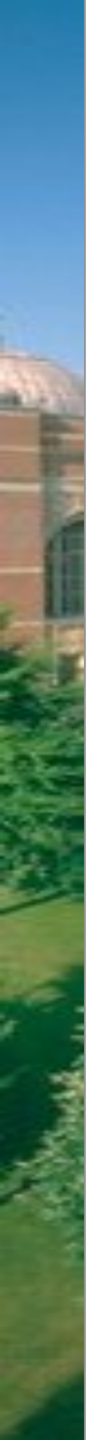


Automated Game-theoretic Verification for Probabilistic Systems

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

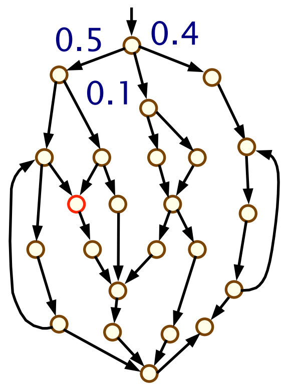
This talk

- 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

System

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



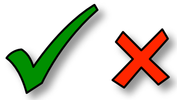
System require-ments

P$_{\leq 0.01}$ [F crash]

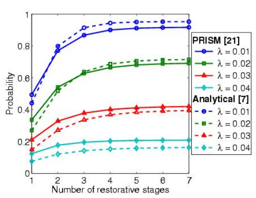
Probabilistic temporal logic specification
e.g. PCTL, CSL, LTL, …

Probabilistic model checker
e.g. PRISM

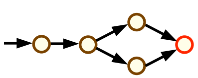
Result



Quantitative results

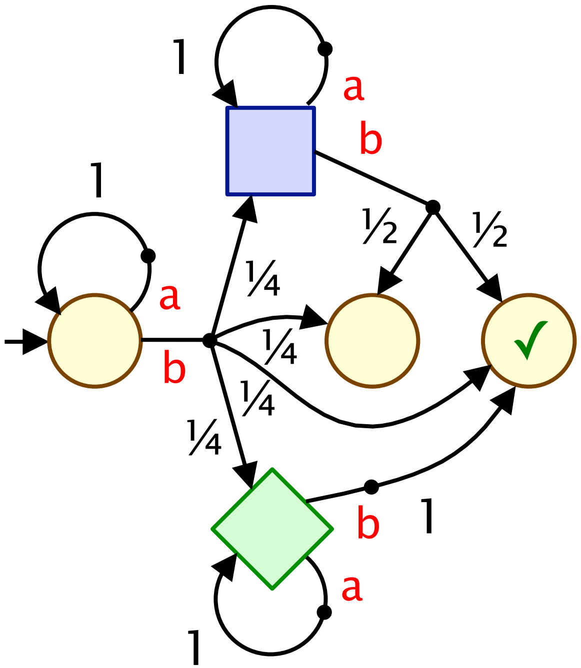


Witness / counter-example



Stochastic multi-player games

- Stochastic multi-player game (SMGs)
  - nondeterminism + multiple players + probability
- A (turn-based) SMG is a tuple $(\Pi, S, \langle S_i \rangle_{i \in \Pi}, A, \Delta, L)$:
  - Π 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
  - $\Delta : S \times A \rightarrow Dist(S)$ is a (partial) transition probability function
  - $L : S \rightarrow 2^{AP}$ is a labelling with atomic propositions from AP



Strategies, probabilities & rewards

- Strategy for player i: resolves choices in S_i states
  - based on execution history, i.e. $\sigma_i : (SA)^*S_i \rightarrow Dist(A)$
  - can be: deterministic (pure), randomised, memoryless, finite-memory, …
  - Σ_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^σ 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:
  - $\langle\langle \{1,2\} \rangle\rangle\ P_{<0.01}\ [\ F^{\leq 10}\ error\ ]$
  - "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:

$$\phi ::= \top \mid a \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle C\rangle\rangle P_{\bowtie q}[\psi] \mid \langle\langle C\rangle\rangle R^{r}_{\bowtie x}\ [F^{\star}\phi]$$

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

- where:
  - $a \in AP$ 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 $\star \in \{0,\infty,c\}$ is a reward type

- Semantics:

- P operator: $s \vDash \langle\langle C\rangle\rangle P_{\bowtie q}[\psi]$ iff:
  - "there exist strategies for players in coalition C such that, for all strategies of the other players, the probability of path formula ψ being true from state s satisfies $\bowtie q$"

rPATL semantics (rewards)

