Simulative Model Checking of Colored Stochastic Petri Nets

Motivation

- Stochastic models used to model
- Technical systems
- Biochemical networks
- Compulsory, if the stochastic noise is crucial
- Systems are getting more and more complex
- Number of states grows
- Unbounded models
- Number of states is \( \infty \)
- ⇒ Simulative approach to handle such models

Colored Stochastic Petri Net

- \( \text{SPN} c = \langle P, T, F, \Sigma, g, f, v, m_0 \rangle \)
- \( P = \text{finite, non-empty set of places} \)
- \( T = \text{finite, non-empty set of transitions} \)
- \( F = \text{finite set of directed arcs} \rightarrow \)
- \( F \subseteq (P \times T) \cup (T \times P) \)
- \( \Sigma = \text{finite, non-empty set of color sets} \)
- \( C : P \rightarrow \Sigma \text{ assign each place a color set} \)
- \( g : T \rightarrow \text{EXP assign each transition a guard expression} \)
- \( f : F \rightarrow \text{EXP assign each arc an expression} \)
- \( v : T \rightarrow \text{H assign a stochastic hazard function to each transition} \)
- \( h := \bigcup_{h_i} \{ h_i | N \rightarrow R \} \)
- \( m_0 : P \rightarrow \text{EXP assign each place an initialization expression} \)
- \( \text{EXP} = \text{expression over colors } \varsigma \in \Sigma \)
- Semantic: Continuous Time Markov Chain

Cyclic Server Polling System

- Cyclic server polling system (CSPS) by Ibe & Trivedi (1990)
- Comprises a set of waiting lines that receive arrivals from the external world
- One server that cyclically visits the queues, providing service to customers
- Use \( N \) to denote the number of stations handled by the polling server

Stochastic Simulation

- Creates a single finite path through the possibly infinite CTMC
- Direct method introduced by Gillespie 1977, computational complexity is \( O(T) \)
- Next reaction method by Gibson & Bruck 2000, computational complexity is \( O(\log T) \)
- Probability that transition \( i, \in T \) will occur in the infinitesimal time interval \( [t, t+\Delta t) \) is given by:
  \[ P(t+\Delta t, i) | s) = h(s) e^{-h(s) \Delta t} \]
- So, the enabled transitions in the net compete in a race condition.
- The fastest one determines the next state and the simulation time elapsed.
- In the new state, the race condition starts anew.

Model Checking

- Automatically determines whether a system satisfies a specific
- Property expressed in some kind of temporal logic
- Probabilistic Linear-time Temporal Logic with numerical constraints – PLTLc
- Transient analysis
  - \( N = 10 \) stations, time point \( T = 10 \)
  - Probability that at time point 10 station \( x \) is
    awaiting service:
    \[ P_x(\sum_{i=1}^{N} s = 3\land a = 1) \]
    \[ x_i = 0.07606892732, x_i = 0.0760200225 \]
    \[ x_i = 0.07604394239, x_i = 0.0761958408 \]
    \[ x_i = 0.0762871845, x_i = 0.0744907591 \]
    \[ x_i = 0.0763024084, x_i = 0.0750540450 \]
    \[ x_i = 0.0761197211, x_i = 0.076677831 \]
- Steady state analysis
  - \( N = 10 \) stations
  - Probability that in the long run station \( 1 \) is
    awaiting service:
    \[ P_1(\sum_{i=1}^{N} s = 1\land -(s = 1 \land a = 1)) \]
    \[ S = 0.14021 \]
- Probability that in the long run station \( 1 \) is idle:
  \[ P_1(\sum_{i=1}^{N} s = 0) \]
  \[ S = 0.78162 \]

Comparison

- Efficient stochastic simulation
- Direct, next reaction method
- Simulative PLTLc model checking
- Simulative steady state computation
- Time-unbounded temporal operators
- On-the-fly and offline
- Multi-threaded and distributed computation

Conclusions

- Simulation & model checking at the colored level
- Numerical PLTLc model checking
- Model checking of hybrid Petri nets
- Reinvestigate GPGPU for simulative model checking

References


International Graduiertenschule der BTU Cottbus

Fakultät ZUSYS

Kontakt

Name: Christian Rohr
Email: Christian.Rohr@tu-cottbus.de