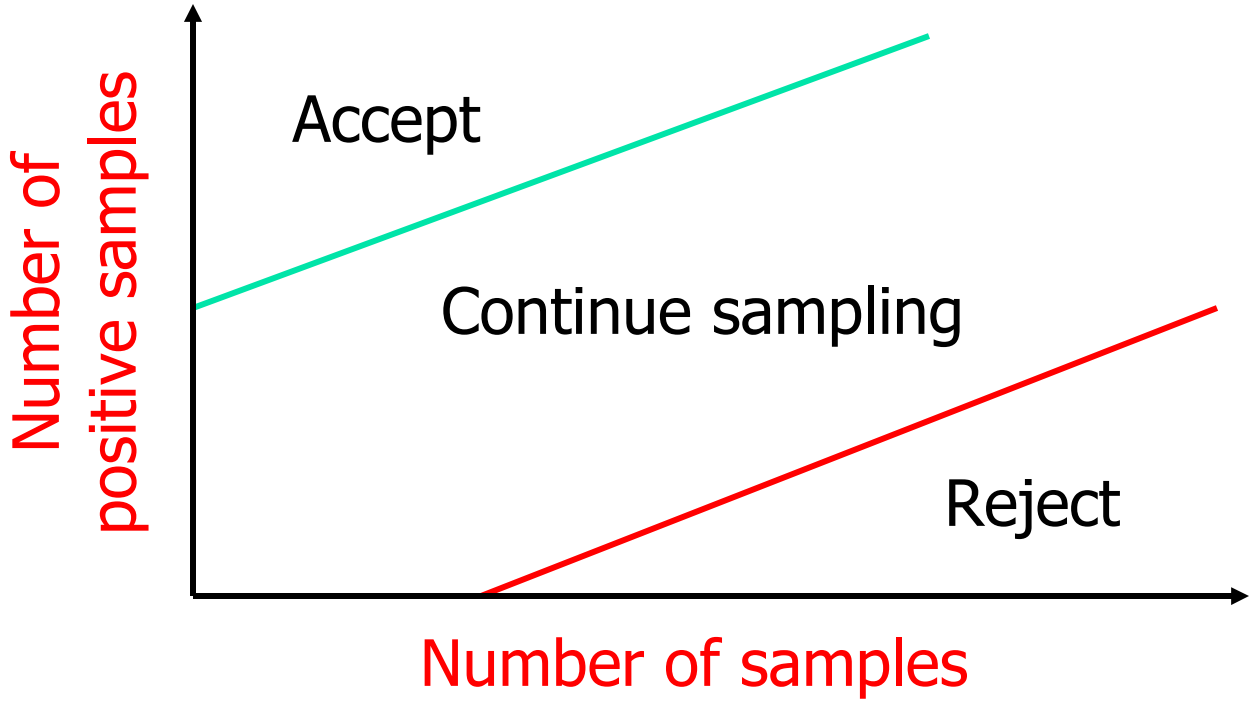


Graphical Representation of Sequential Test



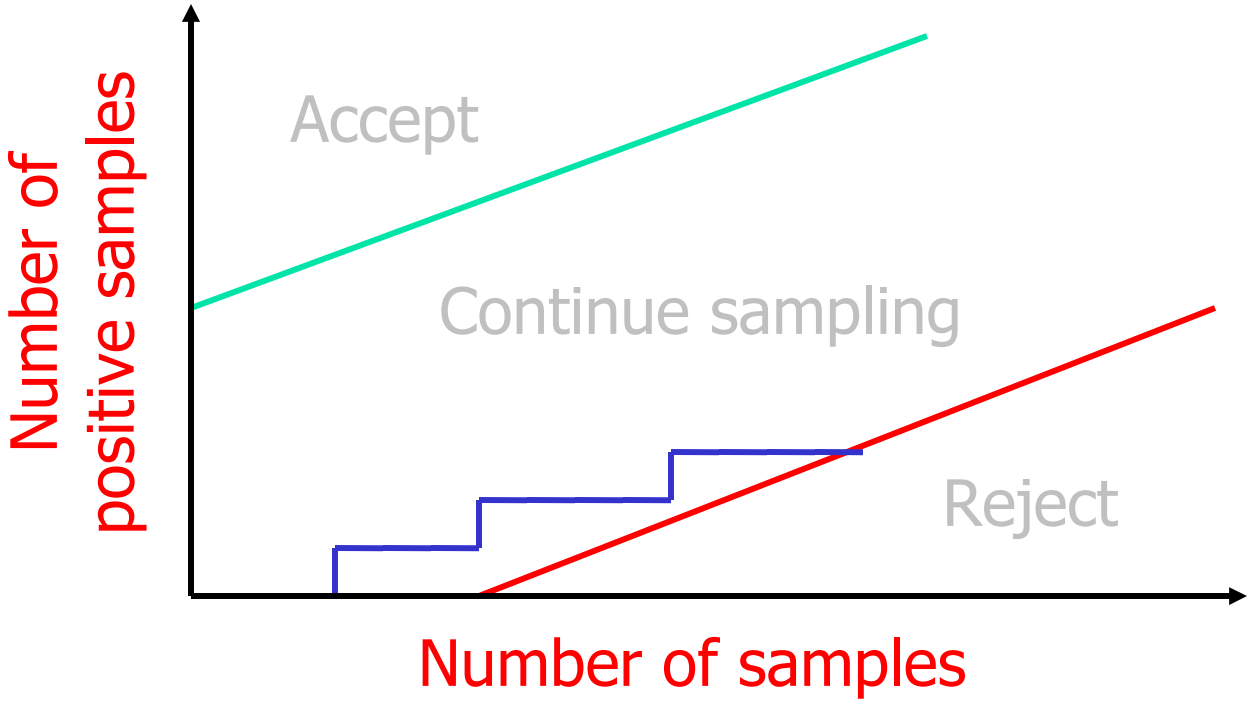
Graphical Representation of Sequential Test

