Probabilistic Model Checking with Perfect Simulation

Diana EL RABIH   Nihal PEKERGIN

LACL, University of Paris Est

This work is supported by Checkbound, ANR-06-SETI-002

9th Workshop on Performance Evaluation, 2008

Outline

1. Introduction
   - Probabilistic Model Checking
   - Model Checking of CTMCs using CSL
   - CSL formulas
   - Numerical vs Statistical
2. Previous Work
   - Statistical Model Checking
   - Perfect Simulation
3. Motivations and Objective
4. Our Contribution
   - SMC Decision and Precision
   - SMC of CSL Steady State Operator
   - SMC of CSL Unbounded Until formula

Outline

Probabilistic Model Checking

- Automatic formal verification technique for the analysis of systems which exhibit stochastic behavior.
- Given a model M, a state s, and a property Φ, does Φ hold in s for M?
  - Model: Continuous-time Markov Chain
  - Property: Continuous Stochastic Logic (CSL) formula
- Solution methods:
  - Numerical: computation of distributions
  - Statistical:
    - Sampling (by simulation or by measurement)
    - Hypothesis Testing

Probabilistic Model Checking

- Automatic formal verification technique for the analysis of systems which exhibit stochastic behavior.
- Given a model M, a state s, and a property Φ, does Φ hold in s for M?
  - Model: Continuous-time Markov Chain
  - Property: Continuous Stochastic Logic (CSL) formula
- Solution methods:
  - Numerical: computation of distributions
  - Statistical:
    - Sampling (by simulation or by measurement)
    - Hypothesis Testing

Probabilistic Model Checking

- Automatic formal verification technique for the analysis of systems which exhibit stochastic behavior.
- Given a model M, a state s, and a property Φ, does Φ hold in s for M?
  - Model: Continuous-time Markov Chain
  - Property: Continuous Stochastic Logic (CSL) formula
- Solution methods:
  - Numerical: computation of distributions
  - Statistical:
    - Sampling (by simulation or by measurement)
    - Hypothesis Testing

Outline

Model Checking of CTMCs using CSL

- Model checking of stochastic systems
  - Continuous-time Markov chains CTMC
  - Continuous Stochastic Logic (CSL)
  - State formulas
    - Truth value is determined in a single state
  - Path formulas
    - Truth value is determined over a path

Model Checking of CTMCs using CSL

- Model checking of stochastic systems
  - Continuous-time Markov chains CTMC
  - Continuous Stochastic Logic (CSL)
  - State formulas
    - Truth value is determined in a single state
  - Path formulas
    - Truth value is determined over a path

Model Checking of CTMCs using CSL

- Model checking of stochastic systems
  - Continuous-time Markov chains CTMC
  - Continuous Stochastic Logic (CSL)
  - State formulas
    - Truth value is determined in a single state
  - Path formulas
    - Truth value is determined over a path

Outline

CSL formulas

- Standard logic operators: $\neg\phi \mid \phi_1 \wedge \phi_2 \ldots$
- Probabilistic operator: $\mathcal{P}_{\geq\theta}(\rho)$
  - Holds in state s iff probability is at least θ that ρ holds over paths starting in s
- Time bounded Until: $s \models \mathcal{P}_{\geq\theta}(\phi\,\mathcal{U}^T\,\psi)$
  - Holds over path σ *iff* ψ becomes true along σ within time T, and ϕ is true until then
  - If T=[0,∞) then $s \models \mathcal{P}_{\geq\theta}(\phi\,\mathcal{U}\,\psi)$ is unbounded until
- Steady State Operator: $\mathcal{S}_{\geq\theta}(\phi)$

CSL formulas

- Standard logic operators: $\neg\phi \mid \phi_1 \wedge \phi_2 \ldots$
- Probabilistic operator: $\mathcal{P}_{\geq\theta}(\rho)$
  - Holds in state s iff probability is at least θ that ρ holds over paths starting in s
- Time bounded Until: $s \models \mathcal{P}_{\geq\theta}(\phi\,\mathcal{U}^T\ \psi)$
  - Holds over path σ *iff* ψ becomes true along σ within time T, and ϕ is true until then
  - If T=[0,∞) then $s \models \mathcal{P}_{\geq\theta}(\phi\,\mathcal{U}\ \psi)$ is unbounded until
- Steady State Operator: $\mathcal{S}_{\geq\theta}(\phi)$

