

# PRISM: A Tool for Automatic Verification of Probabilistic Systems



Andrew Hinton, Marta Kwiatkowska,  
Gethin Norman, **Dave Parker**



University of Birmingham

# Overview

- Probabilistic model checking
- The PRISM tool
  - functionality + features
  - modelling language
  - property specification
  - brief demo
  - case studies
  - resources

# Probabilistic Model Checking

- **Formal methods for analysing probabilistic systems**
  - randomisation, e.g. Bluetooth, FireWire, ...
  - uncertainty, e.g. message losses/delays, failures, ...
- **Exhaustive exploration/analysis of probabilistic models**
  - discrete/continuous-time Markov chains (DTMCs/CTMCs)
  - Markov decision processes (MDPs) (nondeterminism + prob.)
- **Analysis of quantitative (or qualitative) properties**
  - probabilistic extensions of temporal logic (e.g. PCTL)
  - graph-based algorithms + numerical computation

# Probabilistic model checking

- + wide range of quantitative measures
- + exact answers computed
- + fully automatic process
  - model construction + numerical solution
- no counterexamples
  - but can identify patterns, trends, anomalies in quantitative results
- + exhaustive analysis, good for 'corner cases', e.g.
  - all possible initial configurations/model parameter values
  - all possible process schedulings (nondeterminism)
- state space explosion: time and memory constraints
- + efficient algorithms and implementations

# PRISM: Probabilistic Model Checker

- Constructs three types of probabilistic models:
  - DTMCs, CTMCs, MDPs
  - augmented with costs/rewards (new)
  - PRISM modelling language
- Variety of import/export functionality
  - model output: text files, Dot graphs, Matlab, ETMCC/MRMC
  - model import: text files (new)
  - other input formalisms via language translation
    - PEPA (now), CSP, PROBMELA, pi-calculus (in progress)
  - connections to other tools:
    - APMC, ProVer, Ymer, CASPA, KRONOS

# PRISM: Probabilistic Model Checker

- Supports verification of:
  - PCTL, CSL + extensions
  - cost/reward-based properties (new)
- Multiple computation engines:
  - efficient symbolic (BDD-based) implementations
  - sampling-based computation (discrete-event simulation) (new)
    - distributed implementation under development
- Graphical user interface
  - model/property editor
  - easy automation + graphical visualisation of results
  - debugging tool: simulation engine (new)
    - manual/automatic generation of model traces

# PRISM modelling language

- **Simple, state-based** language for DTMCs/CTMCs/MDPs
  - based on Reactive Modules [Alur/Henzinger]
- **Modules** (system components, composed in parallel)
- **Variables** (finite-valued, local or global)
- **Guarded commands** (labelled with probabilities/rates)
- **Synchronisation** (CSP-style) + **process-algebraic operators** (parallel composition, action hiding/renaming)

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



# PRISM Language Example

```
// hermans self-stabilisation algorithm [Her90]
```

```
dtmc // algorithm is synchronous
```

```
module process1 // first of N=5 symmetric processes
```

```
    x1 : [0..1]; // one bit per process; xi=x(i-1) means process i has a token
```

```
    [step] (x1=x5) -> 0.5 : (x1'=0) + 0.5 : (x1'=1);
```

```
    [step] !x1=x5 -> (x1'=x5);
```

```
endmodule
```

```
// add further processes through renaming
```

```
module process2 = process1[ x1=x2, x5=x1 ] endmodule
```

```
module process3 = process1[ x1=x3, x5=x2 ] endmodule
```

```
module process4 = process1[ x1=x4, x5=x3 ] endmodule
```

```
module process5 = process1[ x1=x5, x5=x4 ] endmodule
```

```
// can start in any possible configuration
```

```
init true endinit
```

```
// cost - 1 in each state (expected number of steps)
```

