On Model Checking Techniques for Randomized Distributed Systems

Christel Baier

Technische Universität Dresden, Faculty of Computer Science, Germany

Abstract. The automata-based model checking approach for randomized distributed systems relies on an operational interleaving semantics of the system by means of a Markov decision process and a formalization of the desired event E by an ω-regular linear-time property, e.g., an LTL formula. The task is then to compute the greatest lower bound for the probability for E that can be guaranteed even in worst-case scenarios. Such bounds can be computed by a combination of polynomially time-bounded graph algorithm with methods for solving linear programs. In the classical approach, the "worst-case" is determined when ranging over all schedulers that decide which action to perform next. In particular, all possible interleavings and resolutions of other nondeterministic choices in the system model are taken into account. The worst-case analysis relying on this general notion of schedulers is often too pessimistic and leads to extreme probability values that can be achieved only by schedulers that are unrealistic for parallel systems. This motivates the switch to more realistic classes of schedulers that respect the fact that the individual processes only have partial information about the global system states. Such classes of partial-information schedulers yield more realistic worst-case probabilities, but computationally they are much harder. A wide range of verification problems turns out to be undecidable when the goal is to check that certain probability bounds hold under all partial-information schedulers.

Probabilistic phenomenon appear rather natural in many areas of computer science. Randomized algorithms, performance evaluation, security protocols, control theory, stochastic planning, operations research, system biology or resilient systems are just a few examples. Although a wide range of different stochastic models are used in these areas, it is often possible to deal with Markovian models. These rely on the memoryless property stating that the future system behavior only depends on the current state, but not on the past. If the state space is finite, then Markovian models can be viewed as a variant of classical finite automata augmented with distributions which makes them best suited for the application of model checking techniques.

In this extended abstract, we summarize the main steps of the automata-based model checking approach for the quantitative analysis of Markov decision processes in worst-case scenarios, and point out the difficulties that arise when taking the local views of the processes into account.

D. Méry and S. Merz (Eds.): IFM 2010, LNCS 6396, pp. 1–11, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Markov decision processes (MDPs) can be understood as a probabilistic extension of labeled transition systems. Nondeterminism can be represented in an MDP by the choice between different actions. The actions of an MDP can have a probabilistic effect, possibly depending on the state in which they are executed.

The coexistence of nondeterminism and probabilism in an MDP allows for representing concurrent (possibly randomized) activities of different processes by interleaving, i.e., the choice which process performs the next step. Besides interleaving, the nondeterminism in an MDP can also be useful for abstraction purposes, for underspecification to be resolved in future refinement steps, or for modeling the interface with an unknown environment. Formally, an MDP is a tuple $\mathcal{M} = (S, \mathsf{Act}, \mathsf{P}, s_0, \ldots)$ where

– S is a finite nonempty set of states,
– Act is a finite nonempty set of actions,
– $\mathsf{P} : S \times \mathsf{Act} \times S \to [0,1]$ is a function, called *transition probability function*, such that for all states $s \in S$ and actions $\alpha \in \mathsf{Act}$, the function $s \mapsto \mathsf{P}(s, \alpha, \cdot)$ is either the null-function or a probabilistic distribution, i.e.,

$$\sum_{s' \in S} \mathsf{P}(s, \alpha, s') \in \{0, 1\} \text{ for all states } s \in S \text{ and actions } \alpha \in \mathsf{Act},$$

– $s_0 \in S$ is the initial state.

Alternatively, one can deal with a distribution over initial states. Further components can be added, such as reward or cost functions or atomic propositions. MDPs with a rewards for the states and actions can be useful to model sojourn times in states or other quantitative measures such as energy consumption. Atomic propositions can serve as state predicates and are often used to formalize properties in some temporal or modal logic.

If $s \in S$ then $\mathsf{Act}(s)$ denotes the set of actions that are *enabled* in state s, i.e.,

$$\mathsf{Act}(s) \overset{\text{def}}{=} \big\{\alpha \in \mathsf{Act} : \mathsf{P}(s, \alpha, s') > 0 \text{ for some } s' \in S\big\}.$$

For technical reasons, it is often useful to assume that there are no terminal states, i.e., for each state $s \in S$ the set $\mathsf{Act}(s)$ in nonempty.

