Probabilistic model checking: theory and practice

Gethin Norman

SICSA Modelling & Abstraction meeting

University of Stirling, January 2010
Why probability?

• many systems we want to verify are inherently probabilistic

Randomisation, e.g. in distributed coordination algorithms

• random delays/back-off in Bluetooth, CSMA/CD, WLAN, ...
• random IP address selection in Zeroconf/Bonjour, ...
• randomised algorithms for anonymity, contract signing, ...

Uncertainty, e.g. communication failures/delays

• prevalence of wireless communication, low-power devices

Need formal techniques for quantitative guarantees of:

• safety, reliability, performance, dependability, resource usage, security, privacy, trust, anonymity, fairness, ...
Probabilistic models

Probabilistic specifications

Probabilistic model checking

Combating the state space explosion problem
  • quantitative abstraction refinement framework
  • quantitative assume guarantee reasoning

Current and future research directions
Features:

- discrete state space
- discrete time-steps
- discrete transition probabilities

Discrete-time Markov chains (DTMCs)

Graphical representation of the DTMC with states $s_0$, $s_1$, $s_2$, and $s_3$. The transition probabilities are as follows:

- From $s_0$ to $s_1$: 0.98
- From $s_0$ to $s_2$: 0.01
- From $s_0$ to $s_3$: 1
- From $s_1$ to $s_0$: 1
- From $s_1$ to $s_2$: 0.01
- From $s_1$ to $s_3$: 0.98
- From $s_2$ to $s_0$: 1
- From $s_2$ to $s_1$: 0.01
- From $s_2$ to $s_3$: 1
- From $s_3$ to $s_0$: 0.01
- From $s_3$ to $s_1$: 0.98
- From $s_3$ to $s_2$: 1
Discrete-time Markov chains (DTMCs)

Features:
- discrete state space
- discrete time-steps
- discrete transition probabilities

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

Case studies:
- probabilistic contract signing
- leader election/self-stabilisation protocols
- nanotechnology (NAND multiplexing)
Continuous–time Markov chains (CTMCs)

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

**Transitions labelled with rates**
- parameters of the exponential distribution
- give probability the transition is triggered before $t$ time units have elapsed
  - i.e. for transition with rate $\lambda$ probability equals $1 - \exp(-\lambda \times t)$
- race condition: if more than one transition from a state then first transition triggered determines next state
Continuous–time Markov chains (CTMCs)

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

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

Case studies:
- dynamic power management schemes
- queueing and manufacturing systems
- groupware systems
- biological pathways, molecular reactions, …
Markov decision processes (MDPs)

Features:
- discrete state space, time-steps
- probability and nondeterminism
- nondeterministic choice between multiple discrete transition probability distributions
Markov decision processes (MDPs)

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

An adversary (aka scheduler or policy) is a resolution of the nondeterminism in the MDP
- under a fixed adversary the behaviour is fully probabilistic (a DTMC)
- to reasoning about best or worst case behaviour we consider the minimum or maximum values over all adversaries
  - e.g. the minimum probability of terminating within $K$ rounds
  - e.g. the maximum probability of an error occurring
Markov decision processes (MDPs)

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

Well suited to modelling:
- general parallel composition of components, e.g. distributed algorithms
- environmental factors, e.g. attacker’s behaviour against security protocols
- under-specification, e.g. unknown parameters

Case studies:
- randomised algorithms for byzantine agreement, consensus, …
- security protocols: anonymity, fair exchange, pin cracking, …
- power management, …
Probabilistic timed automata (PTAs)

Features:
- probability, nondeterminism and real-time
- extends MDPs with real-valued clocks
- alternatively, extends timed automata with discrete probability distributions
Features:

- probability, nondeterminism and real-time
- extends MDPs with real-valued clocks
- alternatively, extends timed automata with discrete probability distributions

Assume we have a finite set of clocks $X$ and locations $L$

- a clock valuation is real-valued vector $v \in \mathbb{R}^X$ over the clocks
- a clock constraint is a convex sets of clock valuations $\zeta \subseteq \mathbb{R}^X$
  - say a clock valuation $v$ satisfies a clock constraint $\zeta$ when $v \in \zeta$
- invariants: clock constraints associated with locations of the PTA
- enabling conditions: clock constraints associated with transitions of the PTA
- a state of the PTA is a location-clock valuation pair $(l,v)$ such that $v$ satisfies the invariant of the location $l$
Features:
- probability, nondeterminism and real-time
- extends MDPs with real-valued clocks
- alternatively, extends timed automata with discrete probability distributions

