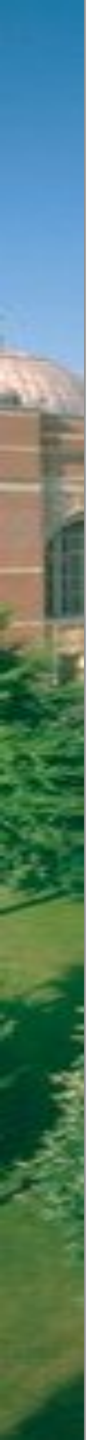


Automated Game-theoretic Verification for Probabilistic Systems

Dave Parker

University of Birmingham

Joint work with:

Taolue Chen, Vojtěch Forejt, Marta Kwiatkowska, Aistis Simaitis

Imperial College London, December 2012

Verifying stochastic systems

- Quantitative verification
  - probability, time, costs/rewards, …
  - in particular: systems with stochastic behaviour
  - e.g. due to unreliability, uncertainty, randomisation, …
  - often: subtle interplay between probability/nondeterminism
- Automated verification
  - probabilistic model checking
  - efficiency and scalable algorithms/techniques
  - tool support: PRISM model checker
- Practical applications
  - wireless communication protocols, security protocols, systems biology, DNA computing, robotic planning, …

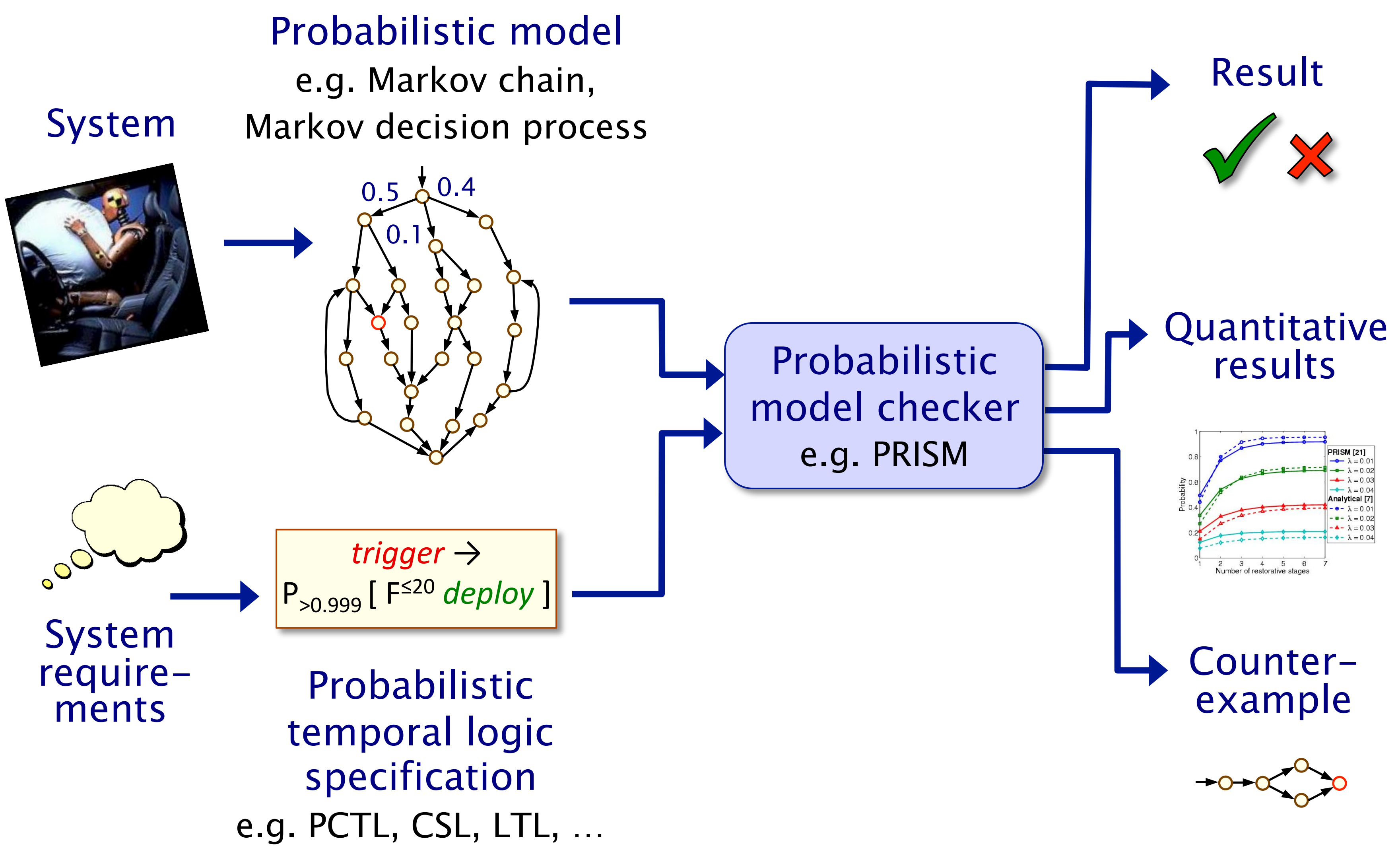
Competitive/collaborative behaviour

- Open systems
  - need to account for the behaviour of system components not under our control, possibly with differing/opposing goals