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



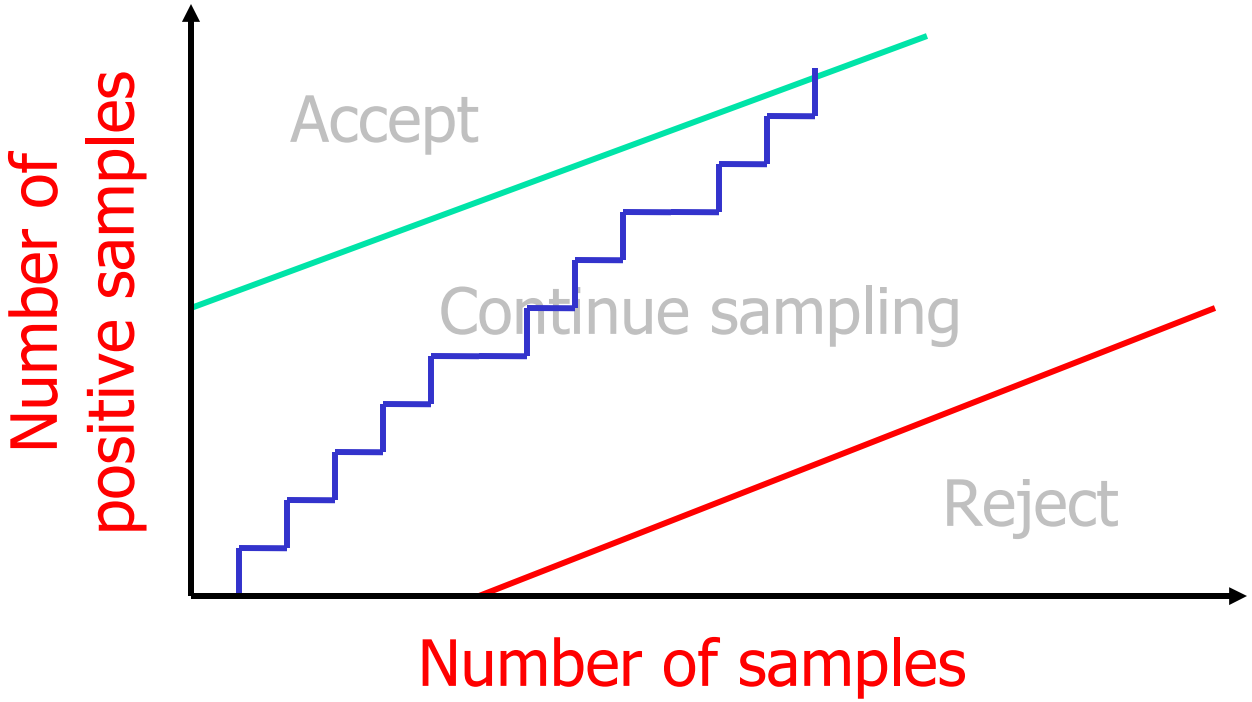
Graphical Representation of Sequential Test

- Reject hypothesis

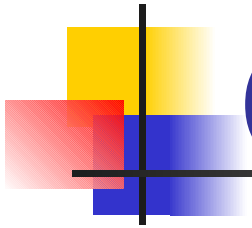


Graphical Representation of Sequential Test

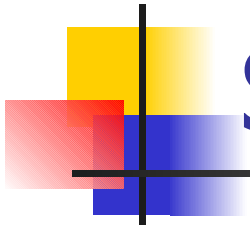
- Accept hypothesis



Continuous Stochastic Logic (CSL)

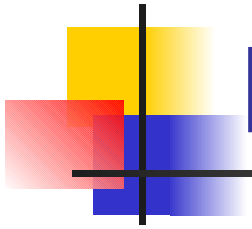


- State formulas
  - Truth value is determined in a single state
- Path formulas
  - Truth value is determined over an execution path



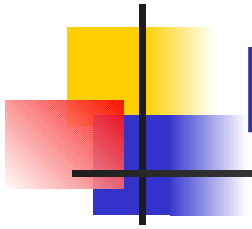
State Formulas

- 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
  - $Pr_{\geq 0.7}$(“The stork satisfies its hunger within 180 seconds”)



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
  - "The stork satisfies its hunger within 180 seconds": $\text{true} \; U^{\leq 180} \; \neg\text{hungry}$