- R operator: $s \models \langle\langle C \rangle\rangle R^{r}_{\bowtie x}\ [F^{\star}\phi]$ 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 ϕ-state (type $\star$) satisfies $\bowtie x$"
- 3 reward types $\star \in \{\infty, c, 0\}$
  - defining reward if a ϕ-state is never reached
  - reward is: infinite ($\star=\infty$), cumulated sum ($\star=c$), zero ($\star=0$)
  - ∞: 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. $\langle\langle C\rangle\rangle P_{\geq q}[\psi] \;\Leftrightarrow\; \sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}(\psi) \geq q$
  - complexity: NP ∩ coNP (without any $R[F^0]$ operators)
  - complexity for full logic: NEXP ∩ 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:
  - $\langle\langle\{1,2\}\rangle\rangle\ P_{max=?}$ [F error]
  - "what is the maximum probability of reaching an error state that players 1 and 2 can guarantee?"
  - i.e. $\sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}(\text{F error})$
- 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, …

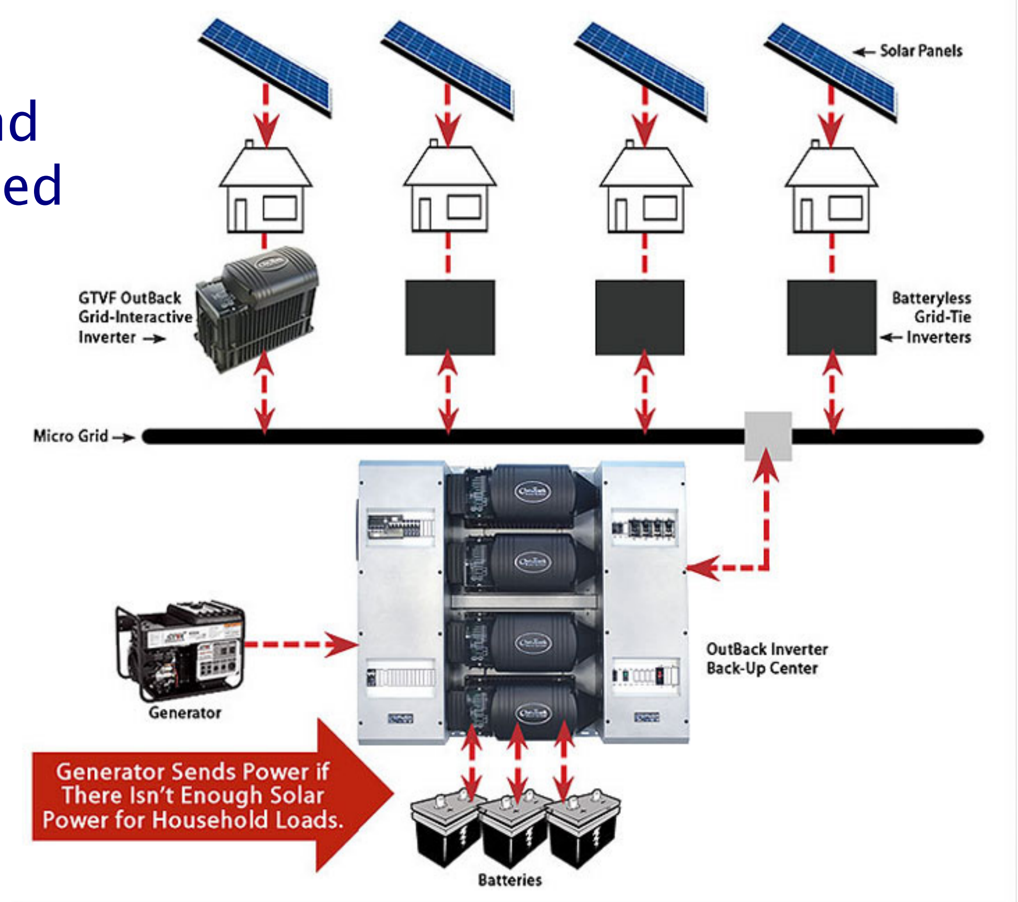
- Availability
  - download: http://www.prismmodelchecker.org/games/
  - free, open source (GPL)
  - benchmark suite