The intuitive operational behavior of an MDP $\mathcal{M}$ as above is the following. The computation starts in the initial state s_0. If after n steps the current state is s_n then first an enabled action $\alpha_{n+1} \in \mathsf{Act}(s_n)$ is chosen nondeterministically. The effect of action α_{n+1} in state s_n is given by the distribution $\mathsf{P}(s_n, \alpha_{n+1}, \cdot)$. Thus, the next state s_{n+1} belongs to the support of $\mathsf{P}(s_n, \alpha_{n+1}, \cdot)$ and is chosen probabilistically. The resulting sequence of states $\pi = s_0\, s_1\, s_2 \ldots \in S^\omega$ is called a path of $\mathcal{M}$.

Markov decision processes are widely used as an operational model for parallel systems where some components behave probabilistically, e.g., if they rely on a randomized algorithms or communicate via an unreliable fifo channel that looses or corrupts messages with some small (fixed) probability.

Example 1 (MDP for a randomized mutual exclusion protocol). Figure 1 shows an MDP modeling a simple mutual exclusion protocol for two processes P_1, P_2 that compete to access a certain shared ressource. Process P_i is represented by three local states n_i, w_i, c_i. Local state n_i stands for the non-critical phase of process P_i, w_i represents the location where process P_i is waiting and c_i denotes the critical section. The local states n_i and c_i can be left by performing a request or release action, respectively. If P_i is waiting, i.e., its current local state is w_i, and the other process P_j is not its critical section then P_i can enter the local state c_i (action *enter*$_i$). If both P_1 and P_2 are waiting then the competition is resolved by a randomized arbiter who tosses a fair coin to decide whether P_1 will enter its critical section (if the outcome is head) or P_2 gets the grant (if the outcome is tail). All other actions *request*$_i$, *release*$_i$, and *enter*$_i$ have a deterministic effect in the sense that, in all states s where they are enabled, the associated distribution is a Dirac distribution, i.e., assigns probability 1 to the unique successor state.

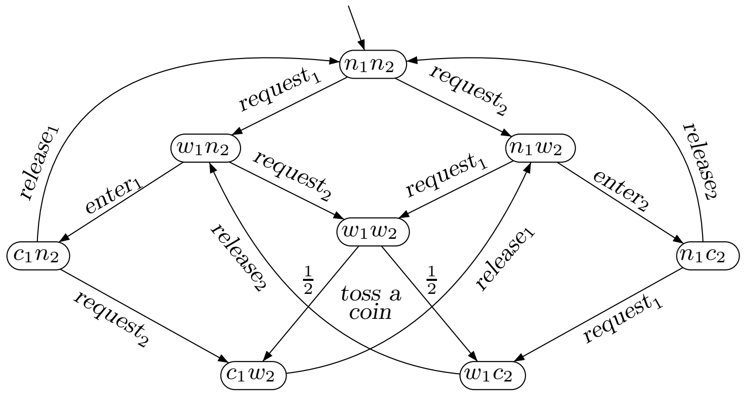


Fig. 1. MDP for mutual exclusion with randomized arbiter

Schedulers. Reasoning about the probabilities for path events (i.e., conditions that might or might not hold for a path) in an MDP requires the concept of *schedulers*, also often called *policies* or *adversaries*. A scheduler refers to any, possibly history-dependent, strategy for resolving the nondeterministic choices. Formally, a scheduler for an MDP $\mathcal{M} = (S, \mathsf{Act}, \mathsf{P}, s_0, \ldots)$ can be defined as a function

$$D : S^+ \to \mathsf{Act} \text{ such that } D(s_0\, s_1 \ldots s_n) \in \mathsf{Act}(s_n).$$

The input $s_0\, s_1 \ldots s_n$ of D stands for the "history", i.e., the sequence of states that have been visited in the past. The last state s_n represents the current state s_n. (The values of D are only relevant for finite D-paths, i.e., sequences $s_0\, s_1 \ldots s_n$ such that $\mathsf{P}(s_i, D(s_0, \ldots, s_n), s_{i+1}) > 0$ for $0 \leq i < n$.)