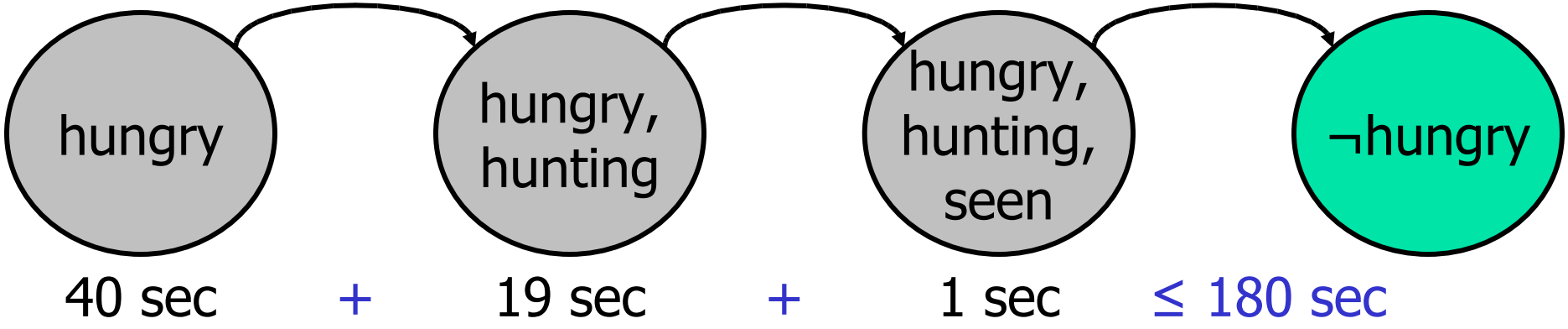


Expressing Properties in CSL

- "The probability is at least 0.7 that the stork satisfies its hunger within 180 seconds"
  - $Pr_{\geq 0.7}(\text{true } U^{\leq 180} \neg\text{hungry})$
- "The probability is at least 0.9 that the customer is served within 60 seconds and remains happy while waiting"
  - $Pr_{\geq 0.9}(\text{happy } U^{\leq 60} \text{ served})$

Semantics of Until

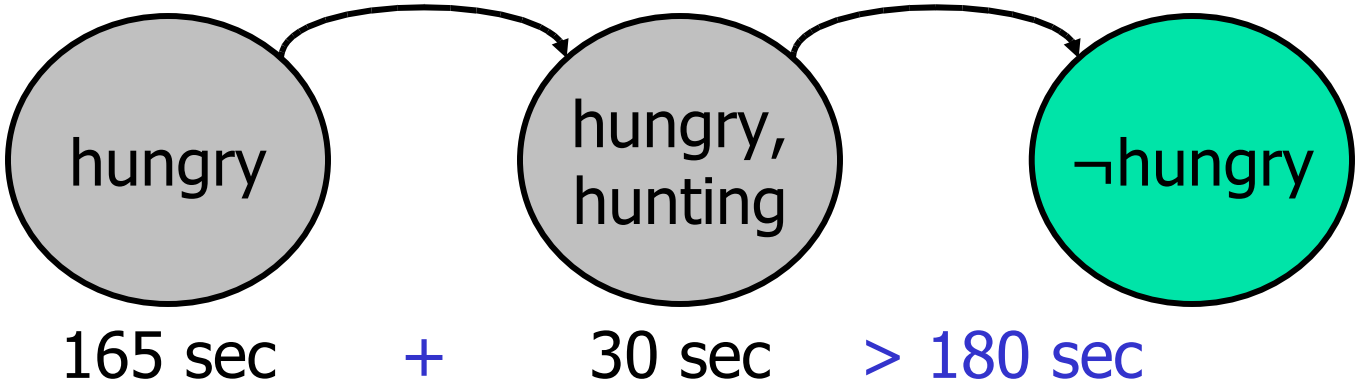
- true $U^{\leq 180}$ ¬hungry



True!

Semantics of Until

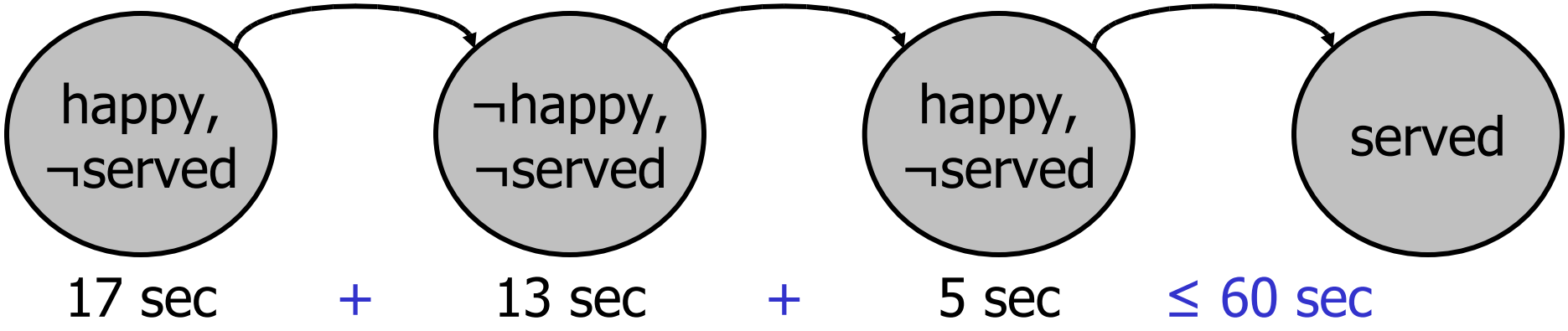
- true $U^{\leq 180}$ ¬hungry



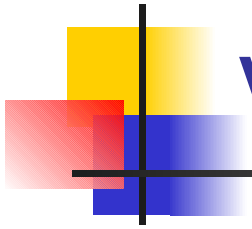
False!

Semantics of Until

- happy $U^{\leq 60}$ served



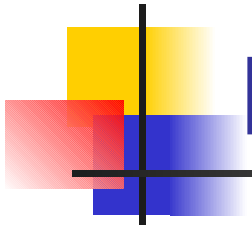
False!



