Model Checking and Strategy Synthesis for Stochastic Games: From Theory to Practice[*]

Marta Z. Kwiatkowska

University of Oxford, Oxford, UK
marta.kwiatkowska@cs.ox.ac.uk

Abstract

Probabilistic model checking is an automatic procedure for establishing if a desired property holds in a probabilistic model, aimed at verifying quantitative probabilistic specifications such as the probability of a critical failure occurring or expected time to termination.

Much progress has been made in recent years in algorithms, tools and applications of probabilistic model checking, as exemplified by the probabilistic model checker PRISM (`http://www.prismmodelchecker.org`). However, the unstoppable rise of autonomous systems, from robotic assistants to self-driving cars, is placing greater and greater demands on quantitative modelling and verification technologies. To address the challenges of autonomy we need to consider collaborative, competitive and adversarial behaviour, which is naturally modelled using game-theoretic abstractions, enhanced with stochasticity arising from randomisation and uncertainty. This paper gives an overview of quantitative verification and strategy synthesis techniques developed for turn-based stochastic multi-player games, summarising recent advances concerning multi-objective properties and compositional strategy synthesis. The techniques have been implemented in the PRISM-games model checker built as an extension of PRISM.

1998 ACM Subject Classification F.3.1 Specifying and Verifying and Reasoning about Programs. D.2.4 Software/Program Verification

Keywords and phrases Quantitative verification, Stochastic games, Temporal logic, Model checking, Strategy synthesis

Digital Object Identifier 10.4230/LIPIcs.ICALP.2016.4

Category Invited Talk

1 Introduction

Probability is pervasive: it is used to quantify component failures, unreliable communication media and sensor imprecision, in randomised schemes to ensure efficiency of distributed coordination and resource management mechanisms, when dealing with environmental uncertainty, and to model performance and biological systems. Probabilistic model checking [48], also referred to as quantitative probabilistic verification, is an automated technique for verifying quantitative temporal logic properties of probabilistic models, which typically arise as extensions of Markov chains or Markov decision processes, additionally annotated with quantitative information such as time, energy or cost. The properties capture a variety of quantitative correctness, reliability or performance properties, and provide formal guarantees for properties such as "the maximum probability of the airbag failing to deploy within 0.02

[*] This work was partially supported by ERC Advanced Grant VERIWARE and EPSRC Mobile Autonomy Programme Grant EP/M019918/1.

seconds is at most 10^{-10}". Probabilistic model checking involves conventional model checking techniques, for example symbolic and automata-theoretic methods, in combination with numerical or statistical analysis. For models that allow nondeterministic choices, it also enables strategy synthesis from temporal logic specifications to automate decision-making in applications such as robotics and security, where it can be employed to generate controllers or counter-attacks. This is similar to motion planning and optimal control, except that temporal logics are used to rigorously specify high-level goals, yielding controllers that are guaranteed to be correct.

Quantitative verification has been the focus of much interest in recent years, resulting in the adoption of tools such as the real-time model checker UPPAAL [3] and the probabilistic model checker PRISM [51] in several application domains. While UPPAAL is based on the theory of timed automata developed in the 1990s [2], the algorithms underlying PRISM have been known since the mid-1980s [69, 31], but have not entered the mainstream until around year 2000, enabled by symbolic techniques [7, 33, 50, 60] implemented in PRISM's first release [49]. Since then several new models and features have been added, including continuous-time Markov chains and probabilistic timed automata. PRISM has been successfully used to verify quantitative properties of a wide variety of real-life systems, automatically discovering design flaws in some of them. These include anonymity protocols [65], randomised distributed algorithms [53], wireless communication protocols [37], security [8], nanotechnology designs [59], probabilistic software [47], self-adaptive systems [15], molecular signalling pathways [45] and computational DNA circuits [57, 32].

Despite much success of probabilistic model checking, the accelerating technological advances in autonomous systems, to mention robotic assistants, self-driving cars and closed-loop medical devices, place greater and greater demands on quantitative verification technologies. Increasingly, we wish to delegate day-to-day tasks to autonomous, wirelessly connected, Internet-enabled and software-controlled devices, whose aim is to achieve certain objectives, such as safe and fuel efficient autonomous driving on a motorway. This incurs the need to consider cooperative, competitive and adversarial behaviour due to conflicting goals, e.g., safety versus fuel efficiency, while also taking into account their (possibly adversarial) interaction with environment, e.g., other cars. We also need to consider communities comprising digital and human agents, able to represent typical interactions and relationships present in scenarios such as semi-autonomous driving scenarios or collaborating with robotic assistants. Game-theoretic approaches, which explicitly represent the moves and countermoves of players and adversaries, are a natural model to represent and analyse such behaviours through the study of winning strategies. Strategy synthesis from quantitative specifications, in particular, can be viewed as constructing a strategy that is winning against all adversarial players. In these data-rich dynamic environments, stochasticity is needed not only to quantify aspects such as failure, safety and sensor uncertainty, but also to facilitate model derivation through inference from data, for example GPS sensor data.

In this paper we summarise recent progress made towards development of algorithmic techniques, tool implementation and real-life applications for turn-based stochastic multi-player games, implemented in an extension of the PRISM model checker called PRISM-games [25, 54]. Game theory is widely studied in areas such as economics and mechanism design, though is less well established as a modelling abstraction for autonomy. In the verification community, the majority of current research activity is focused on the study of algorithmic techniques and their computational complexity. In contrast, we report on ongoing effort towards the "theory to practice" transfer of quantitative verification and synthesis techniques to the challenging application domain of mobile autonomy. The overview will cover verification and strategy synthesis from quantitative temporal logic specifications, with a focus on zero-sum, complete observation stochastic games, and will include both single- and multi-objective properties, together with a wide range of quantitative objectives (expected total reward, longrun average and ratio objectives) and compositional assume-guarantee strategy synthesis. We will also briefly describe a number of real-life examples from autonomous systems and lessons learnt through applying PRISM-games in these scenarios. We conclude the paper with a discussion of work in progress and future research in this area.

2 Stochastic Multi-Player Games

Stochastic multi-player games (SMGs) [64, 30] are a generalisation of Markov decision processes, where we distinguish between several types of nondeterministic choices, each corresponding to a different player. SMGs thus allow us to reason about strategic decisions of multiple players competing or collaborating to achieve the same objective. Several stochastic game models exist, which include concurrent games [64, 22] and partial-observation games [18, 20]. Here we focus on *turn-based* games as studied in [30], in which a single player controls the choices made in a given state. The presentation is based on [23, 24, 26, 11, 10, 9, 67], where we refer the reader for further details.

Let $\mathcal{D}(X)$ denote the family of all probability distributions over a set X.

▶ **Definition 1** (Stochastic multi-player game (SMG)). A *stochastic multi-player game* (SMG) is a tuple $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$, where Π is a finite set of players; S is a finite nonempty set of states partitioned into player states $S_\Pi = \bigcup S_{i \in \Pi}$ and probabilistic states S_p; $s_{\text{init}} \in S_\Pi$ is an initial state; $\Delta \colon S \times S \to [0,1]$ is a probabilistic transition function such that for all player states $s, t \in S_\Pi$ and probabilistic states $s' \in S_p$ we have $\Delta(s,t) = 0$, $\Delta(s,s') \in \{0,1\}$ and $\sum_{t \in S_\Pi} \Delta(s',t) = 1$; A is a finite nonempty set of actions; and $L : S_p \to \mathrm{A}$ is an action labelling function.

Note that each state of an SMG is controlled by a single player. The game proceeds as follows. It starts in the initial player state s_{init} and, when in a player $i \in \Pi$ state s, the player chooses the next state s' such that $\Delta(s,s') = 1$, and when in a probabilistic state $s \in S_p$ the next state is sampled according to the probability distribution $\Delta(s', \cdot)$.

Stochastic games can be equivalently defined, see e.g. [24], by partitioning the state space into player states S_Π only, and associating with each such state $s \in S_\Pi$ a set of action-distribution pairs (a, μ) called *moves*, where $a = L(s')$ for some $s' \in S_p$ such that $\Delta(s,s') = 1$ and the distribution μ, defined by $\mu(t) = \Delta(s',t)$ for all $t \in S$, gives the probability of moving to a successor state. If the sets of all players but one are empty, then a stochastic turn-based game is a probabilistic automaton in the sense of Segala [63], a mild generalisation of a Markov decision process (MDP).