CSL formulas

- Standard logic operators: $\neg\phi \mid \phi_1 \wedge \phi_2 \ldots$
- Probabilistic operator: $\mathcal{P}_{\geq\theta}(\rho)$
  - Holds in state s iff probability is at least θ that ρ holds over paths starting in s
- Time bounded Until: $s \models \mathcal{P}_{\geq\theta}(\phi\,\mathcal{U}^T\ \psi)$
  - Holds over path σ *iff* ψ becomes true along σ within time T, and ϕ is true until then
  - If T=[0,∞) then $s \models \mathcal{P}_{\geq\theta}(\phi\,\mathcal{U}\ \psi)$ is unbounded until
- Steady State Operator: $\mathcal{S}_{\geq\theta}(\phi)$

CSL formulas

- Standard logic operators: $\neg\phi \mid \phi_1 \wedge \phi_2 \ldots$
- Probabilistic operator: $\mathcal{P}_{\geq\theta}(\rho)$
  - Holds in state s iff probability is at least θ that ρ holds over paths starting in s
- Time bounded Until: $s \models \mathcal{P}_{\geq\theta}(\phi\, \mathcal{U}^T\ \psi)$
  - Holds over path σ *iff* ψ becomes true along σ within time T, and ϕ is true until then
  - If T=[0,∞) then $s \models \mathcal{P}_{\geq\theta}(\phi\, \mathcal{U}\ \psi)$ is unbounded until
- Steady State Operator: $\mathcal{S}_{\geq\theta}(\phi)$

Outline

Numerical vs Statistical Model Checking

- Numerical Method
  - Highly accurate results
  - Expensive for systems with many states
- Statistical Method
  - Low memory requirements (state explosion problem)
  - Expensive if high accuracy is required

Numerical vs Statistical Model Checking

- Numerical Method
  - Highly accurate results
  - Expensive for systems with many states
- Statistical Method
  - Low memory requirements (state explosion problem)
  - Expensive if high accuracy is required

Outline

Concept of SMC

- Statistical approach is based on
  - Generating sample paths by simulation or by measurement
  - Hypothesis Testing
- We cannot guarantee that the verification result is correct
  - But we can at least bound the probability of generating an incorrect answer to a verification problem
- A key observation of SMC interest is that
  - It is not necessary to obtain an accurate estimate of a probability in order to verify probabilistic properties

Concept of SMC

- Statistical approach is based on
  - Generating sample paths by simulation or by measurement
  - Hypothesis Testing
- We cannot guarantee that the verification result is correct
  - But we can at least bound the probability of generating an incorrect answer to a verification problem
- A key observation of SMC interest is that
  - It is not necessary to obtain an accurate estimate of a probability in order to verify probabilistic properties

Concept of SMC

- Statistical approach is based on
  - Generating sample paths by simulation or by measurement
  - Hypothesis Testing
- We cannot guarantee that the verification result is correct
  - But we can at least bound the probability of generating an incorrect answer to a verification problem
- A key observation of SMC interest is that
  - It is not necessary to obtain an accurate estimate of a probability in order to verify probabilistic properties

Hypothesis Testing in SMC

- In SMC approach a model checking problem can be seen as hypothesis testing problem to verify probabilistic properties
- To verify a given property
  - Test the hypothesis $H : p < \theta$ against the alternative hypothesis $K : p \geq \theta$
- SMC approach permits to estimate the probability that a given formula is satisfied on sample paths
  - for specified confidence interval, confidence level and error bounds

Hypothesis Testing in SMC

- In SMC approach a model checking problem can be seen as hypothesis testing problem to verify probabilistic properties
- To verify a given property
  - Test the hypothesis H : $p < \theta$ against the alternative hypothesis K : $p \geq \theta$
- SMC approach permits to estimate the probability that a given formula is satisfied on sample paths
  - for specified confidence interval, confidence level and error bounds

Hypothesis Testing in SMC

- In SMC approach a model checking problem can be seen as hypothesis testing problem to verify probabilistic properties
- To verify a given property
  - Test the hypothesis $H : p < \theta$ against the alternative hypothesis $K : p \geq \theta$
- SMC approach permits to estimate the probability that a given formula is satisfied on sample paths
  - for specified confidence interval, confidence level and error bounds

Existing SMC Approaches

- Randomised approximation scheme proposed by Peyronnet and al.
- Statistical hypothesis testing of Younes and al. was studied CSL time bounded formulas
  - Based on discrete event simulation and on acceptance sampling
  - Extended to the case of black box systems
- Statistical Model Checking of Sen and al. was studied in addition unbounded until CSL formula
  - Based on discrete event simulation and on hypothesis testing
  - Extended to the case of black box systems

Existing SMC Approaches

- Randomised approximation scheme proposed by Peyronnet and al.
- Statistical hypothesis testing of Younes and al. was studied CSL time bounded formulas
  - Based on discrete event simulation and on acceptance sampling
  - Extended to the case of black box systems
- Statistical Model Checking of Sen and al. was studied in addition unbounded until CSL formula
  - Based on discrete event simulation and on hypothesis testing
  - Extended to the case of black box systems

