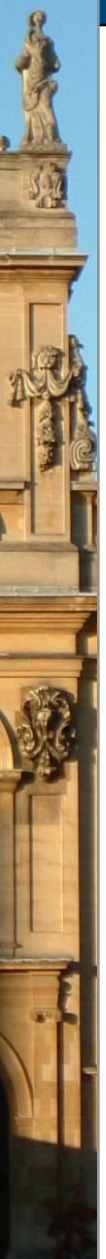
Probabilistic Model Checking in Practice

Dave Parker

Oxford University Computing Laboratory

Quantitative Model Checking PhD School, Copenhagen, March 2010



Overview

- Tool support for probabilistic model checking
- The PRISM model checker
  - functionality, features
  - modelling language
  - property specification
- PRISM tool demo
- PRISM lab session

Probabilistic model checking

- Recap…
- Probabilistic models
  - discrete-time Markov chains (DTMCs)
  - Markov decision processes (MDPs)
  - continuous-time Markov chains (CTMCs)
- Probabilistic temporal logics
  - PCTL, LTL, PCTL* (discrete-time models)
  - CSL (continuous-time models)

Probabilistic model checkers

- PRISM (this session)
  - DTMCs, MDPs, CTMCs + costs/rewards
- Markov chain model checkers
  - MRMC: explicit-state engine for DTMCs, CTMCs + rewards
  - PEPA Plug-in Project: CSL model checking for PEPA (CTMCs)
  - CASPA: symbolic model checking of stochastic process algebra
- MDP model checkers
  - LiQuor: LTL verification for MDPs (Probmela language)
  - RAPTURE: abstraction/refinement tool for MDPs
- Many other interesting tools being developed:
  - e.g. for PTAs: UPPAAL PRO, PRISM (soon), mcpta, Fortuna

The PRISM tool

- PRISM: Probabilistic symbolic model checker
  - developed at Universities of Birmingham/Oxford, since 1999
  - free, open source (GPL), versions for all major OSs

- Modelling of:
  - DTMCs, CTMCs, MDPs + costs/rewards
  - simple, state-based modelling language
- Model checking of:
  - PCTL, CSL, LTL, PCTL* + extensions + costs/rewards

- Features
  - efficient symbolic/explicit implementation techniques
  - approximate verification using simulation + sampling
  - GUI: model editor, simulator/debugger, result visualisation

PRISM modelling language

- Simple, textual, state-based language
  - modelling of DTMCs, CTMCs and MDPs
  - based on Reactive Modules [Alur/Henzinger]

- Basic components:
  - modules: components of system being modelled
    - combined through parallel composition
  - variables: local/global, finite-ranging (integers/Booleans)
  - guarded commands: probabilistic updates to variables
    - optional action labels for synchronisation

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

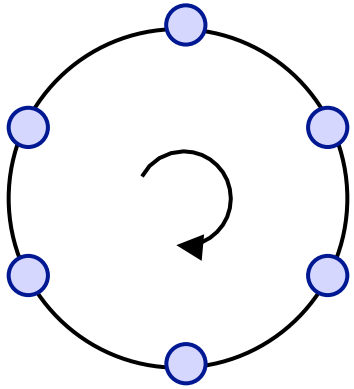
action guard probability update probability update

PRISM modelling language

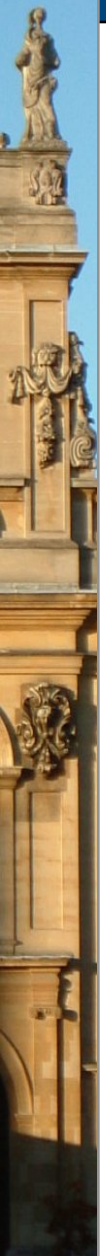
- Parallel composition
  - synchronous or asynchronous composition of modules
  - process-algebraic operators for e.g. action hiding/renaming
- Module renaming
  - easy construction of identical/symmetric modules
- Rewards (or equivalently: costs, prices, …)
  - real-valued quantities assigned to states and/or transitions
  - these can have a wide range of possible interpretations, e.g.:
  - elapsed time, power consumption, size of message queue, number of messages successfully delivered, net profit, …

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 chooses a (uniformly) random id $\in \{0,\ldots,k-1\}$
  - (k is a parameter of the protocol)
  - all nodes pass their id around the ring
  - the node with the maximum unique id becomes the leader
  - if no unique id exists, try again with a new round