We unfold an SMG $\mathcal{G}$ into *paths*, namely possibly infinite sequences $\lambda = s_0 s_1 s_2 \ldots$ such that $\Delta(s_i, s_{i+1}) > 0$ for all $i \geq 0$. Note that player states and probabilistic states alternate in a path. For a finite path $\lambda = s_0 s_1 \ldots s_k$ we use $|\lambda| = k+1$ to denote the length of the path and $\text{last}(\lambda) = x_k$ denotes its last element. A *trace* of λ is the sequence of actions that label the probabilistic states within λ. We use $\text{Path}_{\mathcal{G},s}$ to denote the set of all infinite paths originating in a state $s \in S$ and $\text{Path}_{\mathcal{G}} = \bigcup_{s \in S} \text{Path}_{\mathcal{G},s}$. The sets $\text{FPath}_{\mathcal{G},s}, \text{FPath}_{\mathcal{G}}$ of finite paths are defined analogously.

SMGs can be annotated with rewards, which allows us to formulate a variety of quantitative analyses that assign reward values to paths. We consider cumulative, longrun average and ratio rewards. The analysis typically involves ensuring that the game achieves a certain reward bound or optimising these values in the game.

▶ **Definition 2** (Reward structure). Given a game $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$, a *reward structure* for $\mathcal{G}$ is a function $r : S \to \mathbb{R}$.

To resolve the nondeterminism in an SMG, similarly to MDPs we use strategies, except we now have a strategy for each player $i \in \Pi$. We work with an explicit memory representation of strategies due to [14].

▶ **Definition 3** (Strategy). For an SMG $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$, a *strategy* σ_i for player i of $\mathcal{G}$ is a tuple $\sigma_i = (M, \sigma_i^u, \sigma_i^n, \sigma_i^{\mathsf{init}})$, where M is a countable set of memory elements, $\sigma_i^u \colon M \times S \to \mathcal{D}(M)$ is a memory update function, $\sigma_i^n \colon M \times S_i \to \mathcal{D}(S)$ is a next move function such that $\sigma_i^n(m, s)(s') > 0$ only if $\Delta(s, s') > 0$, and $\sigma_i^{\mathsf{init}} \colon S \to \mathcal{D}(M)$ is an initial memory element function. If the memory update function maps to point distributions, i.e. is of type $\sigma_i^u \colon M \times S \to M$, the strategy σ_i is *deterministic memory update*, and otherwise it is *stochastic memory update*. The set of all strategies for player $i \in \Pi$ is denoted by $\Sigma_{\mathcal{G}}^i$. A *strategy profile* $\sigma = \sigma_1, \ldots \sigma_{|\Pi|}$ comprises a strategy for every player in $\mathcal{G}$.

For a given strategy σ_i of player i, the game proceeds as follows. It starts in a player state with memory sampled according to the initial distribution function σ_i^{init}. Then, in every step of the game, player i updates the current memory element based on the current state of the game using the memory update function σ_i^u. Moreover, if the game is in a player i state, player i chooses the next state of the game using the next move action σ_i^n.

Strategies can be classified according to the use of randomisation or memory. A given strategy σ_i of player i is *deterministic* (pure) if the next move action σ_i^n is of type $\sigma_i^n \colon M \times S_i \to S$, and otherwise it is *randomised.* Similarly, σ_i is *memoryless* if M is a singleton, *finite memory* if M is finite, and *infinite memory* otherwise. An alternative, standard representation of a deterministic player i strategy is as a function from finite paths terminating in a player i state to distributions over the available moves (a, μ) in the given state. Deterministic memoryless strategies can be simply represented as functions $\sigma_i^n \colon S_i \to S$.

Stochastic memory update strategies have the same power as deterministic memory update but are exponentially more succinct than deterministic memory update [66]. Stochastic memory update strategies can be determinised if their memory is not restricted [10, 9]; in fact, a finite stochastic memory update strategy can result in a finite or infinite deterministic update strategy. Memory elements of the determinised strategies are probability distributions over memory elements, which can be interpreted as the belief that the player has about the memory element, knowing only the history and rules to update them, while the actual memory based on sampling is kept secret.

For a given strategy profile $\sigma = \sigma_1, \ldots \sigma_{|\Pi|}$, the behaviour of an SMG $\mathcal{G}$ is fully probabilistic and we use $\mathrm{Path}_{\mathcal{G},s}^\sigma$, $\mathrm{Path}_{\mathcal{G}}^\sigma$, $\mathrm{FPath}_{\mathcal{G},s}^\sigma$ and $\mathrm{FPath}_{\mathcal{G}}^\sigma$, respectively, to denote paths obtained under the strategy profile σ. Following standard methods [13], we can define a probability measure $\mathrm{Pr}_{\mathcal{G},s}^\sigma$ over the set of infinite paths $\mathrm{Path}_{\mathcal{G},s}^\sigma$. Given a random variable ρ over this probability space, the expected value of ρ is defined as $\mathbb{E}_{\mathcal{G},s}^\sigma(\rho) = \int_{\mathrm{Path}_{\mathcal{G},s}^\sigma} \rho \; \mathrm{dPr}_{\mathcal{G},s}^\sigma$.

We will sometimes require restrictions on SMGs. One such restriction to so called *stopping games* was introduced to avoid infinite accumulation of rewards.

▶ **Definition 4** (Stopping game). A state $s_f \in S$ of a stochastic multi-player game $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$ is called terminal if and only if $\Delta(s_f, s_f) = 1$ and $\Delta(s_f, s') = 0$ for all $s' \neq s_f, s' \in S$. A game is called *stopping* if it has at least one terminal state and if it holds that, for every strategy profile $\sigma = \sigma_1, \ldots \sigma_{|\Pi|}$ and starting from the initial state, with probability 1 the game eventually stops, *i.e.,* a terminal state is reached.

3 Property Specification

We now introduce a notation for specifying temporal and reward-based properties of stochastic games. The presentation employs a variant of the probabilistic temporal logic called RPATL, based on Probabilistic Computation Tree Logic (PCTL) [12] with rewards [41], enhanced with the coalition operator from Alternating Temporal Logic (ATL) [1]. We discuss both single and multi-objective properties. The full logics RPATL and RPATL* are studied in [23, 29].

▶ **Definition 5** (RPATL property)**.** A *single-objective*, respectively *multi-objective*, RPATL property is a formula ϕ, respectively Φ, in the following grammar:

$$\phi ::= \langle\!\langle C \rangle\!\rangle \mathtt{P}_{\bowtie p}[\psi] \mid \langle\!\langle C \rangle\!\rangle \mathtt{R}^{r}_{\bowtie x}[\rho] \mid \langle\!\langle C \rangle\!\rangle \mathtt{R}^{r/c}_{\bowtie x}[\mathtt{S}]$$

$$\Phi ::= \langle\!\langle C \rangle\!\rangle (\bigwedge_{i=1}^{n} \mathtt{P}_{\bowtie p_i}[\psi_i]) \mid \langle\!\langle C \rangle\!\rangle (\bigwedge_{j=1}^{m} \mathtt{R}^{r_j}_{\bowtie x_j}[\rho_j]) \mid \langle\!\langle C \rangle\!\rangle (\bigwedge_{j=1}^{m} \mathtt{R}^{r_j/c_j}_{\bowtie x_j}[\mathtt{S}])$$

$$\psi ::= \mathtt{F}\, a$$

$$\rho ::= \mathtt{C} \mid \mathtt{S}$$

where $a \in \mathrm{AP}$ is an atomic proposition, $C \subseteq \Pi$ is a coalition of players, $\bowtie \in \{\leq, \geq\}$, $p \in [0,1]$, r, c are reward structures, and $x \in \mathbb{R}$.

The operator $\mathtt{P}_{\bowtie p}[\psi]$ is the PCTL probabilistic operator, where ψ is a CTL path formula. To simplify the presentation we only consider the reachability path formulas $\mathtt{F}\ a$. The operators $\mathtt{R}^{r}_{\bowtie x}[\mathtt{C}]$ and $\mathtt{R}^{r}_{\bowtie x}[\mathtt{S}]$ respectively denote the expected *total reward* and *longrun average reward* (mean payoff), whereas $\mathtt{R}^{r/c}_{\bowtie x}[\mathtt{S}]$ is a *longrun average ratio* reward. We combine these operators with the coalition operator $\langle\!\langle \cdot \rangle\!\rangle$ of ATL as follows. Intuitively, $\langle\!\langle C \rangle\!\rangle \mathtt{P}_{\bowtie p}[\psi]$ means that the players in coalition C can collectively ensure that $\mathtt{P}_{\bowtie p}[\psi]$ is satisfied, no matter what the players outside the coalition do. For example, in a game with players 1, 2 and 3, the property $\langle\!\langle \{1,3\} \rangle\!\rangle \mathtt{P}_{\geq 1}[\mathtt{F}\ \mathtt{end}]$ states that players 1 and 3 have a strategy to ensure that the game reaches an $\mathtt{end}$-state almost surely, irrespective of what player 2 does. $\langle\!\langle C \rangle\!\rangle \mathtt{R}^{r}_{\bowtie x}[\mathtt{C}]$ means that the players in coalition C can collectively ensure that the expected total reward is in relation $\bowtie$ to x. $\langle\!\langle C \rangle\!\rangle \mathtt{R}^{r}_{\bowtie x}[\mathtt{S}]$ and $\langle\!\langle C \rangle\!\rangle \mathtt{R}^{r/c}_{\bowtie x}[\mathtt{S}]$ are defined similarly, except that they respectively concern expected average rewards cumulated over infinite paths and expected ratio rewards. Both probabilistic and reward properties can be interpreted in *quantitative* fashion, e.g. $\langle\!\langle C \rangle\!\rangle \mathtt{P}_{\max=?}[\mathtt{F}\ a]$, meaning the maximum probability of reaching an a-state that players in C can ensure, regardless of the other players, and similarly for the minimum.