Existing SMC Approaches

- Randomised approximation scheme proposed by Peyronnet and al.
- Statistical hypothesis testing of Younes and al. was studied CSL time bounded formulas
  - Based on discrete event simulation and on acceptance sampling
  - Extended to the case of black box systems
- Statistical Model Checking of Sen and al. was studied in addition unbounded until CSL formula
  - Based on discrete event simulation and on hypothesis testing
  - Extended to the case of black box systems

Outline

Perfect Simulation Global Idea

- Perfect Simulation based on coupling from the past
  - Monte Carlo method
    - directly generates a sample according to the stationary distribution of Markov Chains
  - Avoids burn-in time period
- Perfect simulation is efficient when the model is monotone
  - Trajectories initiating from set of maximal and minimal states
- When all sample-paths couple, a sample state is obtained
  - by running simulation from distant point in the past until the present
  - in order to obtain a perfect sample at coupling

Perfect Simulation Global Idea

- Perfect Simulation based on coupling from the past
  - Monte Carlo method
    - directly generates a sample according to the stationary distribution of Markov Chains
  - Avoids burn-in time period
- Perfect simulation is efficient when the model is monotone
  - Trajectories initiating from set of maximal and minimal states
- When all sample-paths couple, a sample state is obtained
  - by running simulation from distant point in the past until the present
  - in order to obtain a perfect sample at coupling

Perfect Simulation Global Idea

- Perfect Simulation based on coupling from the past
  - Monte Carlo method
    - directly generates a sample according to the stationary distribution of Markov Chains
  - Avoids burn-in time period
- Perfect simulation is efficient when the model is monotone
  - Trajectories initiating from set of maximal and minimal states
- When all sample-paths couple, a sample state is obtained
  - by running simulation from distant point in the past until the present
  - in order to obtain a perfect sample at coupling

Perfect Sampler ψ^2

- ψ^2 proposed in MESCAL Project is a sampler designed for the steady state evaluation of various monotone queueing networks
  - Following a sampler ψ of Markov chains for the perfect sampling of Markov chains without monotonicity properties
- ψ^2 permits to simulate stationary distribution or directly a cost function of large Markov chains
  - By keeping only trajectories issued from the minimal and maximal states

Perfect Sampler ψ^2

- ψ^2 proposed in MESCAL Project is a sampler designed for the steady state evaluation of various monotone queueing networks
  - Following a sampler ψ of Markov chains for the perfect sampling of Markov chains without monotonicity properties
- ψ^2 permits to simulate stationary distribution or directly a cost function of large Markov chains
  - By keeping only trajectories issued from the minimal and maximal states

Our Motivations

- Numerical methods suffer from state space explosion problem
- Statistical methods have low memory requirements and then do not suffer from the state space explosion problem
- CSL Steady State operator was not studied before in other SMC approaches
- CSL Unbounded Until was studied by Sen and al in their SMC method
  - But suffering from stopping probability problem because they cannot detect the steady state in their approach
- While Perfect Simulation permits to detect the steady state
  - Then permits to avoid stopping probability problem

Our Motivations

- Numerical methods suffer from state space explosion problem
- Statistical methods have low memory requirements and then do not suffer from the state space explosion problem
- CSL Steady State operator was not studied before in other SMC approaches
- CSL Unbounded Until was studied by Sen and al in their SMC method
  - But suffering from stopping probability problem because they cannot detect the steady state in their approach
- While Perfect Simulation permits to detect the steady state
  - Then permits to avoid stopping probability problem

Our Motivations

- Numerical methods suffer from state space explosion problem
- Statistical methods have low memory requirements and then do not suffer from the state space explosion problem
- CSL Steady State operator was not studied before in other SMC approaches
- CSL Unbounded Until was studied by Sen and al in their SMC method
  - But suffering from stopping probability problem because they cannot detect the steady state in their approach
- While Perfect Simulation permits to detect the steady state
  - Then permits to avoid stopping probability problem

Our Motivations

- Numerical methods suffer from state space explosion problem
- Statistical methods have low memory requirements and then do not suffer from the state space explosion problem
- CSL Steady State operator was not studied before in other SMC approaches
- CSL Unbounded Until was studied by Sen and al in their SMC method
  - But suffering from stopping probability problem because they cannot detect the steady state in their approach
- While Perfect Simulation permits to detect the steady state
  - Then permits to avoid stopping probability problem

Our Motivations

- Numerical methods suffer from state space explosion problem
- Statistical methods have low memory requirements and then do not suffer from the state space explosion problem
- CSL Steady State operator was not studied before in other SMC approaches
- CSL Unbounded Until was studied by Sen and al in their SMC method
  - But suffering from stopping probability problem because they cannot detect the steady state in their approach
- While Perfect Simulation permits to detect the steady state
  - Then permits to avoid stopping probability problem