PRISM code…



PRISM property specifications

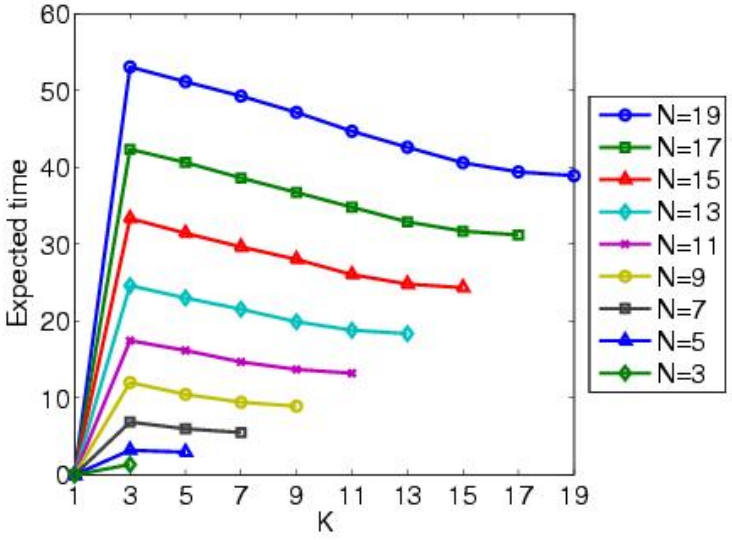
- Based on (probabilistic extensions of) temporal logic
  - incorporates PCTL, CSL, LTL, PCTL*
  - also includes: quantitative extensions, costs/rewards

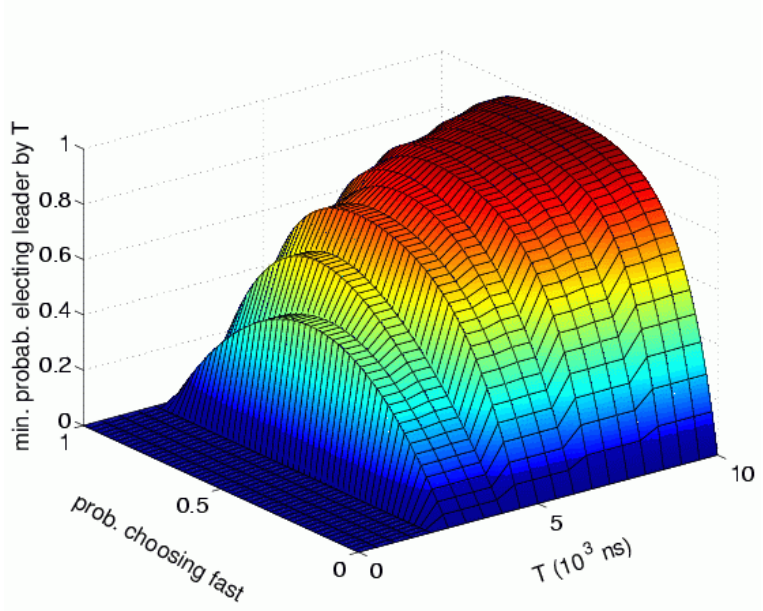
- Example properties (leader election)
  - $P_{\geq 1}$ [F "elected"]
    "with probability 1, a leader is eventually elected"
  - $P_{\geq 1}$ [F G "elected"]
    "with probability 1, a leader is eventually elected permanently"
  - $P_{>0.8}$ [$F^{\leq T}$ "elected"]
    "with probability $>$ 0.8, a leader is elected within T steps"

- Usually focus on quantitative properties:
  - $P_{=?}$ [$F^{\leq T}$ "elected"]
    "what is the probability that a leader is elected within T steps?"

PRISM property specifications

- Experiments:
  - ranges of model/property parameters
  - e.g. $P_{=?}$ [$F^{\leq T}$ "elected"] 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



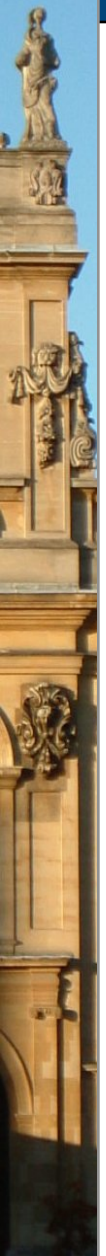


PRISM property specifications

- Rewards/costs
  - expected (instantaneous/cumulative) value of reward
  - e.g. “the expected time for a leader to be elected”
  - e.g. “the expected power consumption over one hour”
  - e.g. “the expected queue size after exactly 90 seconds”

- Best/worst-case scenarios
  - combining “quantitative” and “exhaustive” aspects
  - for MDPs, quantification over all adversaries/schedulers
  - e.g. $P_{min=?}$ [F “terminate”] – “worst-case probability of termination over all possible schedulers”
  - for any model, compute values for a range of states
  - e.g. $R_{=?}$ [F end {“init”}{max}] – “maximum expected run-time over all possible initial configurations”

PRISM demo…



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: manual, tutorial, FAQ
  - support: help forum
  - related publications, talks, tutorials, links
- Practical session using PRISM
  - upstairs in PC labs 2A52 and 2A54
  - http://www.prismmodelchecker.org/courses/qmc10/