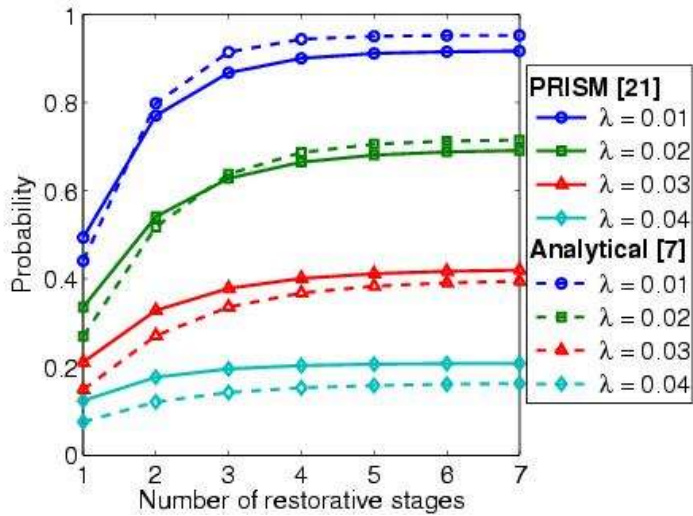
```
rewards true : 1; endrewards
```

# PRISM – Property specifications

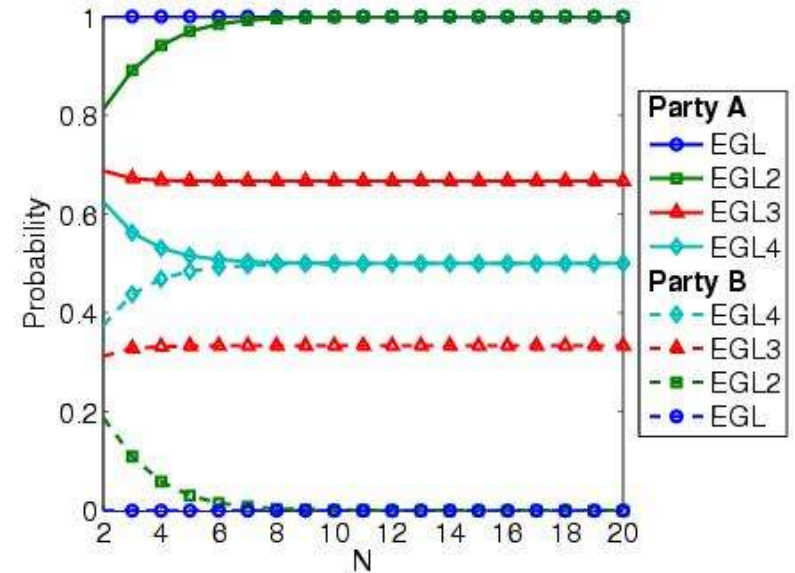
- Formulae in probabilistic extensions of temporal logic
  - PCTL for DTMCs/MDPs, CSL for CTMCs
- Examples:
  - $P < 0.001 [ F \text{ shutdown } ]$  - “shutdown eventually occurs with probability at most 0.001”
  - $P < 0.2 [ F[t,t] (\text{deliv\_rate} < \text{min}) ]$  “the probability that the current packet delivery rate has dropped below minimum at time t is less than 0.2”
  - $P \geq 0.95 [ !\text{repair } U \leq 200 \text{ done } ]$  - “with probability 0.95 or greater, the process will successfully complete within 200 hours and without requiring any repairs”
  - $S > 0.75 [ \text{num\_sensors} \geq \text{min} ]$  - “in the long-run, the probability that an adequate number of sensors are operational is greater than 0.75”

# PRISM – Property specifications

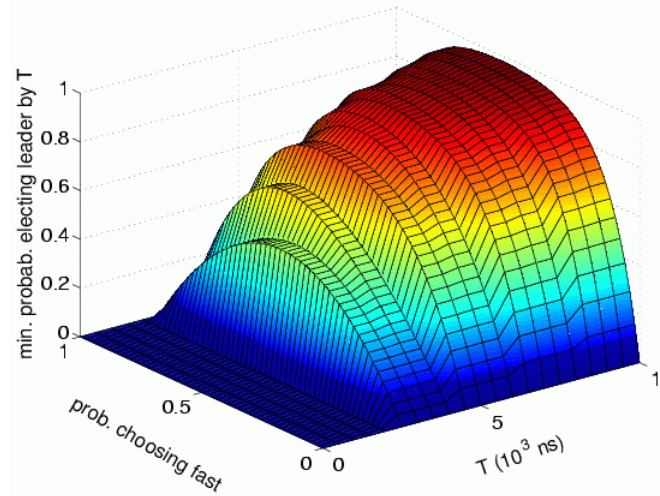
- Focus on quantitative properties, compute actual values
  - $P=? [ F \leq T \text{ "shutdown" } ]$  - “what is the probability of shutdown occurring within T hours?”
  - $S=? [ \text{num\_sensors} \geq 4 ]$  - “what is the long-run probability that 4 or more sensors are operational?”
- Best/worst-case scenarios
  - $P=? [ F \text{ "error" } \{ \text{"init"} \} \{ \text{max} \} ]$  - “what is the worst-case error probability over all possible initial configurations?”
  - $P_{\min}=? [ !\text{end2} \text{ U } \text{end1} ]$  - “what is the minimum probability of process 1 finishing before process 2 over all possible schedulings of the processes?”
- Experiments – ranges of model/property parameters
  - $P=? [ F \leq T \text{ error} ]$  for  $N=1..5, T=1..100$