Verifying Probabilistic Statements

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

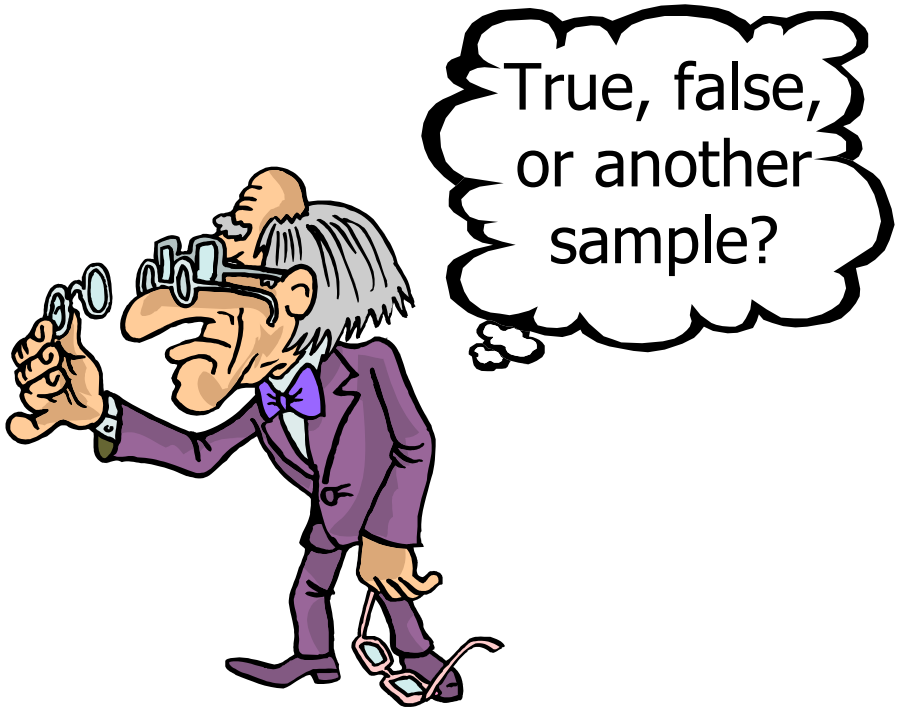
Verification of Nested Probabilistic Statements



- Suppose ρ, in $Pr_{\geq\theta}(\rho)$, contains probabilistic statements
  - $Pr_{\geq 0.8}(\text{true } U^{\leq 60} Pr_{\geq 0.9}(\text{true } U^{\leq 30} \neg\text{hungry}))$
  - Error bounds α' and β' when verifying ρ

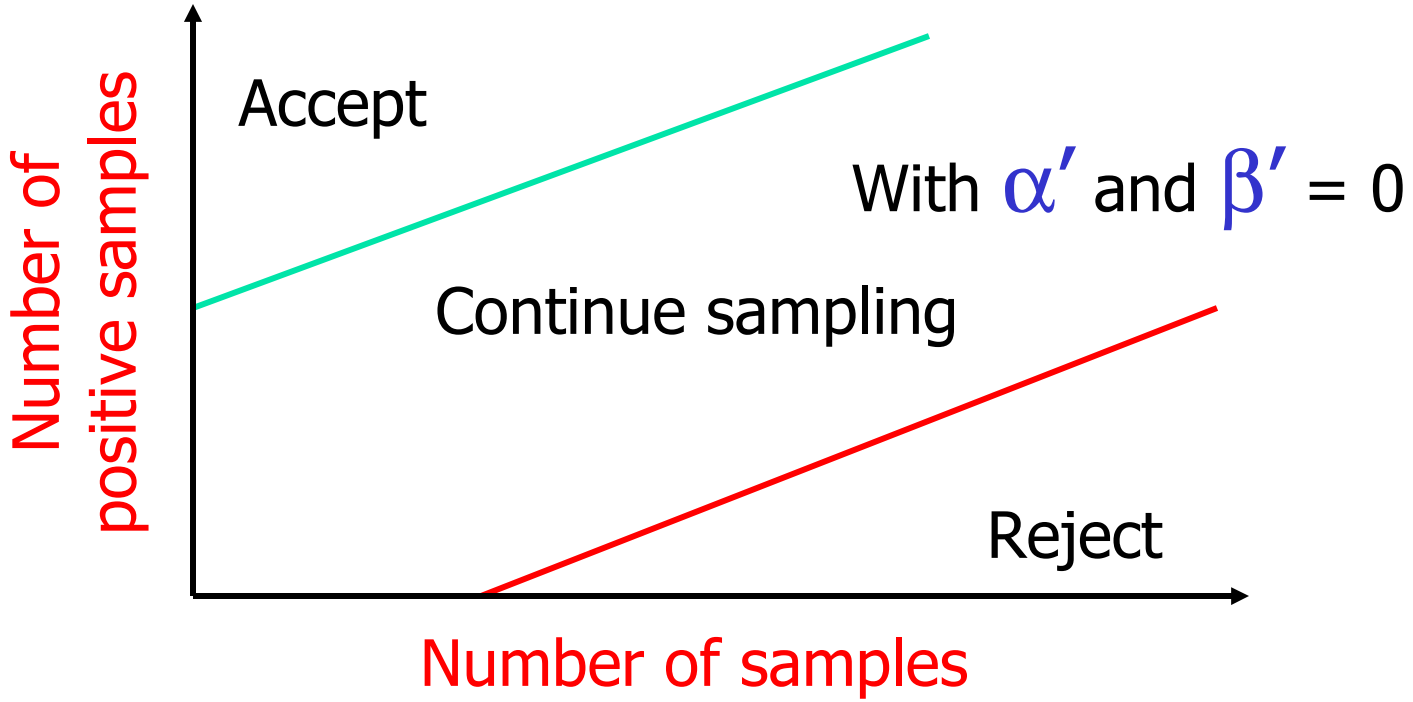
Verification of Nested Probabilistic Statements

