Model Checking of Probabilistic Systems

Dave Parker

University of Oxford

(joint work with Marta Kwiatkowska, Gethin Norman, ...)

ARTIST2 Workshop: Foundations of Component-based Design
September 30th 2007, Salzburg
Overview

- Probabilistic model checking
- Probabilistic models
  - DTMCs, CTMCs, MDPs, PTAs, modelling formalisms
- Property specifications
  - PCTL, quantitative results, costs & rewards
- Implementation/tool support
- Current research directions
  - symmetry reduction, compositionality, abstraction/refinement
Verification through model checking

Finite-state model

Temporal logic specification

$\neg$EF error

Model checker

Result

Error trace
Probabilistic model checking

Finite-state probabilistic model
e.g. Markov chain/process

Probabilistic temporal logic specification
e.g. PCTL

Probabilistic model checker
e.g. PRISM

Result

Quantitative results

$P_{<0.01}$ [ F error ]
Discrete-time Markov chains (DTMCs)

Features:
- discrete state space
- discrete time-steps
- discrete probability distributions

Well suited to modelling:
- randomised algorithms/protocols
- systems with component failures
- assumptions: synchronous (lock-step) parallel composition of components

PRISM case studies:
- Bluetooth device discovery
- probabilistic contract signing
- leader election/self-stabilisation algorithms
- NAND multiplexing
- dynamic power management schemes
Continuous-time Markov chains (CTMCs)

Features:
- discrete state space
- continuous-time (exponentially distributed transition delays)

Well suited to modelling:
- component lifetimes, e.g. network cluster
- inter-arrival times, e.g. queueing systems
- biochemical reaction rates, ...

PRISM case studies:
- Dynamic power management schemes
- Queueing/manufacturing systems
- Workstation clusters
- Groupware systems (thinkteam)
- Biological signalling pathways (FGF, Eukaryotes, ...)
Markov decision processes (MDPs)

Features:
- discrete state space, time-steps
- probability and nondeterminism
- nondeterministic choice between multiple probability distributions

Well suited to modelling:
- asynchronous parallel composition of probabilistic components, e.g. randomised distributed algorithms
- unknown environments, e.g. probabilistic security protocols
- underspecification, e.g. probabilistic communication protocol designed for a range of possible message propagation delays

PRISM case studies:
- Randomised algorithms for self-stabilisation, leader election, consensus, ...
- Firewire/Zeroconf/ZigBee/CSMA/CD, CSMA/CA
- Probabilistic security protocols for anonymity, contract signing, fair exchange, pin cracking, ...
- Dynamic voltage scaling
Probabilistic timed automata (PTAs)

Features:
- probability + nondeterminism + real–time
- real–valued clocks (extension of timed automata with discrete probabilistic choice)

Well suited to modelling:
- real–time communication/network protocols featuring randomisation (e.g. random choice of back–off etc.)

PRISM case studies:
- Firewire (IEEE 1394)
- CSMA/CD
- WiFi (802.11)
- ZigBee (802.15.4)
- Zeroconf
High-level modelling formalisms

- **PRISM modelling language (DTMCs, CTMCs, MDPs)**
  - simple, state-based language
  - based on Reactive Modules [Alur/Henzinger]
  - modules (system components, composed in parallel)
  - variables (finite-valued – integer ranges or booleans)
  - guarded commands (labelled with probabilities/rates)
  - parallel composition of modules – either asynchronous or synchronous (CSP-style)

- **Translations to PRISM language from**
  - process algebras (PEPA, prob-CSP, probabilistic π-calculus)
  - graphical modelling for wireless networks [Fehnker et al.]

- **Probmela – probabilistic extension of Promela**
Property specifications

- **Probabilistic extensions of temporal logic (CTL)**
  - PCTL (for DTMCs/MDPs), CSL (for CTMCs), PTCTL (for PTAs)
  - essentially (timed) probabilistic reachability

- **The P operator – quantitative extension of CTL’s A and E**
  - \( \text{send} \rightarrow P_{\geq 0.95} [ F \text{ deliver} ] \) – “if a message is sent, then the probability of it eventually being delivered is at least 0.95”
  - for MDPs/PTAs, quantity over resolutions of nondeterminism, e.g. “probability of... is at least 0.95 for all possible schedulers”

- **Interesting issues**
  - no (simple) counterexamples, focus is on quantitative results
  - adding costs/rewards
  - combining quantitative and exhaustive analysis
Quantitative properties

• Model properties (and requirements) inherently quantitative
  – quality of service: how reliable is my car’s Bluetooth network?
  – how efficient is my phone’s power management policy?
  – quantifications of trust, anonymity, ...