Our Motivations

- Numerical methods suffer from state space explosion problem
- Statistical methods have low memory requirements and then do not suffer from the state space explosion problem
- CSL Steady State operator was not studied before in other SMC approaches
- CSL Unbounded Until was studied by Sen and al in their SMC method
  - But suffering from stopping probability problem because they cannot detect the steady state in their approach
- While Perfect Simulation permits to detect the steady state
  - Then permits to avoid stopping probability problem

Our Objective

- Probabilistic model checking of stochastic systems modelled by Markov Chains
  - Using statistical approach
  - By applying Perfect Simulation which is a Monte Carlo method
  - To verify CSL Steady State operator and CSL Unbounded Until formula
- This method is efficient
  - when underlying model is monotone
  - when CSL state formula is increasing (functional)
  - These hypothesis are not so restrictive and satisfied in general for performance and reliability models

Our Objective

- Probabilistic model checking of stochastic systems modelled by Markov Chains
  - Using statistical approach
  - By applying Perfect Simulation which is a Monte Carlo method
  - To verify CSL Steady State operator and CSL Unbounded Until formula
- This method is efficient
  - when underlying model is monotone
  - when CSL state formula is increasing (functional)
  - These hypothesis are not so restrictive and satisfied in general for performance and reliability models

Our Objective

- Probabilistic model checking of stochastic systems modelled by Markov Chains
  - Using statistical approach
  - By applying Perfect Simulation which is a Monte Carlo method
  - To verify CSL Steady State operator and CSL Unbounded Until formula
- This method is efficient
  - when underlying model is monotone
  - when CSL state formula is increasing (functional)
  - These hypothesis are not so restrictive and satisfied in general for performance and reliability models

Our Objective

- Probabilistic model checking of stochastic systems modelled by Markov Chains
  - Using statistical approach
  - By applying Perfect Simulation which is a Monte Carlo method
  - To verify CSL Steady State operator and CSL Unbounded Until formula
- This method is efficient
  - when underlying model is monotone
  - when CSL state formula is increasing (functional)
  - These hypothesis are not so restrictive and satisfied in general for performance and reliability models

Our Objective

- Probabilistic model checking of stochastic systems modelled by Markov Chains
  - Using statistical approach
  - By applying Perfect Simulation which is a Monte Carlo method
  - To verify CSL Steady State operator and CSL Unbounded Until formula
- This method is efficient
  - when underlying model is monotone
  - when CSL state formula is increasing (functional)
  - These hypothesis are not so restrictive and satisfied in general for performance and reliability models

Outline

On SMC Approach Precision

- A sample of size n obtained by perfect sampler consists of n observations: $X_1, X_2, ..., X_n$ (Bernoulli variables)
  - Pr[X_i=1]=Pr[pos sample]=p'
  - Pr[X_i=0]=Pr[neg sample]=1-p'
- Hypothesis Testing in SMC approach
  - Testing $H_0 : p' < \theta - \delta \ (s \not\models \phi)$ *against* $H_1 : p' \geq \theta + \delta \ (s \models \phi)$
  - If Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$ then H_1 is accepted and H_0 is rejected , otherwise H_0 is rejected and H_1 is accepted
- Y has binomial distribution then
  - Pr[Y$\leq$ m]= F (m, n, p')= $\sum_{i=1}^{m} C(n,i)(p')^i(1-p')^{n-i}$
  - Where m=n.θ : acceptance threshold
- Then resulting test has the strength depending on error bounds α(significance level) and β

On SMC Approach Precision

- A sample of size n obtained by perfect sampler consists of n observations: $X_1, X_2, ..., X_n$ (Bernoulli variables)
  - Pr[X_i=1]=Pr[pos sample]=p'
  - Pr[X_i=0]=Pr[neg sample]=1-p'
- Hypothesis Testing in SMC approach
  - Testing $H_0 : p' < \theta - \delta$ ($s \not\models \phi$) *against* $H_1 : p' \geq \theta + \delta$ ($s \models \phi$)
  - If Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$ then H_1 is accepted and H_0 is rejected , otherwise H_0 is rejected and H_1 is accepted
- Y has binomial distribution then
  - Pr[Y$\leq$ m]= F (m, n, p')= $\sum_{i=1}^{m} C(n,i)(p')^i(1-p')^{n-i}$
  - Where m=n.θ : acceptance threshold
- Then resulting test has the strength depending on error bounds α(significance level) and β