We also allow for *multi-objective* properties Φ defined as conjunctions of coalition properties of the same type, with the interpretation that all conjuncts are required to be satisfied simultaneously; see [26, 10] for a more general definition of multi-objective properties as Boolean combinations. Intuitively, the property $\langle\!\langle C \rangle\!\rangle (\bigwedge_{i=1}^{n} \mathtt{P}_{\bowtie p_i}[\psi_i])$ means that players in coalition C have a *collective* strategy to ensure that, for all $i = 1, \ldots, n$, we have that $\mathtt{P}_{\bowtie p_i}[\psi_i]$ holds, no matter what the other players do. $\langle\!\langle C \rangle\!\rangle (\bigwedge_{j=1}^{m} \mathtt{R}^{r_j}_{\bowtie x_j}[\rho_j])$ is defined similarly, so, for example, $\langle\!\langle \{4,5\} \rangle\!\rangle (\mathtt{R}^{\mathtt{profit}}_{\geq 5}[\mathtt{S}] \wedge \mathtt{R}^{\mathtt{fuel}}_{\leq 10}[\mathtt{S}])$ states that players 4 and 5 have a collective strategy to ensure that expected longrun average profit is at least 5 *and* expected longrun average fuel usage is at most 10, no matter what the other players do. Property $\langle\!\langle C \rangle\!\rangle (\bigwedge_{j=1}^{m} \mathtt{R}^{r_j/c_j}_{\geq x_j}[\rho_j])$ is a ratio reward, so for example $\langle\!\langle \{1,2\} \rangle\!\rangle (\mathtt{R}^{\mathtt{fuel/time}}_{\leq 10}[\mathtt{S}] \wedge \mathtt{R}^{\mathtt{visit/time}}_{\geq 20}[\mathtt{S}])$ states that players 1 and 2 have a collective strategy to ensure that expected longrun fuel consumption per time unit is at most 10 and expected longrun number of visits is at least 20, no matter what the other players do.

In order to define the semantics of the coalition and multi-objective properties, we require the notion of a coalition game based on the analogous notion for ATL.

▶ **Definition 6** (Coalition game). Given an SMG $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$ and coalition of players $C \subseteq \Pi$, the *coalition game* of $\mathcal{G}$ induced by C is the two-player stochastic game $\mathcal{G}_C = (\Pi, S, (S'_1, S'_2), s_{\text{init}}, \Delta, \mathrm{A}, L)$, where $S'_1 = \bigcup_{i\in C} S_i$ and $S'_2 = \bigcup_{i\in \Pi/C} S_i$. The sets of strategies of player 1 and 2 in the coalition game are respectively denoted $\Sigma^1_{\mathcal{G}_C}$ and $\Sigma^2_{\mathcal{G}_C}$.

▶ **Definition 7** (RPATL semantics). Let $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$ be an SMG whose states are labelled with atomic propositions $a \in AP$. We identify a proposition a with the set of states $S_a = \{s \in S \mid s \models a\}$ satisfying a, also denoted a. The satisfaction relation $\models$ is defined as follows, where formulas ϕ and Φ are interpreted over states of the game, whereas temporal formulas ψ and reward functions ρ are interpreted over paths:

$$\mathcal{G}, \lambda \models \mathtt{F}\, a \iff \exists i \geq 0 : (\lambda_i \lambda_{i+1} \ldots \models a)$$

$$\begin{array}{rcl}
\mathcal{G}, s \models \langle\langle C \rangle\rangle \mathtt{P}_{\bowtie p}[\psi] & \Leftrightarrow & \exists\, \sigma_1 \in \Sigma^1_{\mathcal{G}_C} \text{ s.t. } \forall\, \sigma_2 \in \Sigma^2_{\mathcal{G}_C}. \\
& & \mathrm{Pr}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}\{\lambda \in \mathrm{Path}_{\mathcal{G}_C,s} \mid \mathcal{G}_C, \lambda \models \psi\} \bowtie p \\[1em]
\mathcal{G}, s \models \langle\langle C \rangle\rangle \mathtt{R}^r_{\bowtie x}\rho & \Leftrightarrow & \exists\, \sigma_1 \in \Sigma^1_{\mathcal{G}_C} \text{ s.t. } \forall\, \sigma_2 \in \Sigma^2_{\mathcal{G}_C}. \\
& & \mathbb{E}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(rew(r,\rho)) \bowtie x \\[1em]
\mathcal{G}, s \models \langle\langle C \rangle\rangle \mathtt{R}^{r/c}_{\bowtie x}\mathtt{S} & \Leftrightarrow & \exists\, \sigma_1 \in \Sigma^1_{\mathcal{G}_C} \text{ s.t. } \forall\, \sigma_2 \in \Sigma^2_{\mathcal{G}_C}. \\
& & \mathbb{E}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(rew(r,\mathtt{S})) \bowtie x \\[1em]
\mathcal{G}, s \models \langle\langle C \rangle\rangle (\bigwedge_{i=1}^n \mathtt{P}_{\bowtie p_i}[\psi_i]) & \Leftrightarrow & \exists\, \sigma_1 \in \Sigma^1_{\mathcal{G}_C} \text{ s.t. } \forall\, \sigma_2 \in \Sigma^2_{\mathcal{G}_C},\ 1 \leq i \leq n. \\
& & \mathrm{Pr}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}\{\lambda \in \mathrm{Path}_{\mathcal{G}_C,s} \mid \mathcal{G}_C, \lambda \models \psi_i\} \bowtie p_i \\[1em]
\mathcal{G}, s \models \langle\langle C \rangle\rangle (\bigwedge_{j=1}^m \mathtt{R}^{r_j}_{\bowtie x_j}[\rho_j]) & \Leftrightarrow & \exists\, \sigma_1 \in \Sigma^1_{\mathcal{G}_C} \text{ s.t. } \forall\, \sigma_2 \in \Sigma^2_{\mathcal{G}_C},\ 1 \leq j \leq m. \\
& & \mathbb{E}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(rew(r,\rho_j)) \bowtie x_j \\[1em]
\mathcal{G}, s \models \langle\langle C \rangle\rangle (\bigwedge_{j=1}^m \mathtt{R}^{r_j/c_j}_{\bowtie x_j}[\mathtt{S}]) & \Leftrightarrow & \exists\, \sigma_1 \in \Sigma^1_{\mathcal{G}_C} \text{ s.t. } \forall\, \sigma_2 \in \Sigma^2_{\mathcal{G}_C},\ 1 \leq j \leq m. \\
& & \mathbb{E}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(rew(r/c,\mathtt{S})) \bowtie x_j
\end{array}$$

where $\mathcal{G}_C$=$(\Pi, S, (S'_1, S'_2), s_{\text{init}}, \Delta, \mathrm{A}, L)$ is the coalition game of $\mathcal{G}$ induced by C, r is a reward structure for $\mathcal{G}c$, c is a nonnegative reward structure for $\mathcal{G}_C$ s.t. the probability of its mean payoff under any strategy profile is at least some constant value, and

$$\begin{array}{rcl}
rew^k(r)(\lambda) & = & \sum_{i=0}^k r(\lambda_i) \\
rew(r,\mathtt{C})(\lambda) & = & \liminf_{k\to\infty} rew^k(r)(\lambda) \\
rew(r,\mathtt{S})(\lambda) & = & \liminf_{k\to\infty} \frac{rew^k(r)(\lambda)}{k+1} \\
rew(r/c,\mathtt{S})(\lambda) & = & \liminf_{k\to\infty} \frac{rew^k(r)(\lambda)}{1+rew^k(c)(\lambda)}
\end{array}$$

The properties defined above can be employed for verification as well as strategy synthesis for stochastic multi-player games.