• How to generate counterexamples?
  – counterexample to CTL property $\neg EF$ error is a finite trace
  – what is a good counterexample for $P_{<0.01} [ F \text{ error } ]$?
  – some work on this [Han/Katoen, TACAS'07] [Aljazzar et al.]
  – can also generate best-/worst-case schedulers for MDPs

• So focus on quantitative properties...
Quantitative properties

• Consider a PCTL formula $P_{<p} [ F \text{ error } ]$
  – if the probability is unknown, how to choose the bound $p$?
  – PRISM allows properties of the form $P_{=}? [ F \text{ error } ]$
  – (when the $P$ is the outermost operator of the formula)
  – “what is the probability of reaching an error state?”
  – model checking is no harder: compute the values anyway
  – and for MDPs/PTAs: $P_{\text{min}=}? [ F \text{ error } ]$ and $P_{\text{max}=}? [ F \text{ error } ]$

• Experiments: ranges of model/property parameters
  – e.g. $P_{=}? [ F^{\leq T} \text{ error } ]$ for $T=1..100$, $N=1..5$
  – identify patterns, trends, anomalies in quantitative results
  – investigate trade-offs, e.g. between...
Some real examples

- **NAND multiplexing system**
  - \( P_{\geq?} \left[ F \frac{\text{err/total}}{0.1} \right] \)
  - “what is the probability that 10% of the NAND gate outputs are erroneous?”

- **FireWire communication protocol**
  - \( z \cdot P_{\min=?} \left[ F (z \leq t) \& (\text{done}_1 \mid \text{done}_2) \right] \)
  - “what is the minimum probability that a leader node has been elected before the clock reaches \( t \)”

- **Security: EGL contract signing protocol**
  - \( P_{\geq?} \left[ F (\text{pairs}_a=0 \& \text{pairs}_b>0) \right] \)
  - “what is the probability that the party B gains an unfair advantage during the execution of the protocol?”
Optimum probability of leader election by time $T$ for various coin biases

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
Costs and rewards

• Augment models with rewards (or, conversely, costs)
  – real-valued quantities assigned to states and/or transitions
  – no distinction between rewards (“good”) and costs (“bad”)
  – simple but flexible, many possible interpretations

• Some examples:
  – elapsed time, power consumption, size of message queue, number of messages successfully delivered, net profit, …

• Analyse expected value of these costs/rewards
  – instantaneous or cumulative interpretation, e.g.:
    – $R_{=?} \left[ I=t \right]$ e.g. “expected message queue size at time t?”
    – $R_{=?} \left[ F \end \right]$ e.g. “expected time for protocol termination?”
    – $R_{\max=?} \left[ C\leq2 \right]$ e.g. “maximum expected power consumption during the first 2 hours that the system is in operation?”
Quantitative and exhaustive analysis

• Combining “quantitative” and “exhaustive” aspects
  – analysis of best/worst-case scenarios

• All possible resolutions of nondeterminism (MDPs)
  – $P_{\text{min}=?} \left[ !\text{end2} U \text{end1} \right]$ – “minimum probability of process 1 finishing before process 2, for any scheduling of processes?”

