Automated Game-theoretic Verification for Probabilistic Systems

Dave Parker
University of Birmingham

Joint work with:
Taolue Chen, Vojtěch Forejt, Marta Kwiatkowska, Aistis Simaitis

Dagstuhl, November 2012
Overview

- **Automatic verification (model checking) of systems with:**
  
  1. **Probabilistic behaviour**
     - unreliability, uncertainty, randomisation, ...
  
  2. **Other quantitative aspects**
     - time, costs (e.g. energy), rewards (e.g. profit), ...
  
  3. **Competitive/collaborative behaviour**
     - open systems, controller synthesis, ...

- **Focus:**
  - probabilistic model checking of stochastic multi-player games
  - scalable/efficient techniques/tools for modelling real systems

- **Applications:**
  - e.g. security protocols, algorithms for distributed consensus, sensor network co-ordination or energy management
This talk

• Probabilistic model checking

• Stochastic multi-player games (SMGs)

• Property specification: rPATL

• rPATL model checking

• Tool support: PRISM-games

• Case study: energy management in microgrids
Probabilistic model checking

- Probabilistic model
  - e.g. Markov chain, Markov decision process

- System
  - System requirements
  - System

- Probabilistic temporal logic specification
  - e.g. PCTL, CSL, LTL, ...

- Probabilistic model checker
  - e.g. PRISM

Result
- Quantitative results
- Witness/counter-example

P ≤ 0.01 \[ F \text{ crash} \]
Stochastic multi-player games

- Stochastic multi-player game (SMGs)
  - nondeterminism + multiple players + probability

- A (turn-based) SMG is a tuple \((\Pi, S, \langle S_i \rangle_{i \in \Pi}, A, \Delta, L)\):
  - \(\Pi\) is a set of \(n\) players
  - \(S\) is a (finite) set of states
  - \(\langle S_i \rangle_{i \in \Pi}\) is a partition of \(S\)
  - \(A\) is a set of action labels
  - \(\Delta : S \times A \rightarrow \text{Dist}(S)\) is a (partial) transition probability function
  - \(L : S \rightarrow 2^{AP}\) is a labelling with atomic propositions from \(AP\)

![Diagram of a SMG](image)
• **Strategy for player i:** resolves choices in $S_i$ states
  – based on execution history, i.e. $\sigma_i : (SA)^*S_i \rightarrow \text{Dist}(A)$
  – can be: deterministic (pure), randomised, memoryless, finite-memory, …
  – $\Sigma_i$ denotes the set of all strategies for player $i$

• **Strategy profile:** strategies for all players: $\sigma = (\sigma_1, \ldots, \sigma_n)$
  – induces a set of (infinite) paths from some start state $s$
  – a probability measure $\Pr^s_\sigma$ over these paths

• **Rewards (or costs)**
  – non-negative integers on states/transitions
  – e.g. elapsed time, energy consumption, number of packets lost, net profit, …
  – this talk: expected cumulated value of rewards
Property specification: rPATL

- New temporal logic rPATL:
  - reward probabilistic alternating temporal logic

- CTL, extended with:
  - coalition operator $\langle\langle C \rangle\rangle$ of ATL
  - probabilistic operator $P$ of PCTL
  - generalised (expected) reward operator $R$ from PRISM

- In short:
  - zero-sum, probabilistic reachability + expected total reward

- Example:
  - $\langle\langle\{1,2\}\rangle\rangle P_{<0.01} [ F_{\leq 10} \text{ error } ]$
  - “players 1 and 2 have a strategy to ensure that the probability of an error occurring within 10 steps is less than 0.01, regardless of the strategies of other players”
rPATL syntax/semantics

• Syntax:

\[ \phi ::= \top \mid a \mid \neg \phi \mid \phi \land \phi \mid \langle\langle C \rangle\rangle P_{\bowtie q}[\psi] \mid \langle\langle C \rangle\rangle R_{\bowtie x}^{r} [F^* \phi] \]

\[ \psi ::= X \phi \mid \phi U \phi \mid F \phi \mid G \phi \mid \phi U^{\leq k} \phi \mid F^{\leq k} \phi \mid G^{\leq k} \phi \]

• where:

- \( a \in \text{AP} \) is an atomic proposition, \( C \subseteq \Pi \) is a coalition of players,
  \( \bowtie \in \{\leq, <, >, \geq\} \), \( q \in [0, 1] \cap \mathbb{Q} \), \( x \in \mathbb{Q}_{\geq 0} \), \( k \in \mathbb{N} \)
- \( r \) is a reward structure and \( * \in \{0, \infty, c\} \) is a reward type

• Semantics:

• \( P \) operator: \( s \models \langle\langle C \rangle\rangle P_{\bowtie q}[\psi] \) iff:

  - “there exist strategies for players in coalition \( C \) such that, for all strategies of the other players, the probability of path formula \( \psi \) being true from state \( s \) satisfies \( \bowtie q \)”
rPATL semantics (rewards)

- **R operator:** $s \models \langle\langle C \rangle \rangle R^r \bowtie^x [F^\ast \phi]$ iff:
  - “there exist strategies for players in coalition $C$ such that, for all strategies of the other players, the expected cumulated reward $r$ to reach a $\phi$–state (type $\ast$) satisfies $\bowtie x$”

- **3 reward types** $\ast \in \{\infty, c, 0\}$
  - defining reward if a $\phi$–state is never reached
  - reward is: infinite ($\ast = \infty$), cumulated sum ($\ast = c$), zero ($\ast = 0$)
  - $\infty$: e.g. expected time for algorithm execution
  - $c$: e.g. expected resource usage (energy, messages sent, ...)
  - $0$: e.g. reward incentive awarded on algorithm completion

- **Note:** $F^0$ operator needs finite–memory strategies
  - (for $P$ and other $R$ operators, pure memoryless strat.s suffice)
Model checking rPATL

- **Main task:** checking individual P and R operators
  - reduction to solution of zero-sum stochastic 2-player game
  - (probabilistic reachability + expected total reward)
  - e.g. $\langle \langle C \rangle \rangle P_{\geq q}[\psi] \iff \sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} \Pr_{s,\sigma_1,\sigma_2}(\psi) \geq q$
  - complexity: $\text{NP} \cap \text{coNP}$ (without any $R[F^0]$ operators)
  - complexity for full logic: $\text{NEXP} \cap \text{coNEXP}$ (due to $R[F^0]$ op.)

