Probabilistic Model Checking and Controller Synthesis

Dave Parker

University of Birmingham

AVACS Autumn School, October 2015
Overview

• Probabilistic model checking
  – verification vs. strategy/controller synthesis
  – Markov decision processes (MDPs)
  – example: robot navigation

• Multi-objective probabilistic model checking
  – examples: power management/team-formation

• Stochastic (multi-player) games
  – example: energy management

• Permissive controller synthesis
Motivation

• Verifying probabilistic systems...
  − unreliable or unpredictable behaviour
    • failures of physical components
    • message loss in wireless communication
    • unreliable sensors/actuators
  − randomisation in algorithms/protocols
    • random back–off in communication protocols
    • random routing to reduce flooding or provide anonymity

• We need to verify quantitative system properties
  − “the probability of the airbag failing to deploy within 0.02 seconds of being triggered is at most 0.001”
  − not just correctness: reliability, timeliness, performance, …
  − not just verification: correctness by construction
Probabilistic model checking

• Construction and analysis of probabilistic models
  – state–transition systems labelled with probabilities (e.g. Markov chains, Markov decision processes)
  – from a description in a high–level modelling language

• Properties expressed in temporal logic, e.g. PCTL:
  – trigger → P_{≥0.999} [ F_{≤20} deploy ]
  – “the probability of the airbag deploying within 20ms of being triggered is at least 0.999”
  – properties checked against models using exhaustive search and numerical computation
Probabilistic model checking

- Many types of probabilistic models supported
- Wide range of quantitative properties, expressible in temporal logic (probabilities, timing, costs, rewards, ...)
- Often focus on numerical results (probabilities etc.)
  - analyse trends, look for system flaws, anomalies

  - \( P_{\leq 0.1} \{ F \text{ fail} \} \) – “the probability of a failure occurring is at most 0.1”
  - \( P = \{ F \text{ fail} \} \) – “what is the probability of a failure occurring?”
Probabilistic model checking

- Many types of probabilistic models supported
- Wide range of quantitative properties, expressible in temporal logic (probabilities, timing, costs, rewards, …)
- Often focus on numerical results (probabilities etc.)
  - analyse trends, look for system flaws, anomalies
- Provides "exact" numerical results/guarantees
  - compared to, for example, simulation
- Combines numerical & exhaustive analysis
  - especially useful for nondeterministic models
- Fully automated, tools available, widely applicable
  - network/communication protocols, security, biology, robotics & planning, power management, …
Markov decision processes (MDPs)

- Markov decision processes (MDPs)
  - widely used also in: AI, planning, optimal control, ...
  - model nondeterministic as well as probabilistic behaviour

- Nondeterminism for:
  - control: decisions made by a controller or scheduler
  - adversarial behaviour of the environment
  - concurrency/scheduling: interleavings of parallel components
  - abstraction, or under-specification, of unknown behaviour
• **A strategy** (or “policy”, “scheduler”, “adversary”)  
  – is a resolution of nondeterminism, based on history  
  – is (formally) a mapping $\sigma$ from finite paths to distributions  
  – induces an (infinite-state) discrete-time Markov chain

![Diagram of a Markov chain with states $s_0$, $s_1$, $s_2$, and $s_3$ and transitions labeled with probabilities and actions.]

• **Classes of strategies:**  
  – randomisation: deterministic or randomised  
  – memory: memoryless, finite-memory, or infinite-memory
Example strategy

- Strategy $\sigma$ which picks $b$ then $c$ in $s_1$
  - $\sigma$ is finite-memory and deterministic

- Fragment of induced Markov chain:
Verification vs. Strategy synthesis

1. Verification
   - quantify over all possible strategies (i.e. best/worst-case)
   - $P_{\leq 0.1} [\text{F err}]$: “the probability of an error occurring is $\leq 0.1$ for all strategies”
   - applications: randomised communication protocols, randomised distributed algorithms, security, ...

2. Strategy synthesis
   - generation of "correct-by-construction" controllers
   - $P_{\leq 0.1} [\text{F err}]$: "does there exist a strategy for which the probability of an error occurring is $\leq 0.1$?"
   - applications: robotics, power management, security, ...