4 Single-objective Properties

Model checking and strategy synthesis for single-objective RPATL properties in stochastic games reduces to checking the existence of, respectively finding (if it exists), a winning strategy in two-player stochastic games, and specifically coalition games. Formally, given a stochastic multi-player game $\mathcal{G}$ and a single-objective RPATL property ϕ, e.g., $\phi = \langle\langle C \rangle\rangle \mathtt{P}_{\bowtie p}[\psi]$ or

$\phi = \langle\langle C \rangle\rangle \mathtt{R}_{\bowtie x}[\rho]$, the model checking problem is to establish whether $s_{\text{init}} \models \phi$ in the coalition game $\mathcal{G}_C$. The strategy synthesis problem, on the other hand, for a stochastic game $\mathcal{G}$ and a property ϕ as above is to construct a strategy σ_1 for player 1 in the coalition game $\mathcal{G}_C$ (if it exists) that is a witness to the satisfaction $s_{\text{init}} \models \phi$.

Similarly to MDPs, deciding the verification and strategy synthesis problems involves computing the *optimal values* of path formulas ψ and reward functions ρ defined for the minimum in the coalition game $\mathcal{G}_C$ as follows:

$$\begin{aligned} \mathrm{Pr}^{\min}_{\mathcal{G}_C,s}(\psi) &= \inf_{\sigma_1 \in \Sigma^1_{\mathcal{G}_C}} \sup_{\sigma_2 \in \Sigma^2_{\mathcal{G}_C}} \mathrm{Pr}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(\psi), \\ \mathbb{E}^{\min}_{\mathcal{G}_C,s}(\mathit{rew}(r,\rho)) &= \inf_{\sigma_1 \in \Sigma^1_{\mathcal{G}_C}} \sup_{\sigma_2 \in \Sigma^2_{\mathcal{G}_C}} \mathbb{E}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(\mathit{rew}(r,\rho)), \end{aligned} \tag{1}$$

where we swap infimum and supremum to compute the maximum value. A strategy $\sigma_1 \in \Sigma^1_{\mathcal{G}_C}$ of player 1 starting from state s is called *optimal* if it achieves the optimal value, *e.g.*, $\sup_{\sigma_2 \in \Sigma^2_{\mathcal{G}_C}} \mathrm{Pr}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(\psi) = \mathrm{Pr}^{\min}_{\mathcal{G}_C,s}(\psi)$. Similarly, the strategy is called ε*-optimal*, for $\varepsilon > 0$, if it achieves a value deviating by at most ε from the optimum, *e.g.*, $\sup_{\sigma_2 \in \Sigma^2_{\mathcal{G}_C}} \mathrm{Pr}^{\sigma_1,\sigma_2}_{\mathcal{G}_C,s}(\psi) \geq \mathrm{Pr}^{\min}_{\mathcal{G}_C,s}(\psi) + \varepsilon$.

The optimal values and strategies can be used to solve the RPATL single-objective model checking problem in the following way. For example, to establish the verification problem for $\mathcal{G}, s$ and property $\langle\langle C \rangle\rangle \mathtt{P}_{\geq p}[\psi]$, it suffices to verify that in the coalition game $\mathcal{G}_C$ player 1 can ensure $\mathrm{Pr}^{\max}_{\mathcal{G}_C,s}(\psi) \geq p$. The remaining single-objective properties can be addressed in a similar fashion. To solve the strategy synthesis problem, we compute an optimal or a suitable ε-optimal strategy for the coalition, that is, for player 1. Since the games are zero-sum, for every single-objective RPATL property ϕ there exists a winning strategy for one of the players.

The problems of computing the optimal values and strategies for the variant of RPATL discussed here are in NP $\cap$ coNP [26, 9]. No polynomial-time algorithm is known, and the method used in practice to compute optimal values is a *value iteration* algorithm [30]. For probabilistic reachability properties, given the quantitative probabilistic reachability property $\langle\langle C \rangle\rangle \mathtt{P}_{\max=?}[\mathtt{F}\, a]$, the value iteration algorithm [30] computes the optimal values for player 1 states $s \in S_1$ in the coalition game as

$$\mathrm{Pr}^{\max}_{\mathcal{G}_C,s}(\psi) = v^*(s) = \lim_{n \to \infty} v^*_n(s), \tag{2}$$

where $v^*_n(s)$ is computed iteratively as indicated in Fig. 1, with the computation terminated when a suitable precision threshold α is reached, i.e. the maximum difference between $v^*_n(s)$ and $v^*_{n+1}(s)$, for $s \in S$, is not more than α. The computation of minimum values proceeds analogously.

It has been shown that for single-objective probabilistic reachability properties both players have optimal strategies and memoryless deterministic strategies suffice, and an optimal player 1 strategy can be constructed from the optimal values in time linear in the size of the game [4]. For more complex RPATL* properties not discussed here, where ψ is an LTL formula, pure finite-memory strategies may be required, see [4, 21, 23] and references therein. LTL properties ψ involve converting the formula to a Rabin or parity automaton, building the product of the automaton with the coalition game, and computing optimal values for a probabilistic reachability property on the product.

For single-objective cumulative reward properties $\rho = \mathtt{C}$, if a game is non-stopping then the set of states that receive infinite total reward can be computed by solving the game with

$$v_n^*(s) = \begin{cases} 1 & \text{if } s \models a \\ 0 & \text{if } s \not\models a \text{ and } n = 0, \\ \max\{v_{n-1}^*(s), v_n'(s)\} & \text{if } n > 0, \end{cases}$$

$$v_n'(s) = \begin{cases} \max_{s' \in S}\{\Delta(s,s') \cdot v_{n-1}^*(s')\} & \text{if } s \not\models a \text{ and } s \in S_1, \\ \min_{s' \in S}\{\Delta(s,s') \cdot v_{n-1}^*(s')\} & \text{if } s \not\models a \text{ and } s \in S_2, \\ \sum_{s' \in S} \Delta(s,s') \cdot v_{n-1}^*(s') & \text{if } s \not\models a \text{ and } s \in S_p. \end{cases}$$

Figure 1 Value iteration algorithm for the quantitative probabilistic reachability property $\langle\langle C \rangle\rangle \mathtt{P}_{\max=?}[\mathtt{F}\, a]$ computed on the coalition game $\mathcal{G}_C$.

respect to a parity condition [66]. After removing these states, value iteration algorithm similar to that for probabilistic reachability can be applied to compute the (bounded) optimal values for the remaining states. An optimal (memoryless deterministic) player 1 strategy for a stopping game can be constructed from the optimal values in time linear in the size of the game [4].

Longrun average reward properties $\rho = \mathtt{S}$ are more involved since average reward disregards all transient behaviour. Nevertheless, memoryless deterministic strategies still suffice for both players to win [43, 58] and an optimal strategy can be constructed by reduction to the discounted reward problem. An alternative, more practical, method, which also extends to expected and almost sure ratio rewards, was formulated for multi-objective properties [10, 9] under certain restriction on the models. The method employs stochastic memory update strategies and reduction to expected energy objectives, and is discussed in the next section.

5 Multi-objective Properties

In this section, we discuss the problem of multi-objective verification and strategy synthesis for stochastic multi-player games, previously also studied for MDPs [38, 42], where the goal is to simultaneously satisfy a certain combination of properties. Recall that we defined a multi-objective RPATL property Φ as a conjunction of properties of the same type, e.g. $\langle\langle C \rangle\rangle(\bigwedge_{i=1}^{n} \mathtt{P}_{\bowtie p_i}[\psi_i])$ or $\langle\langle C \rangle\rangle(\bigwedge_{j=1}^{m} \mathtt{R}^{r_j}_{\bowtie x_j}[\rho_j])$ (note that in [26] any positive Boolean combinations are allowed). As for single-objective properties, for a given coalition C we reduce the analysis of a multi-player stochastic game $\mathcal{G}$ to the coalition game $\mathcal{G}_C$, and thus it suffices to consider two-player stochastic games.

Let Φ_C be a multi-objective property for coalition C involving m probabilistic or reward properties and $\mathcal{G}_C$ be the induced two-player coalition game. Let $\mathbf{r} = (r_1, \ldots, r_m)$ denote the vector of reward structures and $\mathbf{r}(s) = (r_1(s), \ldots, r_m(s))$, for every $s \in S$. Similarly, $\mathbf{p} = (p_1, \ldots, p_m)$ and $\mathbf{x} = (x_1, \ldots, x_m)$ denote the vectors of probability and reward bounds. We say that the vector of bounds $(\mathbf{p}, \mathbf{x})$ for Φ_C, denoted $\Phi_C(\mathbf{p}, \mathbf{x})$, is *achievable* if and only if there exists a winning strategy for player 1 that guarantees all properties in Φ_C with bounds $\mathbf{p}, \mathbf{x}$. The optimal achievable vectors of bounds are called Pareto vectors.

