Lecture 6
PRISM

Dr. Dave Parker

Department of Computer Science
University of Oxford
Practicals

• 4 practical exercises
• 4 scheduled 2 hour practical sessions:
  – Tuesday 4–6pm, room 379, weeks 3, 4, 6 and 7
  – demonstrator: Aistis Simaitis

• Note:
  – you will also be expected to complete some of the practical work outside these hours
  – final assignment will include practical (PRISM) exercises

http://www.prismmodelchecker.org/courses/pmc1112/
Overview

• Tool support for probabilistic model checking
  – motivation, existing tools

• The PRISM model checker
  – functionality, features
  – modelling language
  – property specification

• Running example
  – leader election protocol

• PRISM tool demo
Motivation

- **Complexity of PCTL model checking**
  - generally polynomial in model size (number of states)

- **State space explosion problem**
  - models for realistic case studies are typically huge

- **Clearly (efficient) tool support is required**

- **Benefits:**
  - fully automated process
  - high-level languages/formalisms for building models
  - visualisation of quantitative results
Probabilistic model checkers

- PRISM (this lecture): DTMCs, MDPs, CTMCs, PTAs + rewards
- Markov chain model checkers
  - MRMC: DTMCs, CTMCs + reward extensions
  - PEPA toolset: CTMCs + CSL
- Markov decision process (MDP) tools
  - LiQuor: LTL verification for MDPs (Probmela language)
  - RAPTURE: prototype for abstraction/refinement of MDPs
  - ProbDiVinE: parallel/distributed LTL model checking of MDPs
- Simulation-based probabilistic model checking:
  - APMC, Ymer (both based on PRISM language), VESTA
- And more
  - APNN-Toolbox, SMART, CADP, Möbius, PASS, PARAM, ...
  - see: http://www.prismmodelchecker.org/other-tools.php
The PRISM tool

- **PRISM**: Probabilistic symbolic model checker
  - developed at Birmingham/Oxford University, since 1999
  - free, open source (GPL)
  - versions for Linux, Unix, Mac OS X, Windows, 64-bit OSs

- **Modelling of:**
  - DTMCs, CTMCs, MDPs + costs/rewards
  - probabilistic timed automata (PTAs) (not covered here)

- **Model checking of:**
  - PCTL, CSL, LTL, PCTL* + extensions + costs/rewards
PRISM functionality

• High-level modelling language
• Wide range of model analysis methods
  – efficient symbolic implementation techniques
  – also: approximate verification using simulation + sampling
• Graphical user interface
  – model/property editor
  – discrete-event simulator – model traces for debugging, etc.
  – easy automation of verification experiments
  – graphical visualisation of results
• Command-line version
  – same underlying verification engines
  – useful for scripting, batch jobs
Probabilistic model checking

- Overview of the probabilistic model checking process
  - two distinct phases: model construction, model checking
Model construction

High-level model → Model construction

- Translation from high-level language
- Reachability: building set of reachable states

Model construction

- Matrix manipulation
- Graph-based algorithm

Model → DTMC, MDP, or CTMC
Modelling languages/formalisms

- Many high-level modelling languages, formalisms available

- For example:
  - probabilistic/stochastic process algebras
  - stochastic Petri nets
  - stochastic activity networks

- Custom languages for tools, e.g.:
  - PRISM modelling language
  - Probmela (probabilistic variant of Promela, the input language for the model checker SPIN) – used in LiQuor
PRISM modelling language

- **Simple, textual, state-based language**
  - modelling of DTMCs, CTMCs, MDPs, ...
  - based on Reactive Modules [AH99]

- **Basic components...**

- **Modules:**
  - components of system being modelled
  - composed in parallel

- **Variables**
  - finite (integer ranges or Booleans)
  - local or global
  - all variables public: anyone can read, only owner can modify
PRISM modelling language

- **Guarded commands**
  - describe behaviour of each module
  - i.e. the changes in state that can occur
  - labelled with probabilities (or, for CTMCs, rates)
  - (optional) action labels

\[
[\text{send}] (s=2) \rightarrow p_{\text{loss}} : (s'=3) \& (\text{lost}'=\text{lost}+1) + (1-p_{\text{loss}}) : (s'=4);
\]
PRISM modelling language

• **Parallel composition**
  – model multiple components that can execute independently
  – for DTMC models, mostly assume components operate synchronously, i.e. move in lock-step

• **Synchronisation**
  – simultaneous transitions in more than one module
  – guarded commands with matching action-labels
  – probability of combined transition is product of individual probabilities for each component

• **More complex parallel compositions can be defined**
  – using process-algebraic operators
  – other types of parallel composition, action hiding/renaming
Simple example

module M1
    x : [0..3] init 0;
    [a] x=0 -> (x'=1);
    [b] x=1 -> 0.5:(x'=2) + 0.5:(x'=3);
endmodule

module M2
    y : [0..3] init 0;
    [a] y=0 -> (y'=1);
    [b] y=1 -> 0.4:(y'=2) + 0.6:(y'=3);
endmodule
Example: Leader election

- Randomised leader election protocol
  - due to Itai & Rodeh (1990)
- Set-up: N nodes, connected in a ring
  - communication is synchronous (lock-step)
- Aim: elect a leader
  - i.e. one uniquely designated node
  - by passing messages around the ring
- Protocol operates in rounds. In each round:
  - each node choose a (uniformly) random id $\in \{0,\ldots,k-1\}$
  - ($k$ is a parameter of the protocol)
  - all nodes pass their id around the ring
  - if there is (maximum) unique id, node with this id is the leader
  - if not, try again with a new round
PRISM code
PRISM property specifications

- Based on (probabilistic extensions of) temporal logic
  - incorporates PCTL, CSL, LTL, PCTL*
  - also includes: quantitative extensions, costs/rewards

- Leader election properties
  - $P_{\geq 1} [ F \text{ elected } ]$
    - with probability 1, a leader is eventually elected
  - $P_{>0.8} [ F^{\leq k} \text{ elected } ]$
    - with probability greater than 0.8, a leader is elected within k steps

- Usually focus on quantitative properties:
  - $P_{\leq ?} [ F^{\leq k} \text{ elected } ]$
    - what is the probability that a leader is elected within k steps?
PRISM property specifications

- **Best/worst-case scenarios**
  - combining “quantitative” and “exhaustive” aspects

- e.g. computing values for a range of states...

- **P_≤? [ F≤t elected {tokens≤k}{min} ]** –
  - “minimum probability of the leader election algorithm completing within t steps from any state where there are at most k tokens”

- **R_≤? [ F end {“init”}{max} ]** –
  - “maximum expected run-time over all possible initial configurations”
PRISM property specifications

- **Experiments:**
  - ranges of model/property parameters
  - e.g. $P=? [ F \leq T \text{ error }]$ for $N=1..5$, $T=1..100$
    where $N$ is some model parameter and $T$ a time bound
  - identify patterns, trends, anomalies in quantitative results
PRISM…
More info on PRISM

• PRISM website: http://www.prismmodelchecker.org/
  – tool download: binaries, source code (GPL)
  – on-line example repository (50+ case studies)
  – on-line documentation:
    • PRISM manual
    • PRISM tutorial
  – support: help forum, bug tracking, feature requests
  – related publications, talks, tutorials, links

• Course practicals info at:
  – http://www.prismmodelchecker.org/courses/pmc1112/