On SMC Approach Precision

- A sample of size n obtained by perfect sampler consists of n observations: $X_1, X_2, ..., X_n$ (Bernoulli variables)
  - Pr[X_i=1]=Pr[pos sample]=p'
  - Pr[X_i=0]=Pr[neg sample]=1-p'
- Hypothesis Testing in SMC approach
  - Testing $H_0 : p' < \theta - \delta$ ($s \not\models \phi$) *against* $H_1 : p' \geq \theta + \delta$ ($s \models \phi$)
  - If Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$ then H_1 is accepted and H_0 is rejected , otherwise H_0 is rejected and H_1 is accepted
- Y has binomial distribution then
  - Pr[Y$\leq$ m]= F (m, n, p')= $\sum_{i=1}^{m} C(n, i)(p')^i(1 - p')^{n-i}$
  - Where m=n.θ : acceptance threshold
- Then resulting test has the strength depending on error bounds α(significance level) and β

On Perfect Simulation Precision

- We look for bounds on true mean μ, with finite number of samples
  - By finding b_1 and b_2 such that $\Pr(b_1 < \mu < b_2)$ = 1-γ
    - $[b_1, b_2]$: confidence interval
    - 100(1-γ): confidence level
- The confidence interval of a simulation output is given by $M \pm t.s/\sqrt{n}$
  - M : sample mean, s : estimation of the standard deviation and t : constant determined from t distribution table
- In perfect simulation, because of the independence of generated values:
  - The length of the confidence interval at 95% level is majorized by 1.68 s/$\sqrt{n}$ where n is the sample size

On Perfect Simulation Precision

- We look for bounds on true mean μ, with finite number of samples
  - By finding b_1 and b_2 such that $\Pr(b_1 < \mu < b_2) = 1\text{-}\gamma$
    - $[b_1, b_2]$: confidence interval
    - $100(1\text{-}\gamma)$: confidence level
- The confidence interval of a simulation output is given by $M \pm t.s/\sqrt{n}$
  - M : sample mean, s : estimation of the standard deviation and t : constant determined from t distribution table
- In perfect simulation, because of the independence of generated values:
  - The length of the confidence interval at 95% level is majorized by 1.68 s/$\sqrt{n}$ where n is the sample size

On Perfect Simulation Precision

- We look for bounds on true mean μ, with finite number of samples
  - By finding b_1 and b_2 such that $\Pr(b_1 < \mu < b_2) = 1\text{-}\gamma$
    - $[b_1, b_2]$: confidence interval
    - $100(1\text{-}\gamma)$: confidence level
- The confidence interval of a simulation output is given by $M \pm t.s/\sqrt{n}$
  - M : sample mean, s : estimation of the standard deviation and t : constant determined from t distribution table
- In perfect simulation, because of the independence of generated values:
  - The length of the confidence interval at 95% level is majorized by 1.68 $s/\sqrt{n}$ where n is the sample size

Proposed SMC Decision Algorithm

Input: Property ψ, Model M, threshold θ, total nb of samples nbsamptotal, indifference region δ
Output: YES or NO or Don't Know

1. Initialize nbsampos to zero
2. Test of the positive samples from 1 to total number of samples and then calculate number of positive samples
3. Let Y=nbsampos/nbsamptotal,$H_0 : p' < \theta - \delta$ and $H_1 : p' \geq \theta + \delta$ where p'=prob[Xi=1]
4. If Y $\geq \theta$ then deciding YES and making decision by accepting H_1 with c% confidence level, otherwise deciding NO and making decision by accepting H_0 with (1-c)% confidence level

Proposed SMC Decision Algorithm

Input: Property ψ, Model M, threshold θ, total nb of samples nbsamptotal, indifference region δ
Output: YES or NO or Don't Know

1. Initialize nbsampos to zero
2. Test of the positive samples from 1 to total number of samples and then calculate number of positive samples
3. Let Y=nbsampos/nbsamptotal,$H_0 : p' < \theta - \delta$ and $H_1 : p' \geq \theta + \delta$ where p'=prob[Xi=1]
4. If Y $\geq \theta$ then deciding YES and making decision by accepting H_1 with c% confidence level, otherwise deciding NO and making decision by accepting H_0 with (1-c)% confidence level

Proposed SMC Decision Algorithm

Input: Property ψ, Model M, threshold θ, total nb of samples nbsamptotal, indifference region δ
Output: YES or NO or Don't Know

1. Initialize nbsampos to zero
2. Test of the positive samples from 1 to total number of samples and then calculate number of positive samples
3. Let Y=nbsampos/nbsamptotal,$H_0 : p' < \theta - \delta$ and $H_1 : p' \geq \theta + \delta$ where p'=prob[Xi=1]
4. If Y $\geq \theta$ then deciding YES and making decision by accepting H_1 with c% confidence level, otherwise deciding NO and making decision by accepting H_0 with (1-c)% confidence level

Proposed SMC Decision Algorithm

Input: Property ψ, Model M, threshold θ, total nb of samples nbsamptotal, indifference region δ
Output: YES or NO or Don't Know