• Computing values for a range of states
  – $P=? \left[ F^{\leq t} \text{reply\_count}=k \{\text{init}\}\{\text{min}\} \right]$ – “what is the minimum probability, from any initial configuration, that the sender has received $k$ acknowledgements within $t$ clock-ticks?”
  – $P=? \left[ F^{\leq t} \text{elected}\{\text{tokens}\leq k}\{\text{min}\} \right]$ – “minimum probability of the leader election algorithm completing within $t$ steps from any state where there are at most $k$ tokens”
Firewire:
Maximum expected time for leader election for various coin biases

Bluetooth:
Distribution of expected time for two replies to be received, over all possible initial configurations of sender/receiver (1.7x10^10 states)

Self-stabilisation:
Worst-case expected number of steps to stabilise for initial configurations with K tokens amongst N processes
Probabilistic model checking

- Significant overlap between implementation of probabilistic model checking for DTMCs, CTMCs, MDPs:
  - graph-based algorithms on underlying transition system
    - reachability, qualitative probabilistic reachability
  - numerical computation – calculation of probabilities, rewards
    - usually, linear equation systems or linear optimisation problems
    - typically use iterative methods, e.g. Gauss–Seidel, value iteration
  - also: simulation-based sampling for approximate analysis
  - successful development of efficient symbolic implementations
    - see for example PRISM's MTBDD based engines

- For PTAs:
  - probabilistic extensions of: region graph approach, forwards/backwards symbolic reachability, digital clocks
Tool support

• **Tool support provides:**
  – high-level languages/formalisms, automation of verification and experiments, efficient (e.g. symbolic) implementations, discrete event simulators for debugging/approximations

• **PRISM:** symbolic model checking of DTMCs, CTMCs, MDPs
  – Kwiatkowska, Norman, Parker, ... at Birmingham/Oxford
  – indirect support for PTAs (direct translation using digital clocks, combine with Kronos/prototype reachability implementations)

• **ETMCC/MRMC:** DTMCs, CTMCs + reward extensions

• **LiQuor:** LTL verification for MDPs (Probmela language)

• **RAPTURE:** prototype for abstraction/refinement of MDPs

• And more: **APMC, Ymer, VESTA, APNN–Toolbox, SMART, CADP, Möbius**
Current research directions

• **State space explosion problem**
  – scalability is always an important issue
  – as for non-probabilistic (and other) verification techniques
  – current work addressing this includes...

• **Symmetry reduction**
  – exploitation of multiple identical components

• **Compositionality**
  – combining analysis results for individual components

• **Abstraction/refinement**
  – reduction of large/infinite systems (e.g. software verification)

• **And others:**
  – e.g. symbolic (BDD-based) implementations, partial order reduction, ...
Symmetry reduction

- Fully symmetric systems (N identical components)
  - quite common (e.g. symmetric communication protocols)
  - potentially exponential reduction in state space
  - resulting quotient model = strong probabilistic bisimulation

- Two approaches
  - counter abstraction – language level translation of model
    [Donaldson/Miller], GRIP
  - symbolic (MTBDD) implementation – reduction of full model to
    quotient based on state sorting [KNP, CAV'06]

- Also: algorithms to compute (full) bisimulation reduction
  - including symbolic (MTBDD) implementations [Derisavi]
Compositionality

• How to apply to probabilistic model checking?
  – parallel composition of components $M_1 || M_2$ is nondeterministic
  – analysis of $M_1 || M_2$ is of the form “for any (history dependent) scheduler, the probability of $M_1 || M_2$ satisfying $\phi$ is...”
  – introduces causal dependencies between components

• Possible solutions include:
  – partial information schedulers [de Alfaro/Henzinger/...]
  – restricted class of schedulers, e.g. based on token passing, partial information [Cheung]
  – multi-objective criteria for MDPs [Etessami et al., TACAS'07]

• What about quantitative results and compositionality?
Abstraction and Refinement

- Construct abstract model with unimportant info discarded

- What form does the abstract model take?
  - abstract model is “more nondeterministic”
  - DTMCs/MDPs
    - MDPs [D'Argenio et al.]
    - DTMCs + intervals [Fecher et al.]
    - 2-player stochastic games [KNP, QEST'06]
  - CTMCs → CTMDPs [Katoen et al.]

- How to generate/refine abstractions?
  - predicate abstraction? [Wachter et al.]
  - counterexample guided refinement?
Summary

• Probabilistic model checking
  – automated verification of exact quantitative results for a wide range of models and properties
  – efficient implementations and tool support available
  – successfully deployed in wide range of application domains
  – many more challenges remain...

• For more information, see the PRISM web page
  – www.prismmodelchecker.org
  – case studies, related publications, lectures, talks, tutorials, links, tool download/documentation