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



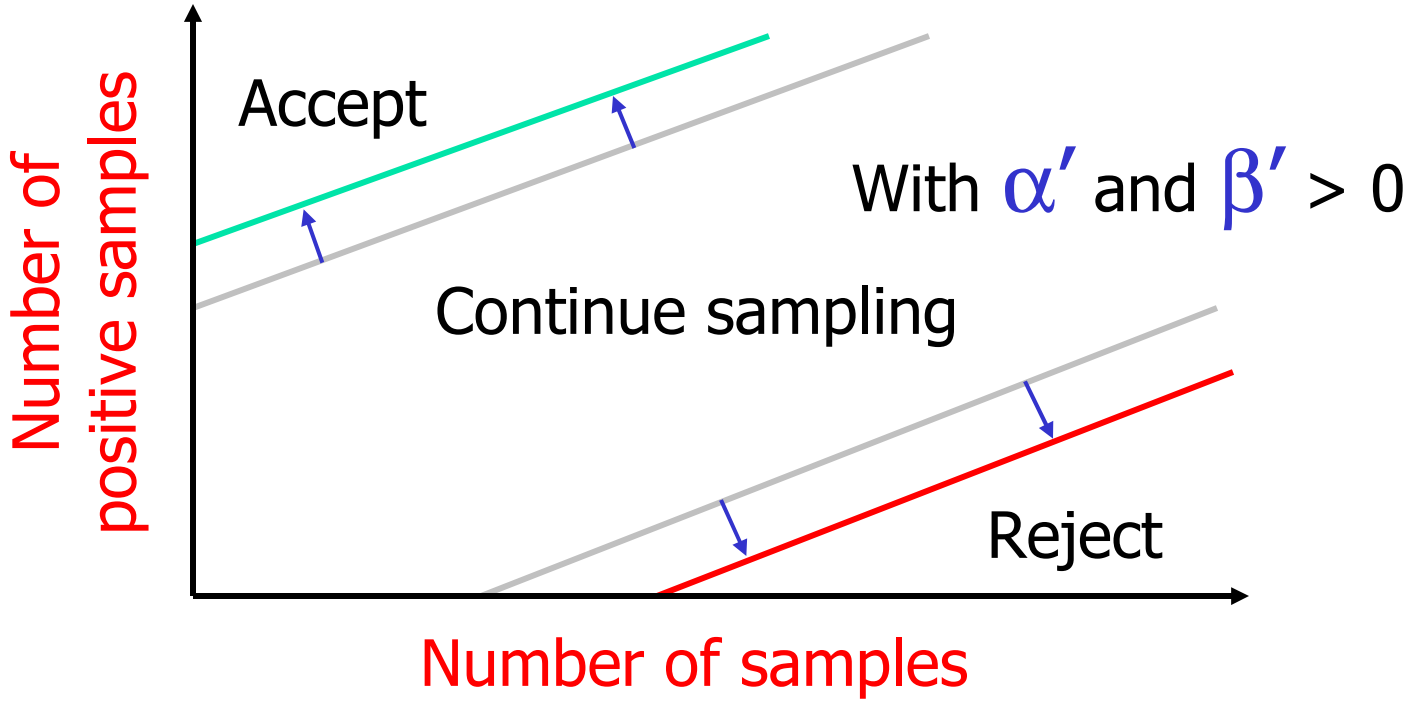
Modified Test

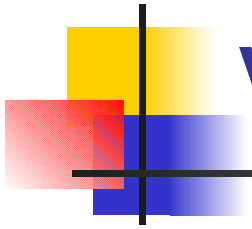
- Find an acceptance line and a rejection line given θ, δ, α, β, α', and β':



Modified Test

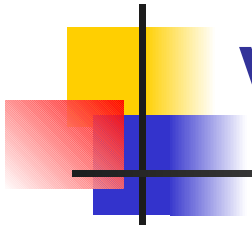
- Find an acceptance line and a rejection line given θ, δ, α, β, α', and β':





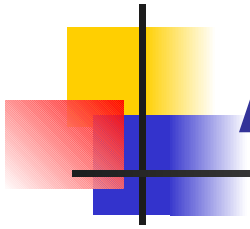
Verification of Negation

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



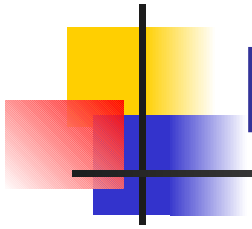
Verification of Conjunction

- Verify $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$ with error bounds α and β
  - Accept if all conjuncts are true
  - Reject if some conjunct is false



Acceptance of Conjunction

- Accept if all conjuncts are true
  - Accept all φ_i with bounds α_i and β_i
    - Probability at most β_i that φ_i is false
    - Therefore: Probability at most $\beta_1 + \ldots + \beta_n$ that conjunction is false
    - For example, choose $\beta_i = \beta/n$
    - Note: α_i unconstrained



Rejection of Conjunction

- Reject if some conjunct is false
  - Reject some φ_i with bounds α_i and β_i
    - Probability at most α_i that φ_i is true
    - Therefore: Probability at most α_i that conjunction is true
    - Choose $\alpha_i = \alpha$
    - Note: β_i unconstrained

Putting it Together

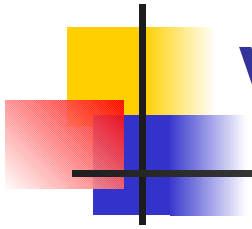
- To verify $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$ with error bounds α and β
  1. Verify each φ_i with error bounds α and β'
  2. Return **false** as soon as any φ_i is verified to be false
  3. If all φ_i are verified to be true, verify each φ_i again with error bounds α and β/n
  4. Return **true** iff all φ_i are verified to be true

"Fast reject"

Putting it Together

- To verify $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$ with error bounds α and β
  1. Verify each φ_i with error bounds α and β'
  2. Return **false** as soon as any φ_i is verified to be false
  3. If all φ_i are verified to be true, verify each φ_i again with error bounds α and β/n
  4. Return **true** iff all φ_i are verified to be true

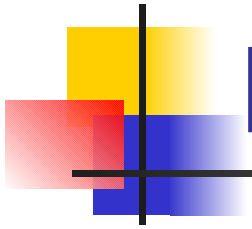
"Rigorous accept"