  - giving rise to competitive/collaborative behaviour
- Many occurrences in practice
  - e.g. security protocols, algorithms for distributed consensus, energy management or sensor network co-ordination
- Natural to adopt a game-theoretic view
  - widely used in computer science, economics, …
- This talk
  - verifying systems with stochastic and game-theoretic aspects
  - stochastic multi-player games
  - temporal logic, model checking, tool support, case studies

Overview

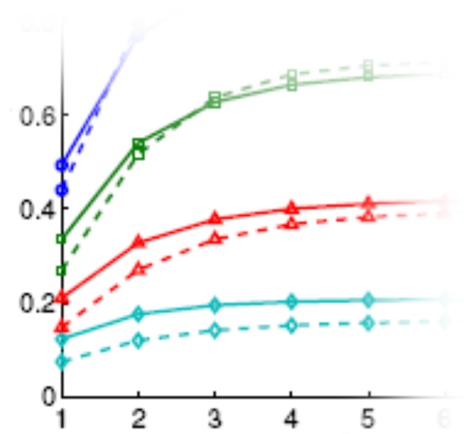
- Probabilistic model checking
- Stochastic multi-player games (SMGs)
  - strategies, probabilities, rewards
- Property specification: rPATL
  - syntax, semantics, subtleties
- rPATL model checking
  - algorithms, tool support
- Case study: Energy management in microgrids

Probabilistic model checking



Probabilistic model checking

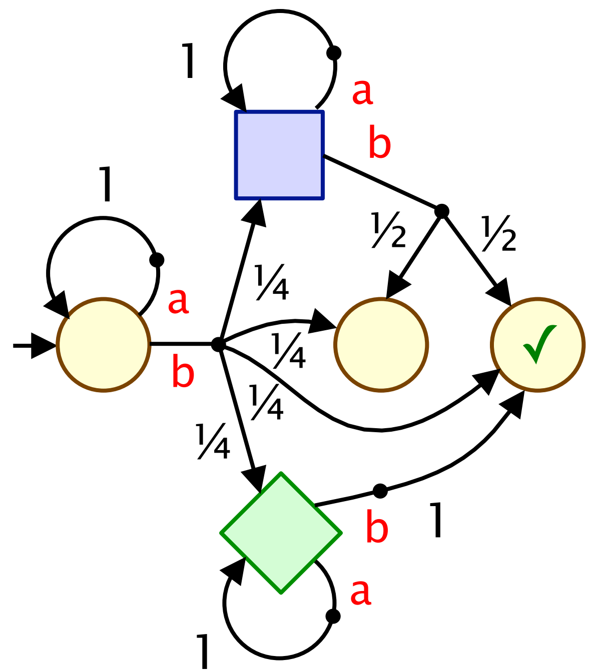
- Usually focus on quantitative (numerical) properties:
  - $P_{=?}$ [$F^{\leq 20}$ deploy] – "what is the probability of the airbag deploying within 20ms?"