Semantics: in state \((l,v)\) the choice of the time \(t\) that elapses and action \(a\) performed is probabilistic under the requirement:

1. the invariant of location \(l\) is continuously satisfied during time \(t\)
2. the enabling condition of action \(a\) is enabled after \(t\) time units have elapsed

When a transition is taken there is a probabilistic choice over both the target location and the clocks that are reset.
**Probabilistic timed automata (PTAs)**

**Features:**
- probability, nondeterminism and real-time
- extends MDPs with real-valued clocks
- alternatively, extends timed automata with discrete probability distributions

**Well suited to modelling:**
- communication/network protocols featuring randomisation
  - e.g. waiting times, backoff schemes or address selection
- security protocols with both timing and probabilistic characteristics

**Case studies:**
- FireWire root contention, Zeroconf dynamic configuration protocol
- network protocols for collision avoidance/detection: CSMA/CD, WiFi, ZigBee,…
- Gossip and broadcast protocols, …
Probabilistic models

Probabilistic specifications

Probabilistic model checking

Combating the state space explosion problem
  - quantitative abstraction refinement framework
  - quantitative assume guarantee reasoning

Current and future research directions
Probabilistic extensions of temporal logic (CTL/LTL)

- essentially (time-bounded) probabilistic reachability
  - \( P_{<0.1}[F \text{ error}] \) “the probability of reaching an error state is less than 0.1”
  - \( P_{<0.1}[F^{\leq T} \text{ error}] \) “the probability of reaching an error state within time \( T \) is less than 0.1”
- for MDPs and PTAs, quantity over all resolutions of nondeterminism
  - “the probability of reaching an error state is less than 0.1 for all adversaries”
However properties (and requirements) inherently quantitative

- want to know the quality of service, quantify the trust, anonymity, …
  - e.g. how reliable is my car’s Bluetooth?
  - e.g. how efficient is the phone’s power management?

Therefore also allow for quantitative queries

- $P_=?[F \text{ error}]$ “what is the probability of an error occurring?”
- for MDPs (and PTAs) consider best or worst case values
  - $P_{\text{min}}=?[F \text{ error}]$ “what is the minimum probability of an error occurring”

Model checking is no harder: compute the values anyway
Reward structures

Augment models with reward (or cost) structures

- real-valued quantities assigned to states and/or transitions
- state rewards can be cumulative (dependent on the time spent in a state)
- simple but flexible approach with many possible interpretations
  - e.g. elapsed time, power consumption, size of message queue, number of successfully delivered messages, …

Analyse the expected reward (or cost) value

- \( R_{=? \to I^T} \) “expected message queue size at time \( T \)”
- \( R_{\min=? \to F \text{ “finished”}} \) “minimum expected time for the protocol to terminate?”
- \( R_{\max=? \to C \leq 2} \) “maximum expected power consumption during the first 2 hours?”
- \( R_{=? \to S} \) “the long run average concentration of a complex”
Experiments: range of property/model parameters

- useful for identify patterns, trends and anomalies
- investigate trade-offs, e.g. between performance and reliability
Experiments: range of property/model parameters

- useful for identifying patterns, trends, and anomalies
- investigating trade-offs, e.g., between performance and reliability

FireWire: analyse for a range coin bias and time-bound values

\[ P_{\text{min}} = ? \quad [F \leq T \text{ “elected”}] \]
“what is the minimum probability a leader is elected by time \( T \)?”

\[ R_{\text{max}} = ? \quad [F \text{ “elected”}] \]
“what is the maximum expected time to elect a leader?”

demonstrates that performance is improved with a biased coin
Probabilistic models

Probabilistic specifications

**Probabilistic model checking**

Combating the state space explosion problem

- quantitative abstraction refinement framework
- quantitative assume guarantee reasoning

Current and future research directions
Significant overlap between implementations for model checking of DTMCs, CTMCs and MDPs

**Graph based algorithms**
- performed on the underlying transition system
- reachability and qualitative (probability 0 or 1) properties

**Numerical computation**
- calculation of probabilities or rewards values
- usually, linear equation systems or linear optimisation problem
- typically use iterative methods, e.g. Gauss-Seidel, value iteration

Also **simulation-based sampling** for approximate analysis
For PTAs more complex (uncountable state space)

Developed a number of approaches for model checking PTAs (based on those developed for timed automata)