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 …

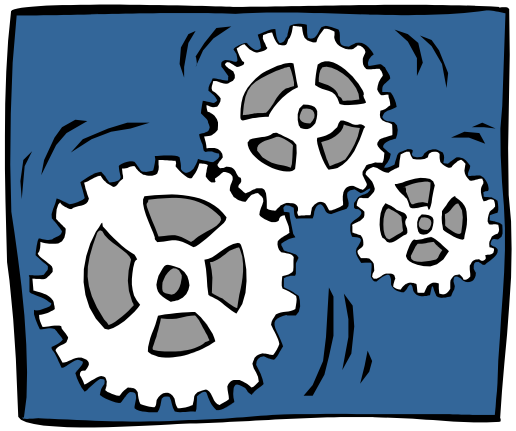
More on Verifying Until



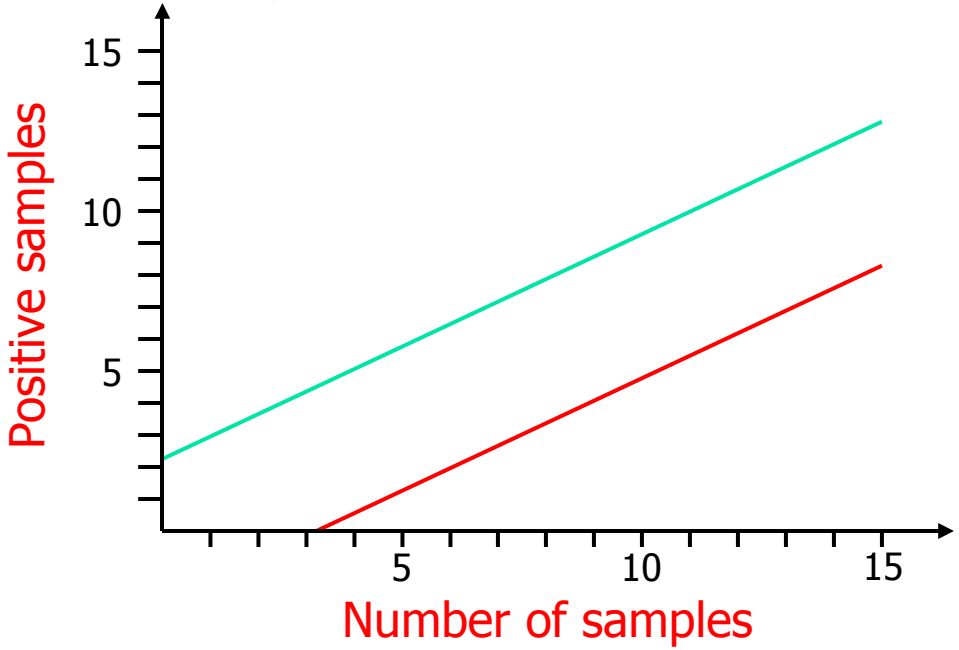
- Given $\varphi_1 \, U^{\leq t} \, \varphi_2$, let n be the index of the first state more than t time units away from the current state
- Disjunction of n conjunctions c_1 through c_n, each of size i
- Simplifies if φ_1 or φ_2, or both, do not contain any probabilistic statements

Example

- Verify $Pr_{\geq 0.7}(\text{true } U^{\leq 180} \neg\text{hungry})$ in with $\alpha = \beta = 0.1$ and $\delta = 0.1$