- **In practice though:**
  - (usual approach taken in probabilistic model checking tools)
  - evaluation of numerical **fixed points** (“value iteration”)
  - and more: graph-algorithms, sequences of fixed points, ...

- **See:** [TACAS’12], [CONCUR’12]
rPATL extensions

- **Quantitative (numerical) properties:**
  - numerical rather than boolean-valued queries

- **Example:**
  - \( \langle \langle \{1,2\} \rangle \rangle \ P_{\text{max}} = ? [ F \ \text{error} ] \)
  - “what is the maximum probability of reaching an error state that players 1 and 2 can guarantee?”
  - i.e. \( \sup_{\sigma_1 \in \Sigma_1} \ inf_{\sigma_2 \in \Sigma_2} \ Pr_{s, \sigma_1, \sigma_2} (F \ \text{error}) \)

- **Other extensions:**
  - rPATL* (i.e. support for LTL formulae in P operator)
  - reward-bounded operators
  - exact probability/reward bounds
Tool support: PRISM-games

- **Model checker for stochastic multi-player games**
  - **PRISM-games**: extension of PRISM model checker
  - using new explicit-state model checking engine

- **Features:**
  - modelling language for SMGs
  - rPATL model checking
  - strategy synthesis and analysis
  - GUI: model editor, simulator, graph-plotting, strategies, ...

- **Availability**
  - free, open source (GPL)
  - benchmark suite
Tool support: PRISM-games

- **Extended PRISM modelling language for SMGs**
  - guarded command language
  - probabilistic extension of (simplified) Reactive Modules
  - finite data types, parallel composition, proc. algebra op.s, ...

- **Strategy synthesis and analysis**
  - synthesise strategy for an rPATL query
  - export, simulate, analyse (verify second rPATL property on)

- **Evaluated on several case studies:**
  - team formation protocol [CLIMA’11]
  - futures market investor model [McIver & Morgan]
  - collective decision making for sensor networks [TACAS’12]
  - energy management in microgrids [TACAS’12]
Energy management in microgrids

- **Microgrid**: proposed model for future energy markets
  - localised energy management

- **Neighbourhoods** use and store electricity generated from local sources
  - wind, solar, ...

- **Needs**: demand-side management
  - active management of demand by users
  - to avoid peaks
Microgrid demand-side management

- **Demand-side management algorithm** [Hildmann/Saffre’11]
  - N households, connected to a distribution manager
  - households submit loads for execution
  - execution cost/step = number of currently running loads

- **Simple algorithm:**
  - upon load generation, if cost is below an agreed limit $c_{lim}$, execute it, otherwise only execute with probability $P_{start}$

- **Analysis of** [Hildmann/Saffre’11]
  - load submission probability: daily demand curve
  - load duration: random, between 1 and D steps
  - define household value as $V = \text{loads-executing}/\text{execution-cost}$
  - simulation-based analysis shows reduction in peak demand and total energy cost reduced, with good expected value $V$
  - (if all households stick to algorithm)
Microgrid demand-side management

• The model
  – SMG with N players (one per household)
  – analyse 3-day period, using piecewise approximation of daily demand curve
  – fix parameters D=4, $c_{\text{lim}}=1.5$
  – add rewards structure for value $V$

• Built/analysed models
  – for $N=2,\ldots,7$ households

• Step 1: assume all households follow algorithm of [HS’11] (MDP)
  – obtain optimal value for $P_{\text{start}}$

• Step 2: introduce competitive behaviour (SMG)
  – allow coalition $C$ of households to deviate from algorithm

<table>
<thead>
<tr>
<th>N</th>
<th>States</th>
<th>Transitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>743,904</td>
<td>2,145,120</td>
</tr>
<tr>
<td>6</td>
<td>2,384,369</td>
<td>7,260,756</td>
</tr>
<tr>
<td>7</td>
<td>6,241,312</td>
<td>19,678,246</td>
</tr>
</tbody>
</table>
Results: Competitive behaviour

- **Expected total value V per household**
  - in rPATL: $\langle \langle C \rangle \rangle R_{C_{\text{max}}}^{r} [F_{0} \text{ time}=\text{max time}] / |C|$
  - where $r_{C}$ is combined rewards for coalition $C$

![Graph showing competitive behavior]

- All follow alg.
- No use of alg.
- Deviations of varying size
Results: Competitive behaviour

- **Algorithm fix: simple punishment mechanism**
  - distribution manager can cancel some loads exceeding $c_{\text{lim}}$
Conclusions

- game-theoretic verification for probabilistic systems
- modelled as stochastic multi–player games
- new temporal logic rPATL for property specification
- rPATL model checking algorithm based on num. fixed points
- model checker PRISM–games
- case studies: energy management for microgrid

Future work

- more realistic classes of strategy, e.g. partial observation, ...
- further objectives, e.g. multiple objectives, Nash equilibria, ...
- more application areas: security, randomised algorithms, ...