1. Initialize nbsampos to zero
2. Test of the positive samples from 1 to total number of samples and then calculate number of positive samples
3. Let Y=nbsampos/nbsamptotal,$H_0 : p' < \theta - \delta$ and $H_1 : p' \geq \theta + \delta$ where p'=prob[Xi=1]
4. If Y $\geq \theta$ then deciding YES and making decision by accepting H_1 with c% confidence level, otherwise deciding NO and making decision by accepting H_0 with (1-c)% confidence level

Precision of SMC Decision Algorithm

- How to determine confidence level c?
  - 2 contraints are required in the hypothesis testing context:
    - $\Pr[H_1 \text{ is accepted} \mid H_0 \text{ is true}] \leq \alpha$
    - $\Pr[H_0 \text{ is accepted} \mid H_1 \text{ is true}] \leq \beta$
  - Practically the true mean μ estimated by Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$, is included in the confidence interval $[Y-\delta, Y+\delta]$ where Y is the sample mean of the perfect sampling and δ=1.68 s/ $\sqrt{n}$
  - α is determined from respecting constraint F (m, n, p_0) = α
    - where F (m, n, p)= $\sum_{i=1}^{m} C(n,i)p^i(1-p)^{n-i}$, $p_0 = \theta - \delta, p_1 = \theta + \delta$ and m=n.θ
- Thus c can be determined by applying c=100(1-α)%

Precision of SMC Decision Algorithm

- How to determine confidence level c?
  - 2 contraints are required in the hypothesis testing context:
    - Pr[H_1 is accepted | H_0 is true] $\leq \alpha$
    - Pr[H_0 is accepted | H_1 is true] $\leq \beta$
  - Practically the true mean μ estimated by Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$, is included in the confidence interval $[Y - \delta, Y + \delta]$ where Y is the sample mean of the perfect sampling and δ=1.68 s/ $\sqrt{n}$
  - α is determined from respecting constraint F (m, n, p_0) = α
    - where F (m, n, p)= $\sum_{i=1}^{m} C(n, i)p^i(1-p)^{n-i}$, $p_0 = \theta - \delta, p_1 = \theta + \delta$ and m=n.θ
- Thus c can be determined by applying c=100(1-α)%

Precision of SMC Decision Algorithm

- How to determine confidence level c?
  - 2 contraints are required in the hypothesis testing context:
    - Pr[H_1 is accepted | H_0 is true] $\leq \alpha$
    - Pr[H_0 is accepted | H_1 is true] $\leq \beta$
  - Practically the true mean μ estimated by Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$, is included in the confidence interval $[Y - \delta, Y + \delta]$ where Y is the sample mean of the perfect sampling and δ=1.68 s/ $\sqrt{n}$
  - α is determined from respecting constraint F (m, n, p_0) = α
    - where F (m, n, p)= $\sum_{i=1}^{m} C(n, i)p^i(1 - p)^{n-i}$, $p_0 = \theta - \delta, p_1 = \theta + \delta$ and m=n.θ
- Thus c can be determined by applying c=100(1-α)%

Precision of SMC Decision Algorithm

- How to determine confidence level c?
  - 2 contraints are required in the hypothesis testing context:
    - Pr[H_1 is accepted | H_0 is true] $\leq \alpha$
    - Pr[H_0 is accepted | H_1 is true] $\leq \beta$
  - Practically the true mean μ estimated by Y=$\frac{\sum_{i=1}^{n} X_i}{n} \geq \theta$, is included in the confidence interval $[Y - \delta, Y + \delta]$ where Y is the sample mean of the perfect sampling and δ=1.68 s/ $\sqrt{n}$
  - α is determined from respecting constraint F (m, n, p_0) = α
    - where F (m, n, p)= $\sum_{i=1}^{m} C(n,i)p^i(1-p)^{n-i}$, $p_0 = \theta - \delta, p_1 = \theta + \delta$ and m=n.θ
- Thus c can be determined by applying c=100(1-α)%

Outline

Verification Principle of CSL Steady State Operator

- States of CTMC M are labelled with AP that will be used to define the underlying property ϕ to check
  - ϕ may represent different performance measures of the underlying model
- The checking procedure consists in
  - Finding the sum of the probabilities of the states verifying ϕ
  - Comparing this sum with the probability threshold θ
  - If the comparison relation between the determined sum and θ is verified
    - Then the steady state operator is verified by M

Verification Principle of CSL Steady State Operator

- States of CTMC M are labelled with AP that will be used to define the underlying property ϕ to check
  - ϕ may represent different performance measures of the underlying model
- The checking procedure consists in
  - Finding the sum of the probabilities of the states verifying ϕ
  - Comparing this sum with the probability threshold θ
  - If the comparison relation between the determined sum and θ is verified
    - Then the steady state operator is verified by M