Probability that 10% of gate outputs are erroneous for varying gate failure rates and numbers of stages



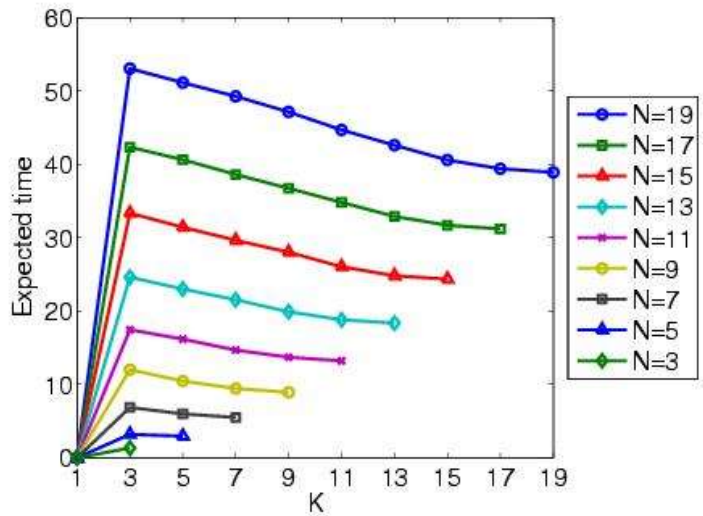
Probability that parties gain unfair advantage for varying numbers of secret packets sent



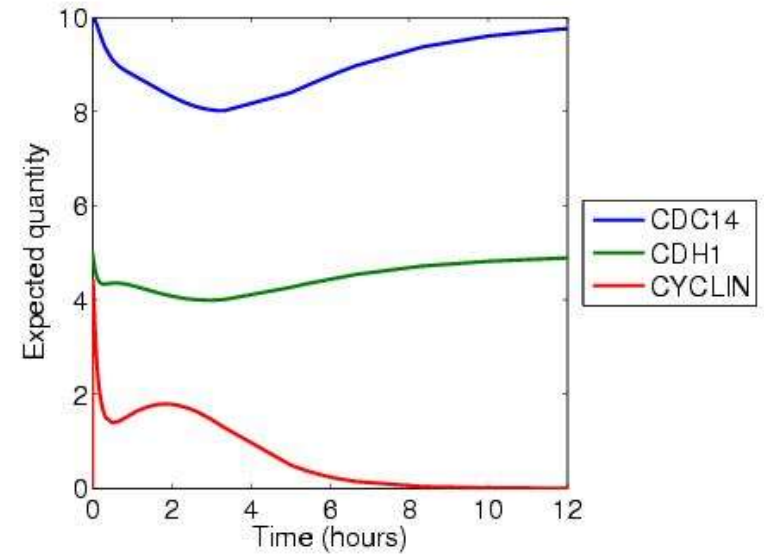
Optimum probability of leader election by time T for various coin biases