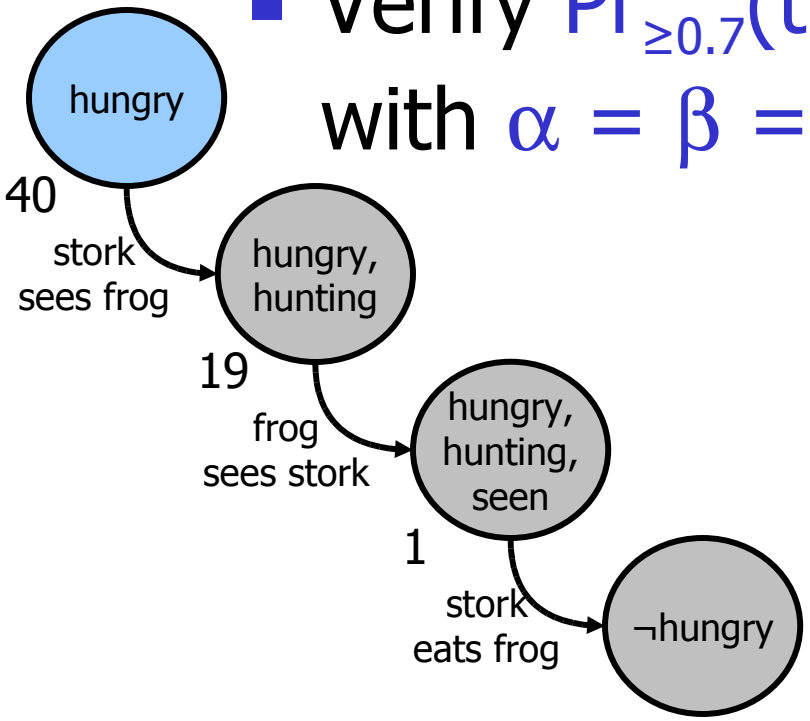


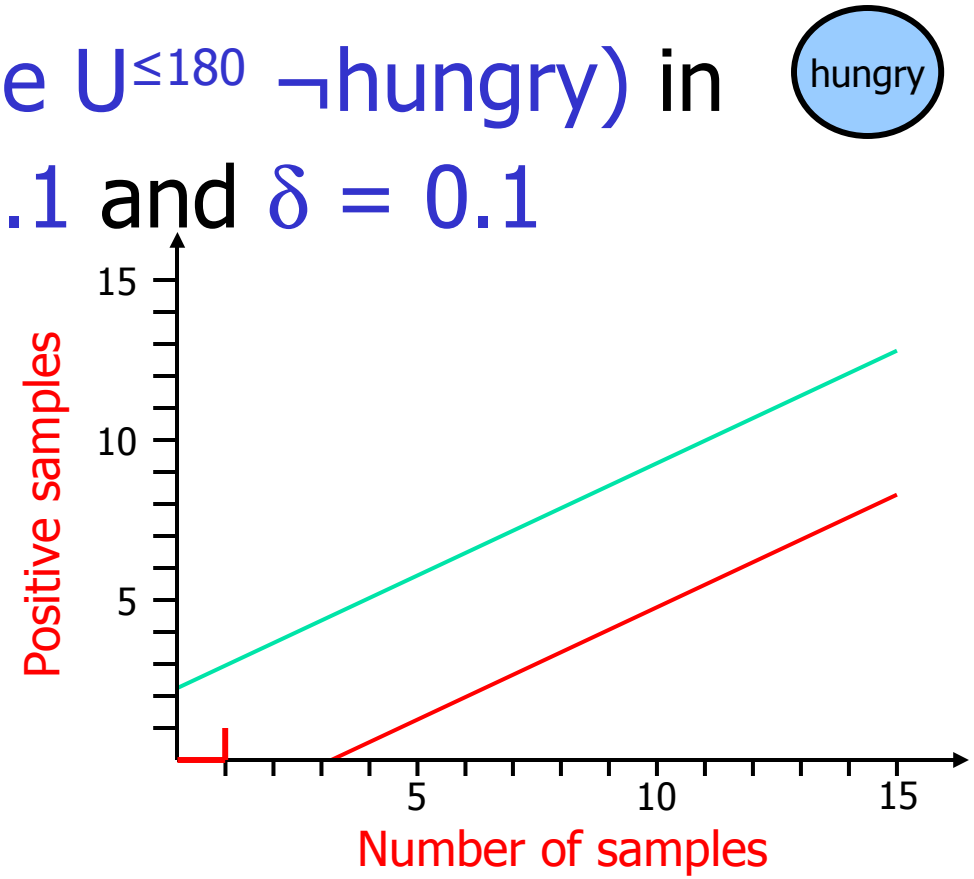
Simulator



Example

- Verify $Pr_{\geq 0.7}(\text{true } U^{\leq 180} \neg\text{hungry})$ in with $\alpha = \beta = 0.1$ and $\delta = 0.1$