Global Idea

- This summation of the probabilities of the states verifying ϕ can be seen
  - as a reward function defined on the state space
  - where r_ϕ=1 if $s \models \phi$ and r_ϕ=0 if $s \not\models \phi$
- Next we propose to apply a method called functional perfect simulation to check the given formula ϕ on each generated sample
  - by means of software ψ^2
  - by supposing the monotonicity of the considered reward function r_ϕ

Global Idea

- This summation of the probabilities of the states verifying ϕ can be seen
  - as a reward function defined on the state space
  - where r_ϕ=1 if $s \models \phi$ and r_ϕ=0 if $s \not\models \phi$
- Next we propose to apply a method called functional perfect simulation to check the given formula ϕ on each generated sample
  - by means of software ψ^2
  - by supposing the monotonicity of the considered reward function r_ϕ

Proposed Algorithm for CSL Steady State Operator

Set time t to 1 and Repeat steps 1, 2, 3 until coupling on reward function (all rewards will be equal)

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 and generate on each step the trajectories Tx for all x $\in$ setof Max U setof min by considering events E[t], E[t-1], , E[1]
4. Finally, if the reward of the perfect sample is equal to 1 then return 1 (studied sample is a positive sample), otherwise return 0 (studied sample is a negative sample)

Proposed Algorithm for CSL Steady State Operator

Set time t to 1 and Repeat steps 1, 2, 3 until coupling on reward function (all rewards will be equal)

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 and generate on each step the trajectories Tx for all x $\in$ setof Max U setof min by considering events E[t], E[t-1], , E[1]
4. Finally, if the reward of the perfect sample is equal to 1 then return 1 (studied sample is a positive sample), otherwise return 0 (studied sample is a negative sample)

Proposed Algorithm for CSL Steady State Operator

Set time t to 1 and Repeat steps 1, 2, 3 until coupling on reward function (all rewards will be equal)

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 and generate on each step the trajectories Tx for all x $\in$ setof Max U setof min by considering events E[t], E[t-1], , E[1]
4. Finally, if the reward of the perfect sample is equal to 1 then return 1 (studied sample is a positive sample), otherwise return 0 (studied sample is a negative sample)

Proposed Algorithm for CSL Steady State Operator

Set time t to 1 and Repeat steps 1, 2, 3 until coupling on reward function (all rewards will be equal)

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 and generate on each step the trajectories Tx for all x $\in$ setof Max U setof min by considering events E[t], E[t-1], , E[1]
4. Finally, if the reward of the perfect sample is equal to 1 then return 1 (studied sample is a positive sample), otherwise return 0 (studied sample is a negative sample)

Outline

Verification Principle of CSL Unbounded Until

- Unbounded until $\phi_1 U \phi_2$ can be obtained as a special case of the bounded ones by taking I= $[0,\infty)$
- Numerically checking principle
  - Probability measure for an until formula is equivalent to the transient probability at time t of the ϕ_2 states on the CTMC M from making every $(\neg\phi_1 \vee \phi_2)$ state absorbing
- Statistically checking principle
  - Testing states s by starting from initial state and continuing test while state s verifies ϕ_1 until we achieve a state s verifying ϕ_2 at steady state or before this state

Verification Principle of CSL Unbounded Until

- Unbounded until $\phi_1 U \phi_2$ can be obtained as a special case of the bounded ones by taking I= $[0,\infty)$
- Numerically checking principle
  - Probability measure for an until formula is equivalent to the transient probability at time t of the ϕ_2 states on the CTMC M from making every $(\neg\phi_1 \vee \phi_2)$ state absorbing
- Statistically checking principle
  - Testing states s by starting from initial state and continuing test while state s verifies ϕ_1 until we achieve a state s verifying ϕ_2 at steady state or before this state

Proposed Algorithm for CSL Unbounded Until

Set time t to 1 and Repeat while STOP=false

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ initial state s_0 U setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 while STOP=false
   - Generate the trajectories $T_{s_0}, T_{min} and T_{Max}$ by considering events E[t], E[t-1], , E[1]
   - For x $\in$ initial state s_0 U setof Max U setof min test
     - If trajectory Tx meets a state non verifying ϕ_1 then STOP will be True and affect the returned test result to 1
     - Else if trajectory Tx meets a state verifying ϕ_2 then STOP will be True and affect the returned test result to 0
4. Finally, if STOP remains false then we have to test the steady state case
   - if all y(x) are equal then STOP will be True and affect the returned test result to 0