Two dual problems; same underlying computation:
   - compute optimal (minimum or maximum) values
• **Example MDP**
  
  – robot moving through terrain divided into a 3x2 grid
Example – Reachability

Verify: $P \leq 0.6 \ [F \ \text{goal}_1 \ ]$

or

Synthesise for: $P \geq 0.4 \ [F \ \text{goal}_1 \ ]$

$\Downarrow$

Compute: $P_{\text{max}}=? \ [F \ \text{goal}_1 \ ]$

Optimal strategies: memoryless and deterministic

Computation:
graph analysis + numerical soln.
(linear programming, value iteration, policy iteration)
Example – Reachability

Verify: \( P_{\leq 0.6} [ F \text{ goal}_1 ] \)

or

Synthesise for: \( P_{\geq 0.4} [ F \text{ goal}_1 ] \)

\( \Downarrow \)

Compute: \( P_{\max} =? [ F \text{ goal}_1 ] = 0.5 \)

Optimal strategies: memoryless and deterministic

Computation:
graph analysis + numerical soln. (linear programming, value iteration, policy iteration)
Example – Reachability

Verify: $P_{\leq 0.6} [ F \text{ goal}_1 ]$

or

Synthesise for: $P_{\geq 0.4} [ F \text{ goal}_1 ]$

⇓

Compute: $P_{\text{max}} = ? [ F \text{ goal}_1 ] = 0.5$

Optimal strategies: memoryless and deterministic

Computation:
graph analysis + numerical soln.
(linear programming, value iteration, policy iteration)

Optimal strategy:
$$
\begin{align*}
\text{s}_0 &: \text{east} \\
\text{s}_1 &: \text{south} \\
\text{s}_2 &: - \\
\text{s}_3 &: - \\
\text{s}_4 &: \text{east} \\
\text{s}_5 &: - 
\end{align*}
$$
Linear temporal logic (LTL)

- **Probabilistic LTL** (multiple temporal operators)
  - e.g. $P_{\text{max}} =? \ [(G\neg\text{hazard}) \land (GF \text{goal}_1)]$ - "maximum probability of avoiding hazard and visiting goal\(_1\) infinitely often?"
  - e.g. $P_{\text{max}} =? \ [\neg\text{zone}_3 \ U (\text{zone}_1 \land F \text{zone}_4)]$ - "max. probability of patrolling zones 1 then 4, without passing through 3".

- **Probabilistic model checking**
  - convert LTL formula $\psi$ to deterministic automaton $A_\psi$ (Buchi, Rabin, finite, ...) 
  - build/solve product MDP $M \otimes A_\psi$
  - reduces to reachability problem 
  - optimal strategies are: 
    - deterministic
    - finite-memory

Det. Buchi automaton $A_\psi$
for $\psi = G\neg h \land GF g_1$
Example: Product MDP construction

\[ M \otimes A_\psi \]

\[ \psi = G\neg h \land GF g_1 \]

\[ M \]

\[ s_0 \xrightarrow{0.4, \text{east}} s_1 \]
\[ s_0 \xrightarrow{0.1, \text{south}} s_3 \]
\[ s_1 \xrightarrow{0.6, \text{east}} s_2 \]
\[ s_1 \xrightarrow{0.1, \text{south}} s_4 \]
\[ s_2 \xrightarrow{0.5, \text{east}} s_3 \]
\[ s_2 \xrightarrow{0.9, \text{stuck}} s_4 \]
\[ s_3 \xrightarrow{0.8, \text{south}} s_0 \]
\[ s_3 \xrightarrow{0.6, \text{west}} s_4 \]
\[ s_4 \xrightarrow{0.4, \text{west}} s_5 \]
\[ s_5 \xrightarrow{0.1, \text{north}} s_2 \]

\[ \{\text{goal}_2\} \]
\[ \{\text{goal}_1\} \]
\[ \{\text{hazard}\} \]

\[ A_\psi \]

\[ \psi = G\neg h \land GF g_1 \]

\[ q_0 \xrightarrow{g_1 \land \neg h} q_1 \]
\[ q_0 \xrightarrow{\neg g_1 \land \neg h} q_2 \]
\[ q_1 \xrightarrow{h} q_2 \]
\[ q_2 \xrightarrow{\text{true}} q_1 \]

\[ \{\text{goal}_1\} \]
\[ \{\text{goal}_2\} \]
\[ \{\text{hazard}\} \]
Example: Product MDP construction

\[ M \otimes A_\psi \]

\[ \psi = G \neg h \land GF g_1 \]
MDPs – Other properties

• **Costs and rewards** (expected, accumulated values)
  