# Cost- and reward-based properties

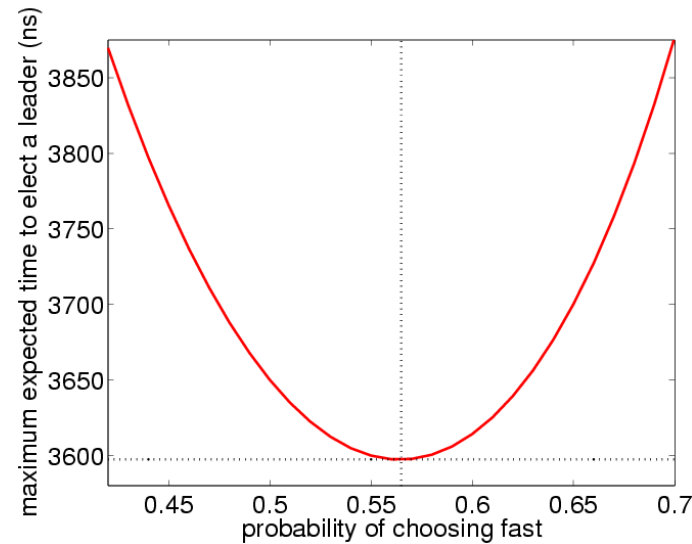
- **Costs and rewards**
  - real-valued quantities assigned to states/transitions
- **Instantaneous** – state-based measures
  - current queue size, number of operational channels, ...
  - “what is the **expected size** of the message queue at time  $t$ ?”
  - “what is the **long-run expected size** of the queue?”
- **Cumulative** – state or transition (impulse) costs/rewards
  - time, power consumption, messages lost, ...
  - “what is the **expected power consumption** during the first 2 hours of operation?”
  - “what is the **worst-case expected time** taken for the protocol to terminate?”



Worst-case expected number of steps to stabilise for initial configurations with K tokens amongst N processes



Expected reactant concentrations over the first 12 hours



Maximum expected time for leader election for various coin biases

# PRISM Demo

# PRISM Screenshots

The screenshot displays the PRISM 3.0.beta1 software interface. The main window shows the source code for a model named 'cluster.sm'. The code is written in a stochastic Petri net style, defining constants for the number of workstations (N, left\_mx, right\_mx), a minimum operational threshold (k), and transition rates (line\_rate, Toleft\_rate, Toright\_rate). It defines two modules, 'Left' and 'Right', which represent the workstation clusters. The 'Left' module has a state variable 'left\_n' (number of operational workstations) and a boolean 'left' (being repaired). Transitions include 'startLeft', 'repairLeft', and a repair action. The 'Right' module is defined similarly but is not fully shown in the code snippet.

```
// workstation cluster [HHK00]
// dxp/gxn 11/01/00

stochastic

const int N; // number of workstations in each cluster
const int left_mx = N; // number of work stations in left cluster
const int right_mx = N; // number of work stations in right cluster

// minimum QoS requires 3/4*N connected workstations operational
const int k=floor(0.75*N);
formula minimum = (left_n>=k & Toleft_n) |
                  (right_n>=k & Toright_n) |
                  ((left_n+right_n)>=k & Toleft_n & Toright_n);

// rates
const double line_rate = 0.0002;
const double Toleft_rate = 0.00025;
const double Toright_rate = 0.00025;

// left cluster
module Left

    left_n : [0..left_mx] init left_mx; // number of workstations operational
    left : bool; // being repaired?

    [startLeft] !left & (left_n<left_mx) -> 1 : (left'=true);
    [repairLeft] left & (left_n<left_mx) -> 1 : (left'=false) & (left_n'=left_n+1);
    [] (left_n>0) -> 0.002*left_n : (left_n'=left_n-1);

endmodule

// right cluster
module Right = Left[left_n=right_n,
                    left=right,
                    left_mx=right_mx,
                    startLeft=startRight,
                    repairLeft=repairRight ]
```

On the left side, a tree view shows the model structure: Model: cluster.sm (Type: Stochastic (CTMC)) with Modules (Left, Right, Repairman, Line, ToLeft, ToRight) and Constants (N, left\_mx, right\_mx, k, line\_rate, Toleft\_rate, Toright\_rate, OPERATIONAL, MINIMUM, REPAIR).

At the bottom left, the 'Built Model' section shows: No of states: 4180, No of transitions: 19552.

The bottom status bar indicates 'Building model... done.'