- the region graph [ARTS’99]
- forwards symbolic reachability [TCS’02]
- backwards symbolic reachability [IC’07]
- digital clocks [FMSD’06]

In each approach the model checking procedure has two steps:

1. an exploration of the transitions of the PTA to construct a finite state MDP
2. model checking the constructed MDP to infer properties of the PTA
State space explosion problem

- as for non-probabilistic verification techniques
- scalability is the main issue to verifying complex/real-life systems

A number of approaches addressing this problem including:

- abstraction refinement
- assume-guarantee reasoning
- many others: symmetry reduction, partial order reduction, bisimulation, symbolic (BDD-based) implementations, …
Probabilistic models

Probabilistic specifications

Probabilistic model checking

**Combating the state space explosion problem**
- quantitative abstraction refinement framework
- quantitative assume guarantee reasoning

Current and future research directions
Abstraction

- essential for verification of large/infinite-state systems
- eliminate details irrelevant to the property of interest
- yields smaller/finite model which is easier/feasible to verify
- loss of precision: verification can return ‘don’t know’

Abstraction refinement

- automatic process for constructing sufficiently precise abstractions
- start with simple coarse abstraction
- when property cannot be validated or refuted use information from verification process to refine the abstraction and repeat the analysis
Abstraction increases the degree of nondeterminism

Key idea: separate the nondeterminism caused by abstraction

- abstract DTMCs to MDPs (since only one form on nondeterminism)
- however abstract MDPs to stochastic two player games (since two separate forms of nondeterminism: from the MDP and from the abstraction process)

Analysis of abstraction yields lower and upper bounds together with adversaries that achieve the bounds

$$\text{Prob}_{\min}(\phi)$$  $$\text{Prob}_{\max}(\phi)$$

MDP: gives a quantitative measure of the utility or precision of the abstraction
Difference between bounds gives a *quantitative measure* of the abstraction’s *precision*

- if the difference (or *error*) is too great, then *refine* the abstraction
- use bounds and adversaries which achieve the bounds to guide refinement
Difference between bounds gives a **quantitative measure of the abstraction’s precision**

- if the difference (or error) is too great, then refine the abstraction
- use bounds and adversaries which achieve the bounds to guide refinement

```
Initial abstraction
```

```
Abstract model
```

```
New abstraction
```

```
界限 & 威胁
```

```
返回界限
```

```
abstract
```

```
model check
```

```
[error≥ε]
```

```
refine
```

```
[error<ε]
```
Quantitative abstraction refinement

Difference between bounds gives a **quantitative measure of the abstraction’s precision**

- if the difference (or *error*) is too great, then **refine** the abstraction
- use bounds and adversaries which achieve the bounds to guide refinement

Initial abstraction $\xrightarrow{\text{abstract}}$ Abstract model $\xrightarrow{\text{model check}}$ Bounds & adversaries $\xrightarrow{[\text{error}\geq \varepsilon]}$ New abstraction $\xrightarrow{\text{refine}}$

[error$\geq \varepsilon$]

Guaranteed to converge for finite-state models

Guaranteed to converge for infinite-state models with finite bisimulation quotients

Return bounds $\xrightarrow{[\text{error}<\varepsilon]}$
Explicit-state prototype [QEST’06,FMSD’10]
- demonstrates the feasibility of the framework

Verification of software [VMCAI’09]
- predicate abstraction and SAT-based techniques
- successfully applied to Linux network utilities (1KLOC non-trivial C)
- probability is used to model loss of packets
- nondeterminism is used to model kernel calls

Verification of PTAs [FORMATS’09]
- use DBMs (difference bound matrices) to symbolically represent and manipulate infinite sets of clock values
- initial abstraction constructed via forwards symbolic reachability algorithm
- outperforms existing methods for verifying PTAs
Extend framework to **CTMCs**

- abstract model: **CTMDPs** (continuous time Markov decision processes)
- has been demonstrated that lower and upper bounds are generated
- question remains how to refine abstractions
  - current refinement schemes require simple (or memoryless) adversaries
  - however time-bounden properties for CTMDPs do not yield such simple adversaries

Extend framework to **priced PTAs and hybrid systems**

Improve **refinement schemes**, e.g. through counterexamples

Develop methods for abstraction construction

- language level construction
- use of **imprecise abstractions** to allow for faster model construction
Probabilistic models

Probabilistic specifications

Probabilistic model checking

**Combating the state space explosion problem**
- quantitative abstraction refinement framework
- quantitative assume guarantee reasoning