In the literature, more general types of schedulers have been defined, e.g., randomized schedulers that assign distributions of actions to finite paths rather than single actions. Furthermore, the input of a scheduler can also include the actions that have been taken in the past. Vice versa, one can also restrict the class of schedulers to those that are realizable by a finite-state machine. For reasoning about probabilities for ω-regular path events in worst-case scenarios, these differences in the definition of schedulers are irrelevant, as long as they rely on complete information about the states of $\mathcal{M}$.

Given a scheduler D, the operational behavior of $\mathcal{M}$ under D can be unfolded into a tree-like infinite Markov chain. This allows to apply standard techniques for Markov chains to define a σ-algebra over infinite paths and the probability of path events, i.e., measurable sets of infinite paths. Details can be found in any textbook on Markov decision processes, see e.g. [27].

Quantitative worst-case analysis. The typical task of the *quantitative analysis* of an MDP is to determine the probabilities for a certain path event E in a worst-case scenario, i.e., the maximal or minimal probabilities for E when ranging over all schedulers. Thus, the purpose of a quantitative analysis is to provide lower or upper probability bounds that are guaranteed for all interleavings of concurrent activities, all refinements of nondeterministic choices that have been obtained from abstraction techniques, and no matter how the environment behaves, provided that the environment has been modeled nondeterministically. When Markov decision processes are augmented with cost functions, then the quantitative analysis can also establish lower or upper bounds on expected values (e.g., costs for reaching a certain goal set or long-run averages) that can be guaranteed for all schedulers. The notion *qualitative analysis* refers to the task where one has to check whether a given event E holds almost surely, i.e., with probability 1 for all schedulers, or whether E holds with zero probability, no matter which scheduler is used.

For an example, let us regard the mutual exclusion protocol modeled by the MDP shown in Figure 1 again.

- The safety property E_{safe} stating that the two processes are never simultaneously in their critical section needs no probabilistic features and can be establsihed by standard (non-probabilistic) model checking techniques, as the mutual exclusion property holds along all paths. Note that the state $\langle c_1, c_2 \rangle$ is not reachable.
- The liveness property E_{live} stating that each waiting process will eventually enter its critical section does not hold along all paths. E.g., in any infinite path that runs forever through the cycle $\langle n_1, w_2 \rangle \langle w_1, w_2 \rangle \langle c_1, w_2 \rangle \langle n_1, w_2 \rangle$ the second process is waiting forever. However, such paths have probability measure 0 under all schedulers. Hence, event E_{live} holds almost surely (i.e., with probability 1) under each scheduler. This yields that the minimal probability for E_{live} is 1.
- Suppose now that process P_2 is waiting in the current state, i.e., we treat state $\langle n_1, w_2 \rangle$ as initial state.
  For each scheduler, the probability that P_2 will enter its critical section within the next n rounds is at least $1 - \frac{1}{2^n}$. Here, by a "round" we mean

any simple cycle containing the state $\langle w_1, w_2\rangle$ where the randomized arbiter tosses a coin. The worst case scenario for process P_2 is a scheduler that always schedules action *request*$_1$ in state $\langle n_1, w_2\rangle$. Under this scheduler, process P_1 is the winner of the first n coin tossing experiment with probability $\frac{1}{2^n}$. It sould be noticed that under other schedulers, process P_2 can have better chances to enter its critical section within the next n rounds. For example, if action *enter*$_2$ is scheduled in state $\langle n_1, w_2\rangle$ then process P_2 will enter its critical section in the first round.
The expected number of rounds that P_2 has to wait after having requested its critical section is less or equal

$$\sum_{n=1}^{\infty} n \cdot \frac{1}{2^n} \;=\; 2$$

for each scheduler. Value 2 is obtained under the scheduler which always schedules the action *request*$_1$ in state $\langle n_1, w_2\rangle$. Hence, the MDP in Figure 1 enjoys the property that the minimal expected number of rounds that P_2 has to wait before entering its critical section is 2.