  – e.g. $R_{\text{max}}=? \ [ \text{F end} ]$ – "what is the worst-case (maximum) expected time for the protocol to complete?"
  
  – e.g. $R_{\text{min}}=? \ [ \text{F goal}_2 ]$ – "what is the optimal (minimum) expected number of moves needed to reach goal$_2$?"
  
  – optimal strategies: memoryless and deterministic
  
  – similar computation to probabilistic reachability

• **Expected cost/reward to satisfy** (co-safe) LTL formula
  
  – e.g. $R_{\text{min}}=? \ [ \neg \text{zone}_3 \ U \ (\text{zone}_1 \land \text{F zone}_4) ]$ – "minimise exp. time to patrol zones 1 then 4, without passing through 3"
  
  – optimal strategies: finite-memory and deterministic
  
  – build/solve product of MDP and det. finite automaton

• **Nested properties**, e.g. using PCTL (branching time logic)
Application: Robot navigation

- **Navigation planning:** [IROS'14]
  - MDP models navigation through an uncertain environment
  - LTL used to formally specify tasks to be executed
  - synthesise finite-memory strategies to construct plans/controllers
  - links to continuous-space planner
Application: Robot navigation

• Navigation planning MDPs
  – expected timed on edges + probabilities
  – learnt using data from previous explorations

• LTL-based task specification
  – expected time to satisfy (one or more) co-safe LTL formulas
  – c.f. ad-hoc reward structures, e.g. with discounting
  – also: efficient re-planning [IROS’14]; progress metric [IJCAI’15]

• Implementation
  – MetraLabs Scitos A5 robot + ROS module based on PRISM
Overview

• Probabilistic model checking
  – verification vs. strategy synthesis
  – Markov decision processes (MDPs)
  – example: robot navigation

• Multi-objective probabilistic model checking
  – examples: power management/team-formation

• Stochastic (multi-player) games
  – example: energy management

• Permissive controller synthesis
Multi-objective model checking

- **Multi-objective probabilistic model checking**
  - investigate trade-offs between conflicting objectives
  - in PRISM, objectives are probabilistic LTL or expected rewards

- **Achievability queries**: \( \text{multi}(P_{>0.95}[F \text{ send }], R_{time>10}[C]) \)
  - e.g. “is there a strategy such that the probability of message transmission is > 0.95 and expected battery life > 10 hrs?”

- **Numerical queries**: \( \text{multi}(P_{\text{max=?}}[F \text{ send }], R_{time>10}[C]) \)
  - e.g. “maximum probability of message transmission, assuming expected battery life-time is > 10 hrs?”

- **Pareto queries**:
  - \( \text{multi}(P_{\text{max=?}}[F \text{ send }], R_{\text{time}\text{max=?}}[C]) \)
  - e.g. "Pareto curve for maximising probability of transmission and expected battery life-time"
Multi-objective model checking

- Multi-objective probabilistic model checking
  - investigate trade-offs between conflicting objectives
  - in PRISM, objectives are probabilistic LTL or expected rewards

- Achievability queries: \( \text{multi}(P_{>0.95}[F \text{ send}], R_{\text{time} > 10}[C]) \)
  - e.g. “is there a strategy such that the probability of message transmission is > 0.95 and expected battery life > 10 hrs?”

- Numerical queries: \( \text{multi}(P_{\text{max}=?}[F \text{ send}], R_{\text{time} > 10}[C]) \)
  - e.g. “maximum probability of message transmission, assuming expected battery life-time is > 10 hrs?”