Current and future research directions
Assume guarantee reasoning for MDPs [TACAS’10]

- verify a system through the analysis of its subcomponents in isolation
- based on quantitative multi-objective model checking

Limited to the analysis of regular safety properties

- i.e. the set of bad prefixes (finite violations) form a regular language
- e.g. “at least one sensor is always operational”
  - finite violations: finite paths where in the final state no sensors are operational
- e.g. “at most two failures occur”
  - finite violations: finite paths with three failures

Key property: for an MDP terminating early cannot decrease the probability of satisfying a safety property
Given MDP $M$, (LTL) properties $\phi_1, \ldots, \phi_k$ and bounds $\sim r_1, \ldots, \sim r_k$

Quantitative multi-objective model checking problem:
check for the existence of an adversary under which the probability of satisfying $\phi_i$ meets the bound $\sim r_i$ for all $i \leq k$

- can be solved using linear programming [Etessami et al TACAS’07]
Assume-guarantee statements

Given safety properties $A$ and $G$ and bounds $\geq p_A$ and $\geq p_G$

Assume-guarantees statements for an MDP $M$ are of the form:

for all adversaries $\sigma$, if under $\sigma$ the probability of satisfying $A$ is $\geq p_A$, then under $\sigma$ the probability of satisfying $G$ is $\geq p_G$

• when statement is satisfied we write $(A)_{\geq p_A} M (G)_{\geq p_G}$

Statements verified through multi-objective model checking

• sufficient to show there does not exist an adversary such that the probability of satisfying $A$ is $\geq p_A$ and the probability of satisfying $\neg G$ is $> 1 - p_G$

Write $(true) M (G)_{\geq p_G}$ when there is no assumption

• equivalent to a standard probabilistic model checking query
• i.e. for all adversaries of $M$ the probability of satisfying $G$ is $\geq p_G$
Simple asymmetric rule

\[
\begin{align*}
\text{(true) } M_1 \ (A) & \geq p_A \\
(A) & \geq p_A \quad M_2 \ (G) & \geq p_G \\
\hline
\text{(true) } M_1 \ || \ M_2 \ (G) & \geq p_G
\end{align*}
\]

To verify that under all adversaries the probability of \( M_1 \ || \ M_2 \) satisfying \( G \) is \( \geq p_G \) it is sufficient to check:

- one (standard) probabilistic model checking query on the subcomponent \( M_1 \)
  - i.e. verify that under any adversary of \( M_1 \) the probability of satisfying \( A \) is \( \geq p_A \)
- one multi-objective query on the subcomponent \( M_2 \)
  - i.e. verify that for any adversary of \( M_2 \), if the probability of satisfying \( A \) is \( \geq p_A \), then the probability of satisfying \( G \) is \( \geq p_G \)
Generalised rules to allow for

- multiple assumptions on $M_1$, e.g. $(A_1)_{\geq p_1}, \ldots, (A_k)_{\geq p_k}$
- analysis of more complex systems, e.g. $M_1 \parallel \ldots \parallel M_k$
- making assumptions on $M_2$ when proving assumptions of $M_1$
- the independence caused through asynchronous composition

Has been implemented and successfully applied to a number of large case studies

- distributed consensus protocol and Zeroconf dynamic configuration protocol
- verified instances for which conventional probabilistic verification is infeasible

Future work includes

- expanding the range of rules/properties, e.g. rewards
- investigating techniques to generate assumptions, e.g. learning
Probabilistic models

Probabilistic specifications

Probabilistic model checking

Combating the state space explosion problem
  • quantitative abstraction refinement framework
  • quantitative assume guarantee reasoning

Current and future research directions
Extend range of properties that can be analysed

- oscillations in biological pathways
- conditional probabilities/expectations (e.g. entropy used in anonymity metrics)

Model checking against partial-information adversaries

- adversaries make choices based on a limited knowledge/view of the system
- vital for security analysis, e.g. attacker cannot decrypt certain messages

Cost-benefit analysis

- e.g. minimise power consumption subject to constraints on performance
- analysing the Pareto curve using multi-objective model checking

Parametric model checking

- analyse models where probabilities or rates are given as parameters

Counterexample generation and analysis, ...
Marta Kwiatkowska (Oxford University Computing Laboratory)
Dave Parker (Oxford University Computing Laboratory)
Mark Kattenbelt (Oxford University Computing Laboratory)
Jermey Sproston (Università di Torino)

...