▶ **Definition 8** (Pareto set). Let Φ_C be a multi-objective property for coalition C involving n probabilistic or reward properties. A vector $(\mathbf{p}, \mathbf{x}) \in \mathbb{R}^m$ is called a *Pareto vector* if the property $\Phi_C(\mathbf{p} - \varepsilon, \mathbf{x} - \varepsilon)$ is achievable in $\mathcal{G}_C$ for every $\varepsilon > 0$ and $\Phi_C(\mathbf{p} + \varepsilon, \mathbf{x} + \varepsilon)$ is not achievable for any $\varepsilon > 0$. The set P of all Pareto vectors for Φ_C is called a *Pareto set.*

$$V_n^*(s) = \begin{cases} \{\mathbf{x} \in \mathbb{R}^m_{\geq 0} \mid \mathbf{x} \leq \mathbf{r}(s)\} & \text{if } n = 0, \\ \mathsf{dwc}(\mathbf{r}(s) + \mathsf{conv}(\bigcup_{\Delta(s,s')=1} V^*_{n-1}(s'))) & \text{if } n > 0 \text{ and } s \in S_1, \\ \mathsf{dwc}(\mathbf{r}(s) + \bigcap_{\Delta(s,s')=1} V^*_{n-1}(s')) & \text{if } n > 0 \text{ and } s \in S_2, \\ \mathsf{dwc}(\mathbf{r}(s) + \sum_{\Delta(s,s')>0} \Delta(s,s') \cdot V^*_{n-1}(s')) & \text{if } n > 0 \text{ and } s \in S_p, \end{cases}$$

$$\begin{aligned} x \cdot X &= \{x \cdot \mathbf{x} \mid \mathbf{x} \in X\}, \\ \mathbf{x} + X &= \{\mathbf{x} + \mathbf{x}' \mid \mathbf{x}' \in X\}, \\ \mathsf{dwc}(X) &= \{\mathbf{y} \mid \exists \mathbf{x} \in X : \mathbf{y} \leq \mathbf{x}\}, \\ \mathsf{conv}(X) &= \{\mathbf{y} \mid \exists \mathbf{x}, \mathbf{x}' \in X, \alpha \in [0,1] : \mathbf{y} = \alpha\mathbf{x} + (1-\alpha)\mathbf{x}'\}. \end{aligned}$$

Figure 2 Iterative computation of an ε-approximation of the Pareto set for a multi-objective expected total reward property in a two-player stochastic game. Here, $x \in \mathbb{R}_{\geq 0}$ is a real number, $\mathbf{x} \in \mathbb{R}^m_{\geq 0}$ is a vector, $X \subseteq \mathbb{R}^m_{\geq 0}$ is a set, $\leq$ is the componentwise partial order on $\mathbb{R}^m_{\geq 0}$, $\mathsf{dwc}(X)$ is the downward closure of the set X, and $\mathsf{conv}(X)$ is the convex closure of X. Given a two-player stopping game $\mathcal{G}$ with multiple reward structures $\mathbf{r}$ and a multi-objective total reward property $\Phi(\mathbf{x})$, the approximation is computed for every state $s \in S$ in $k = |S| + \lceil |S| \cdot \frac{\ln(\varepsilon \cdot (n \cdot M)^{-1})}{\ln(1-\delta)} \rceil$ iterations, where $M = |S| \cdot \frac{\max_{i,s \in S} r_i(s)}{\delta}$, $\delta = \Delta_{\min}^{|S|}$, and $\Delta_{\min}$ is the smallest positive probability in $\mathcal{G}$.

The problems of multi-objective verification and strategy synthesis are formulated similarly to the single-objective case discussed in the previous section. However, unlike in the single-objective case, optimal strategies might not exist, as shown in [27] for conjunctions. Further, an infinite-memory strategy may be required, even for stopping games with reachability objectives [26]. Existing solutions therefore compute ε-approximations of Pareto sets and the corresponding ε-optimal strategies.

▶ **Definition 9** (Pareto set approximation). For $\varepsilon > 0$, an ε-approximation of the Pareto set is a set of vectors Q such that for every $(\mathbf{q}, \mathbf{y}) \in Q$ there exists a Pareto vector $(\mathbf{p}, \mathbf{x}) \in P$ with $\|(\mathbf{q}, \mathbf{y}) - (\mathbf{p}, \mathbf{x})\| \leq \varepsilon$, and vice versa, for every Pareto vector $(\mathbf{p}, \mathbf{x}) \in P$ there exists a vector $(\mathbf{q}, \mathbf{y}) \in Q$ with $\|(\mathbf{q}, \mathbf{y}) - (\mathbf{p}, \mathbf{x})\| \leq \varepsilon$, where $\| \cdot \|$ is the Manhattan distance defined as the sum of componentwise differences.

The ε-approximation of the Pareto set for a stopping coalition game $\mathcal{G}_C$ and a multi-objective property $\Phi_C(\mathbf{x})$ can be computed using the iteration algorithm in Fig. 2. The algorithm successively computes, for each state $s \in S$, the sets $V_n^*(s)$, where the nth such set is the downward closure of vectors of bounds achievable by the coalition (player 1), from s, in up to n steps. Since player 1 can randomise between its successor states, the set $V_n^*(s)$ for $s \in S_1$ is computed as a downward, convex closure of the union of $V^*_{n-1}(s')$, for all s' such that $\Delta(s, s') = 1$. For $s \in S_2$, the bounds must be achievable for all successor states, and hence we take the intersection. Finally, for probabilistic states $s \in S_p$, we consider the sum weighted by the corresponding probabilistic distribution.

Once an ε-approximation of the Pareto set has been computed, the corresponding ε-optimal player 1 strategy can be constructed [29, 26, 70], where the stochastic memory update, $\sigma_1 = (M, \sigma_1^u, \sigma_1^n, \sigma_1^{\mathsf{init}})$, representation is employed. In the construction, the vertices of approximation sets $V_n^*(s), s \in S$, act as memory elements M and represent the vector of reward bounds that the strategy currently aims to achieve. The distributions in functions σ_1^u and σ_1^{init} are constructed so that the expected value of the next memory element is an ε-approximation of the target reward bounds $\mathbf{x}$. Employing stochastic memory update representation enables a reduction in the memory required from up to infinite to finite for stopping games [26].

Since probabilistic reachability properties can be reduced to total reward properties, the iterative algorithm in Fig. 2 can be adapted to compute ε-approximations of Pareto sets for any stopping stochastic game with a multi-objective property that involves only probabilistic reachability properties. This can be extended, for stopping games, to probabilistic LTL properties [29] by employing reduction to Rabin automata and building a synchronous product of all the automata and the original SMG $\mathcal{G}$, with a new terminal state which is entered after $\mathcal{G}$ enters any of its terminal states. However, the strategy synthesis problem for multi-objective probabilistic LTL properties in general stochastic games remains open.

Unfortunately, this method cannot be used for multi-objective strategy synthesis for longrun average reward properties. This is because the algorithm in Fig. 2 approximates the Pareto set in a finite number of iterations by combining the achievable values of successive states, but expected average reward disregard all transient behaviour. Nevertheless, for the special case of almost sure average reward properties [10], strategy synthesis for multi-objective properties of this type reduces to synthesis for multi-objective expected energy properties. The corresponding decision problem is in co-NP, as shown in [9] and [19]. In addition, [9] also formulate an algorithm to construct a strategy, if it exists, where stochastically updated memory strategies are generated, which can yield exponentially more compact representations than deterministically updated strategies used in [19]. Further, [9] identify a general class of games for which the synthesis algorithm can be extended to arbitrary Boolean combinations of expected mean-payoff objectives.

Finally, in [10, 9] a method to handle multi-objective ratio reward properties is provided. To solve the strategy synthesis problem for a single-objective ratio reward property in a coalition game, it suffices to solve the problem for an almost sure average reward property, and this can be extended to (conjunctive) multi-objective properties using vectors.

6 Compositional Strategy Synthesis

One difficulty with multi-objective strategy synthesis, in view of high computational complexity [26], is its scalability. To deal with this problem, [11, 70] formulate a *compositional* framework for strategy synthesis, which allows one to derive a strategy for the composed system by synthesising only for the (smaller) individual components. Firstly, recall that probabilistic automata of [63] correspond to coalition games with only one player present. In verification, the nondeterminism that is present in the probabilistic automaton models an adverse, uncontrollable, environment. By applying a coalition strategy to a game to resolve the controllable nondeterminism, we are left with a probabilistic automaton where only uncontrollable nondeterminism for the remaining players remains. This observation allows us to reuse rules for compositional verification of probabilistic automata, such as those in [56], to derive strategy synthesis rules for SMGs.