- Then analyse trends in quantitative properties as system parameters vary
  - looking for flaws, anomalies, …
- Unlike (non-probabilistic) model checking
  - often investigate effect of (known) failures, rather than identifying existence of (unknown) bugs
- Strength: combines numerical and exhaustive aspects
  - "worst-case (maximum) probability of the airbag failing to deploy within 20ms, *from any possible* crash scenario"
  - "worst-case (maximum) expected algorithm execution time *for any possible scheduling* of system components"

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
  - $\Delta : S \times A \to Dist(S)$ is a (partial) transition probability function
  - $L : S \to 2^{AP}$ 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:
  - $\langle\langle \{1,3\} \rangle\rangle \, P_{<0.01} [\, F^{\leq 10} \, \mathrm{error} \,]$
  - "players 1 and 3 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:

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

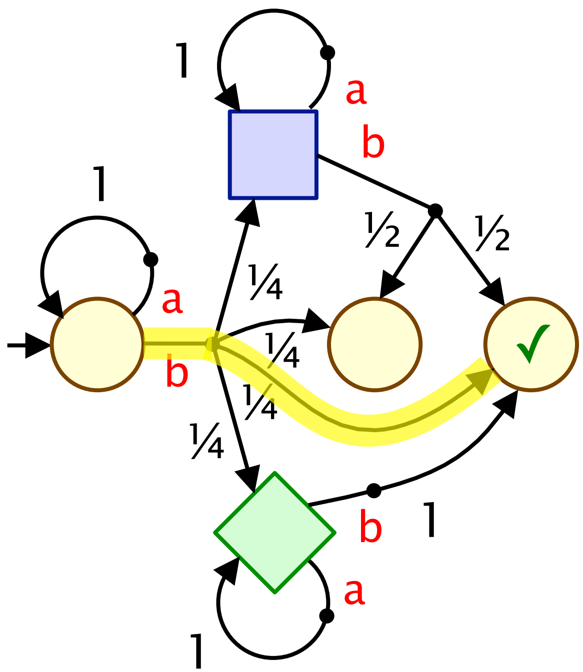
$$\psi ::= X\,\phi \mid \phi\ U\ \phi \mid F\,\phi \mid G\,\phi \mid \phi\ U^{\leq k}\ \phi \mid F^{\leq k}\,\phi \mid G^{\leq k}\,\phi$$

- 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"

Examples



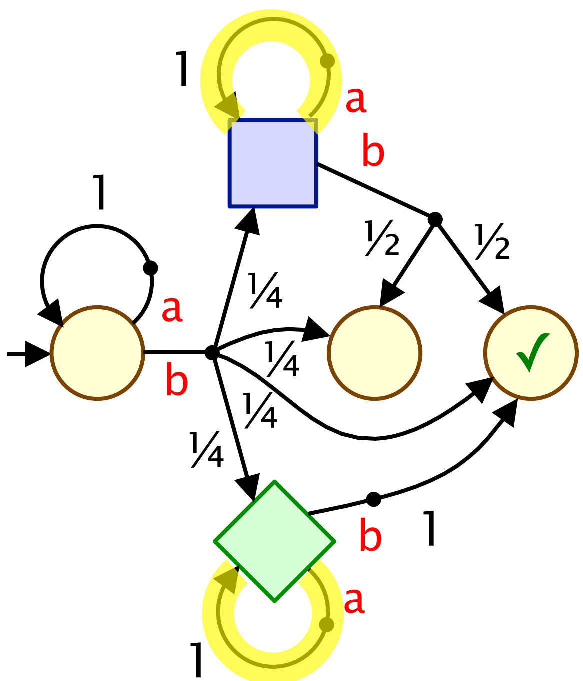
$\langle\langle \bigcirc \rangle\rangle P_{\geq 1/4}[\ F\ ✓\ ]$