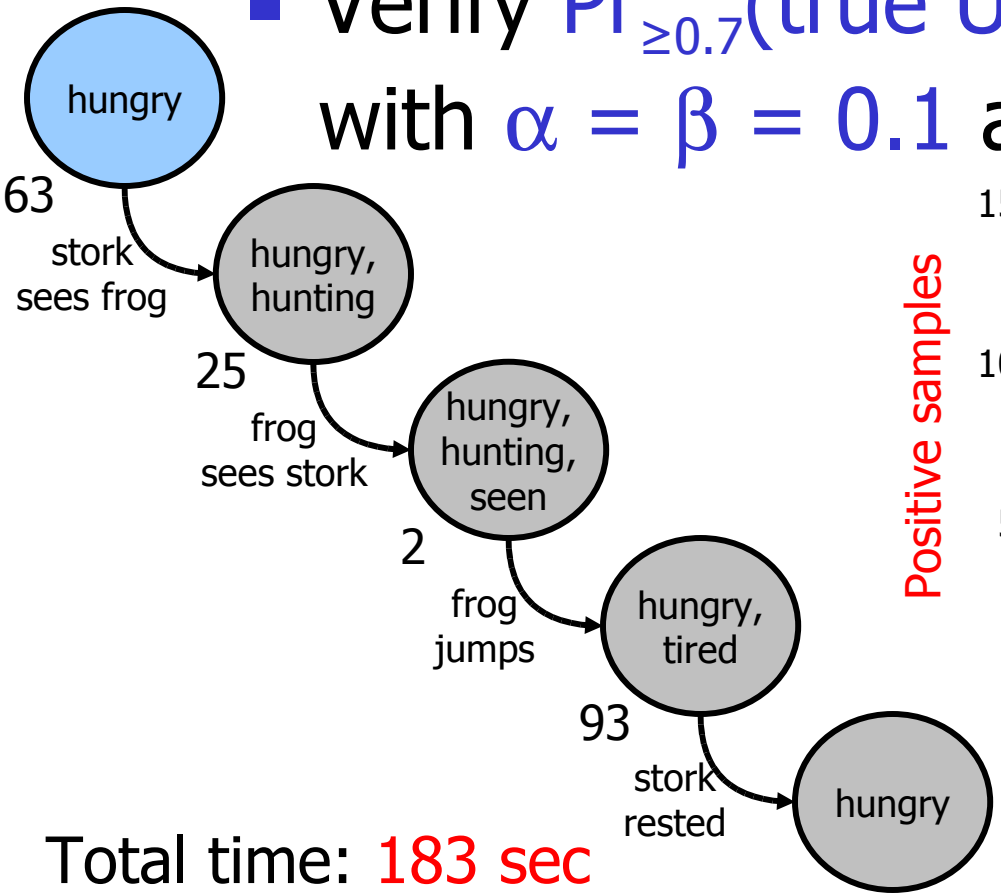


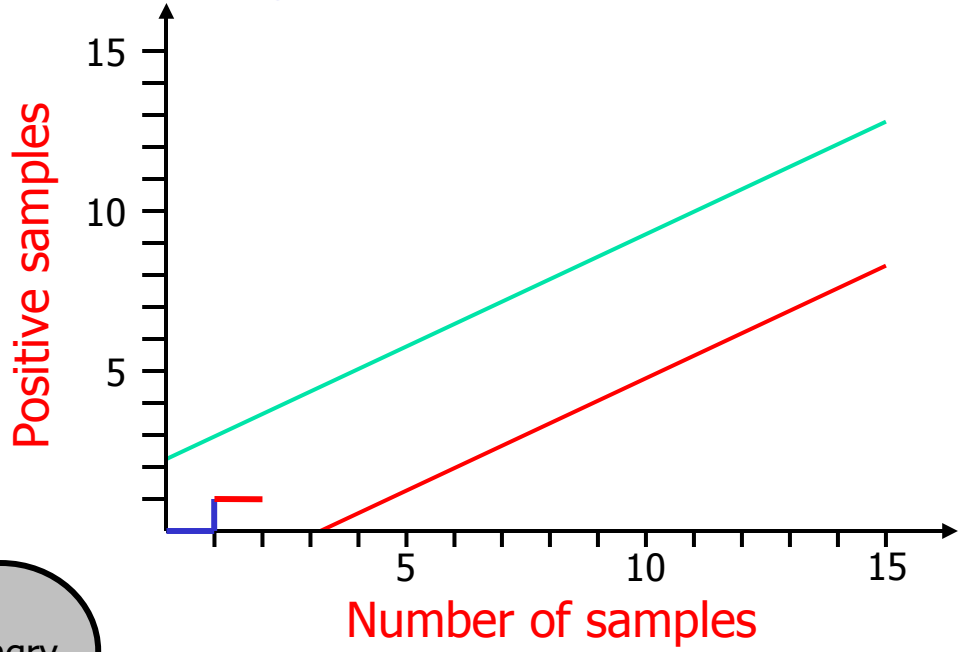


Total time: 60 sec

Example

- Verify $Pr_{\geq 0.7}(\text{true } U^{\leq 180} \neg\text{hungry})$ in with $\alpha = \beta = 0.1$ and $\delta = 0.1$

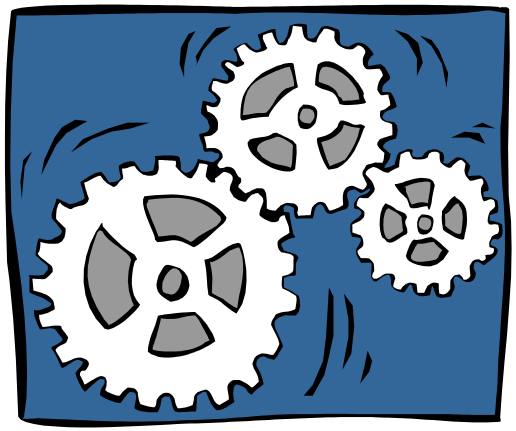


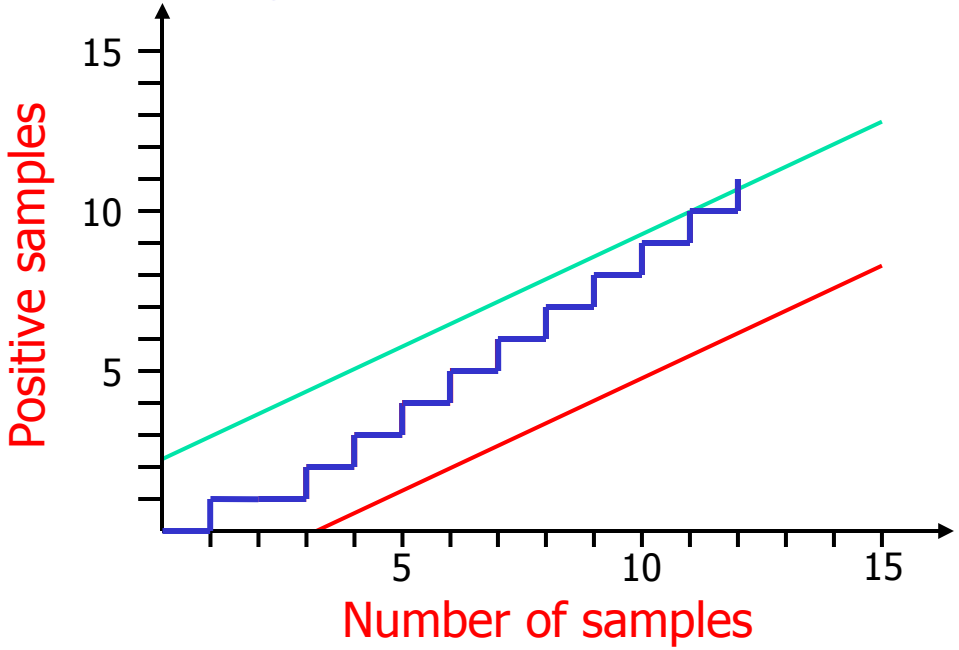


Total time: 183 sec

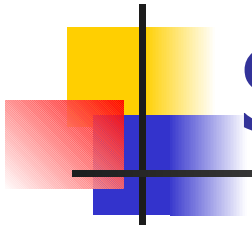
Example

- Verify $Pr_{\geq 0.7}(\text{true } U^{\leq 180} \neg\text{hungry})$ in with $\alpha = \beta = 0.1$ and $\delta = 0.1$



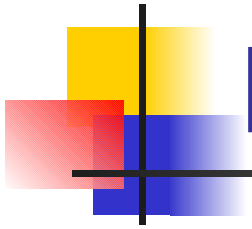


Property holds!



Summary

- Algorithm for probabilistic verification of discrete event systems
- Sample execution paths generated using simulation
- Probabilistic properties verified using sequential acceptance sampling



Future Work

- Apply to hybrid dynamic systems
- Develop heuristics for formula ordering and parameter selection
- Use verification to aid policy generation for real-time stochastic domains