The compositional framework of [11, 9, 70] is based on a parallel composition operation for stochastic games, which is closely related to that of probabilistic automata [63], except that the identity of the player is preserved through composition. When composed, the component SMGs $\mathcal{G} = (\Pi, S, (S_\Pi, S_p), s_{\text{init}}, \Delta, \mathrm{A}, L)$ and $\mathcal{G}' = (\Pi', S', (S'_\Pi, S'_p), s'_{\text{init}}, \Delta', \mathrm{A}', L')$ synchronise on shared actions $A \cup A'$, yielding the composed game $\mathcal{G}'' = \mathcal{G} \parallel \mathcal{G}'$. Properties of the component SMGs, as well as the composed game, are then defined on traces, that is, sequences of actions that label the probabilistic states in the path, rather than over paths. Under the assumption that the component games are compatible, *i.e.,* all actions of player 1 in each composite game are enabled and fully controlled by player 1, the player 1 strategy $\sigma_1'' = \sigma_1 \parallel \sigma_1'$ for $\mathcal{G}''$ that is a composition of player 1 strategies for component games preserves all properties. More precisely, if strategies σ_1 and σ_1' guarantee (possibly multi-objective) properties Φ, Φ' in component games, then the composed strategy σ_1'' guarantees property Φ'' in $\mathcal{G}''$, where Φ'' is *any* property for the composed game that can be derived from Φ and Φ' using, for example, assume-guarantee rules in [56]. In particular, player 1 of different component games can cooperate to achieve a common goal: if in one component game player 1 guarantees a property Φ_2 under some assumption Φ_1 on the environment, *i.e.*, $\Phi_1 \Rightarrow \Phi_2$, and player 1 in a different component game ensures Φ_1, then the composition satisfies property Φ_2. A broad range of such assume-guarantee contracts can be supported for both probabilistic and reward multi-objective properties.

The method for compositional strategy synthesis [11] first computes an under-approximation Q of the Pareto set for Φ'' based on ε-approximations Q, Q' of Pareto sets for Φ, Φ'. For a chosen achievable vector of bounds $(\mathbf{p}, \mathbf{x})$ for Φ'', player 1 strategies σ_1, σ_i' are synthesised for component games that achieve $\Phi(\mathbf{p}, \mathbf{x})$, $\Phi'(\mathbf{p}', \mathbf{x}')$, where $(\mathbf{p}, \mathbf{x})$, $(\mathbf{p}', \mathbf{x}')$ are the respective bounds obtained by projecting $(\mathbf{p}, \mathbf{x})$ from Q'' to Q, Q'. The composed strategy $\sigma_1'' = \sigma_1 \parallel \sigma_1'$ then achieves $\Phi''(\mathbf{p}'', \mathbf{x}'')$. Note that, since assume-guarantee contracts may involve implication, to be able to apply this framework and, in particular, to take full advantage of assume-guarantee rules, we would need to be able to synthesise strategies for arbitrary Boolean combinations of properties, which is possible [9, 70] under certain restrictions on models and properties.

7 Tool Implementation and Applications

All techniques overviewed in this paper, including single- and multi-objective, as well as compositional, strategy synthesis problems for turn-based stochastic multi-player games, have been implemented in the open-source tool called PRISM-games [25, 54], developed as an extension of the probabilistic model checker PRISM [52]. PRISM-games can be used to model, verify, solve and simulate stochastic multi-player games with complex properties. As a modelling notation, PRISM-games uses an extension of PRISM's modelling language based on reactive modules. The specification notation is based on RPATL [23], and includes support for the coalition operator; single-objective properties, namely probabilistic reachability, total reward properties for stopping games, average reward and ratio properties for a special class of games called controllable multichain games (for details, see [70]), and almost sure average reward and ratio properties; and multi-objective properties, and specifically Boolean combinations of the same type of reward properties, except for the almost sure average and ratio reward properties for which only conjunctions are supported.

Currently, PRISM-games is an explicit state model checker, which extends the Java-based engine of PRISM, and relies on the Parma Polyhedra Library [6] for symbolic manipulation of convex sets during ε-approximate computation of Pareto sets, see [66, 70].

Below we report on a variety of case studies of autonomous systems that employed stochastic game models and were analysed using PRISM-games. For more information, see [66, 70, 68, 67] and the PRISM-games website [61].

Microgrid demand-side management [66]. The example models a decentralised energy management protocol for smart grids that draw energy from a variety of sources. The system consists of a set of households, where each household follows a simple probabilistic protocol to execute a load if the current energy cost is below a pre-agreed limit, and otherwise it only executes the load with a pre-agreed probability. The energy cost to execute a load for a single time unit is the number of loads currently being executed in the grid. The analysis of the protocol with respect to the expected load per cost unit for a household, formulated as a single-objective total reward property, exposed a protocol weakness. The weakness was then corrected by disincentivising non-cooperative behaviour.

Human-in-the-loop UAV mission planning [40]. This case study concerns autonomous unmanned aerial vehicles (UAV) performing road network surveillance and reacting to inputs from a human operator. The UAV performs most of the piloting functions, such as selecting the waypoints and flying the route. The operator primarily performs sensor tasks at waypoints but may also pick a road for the UAV at waypoints. The optimal UAV piloting strategy depends on mission objectives, *e.g.,* safety, reachability, coverage, and operator characteristics, *i.e.,* workload, proficiency, and fatigue. The main focus of the case study is on studying a multi-objective property to analyse the trade-off between the mission completion time and the number of visits to restricted operating zones, which have been investigated by computing Pareto sets.

Autonomous urban driving [29]. A stochastic game model of an autonomous car is developed, which considers the car driving through an urban environment and reacting to hazards such as pedestrians, obstacles, and traffic jams. The car does not only decide on the reactions to hazards, which are adversarial, but also chooses the roads to take in order to reach a target location. The presence and probability of hazards is based on statistical information for the road. Through multi-objective strategy synthesis, strategies with optimal trade-off between the probability of reaching the target location, the probability of avoiding accidents and the overall quality of roads on the route are identified.

Aircraft power distribution [10]. An aircraft electrical power network is considered, where power is to be routed from generators to buses through controllable switches. The generators can exhibit failures and switches have delays. The system consists of several components, each containing buses and generators, and the components can deliver power to each other. The network is modelled as a composition of stochastic games, one for each component. These components are physically separated for reliability, and hence allow limited interaction and communication. Compositional strategy synthesis is applied to find strategies with good trade-off between uptime of buses and failure rate. By employing stochasticity, we can faithfully encode the reliability specifications in quantitative fashion, thus improving over previous results. The property is modelled as a conjunction of ratio reward properties.

Self-adaptive software architectures [44, 16]. Self-adaptive software automatically adapts its structure and behaviour according to changing requirements and quantitative goals. Several self-adaptive software architectures, such as adaptive industrial middleware used to monitor and manage sensor networks in renewable energy production plants, have been modelled as stochastic games and analysed. Both single- and multi-objective verification of multi-player stochastic games has been applied to to evaluate their resilience properties and synthesise pro-active adaptation policies.

DNS Bandwidth Amplification Attack [35]. The Domain Name System (DNS) is an Internet-wide hierarchical naming system for assigning IP addresses to domain names, and any disruption of the service can lead to serious consequences. A notable threat to DNS, namely the bandwidth amplification attack, where an attacker attempts to flood a victim DNS server with malicious traffic, is modelled as a stochastic game. Verification and strategy synthesis is used to analyse and generate counter-measures to defend against the attack.

Attack-defence scenarios in RFID goods management system [5]. This case study considers complex attack-defence scenarios, such as an RFID goods management system, translating attack-defence trees to two-player stochastic games. Probabilistic verification is then employed to check security properties of the attack-defence scenarios and to synthesise strategies for attackers or defenders which guarantee or optimise some quantitative property. The properties considered include single-objective properties such as the probability of a successful attack or the incurred cost, as well as their multi-objective combinations.

8 Challenges

Clearly, there has been much progress towards quantitative verification and aspects of quantitative synthesis for autonomy, with a wide variety of relevant case studies serving as proof of concept. However, a number of significant challenges have yet to be overcome. We briefly review a selection of these below.