true in initial state

$\langle\langle \bigcirc \rangle\rangle P_{\geq 1/3}[\ F\ ✓\ ]$

$\langle\langle \bigcirc, \square \rangle\rangle P_{\geq 1/3}[\ F\ ✓\ ]$

Examples



$\langle\langle \bigcirc \rangle\rangle P_{\geq 1/4} [\ F\ \checkmark\ ]$

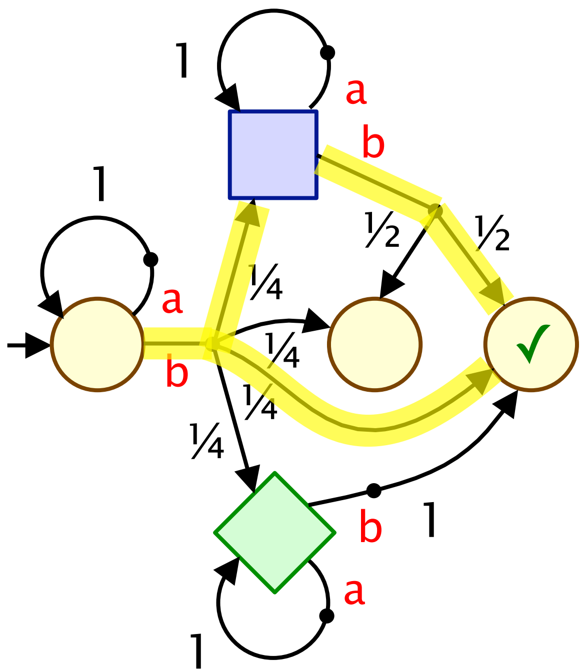
true in initial state

$\langle\langle \bigcirc \rangle\rangle P_{\geq 1/3} [\ F\ \checkmark\ ]$

false in initial state

$\langle\langle \bigcirc, \square \rangle\rangle P_{\geq 1/3} [\ F\ \checkmark\ ]$

Examples



$\langle\langle$ ○ $\rangle\rangle P_{\geq 1/4}$[F ✓]

true in initial state

$\langle\langle$ ○ $\rangle\rangle P_{\geq 1/3}$ [F ✓]

false in initial state

$\langle\langle$ ○, □ $\rangle\rangle P_{\geq 1/3}$ [F ✓]

true in initial state

rPATL semantics (rewards)

- R operator: $s \vDash \langle\langle C \rangle\rangle R^{r}_{\bowtie x}\ [F^{\star}\phi]$ iff:
  - "<u>there exist</u> strategies for players in coalition C such that, <u>for all</u> strategies of the other players, the expected cumulated reward r to reach a ϕ-state (type ⋆) satisfies ⋈ 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)

rPATL extensions

- Quantitative (numerical) properties:
  - numerical rather than boolean-valued queries
- Example:
  - $\langle\langle\{1\}\rangle\rangle\ P_{max=?}$ [F error]
  - "what is the maximum probability of reaching an error state that player 1 can guarantee?" (against player 2)
  - i.e. $\sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}$ (F error)

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

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]

Independence of strategies

- Strategies for each coalition operator are independent
  - for example, in: $\langle\langle 1 \rangle\rangle\ P_{\geq 1}[\ G\ (\ \langle\langle 1,2 \rangle\rangle\ P_{\geq 1/4}[\ F\ ✓\ ]\ )\ ]$
  - no dependencies in player 1 strategies in quantifiers
  - branching-time temporal logic (like ATL, PCTL, …)

- Introducing dependencies is problematic
  - e.g. subsumes existential semantics for PCTL on Markov decision processes (MDPs), which is undecidable
  - (does there exist a single adversary satisfying one formula?)
  - $\langle\langle 1 \rangle\rangle\ P_{\geq 1}[\ G\ \langle\langle 1 \rangle\rangle\ P_{\geq 1/4}[\ F\ ✓\ ]\ ]$

- But nested properties still have natural applications
  - e.g. sensor network, with players: sensor, repairer
  - $\langle\langle sensor \rangle\rangle\ P_{<0.01}[\ F\ (\neg \langle\langle repairer \rangle\rangle\ P_{\geq 0.95}[\ F\ \text{"operational"}\ ]\ )\ ]$

Why do we need multiple players?

- SMGs have multiple (>2) players
  - but model checking (and semantics) reduce to 2-player case
  - due to (zero sum) nature of queries expressible by rPATL
  - so why do we need multiple players?

- 1. Modelling convenience
  - and/or multiple rPATL queries on same model

- 2. May also exploit in nested queries, e.g.:
  - players: sensor1, sensor2, repairer
  - $\langle\langle sensor1 \rangle\rangle \, P_{<0.01}[\, F \, (\neg \langle\langle repairer \rangle\rangle \, P_{\geq 0.95}[\, F \text{ “operational” }] \,) \,]$