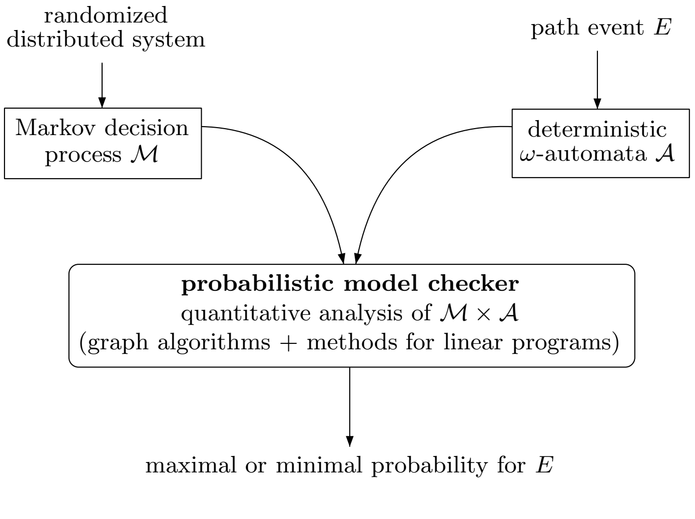

randomized distributed system
path event E
Markov decision process $\mathcal{M}$
deterministic ω-automata $\mathcal{A}$
probabilistic model checker
quantitative analysis of $\mathcal{M} \times \mathcal{A}$
(graph algorithms + methods for linear programs)
maximal or minimal probability for E

Fig. 2. Schema for the quantitative analysis

Probabilistic model checking. Probabilistic model checking techniques for finite-state Markov decision processes have been designed for verifying qualitative and quantitative properties specified in probabilistic computation tree logic [8,15,7] or computing extremal probabilities for ω-regular path events [32,33,13,5]. The schema of the standard model checking approach for computing extremal (i.e., minimal or maximal) probabilities for ω-regular path events in MDPs is shown in Figure 2. The main idea relies on an analysis of the product that results from the given MDP $\mathcal{M}$ with a deterministic ω-automaton $\mathcal{A}$ representing the path event. (See, e.g., [31,19] for an overview of automata over infinite structures.) As $\mathcal{A}$ is deterministic, the product $\mathcal{M} \times \mathcal{A}$ can be understood as an MDP that behaves operationally as $\mathcal{M}$ and additionally mimicks $\mathcal{A}$'s behavior when scanning a path of $\mathcal{M}$. The extremal probabilities for the given path event E in $\mathcal{M}$ agree with the extremal probabilities for the acceptance condition of $\mathcal{A}$ in the product-MDP $\mathcal{M}$. The latter can be computed by means of a graph analysis and linear programming techniques for calculating minimal or maximal reachability probabilities. The details of this procedure can be found in the above mentioned literature.

Several efficient probabilistic model checking tools for MDPs are available that have been used in many application areas. The most prominent one that uses a symbolic approach with BDD-variants is PRISM [20].

Anomalies when ranging over all schedulers. The approach sketched above computes "worst-case" probabilities for path events when ranging over all schedulers. In particular, all possible interleavings and resolutions of other nondeterministic choices in the system model are taken into account. However, the computed worst-case probabilities are often too pessimistic since there are unrealistic schedulers that might yield extremal probabilities.

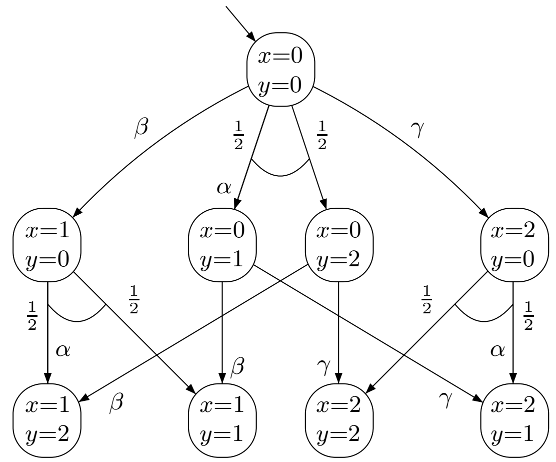
As in the non-probabilistic case, *fairness assumption* might be necessary to rule out schedulers that treat some processes in an unfair way. E.g., for the MDP in Figure 1, the scheduler that never takes an action of process P_2 and only schedules the actions $\mathit{request}_1$, enter_1 and $\mathit{release}_1$ can be viewed to be unrealistic since the request-operation of process P_2 is always enabled. The schema sketched in Figure 2 can be refined to compute the minimal and maximal probabilities of ω-regular path events when ranging over fair schedulers only [7,6]. The major difference is that the graph-based analysis has to be revised.