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

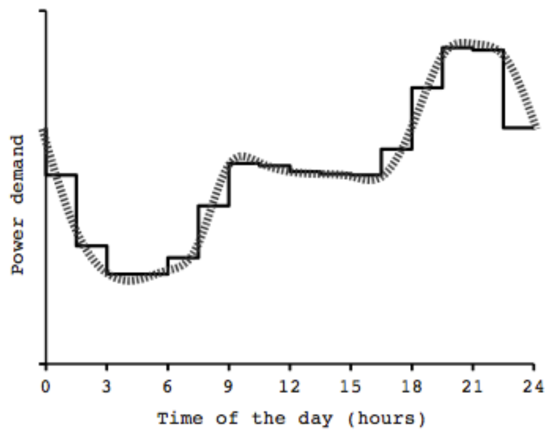


Microgrid demand-side management

- 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=loads_executing/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_{lim}=1.5
  - add rewards structure for value V



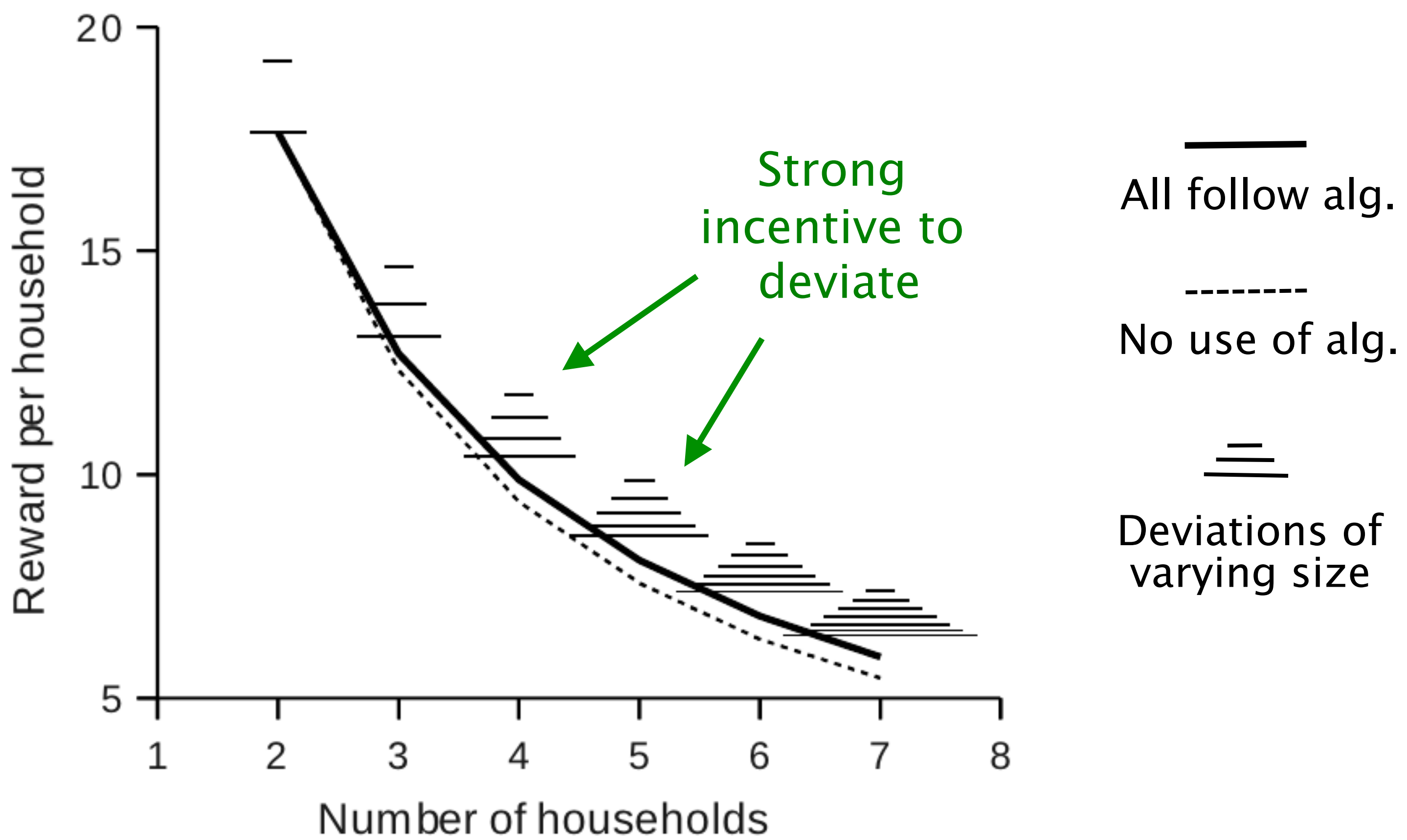
- Built/analysed models
  - for N=2,…,7 households

N	States	Transitions
5	743,904	2,145,120
6	2,384,369	7,260,756
7	6,241,312	19,678,246