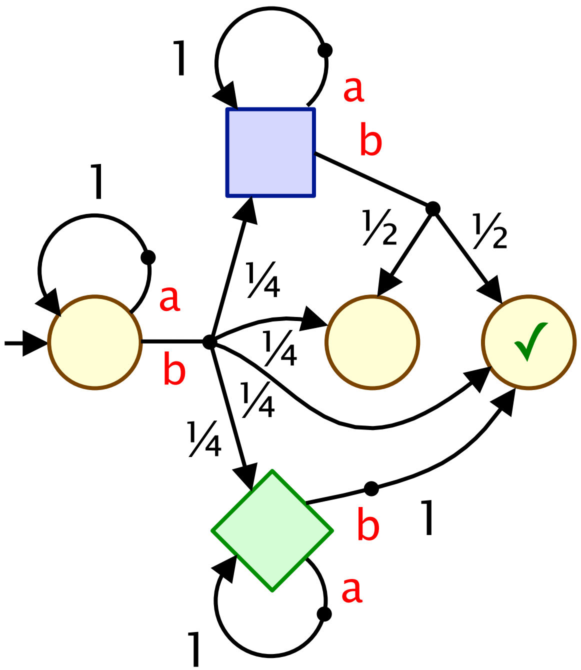
Probabilities for P operator

- E.g. $\langle\langle C \rangle\rangle P_{\geq q}[\ F\ \phi\ ]$: max/min reachability probabilities
  - compute $\sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}(F\ \phi)$ for all states s
  - deterministic memoryless strategies suffice
- Value is:
  - 1 if $s \in Sat(\phi)$, and otherwise least fixed point of:

$$f(s) = \begin{cases} \max_{a \in A(s)} \left(\sum_{s' \in S} \Delta(s,a)(s') \cdot f(s') \right) & \text{if } s \in S_1 \\ \min_{a \in A(s)} \left(\sum_{s' \in S} \Delta(s,a)(s') \cdot f(s') \right) & \text{if } s \in S_2 \end{cases}$$

- Computation:
  - start from zero, propagate probabilities backwards
  - guaranteed to converge

Example



rPATL: $\langle\langle$ ○, □ $\rangle\rangle P_{\geq 1/3}$ [F ✓]

Player 1: ○, □ Player 2: ◇

Compute: $\sup_{\sigma_1 \in \Sigma_1} \inf_{\sigma_2 \in \Sigma_2} Pr_s^{\sigma_1,\sigma_2}$ (F ✓)

Rewards for R[F^c] operator

- E.g. $\langle\langle C\rangle\rangle R^r_{\geq q}[\ F^c\ \phi\ ]$: max/min expected rewards for P1/P2
  - again: deterministic memoryless strategies suffice

- Value is:
  - ∞ if $s \in Sat(\ \langle\langle C\rangle\rangle P_{>0}[\ G\ F\ \text{“pos\_rew”}\ ]\ )$,
  - 0 if $s \in Sat(\phi)$, and otherwise least fixed point of:

$$f(s) = \begin{cases} r(s) + \max_{a \in A(s)} \left(\sum_{s' \in S} \Delta(s,a)(s') \cdot f(s') \right) & \text{if } s \in S_1 \\ r(s) + \min_{a \in A(s)} \left(\sum_{s' \in S} \Delta(s,a)(s') \cdot f(s') \right) & \text{if } s \in S_2 \end{cases}$$

Rewards for R[F$^\infty$] operator

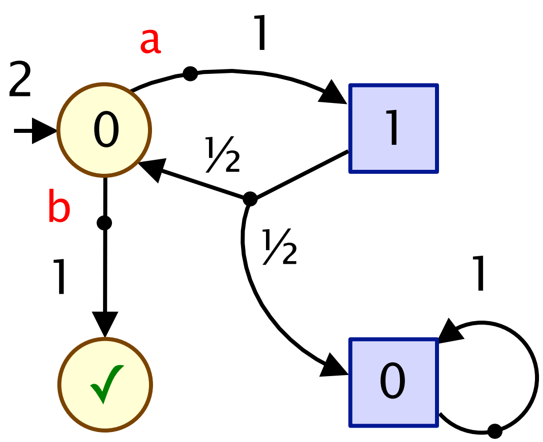
- E.g. $\langle\langle C\rangle\rangle R^{r}_{\geq q}[\ F^{\infty}\ \phi\ ]$: max/min expected rewards for P1/P2
  - again: deterministic memoryless strategies suffice
- Value is:
  - ∞ if $s \in$ Sat($\langle\langle C\rangle\rangle P_{>0}[$ G F "pos_rew" $]$),
  - 0 if $s \in$ Sat(ϕ), and otherwise greatest fixed point over $\mathbb{R}$ of:

$$f(s) = \begin{cases} r(s) + \max_{a\in A(s)} \left(\sum_{s'\in S} \Delta(s,a)(s') \cdot f(s') \right) & \text{if } s \in S_1 \\ r(s) + \min_{a\in A(s)} \left(\sum_{s'\in S} \Delta(s,a)(s') \cdot f(s') \right) & \text{if } s \in S_2 \end{cases}$$

- Computation:
  - 1. set zero rewards to ϵ, compute least fixed point
  - 2. evaluate greatest fixed point, downwards from step 1