Proposed Algorithm for CSL Unbounded Until

Set time t to 1 and Repeat while STOP=false

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ initial state s_0 U setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 while STOP=false
   - Generate the trajectories T_{s_0}, $T_{min} and T_{Max}$ by considering events E[t], E[t-1], , E[1]
   - For x $\in$ initial state s_0 U setof Max U setof min test
     - If trajectory Tx meets a state non verifying ϕ_1 then STOP will be True and affect the returned test result to 1
     - Else if trajectory Tx meets a state verifying ϕ_2 then STOP will be True and affect the returned test result to 0
4. Finally, if STOP remains false then we have to test the steady state case
   - if all y(x) are equal then STOP will be True and affect the returned test result to 0

Proposed Algorithm for CSL Unbounded Until

Set time t to 1 and Repeat while STOP=false

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ initial state s_0 U setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 while STOP=false
   - Generate the trajectories T_{s_0}, $T_{min} and T_{Max}$ by considering events E[t], E[t-1], , E[1]
   - For x $\in$ initial state s_0 U setof Max U setof min test
     - If trajectory Tx meets a state non verifying ϕ_1 then STOP will be True and affect the returned test result to 1
     - Else if trajectory Tx meets a state verifying ϕ_2 then STOP will be True and affect the returned test result to 0
4. Finally, if STOP remains false then we have to test the steady state case
   - if all y(x) are equal then STOP will be True and affect the returned test result to 0

Proposed Algorithm for CSL Unbounded Until

Set time t to 1 and Repeat while STOP=false

1. Initialize time t to 2.t and initiate trajectories for all x $\in$ initial state s_0 U setof Max U setof min
2. Generate new events from t downto t/2+1
3. Loop from t downto 1 while STOP=false
   - Generate the trajectories $T_{s_0}, T_{min} and T_{Max}$ by considering events E[t], E[t-1], , E[1]
   - For x $\in$ initial state s_0 U setof Max U setof min test
     - If trajectory Tx meets a state non verifying ϕ_1 then STOP will be True and affect the returned test result to 1
     - Else if trajectory Tx meets a state verifying ϕ_2 then STOP will be True and affect the returned test result to 0
4. Finally, if STOP remains false then we have to test the steady state case
   - if all y(x) are equal then STOP will be True and affect the returned test result to 0

Conclusion 1

- Our statistical model checking algorithms that we have developed for stochastic models have at least three advantages over previous works
  - can model check CSL formulas which have unbounded untils and steady state
  - do not suffer from memory problem due to state-space explosion
  - CSL unbounded until model checking algorithm does not suffer from stopping probability problem
    - because of possibility of steady state detection in our approach

Conclusion 1

- Our statistical model checking algorithms that we have developed for stochastic models have at least three advantages over previous works
  - can model check CSL formulas which have unbounded untils and steady state
  - do not suffer from memory problem due to state-space explosion
  - CSL unbounded until model checking algorithm does not suffer from stopping probability problem
    - because of possibility of steady state detection in our approach

Conclusion 1

- Our statistical model checking algorithms that we have developed for stochastic models have at least three advantages over previous works
  - can model check CSL formulas which have unbounded untils and steady state
  - do not suffer from memory problem due to state-space explosion
  - CSL unbounded until model checking algorithm does not suffer from stopping probability problem
    - because of possibility of steady state detection in our approach

Conclusion 2

- However, our algorithms also have at least two limitations
  - cannot guarantee the accuracy that numerical techniques achieve
  - running time will increase if we try to increase the accuracy by making the error bounds or confidence level very small
- Thus statistical model checking technique can be seen as
  - an alternative to numerical techniques
  - can be be used when it is infeasible to use numerical techniques, for example, in large-scale systems as ad hoc and sensor networks

Conclusion 2

- However, our algorithms also have at least two limitations
  - cannot guarantee the accuracy that numerical techniques achieve
  - running time will increase if we try to increase the accuracy by making the error bounds or confidence level very small
- Thus statistical model checking technique can be seen as
  - an alternative to numerical techniques
  - can be be used when it is infeasible to use numerical techniques, for example, in large-scale systems as ad hoc and sensor networks

Conclusion 2

- However, our algorithms also have at least two limitations
  - cannot guarantee the accuracy that numerical techniques achieve
  - running time will increase if we try to increase the accuracy by making the error bounds or confidence level very small
- Thus statistical model checking technique can be seen as
  - an alternative to numerical techniques
  - can be be used when it is infeasible to use numerical techniques, for example, in large-scale systems as ad hoc and sensor networks

Conclusion 2

- However, our algorithms also have at least two limitations
  - cannot guarantee the accuracy that numerical techniques achieve
  - running time will increase if we try to increase the accuracy by making the error bounds or confidence level very small
- Thus statistical model checking technique can be seen as
  - an alternative to numerical techniques
  - can be be used when it is infeasible to use numerical techniques, for example, in large-scale systems as ad hoc and sensor networks