- Pareto queries:
  - \( \text{multi}(P_{\text{max}=?}[F \text{ send}], R_{\text{time max}=?}[C]) \)
  - e.g. "Pareto curve for maximising probability of transmission and expected battery life-time"
Multi-objective model checking

• Optimal strategies:
  – usually finite-memory (e.g. when using LTL formulae)
  – may also need to be randomised

• Computation:
  – construct a product MDP (with several automata),
    then reduces to linear programming [TACAS'07,TACAS'11]
  – can be approximated using iterative numerical methods,
    via approximation of the Pareto curve [ATVA'12]

• Extensions [ATVA'12]
  – arbitrary Boolean combinations of objectives
    • e.g. $\psi_1 \implies \psi_2$ (all strategies satisfying $\psi_1$ also satisfy $\psi_2$)
    • (e.g. for assume–guarantee reasoning)
  – time-bounded (finite–horizon) properties
Example – Multi-objective

- Achievability query
  - \( P_{\geq 0.7}[G \neg\text{hazard}] \land P_{\geq 0.2}[GF \text{ goal}_1] \) ? True (achievable)

- Numerical query
  - \( P_{\text{max}=?}[GF \text{ goal}_1] \) such that \( P_{\geq 0.7}[G \neg\text{hazard}] \) ? \(~0.2278\)

- Pareto query
  - for \( P_{\text{max}=?}[G \neg\text{hazard}] \land P_{\text{max}=?}[GF \text{ goal}_1] \) ?
Example – Multi-objective

Strategy 1
(deterministic)

\( s_0 : \text{east} \)
\( s_1 : \text{south} \)
\( s_2 : \text{–} \)
\( s_3 : \text{–} \)
\( s_4 : \text{east} \)
\( s_5 : \text{west} \)

\( \psi_1 = G \neg \text{hazard} \)
\( \psi_2 = GF \text{ goal}_1 \)
Example – Multi-objective

Strategy 2 (deterministic)

\[ s_0 : \text{south} \]
\[ s_1 : \text{south} \]
\[ s_2 : - \]
\[ s_3 : - \]
\[ s_4 : \text{east} \]
\[ s_5 : \text{west} \]

\[ \psi_1 = G \neg \text{hazard} \]
\[ \psi_2 = GF \text{ goal}_1 \]
Example – Multi-objective

Optimal strategy:
(randomised)

\[ s_0 : 0.3226 : \text{east} \]
\[ s_0 : 0.6774 : \text{south} \]

\[ s_1 : 1.0 : \text{south} \]

\[ s_2 : \text{-} \]

\[ s_3 : \text{-} \]

\[ s_4 : 1.0 : \text{east} \]

\[ s_5 : 1.0 : \text{west} \]
Multi-objective: Applications

Synthesis of controllers for dynamic power management [TACAS'11]

IBM TravelStar VP disk drive
- switches between power modes:
- active/idle/idlelp/stby/sleep

MDP model in PRISM:
- power manager
- disk requests
- request queue
- power usage

Multi-objective:
"minimise energy consumption, subject to constraints on:
(i) expected job queue size;
(ii) expected number of lost jobs

Synthesis of team formation strategies [CLIMA'11, ATVA'12]

Pareto curve:
\[ x = \text{"probability of completing task 1"}; \]
\[ y = \text{"probability of completing task 2"}; \]
\[ z = \text{"expected size of successful team"} \]
Overview

• Probabilistic model checking
  – verification vs. strategy synthesis
  – Markov decision processes (MDPs)
  – example: robot navigation

• Multi-objective probabilistic model checking
  – examples: power management/team-formation

• Stochastic (multi-player) games
  – example: energy management

• Permissive controller synthesis
Stochastic multi-player games (SMGs)

- **Stochastic multi-player games**
  - players control states; choose actions
  - models *competitive/collaborative* behaviour
  - applications: security (system vs. attacker), controller synthesis (controller vs. environment), distributed algorithms/protocols, ...