Example: Finite memory for R[F0]

- E.g. $\langle\langle C \rangle\rangle R^{r}_{\geq q}[\ F^{0}\ \phi\ ]$: max/min expected rewards for P1/P2
  - now: deterministic memoryless strategies do not suffice



$\langle\langle ○, □ \rangle\rangle R^{r}_{\geq ½}[\ F^{0}\ ✓\ ]$

b: reward 0
a, b: expected reward 0.5
a, a, b: expected reward 0.5
a, a, a, b: expected reward 0.375

What if incoming reward is 2?

b: reward 2
a, b: expected reward 1.5

Rewards for R[F^0] operator

- E.g. $\langle\langle C \rangle\rangle R^{r}_{\geq q}[\ F^{0}\ \phi\ ]$: max/min expected rewards for P1/P2
  - now: deterministic memoryless strategies do not suffice
- There exists a finite-memory optimal strategy for P1
  - there exists a bound B, beyond which strategy is memoryless
  - B is exponential in worst-case, but can be computed…
- Computation:
  - compute bound B (using simpler rPATL queries)
  - perform value iteration for each level 0,…,B; combine results

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
  - symbolic (BDD-based) implementation in progress

- Features:
  - modelling language for SMGs (guarded command based)
  - rPATL model checking
  - strategy synthesis and analysis
  - GUI: model editor, simulator, graph-plotting, strategies, …

- Available now
  - http://www.prismmodelchecker.org/games/

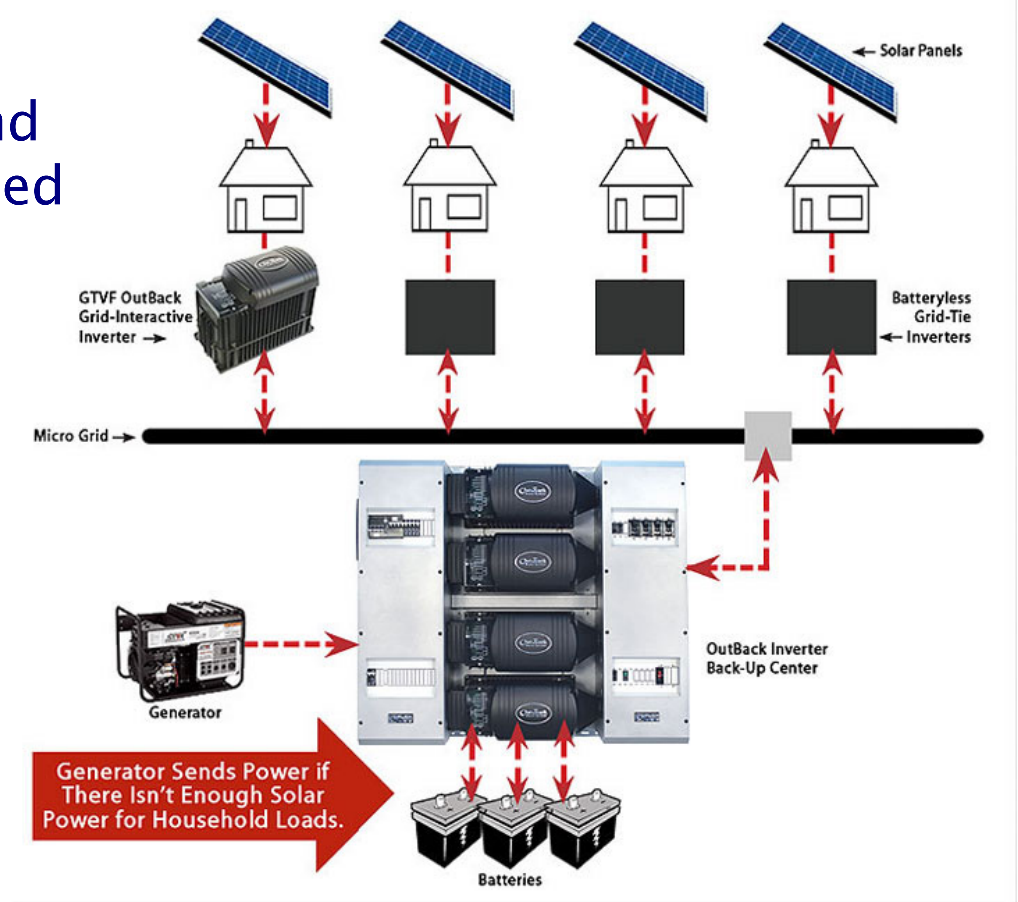
Case studies

- 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]