# PRISM Screenshots

PRISM 3.0.beta1

File Edit Model Properties Options

Properties list: /data/private/luser/prism-examples/cluster/cluster.csl

Properties

```

S=? [ "premium" ]
S=? [ !"minimum" ]
P>=1 [ true U "premium" ]
P=? [ true U<=T !"minimum" ]
P=? [ true U[T,T] !"minimum" {"!minimum"}{max} ]
P=? [ true U<=T "premium" {"!minimum"}{min} ]
P=? [ "minimum" U<=T "premium" {"!minimum"}{min} ]
P=? [ !"minimum" U>=T "minimum" {"!minimum"}{max} ]
R=? [ I=T {"!minimum"}{min} ]
R=? [ C<=T ]
R=? [ C<=T ]

```

Experiments

Property	Defined Const...	Progress	Status	Method
P=? [ true U[T...	T=0.0:1.0E-...	660/660 (100%)	Done	Verification
P=? [ true U[T...	N=3,T=0.0:1...	101/101 (100%)	Done	Simulation
P=? [ true U[T...	N=3,T=0.0:1...	44/101 (43%)	Stopped	Verification
P=? [ true U<...	N=3,T=0.0:1...	21/21 (100%)	Done	Verification
P=? [ true U<...	N=3:1:5,T=0...	63/63 (100%)	Done	Verification

Graph1 Graph2 Graph3 Graph4 Graph5

New Graph

Probability

T

Legend: N=3, N=4, N=5

Model Properties Simulator Log

Running experiment... done.

# PRISM Screenshots

PRISM 3.0.beta1

File Edit Model Properties Options

✂️ 📄 🗑️ 🗑️

Exploration

Auto Update

No. Steps:

Do Update

State time:   Auto

Action	Rate	Update
Left	0.016	left_n'=7
Right	0.02	right_n'=9
Line	2.0E-4	line_n'=false
ToRight	2.5E-4	Toright_n'=fal
[startLeft]	10.0	left'=true, r'='
[startToLeft]	10.0	r'=true, Toleft

Path Modification

Backtrack Remove

To Step:  Before:

Formulae

Path formulae:

- ? true U<=T !"minimum"
- ? true U[T,T] !"minimum"
- ✓ true U<=T "premium"

State labels:

- ✗ init
- ✗ deadlock
- ✓ minimum
- ✓ premium

Simulation Path

New Path Reset Path Export Path

<b>Model Type:</b> Stochastic (CTMC)	<b>Path Length:</b> 18	<b>Total Time:</b> 44.030215385327985
<b>State Rewards:</b> 4373.594264201196	<b>Transition Rewards:</b> 0.0	<b>Total Reward:</b> 4373.594264201196
<b>Defined Constants:</b> N=10,T=100.0		

Step	left_n	left	right_n	right	r	line	line_n	Toleft
0	10	false	10	false	false	false	true	false
1			9					
2				true	true			
3			10	false	false			
4	9							
5		true			true			
6	10	false			false			
7			9					
8				true	true			
9			10	false	false			
10	9							
11								
12			9					
13	8							
14	7							
15		true			true			
16	8	false			false			
17				true	true			
18	8	false	10	false	false	false	true	false

Model Properties Simulator Log

Loading properties... done.

# PRISM Case studies

- Wide range of application domains
  - **communication and multimedia protocols** – Bluetooth, FireWire, Zeroconf, CSMA/CD, WiFi, ...
  - **security protocols** – anonymity, contract signing, fair exchange, negotiation frameworks, PIN cracking, ...
  - **randomised distributed algorithms** – self-stabilisation, consensus, leader election, mutual exclusion, ...
  - **biological processes** – signalling/cell cycle pathways, ..
  - **performance/reliability** of manufacturing systems, computer networks, NAND multiplexing, dynamic power management, embedded systems, quantum cryptography, thinkteam, ...
- See: [www.cs.bham.ac.uk/~dxp/prism/casestudies](http://www.cs.bham.ac.uk/~dxp/prism/casestudies)

# Getting PRISM + Other Resources

- [Free and open source](#) (GPL licensed)
  - Linux, Mac OS X (new), Windows, Unix
- Also available:
  - [online example repository](#) (~40 case studies)
  - [documentation](#) (wiki-based manual online soon)
  - [support: help forum, bug tracking, feature requests](#)
    - hosted on Sourceforge
  - [related publications, links](#)
- See: [www.cs.bham.ac.uk/~dxp/prism](http://www.cs.bham.ac.uk/~dxp/prism)