Partial information. Practical quantitative verification, as exemplified by PRISM, has so far mostly been limited to complete information systems. This restriction is not applicable to many autonomous scenarios, where agents in the system only have partial information. Partial observability [18, 20] raises a number of algorithmic challenges that need to be tackled.

Modelling social interactions. Autonomous systems are increasingly often employed to assist and interact with humans, and operator models have to be taken into account. Though some progress has been made, for example in the context of UAVs [40], models that incorporate cognitive processes and social interactions, such as those based on trust [39, 46], and the corresponding verification techniques are needed.

Model learning and adaptation from data. Quantitative verification has so far mainly focused on modelling system dynamics, but the behaviour of many autonomous and semi-autonomous systems, such as those involving perception, is data-driven [62]. Techniques that integrate model learning from data in order to inform adaptation in real-time and programming with uncertain data within quantitative verification methodologies are needed.

Model synthesis from specifications. Though correct-by-construction synthesis of strategies has been tackled in a range of models, model synthesis from quantitative specifications requires further study. A possible approach is combining template-based and parameter synthesis methods already developed for Markov chains and MDPs [34, 28, 17] and via discretisation for timed and hybrid automata [36, 55], but more effort is required to tackle autonomous systems.

Scalability, efficiency and precision. Existing model checking and strategy synthesis tools for stochastic games are in early stages of development and substantial effort is necessary to ensure their effectiveness in industrial applications. Compositional approaches, symbolic techniques and methodologies based on induction, deduction and machine learning, as well as their judicious combinations, have great potential.

9 Conclusion

As autonomous systems are becoming an integral part of our society, their failure carries potentially unacceptable and life-endangering risks. Rigorous model-based verification technologies incorporated within the design process can improve their safety and reliability and reduce development costs. This paper has briefly summarised quantitative verification and strategy synthesis techniques developed for autonomous systems modelled as turn-based stochastic multi-player games as implemented in the tool PRISM-games and outlined future research challenges in this challenging yet exciting field.

References

1 R. Alur, T. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

2 Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

3 Tobias Amnell, Gerd Behrmann, Johan Bengtsson, Pedro R. D'Argenio, Alexandre David, Ansgar Fehnker, Thomas Hune, Bertrand Jeannet, Kim G. Larsen, M. Oliver Möller, Paul Pettersson, Carsten Weise, and Wang Yi. UPPAAL – Now, Next, and Future. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Modelling and Verification of Parallel Processes*, number 2067 in Lecture Notes in Computer Science Tutorial, pages 100–125. Springer-Verlag, 2001.

4 Daniel Andersson and Peter Bro Miltersen. The Complexity of Solving Stochastic Games on Graphs. In *Algorithms and Computation*, volume 5878 of *LNCS*, pages 112–121. 2009.

5 Zaruhi Aslanyan, Flemming Nielson, and David Parker. Quantitative Verification and Synthesis of Attack-Defence Scenarios. In *Proc. of Computer Security Foundations Symposium CSF*, 2016. to appear.

6 Roberto Bagnara, Patricia M. Hill, and Enea Zaffanella. The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. *Science of Computer Programming*, 72(1–2):3–21, 2008.

7 C. Baier, E. Clarke, V. Hartonas-Garmhausen, M. Kwiatkowska, and M. Ryan. Symbolic model checking for probabilistic processes. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Proc. 24th Int. Colloq. Automata, Languages and Programming (ICALP'97)*, volume 1256 of *LNCS*, pages 430–440. Springer, 1997.

8 S. Basagiannis, P. Katsaros, A. Pombortsis, and N. Alexiou. Probabilistic model checking for the quantification of DoS security threats. *Computers & Security*, 2009.

9 N. Basset, M. Kwiatkowska, and C. Wiltsche. Compositional strategy synthesis for stochastic games with multiple objectives. Technical Report CS-RR-16-03, Department of Computer Science, University of Oxford, 2016.

10 Nicolas Basset, Marta Z. Kwiatkowska, Ufuk Topcu, and Clemens Wiltsche. Strategy Synthesis for Stochastic Games with Multiple Long-Run Objectives. In *Proc. of Tools and Algorithms for the Construction and Analysis of Systems TACAS*, pages 256–271, 2015.

11 Nicolas Basset, Marta Z. Kwiatkowska, and Clemens Wiltsche. Compositional Controller Synthesis for Stochastic Games. In *Proc. of Concurrency Theory CONCUR*, pages 173–187, 2014.

12 Andrea Bianco and Luca de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proc. of Foundations of Software Technology and Theoretical Computer Science FSTTCS*, volume 1026 of *LNCS*, pages 499–513, 1995.

13 P. Billingsley. *Probability and Measure.* Wiley, 1995.

14 Tomás Brázdil, Václav Brozek, Vojtech Forejt, and Antonín Kucera. Stochastic Games with Branching-Time Winning Objectives. In *Proc. of Logic in Computer Science LICS*, pages 349–358, 2006.

15 R. Calinescu, C. Ghezzi, M. Kwiatkowska, and R. Mirandola. Self-adaptive software needs quantitative verification at runtime. *Communications of the ACM*, 55(9):69–77, 2012.

16 Javier Cámara, Gabriel A. Moreno, and David Garlan. Stochastic Game Analysis and Latency Awareness for Proactive Self-adaptation. In *Proc. of Software Engineering for Adaptive and Self-Managing Systems SEAMS*, pages 155–164, 2014.

17 M. Ceska, F. Dannenberg, N. Paoletti, M. Kwiatkowska, and L. Brim. Precise parameter synthesis for stochastic biochemical systems. *Acta Informatica, to appear*, 2016.

18 Krishnendu Chatterjee and Laurent Doyen. Partial-Observation Stochastic Games: How to Win when Belief Fails. *Transactions on Compuational Logic*, 15(2):16, 2014.

19 Krishnendu Chatterjee and Laurent Doyen. Perfect-information Stochastic Games with Generalized Mean-Payoff Objectives. In *Proc. of LICS*. 2016. To appear.

20 Krishnendu Chatterjee, Laurent Doyen, Sumit Nain, and Moshe Y. Vardi. The Complexity of Partial-Observation Stochastic Parity Games with Finite-Memory Strategies. In *Proc. of Foundations of Software Science and Computation Structures FOSSACS*, pages 242–257, 2014.

21 Krishnendu Chatterjee and Thomas A. Henzinger. A survey of stochastic ω-regular games. *Journal of Computer and System Sciences*, 78(2):394–413, 2012.

22 Krishnendu Chatterjee and Rasmus Ibsen-Jensen. Qualitative analysis of concurrent mean-payoff games. *Information and Computation*, 242:2–24, 2015.

23 T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis. Automatic Verification of Competitive Stochastic Systems. In *Proc. of Tools and Algorithms for the Construction and Analysis of Systems TACAS*, volume 7214 of *LNCS*, pages 315–330, 2012.

24 T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis. Automatic verification of competitive stochastic systems. *Formal Methods in System Design*, 43(1):61–92, 2013.

25 T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis. PRISM-games: A Model Checker for Stochastic Multi-Player Games. In *Proc. of Tools and Algorithms for the Construction and Analysis of Systems TACAS*, volume 7795 of *LNCS*, pages 185–191, 2013.

26 Taolue Chen, Vojtech Forejt, Marta Z. Kwiatkowska, Aistis Simaitis, and Clemens Wiltsche. On Stochastic Games with Multiple Objectives. In *Proc. of Mathematical Foundations of Computer Science MFCS*, pages 266–277, 2013.

27 Taolue Chen, Vojtěch Forejt, Marta Kwiatkowska, Aistis Simaitis, Ashutosh Trivedi, and Michael Ummels. Playing Stochastic Games Precisely. In *Proc. of Concurrency Theory CONCUR*, volume 7454 of *LNCS*, pages 348–363. 2012.

28 Taolue Chen, Ernst Moritz Hahn, Tingting Han, Marta Kwiatkowska, Hongyang Qu, and Lijun Zhang. Model repair for Markov decision processes. In *Proc. 7th Int. Symp. Theoretical Aspects of Software Engineering (TASE'13)*, pages 85–92. IEEE Computer Society Press, 2013.

29 Taolue Chen, Marta Z. Kwiatkowska, Aistis Simaitis, and Clemens Wiltsche. Synthesis for Multi-objective Stochastic Games: An Application to Autonomous Urban Driving. In *Proc. of Quantitative Evaluation of Systems QEST*, pages 322–337, 2013.

30 Anne Condon. The Complexity of Stochastic Games. *Information and Computation*, 96:203–224, 1992.

31 C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite state probabilistic programs. In *Proc. 29th Annual Symp. Foundations of Computer Science (FOCS'88)*, pages 338–345. IEEE Computer Society Press, 1988.