- Ongoing applications
  - trust models in user-centric networks
  - (randomised) security protocols

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

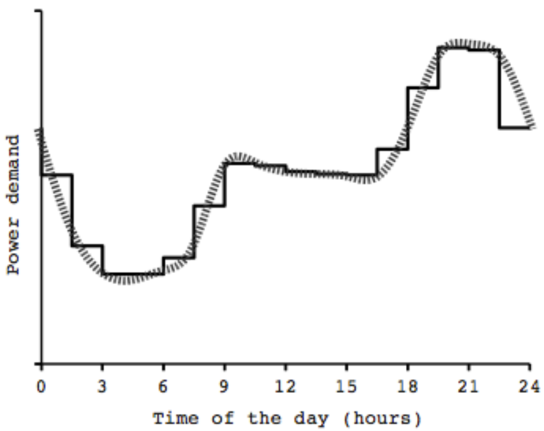


Microgrid demand-side management

- 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



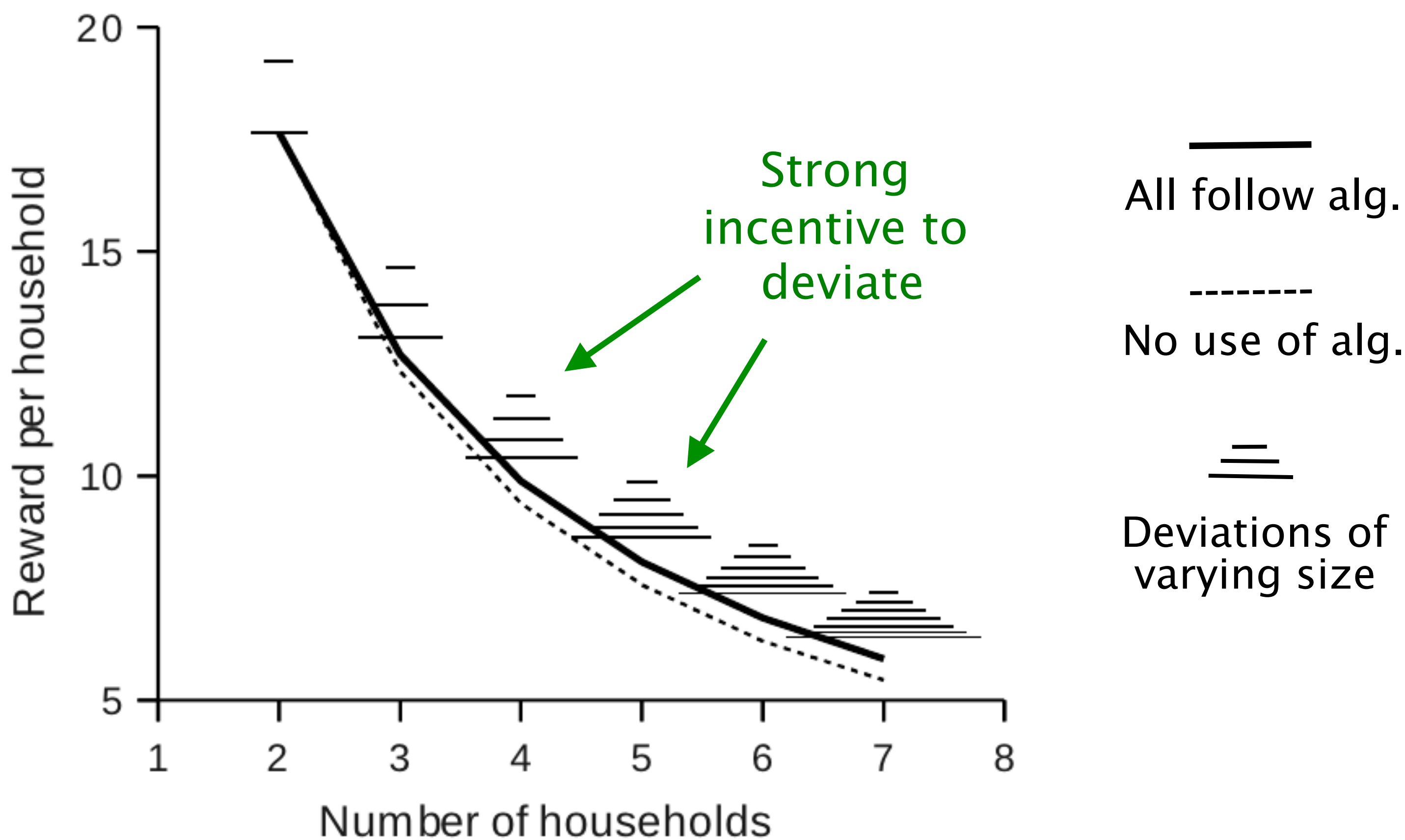
- Built/analysed models
  - for N=2,...,7 households