- Step 1: assume all households follow algorithm of [HS'11] (MDP)
  - obtain optimal value for P_{start}
- Step 2: introduce competitive behaviour (SMG)
  - allow coalition C of households to deviate from algorithm

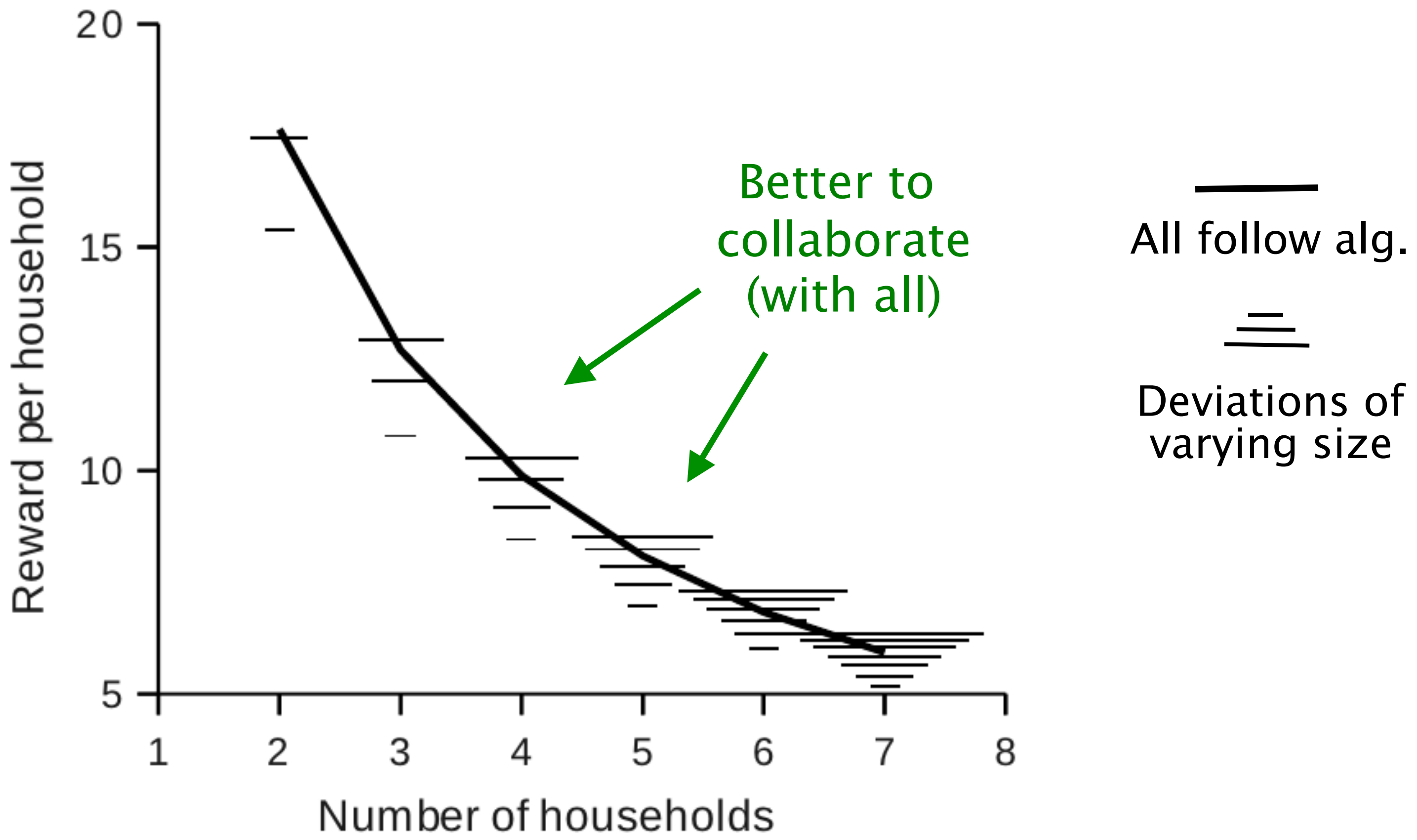
Results: Competitive behaviour

- Expected total value V per household
  - in rPATL: $\langle\langle C \rangle\rangle R^{r_C}_{max=?} [F^0 \text{ time=max time}] \,/\, |C|$
  - where r_C is combined rewards for coalition C



Results: Competitive behaviour

- Algorithm fix: simple punishment mechanism
  - distribution manager can cancel some loads exceeding c_{lim}



Conclusions

- 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, …
- PRISM-games: http://www.prismmodelchecker.org/games/