- **Property specifications: rPATL**
  - \( \langle \langle \{1,2\} \rangle \rangle P_{\geq 0.95} [ F_{\leq 45} \text{ done } ] : \) "can nodes 1,2 collaborate so that the probability of the protocol terminating within 45 seconds is at least 0.95, whatever nodes 3,4 do?"
  - formally: \( \langle \langle C \rangle \rangle \psi : \text{do there exist} \) strategies for players in \( C \) such that, for all strategies of other players, property \( \psi \) holds?

- **Model checking** [TACAS'12,FMSD'13]
  - zero sum properties: analysis reduces to 2-player games
  - PRISM-games: [www.prismmodelchecker.org/games](http://www.prismmodelchecker.org/games)
Example – Stochastic games

- Two players: 1 (robot controller), 2 (environment)
  - probability of $s_1$–south→$s_4$ is in $[p,q] = [0.5-\Delta, 0.5+\Delta]$
Example – Stochastic games

- Two players: 1 (robot controller), 2 (environment)
  - probability of $s_1 \rightarrow$ south → $s_4$ is in $[p,q] = [0.5-\Delta, 0.5+\Delta]$

rPATL: $\langle\langle 1 \rangle\rangle \ P_{\text{max}=?} \ [ F \ \text{goal}_1 ]$

Optimal strategies: memoryless and deterministic

Computation: graph analysis & numerical approximation
Example – Stochastic games

- Two players: 1 (robot controller), 2 (environment)
  - probability of $s_1$–south→$s_4$ is in $[p,q] = [0.5-\Delta, 0.5+\Delta]$
Example: Energy management

• Energy management protocol for Microgrid
  – Microgrid: local energy management
  – randomised demand management protocol
  – random back-off when demand is high

• Original analysis [Hildmann/Saffre'11]
  – protocol increases "value" for clients
  – simulation-based, clients are honest

• Our analysis
  – stochastic multi-player game model
  – clients can cheat (and cooperate)
  – model checking: PRISM–games
Example: Energy management

- Exposes protocol weakness
  - incentive for clients to act selfishly

- We propose a simple fix (and verify it)
  - clients can be punished

Value per client

Value per client, with fix
Overview

• Probabilistic model checking
  – verification vs. strategy synthesis
  – Markov decision processes (MDPs)
  – example: robot navigation

• Multi-objective probabilistic model checking
  – examples: power management/team-formation

• Stochastic (multi-player) games
  – example: energy management

• Permissive controller synthesis
Permissive controller synthesis

- **Multi-strategy synthesis** [TACAS'14]
  - for Markov decision processes and stochastic games
  - choose *sets* of actions to take in each state
  - controller is free to choose any action at runtime
  - flexible/robust (e.g. actions become unavailable or goals change)

- **Example**

  ![Diagram](image)

  **Multi-strategy:**
  - $s_0$ : east or south
  - $s_1$ : south
  - $s_2$ : stuck
  - $s_3$ : stuck
  - $s_4$ : east
  - $s_5$ : west
Permissive controller synthesis

• Multi-strategies and temporal logic
  – multi-strategy $\Theta$ satisfies a property $P_{>p}[F\text{ goal}]$ iff any strategy $\sigma$ that adheres to $\Theta$ satisfies $P_{>p}[F\text{ goal}]$

• We quantify the permissivity of multi-strategies
  – by assigning penalties to each action in each state
  – a multi-strategy is penalised for every action it blocks
  – static and dynamic (expected) penalty schemes

• Permissive controller synthesis
  – $\exists$ a multi-strategy satisfying $P_{\leq 0.6}[F\text{ goal}_1]$ with penalty $< c$?
  – what is the multi-strategy with optimum permissivity?
  – reduction to mixed-integer LP problems
  – applications: energy management, cloud scheduling, …
Conclusion

- **Probabilistic model checking**
  - verification vs. controller synthesis
  - Markov decision processes, temporal logic, applications

- **Recent directions and extensions**
  - multi-objective probabilistic model checking
  - model checking for stochastic games
  - permissive controller synthesis

- **Challenges**
  - stochastic games: multi-objective, equilibria, richer logics
  - partial information/observability
  - probabilistic models with continuous time (or space)
  - scalability, e.g. symbolic methods, abstraction
Thanks for your attention

More info here:
www.prismmodelchecker.org/lectures/avacs15/