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

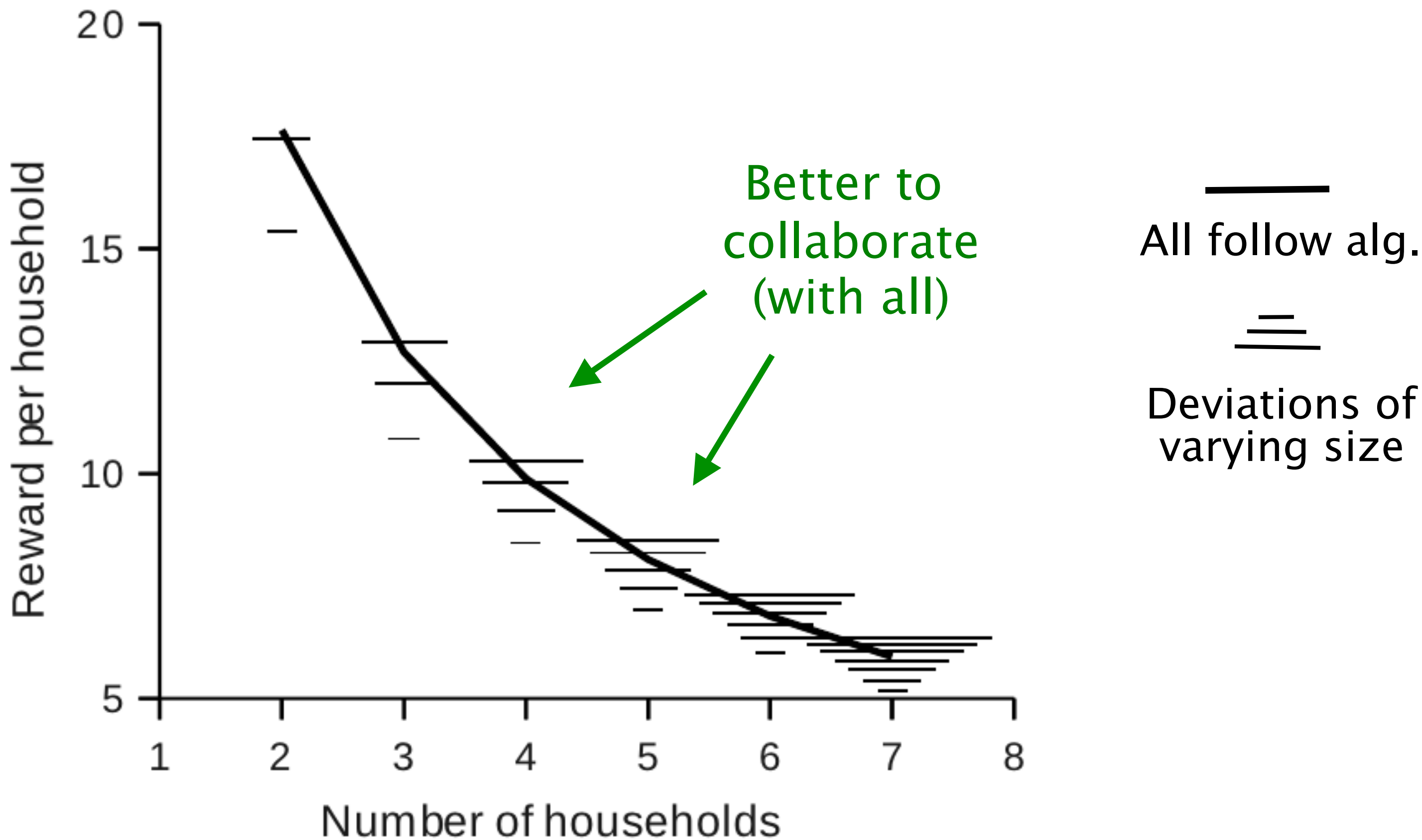
Results: Competitive behaviour

- 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: e.g. 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/