32 F. Dannenberg, M. Kwiatkowska, C. Thachuk, and A. J. Turberfield. Dna walker circuits: Computational potential, design and verification. *Natural Computing*, 14(2):195–211, 2015.

33 L. de Alfaro, M. Kwiatkowska, G. Norman, D. Parker, and R. Segala. Symbolic model checking of probabilistic processes using MTBDDs and the Kronecker representation. In S. Graf and M. Schwartzbach, editors, *Proc. 6th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'00)*, volume 1785 of *LNCS*, pages 395–410. Springer, 2000.

34 C. Dehnert, S. Junges, N. Jansen, F. Corzilius, M. Volk, H. Bruintjes, J-P. Katoen, and E. Ábrahám. PROPhESY: A PRObabilistic ParamEter SYnthesis tool. In *Proc. 27th Int. Conf. Computer Aided Verification (CAV'15)*, volume 9206 of *LNCS*, pages 214–231. Springer, 2015.

35 T. Deshpande, P. Katsaros, S.A. Smolka, and S.D. Stoller. Stochastic Game-Based Analysis of the DNS Bandwidth Amplification Attack Using Probabilistic Model Checking. In *Proc. of European Dependable Computing Conference EDCC*, pages 226–237, 2014.

36 M. Diciolla, C. H. P. Kim, M. Kwiatkowska, and A. Mereacre. Synthesising optimal timing delays for timed I/O automata. In *Proc. 14th International Conference on Embedded Software (EMSOFT'14)*. ACM, 2014.

37 M. Duflot, M. Kwiatkowska, G. Norman, and D. Parker. A formal analysis of Bluetooth device discovery. *Int. Journal on Software Tools for Technology Transfer*, 8(6):621–632, 2006.

38 K. Etessami, M. Kwiatkowska, M. Vardi, and M. Yannakakis. Multi-objective model checking of Markov decision processes. *Logical Methods in Computer Science*, 4(4):1–21, 2008.

39 R. Falcone and C. Castelfranchi. Social trust: A cognitive approach. In *Trust and Deception in Virtual Societies*, pages 55–90. Kluwer, 2001.

40 Lu Feng, Clemens Wiltsche, Laura Humphrey, and Ufuk Topcu. Controller Synthesis for Autonomous Systems Interacting with Human Operators. In *Proc. of Int. Conf. on Cyber-Physical Systems ICCPS*, pages 70–79, 2015.

41 V. Forejt, M. Kwiatkowska, G. Norman, and D. Parker. Automated Verification Techniques for Probabilistic Systems. In *Proc. of Formal Methods for Eternal Networked Software System SFM*, volume 6659 of *LNCS*, pages 53–113, 2011.

42 V. Forejt, M. Kwiatkowska, G. Norman, D. Parker, and H. Qu. Quantitative multi-objective verification for probabilistic systems. In P. Abdulla and K. Leino, editors, *Proc. 17th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'11)*, volume 6605 of *LNCS*, pages 112–127. Springer, 2011.

43 D. Gillette. Stochastic games with zero stop probabilities. *Contributions to the Theory of Games*, 39:179–187, 1957.

44 T.J. Glazier, J. Camara, B. Schmerl, and D. Garlan. Analyzing Resilience Properties of Different Topologies of Collective Adaptive Systems. In *Proc. of Self-Adaptive and Self-Organizing Systems Workshops SASOW*, pages 55–60, 2015.

45 J. Heath, M. Kwiatkowska, G. Norman, D. Parker, and O. Tymchyshyn. Probabilistic model checking of complex biological pathways. *Theoretical Computer Science*, 319(3):239–257, 2008.

46 X. Huang and M. Kwiatkowska. Reasoning about cognitive trust in stochastic multiagent systems. Technical Report CS-RR-16-02, Department of Computer Science, University of Oxford, 2016.

47 M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker. Abstraction refinement for probabilistic software. In N. Jones and M. Muller-Olm, editors, *Proc. 10th Int. Conf. Verification, Model Checking, and Abstract Interpretation (VMCAI'09)*, volume 5403 of *LNCS*, pages 182–197. Springer, 2009.

48 M. Kwiatkowska. Model checking for probability and time: From theory to practice. In *Proc. 18th Annual IEEE Symp. Logic in Computer Science (LICS'03)*, pages 351–360. IEEE Computer Society Press, 2003. Invited Paper.

49 M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic symbolic model checker. In T. Field, P. Harrison, J. Bradley, and U. Harder, editors, *Proc. 12th Int. Conf. Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'02)*, volume 2324 of *LNCS*, pages 200–204. Springer, 2002.

50 M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with PRISM: A hybrid approach. *International Journal on Software Tools for Technology Transfer (STTT)*, 6(2):128–142, 2004.

51 M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Proc. 23rd Int. Conf. Computer Aided Verification (CAV'11)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

52 M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of Probabilistic Real-time Systems. In *Proc. of Computer Aided Verification CAV*, volume 6806 of *LNCS*, pages 585–591, 2011.

53 M. Kwiatkowska, G. Norman, and R. Segala. Automated verification of a randomized distributed consensus protocol using Cadence SMV and PRISM. In G. Berry, H. Comon, and A. Finkel, editors, *Proc. 13th Int. Conf. Computer Aided Verification (CAV'01)*, volume 2102 of *LNCS*, pages 194–206. Springer, 2001.

54 M. Kwiatkowska, D. Parker, and C. Wiltsche. PRISM-games 2.0: A Tool for Multi-Objective Strategy Synthesis for Stochastic Games. In *Proc. of Tools and Algorithms for the Construction and Analysis of Systems TACAS*, 2016. to appear.

55 Marta Kwiatkowska, Alexandru Mereacre, Nicola Paoletti, and Andrea Patanè. Synthesising robust and optimal parameters for cardiac pacemakers using symbolic and evolutionary computation techniques. In *Proceedings of the 4th International Workshop on Hybrid Systems and Biology (HSB 2015)*, volume 9271 of *LNCS/LNBI*, pages 119–140. Springer, 2015.

56 Marta Kwiatkowska, Gethin Norman, David Parker, and Hongyang Qu. Compositional probabilistic verification through multi-objective model checking. *Information and Computation*, 232:38–65, 2013.

57 M. Lakin, D. Parker, L. Cardelli, M. Kwiatkowska, and A. Phillips. Design and analysis of DNA strand displacement devices using probabilistic model checking. *Journal of the Royal Society Interface*, 9(72):1470–1485, 2012.

58 Thomas M. Liggett and Steven A. Lippman. Stochastic Games with Perfect Information and Time Average Payoff. *SIAM Review*, 11(4):604–607, 1969.

59 G. Norman, D. Parker, M. Kwiatkowska, and S. Shukla. Evaluating the reliability of NAND multiplexing with PRISM. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(10):1629–1637, 2005.

60 D. Parker. *Implementation of Symbolic Model Checking for Probabilistic Systems.* PhD thesis, University of Birmingham, 2002.

61 PRISM-games website. `http://www.prismmodelchecker.org/games/`.

62 Dorsa Sadigh, Katherine Driggs-Campbell, Alberto Puggelli, Wenchao Li, Victor Shia, Ruzena Bajcsy, Alberto L. Sangiovanni-Vincentelli, S. Shankar Sastry, and Sanjit A. Seshia. Data-driven probabilistic modeling and verification of human driver behavior. In *Formal Verification and Modeling in Human-Machine Systems, AAAI Spring Symposium*, 2014.

63 R. Segala. *Modelling and Verification of Randomized Distributed Real Time Systems.* PhD thesis, Massachusetts Institute of Technology, 1995.

64 L. S. Shapley. Stochastic games. In *National Academy of Sciences*, pages 1095–1100, 1953.

65 V. Shmatikov. Probabilistic analysis of anonymity. In *Proc. 15th IEEE Computer Security Foundations Workshop (CSFW'02)*, pages 119–128. IEEE Computer Society Press, 2002.

66 A. Simaitis. *Automatic Verification of Competitive Stochastic Systems.* PhD thesis, Department of Computer Science, University of Oxford, 2014.

67 M. Svorenova and M. Kwiatkowska. Quantitative verification and strategy synthesis for stochastic games. *European Journal of Control*, 2016. To appear.

68 M. Ujma. *On Verification and Controller Synthesis for Probabilistic Systems at Runtime.* PhD thesis, University of Oxford, 2015.

69 Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. *Foundations of Computer Science, IEEE Annual Symposium on*, 0:327–338, 1985.

70 C. Wiltsche. *Assume-Guarantee Strategy Synthesis for Stochastic Games.* PhD thesis, Department of Computer Science, University of Oxford, 2015.