But there are still other curious phenomena when ranging over all (possibly fair) schedulers in MDPs that can contort the extremal probabilities.

Example 2. The MDP in Figure 3 arises through the parallel composition of two processes P_1 and P_2 with local integer variables x and y, respectively, that have initial value 0.

Process P_1 consists of a nondeterministic choice between actions β and γ, representing the deterministic assignments $x := 1$ (action β) and $x := 2$ (action γ). Process P_2 probabilistically assigns value 1 or 2 to y, depending on the outcome of a coin tossing experiment (action α). To ensure that no state is terminal, self-loops can be added to all states that have no outgoing transition in Figure 3. These self-loops might represent an internal step performed by a third process.

The maximal probability that a state where $x = y \in \{1, 2\}$ will be reached is 1, since we might first schedule α and then, depending on the outcome of α, we can choose β or γ to ensure that finally $x = y \in \{1, 2\}$. This scheduler, however, is not realizable if there is no central control and process P_1 cannot access the current value of P_2's local variable y. Indeed, intuitively we might expect the answer $\frac{1}{2}$ (rather than 1) as the maximal probability for reaching a



action	effect
α	toss a fair coin; if head then $y := 1$ else $y := 2$
β	$x := 1$
γ	$x := 2$

Fig. 3. MDP for $P_1 \| P_2$ where $P_1 = \beta + \gamma$ and $P_2 = \alpha$

state where $x = y \in \{1, 2\}$ is $\frac{1}{2}$ when ranging over all strategies to resolve the nondeterministic choice between β and γ in process P_1.

Partially-observable Markov decision processes. The above observation motivates to study the worst-case behavior of MDPs for restricted classes of schedulers that take the local view of the processes that run in parallel into account. In the literature, several classes of schedulers have been considered that are more adequate for reasoning about distributed systems [12,18,1]. Partially-observable MDPs, briefly called POMDPs, can be seen as a simple variant that can serve to model the view of one process.

POMDPs are formally defined as MDPs that are augmented by an equivalence relation $\sim$ on the state space which identifies those states that cannot be distinguished by an observer. A scheduler $D : S^+ \to \mathsf{Act}$ for an POMDP is called *observation-based* iff D's decisions only depend on the observables of the history, i.e., if $\pi_1 = s_0\, s_1 \ldots s_n$ and $\pi_2 = t_0\, t_1 \ldots t_n$ are finite paths of the same length such that $[s_i] = [t_i]$ for $0 \leq i \leq n$ then $D(\pi_1) = D(\pi_2)$. Here, $[s] = \{s' \in S : s \sim s'\}$ denotes the $\sim$-equivalence class of state s.

POMDPs are used in many application areas, see e.g. [9], and many algorithms have been proposed for the analysis of the behavior up a fixed number of steps ("finite-horizon") [23,25,21,24]. However, many difficulties arise for path events of the standard safety-liveness spectrum where no restrictions are imposed on the number of relevant steps. The design of algorithms for a quantitative analysis that determines worst-case probabilities for, e.g., a reachability condition, under all observation-based schedulers is impossible. This is due to the close link between POMDPs with an reachability objective, say $\Diamond F$ where F is a set of states and $\Diamond$ denotes the eventually operator of LTL, and *probabilistic finite automata* (PFA) [28,26]. Note that in the extreme case of an POMDP $\mathcal{M}$ where

$\sim$ identifies all states, the observation-based schedulers can be viewed as functions $D : \mathbb{N} \to \mathsf{Act}$, and hence, as infinite words in Act^ω. But then the question whether there exists an observation-based scheduler D for $\mathcal{M}$ such that the probability for $\Diamond F$ under D is larger than a given a threshold $\lambda \in]0,1[$ is equivalent to the *non-emptiness problem* for the PFA that results from $\mathcal{M}$ by treating F as the set of final states and λ as threshold for the accepted language; and the latter is known to be undecidable [26,22].

There is even no approximation algorithm for maximal reachability probabilities in POMDPs and the verification problem for almost all interesting quantitative properties for POMDPs and related models are undecidable [22,18]. Even worse, undecidability results can be established for certain instances of the model checking problem for POMDPs and qualitative properties, such as the question whether there exists an observation-based scheduler such that a repeated reachability condition $\Box\Diamond F$ ("visit infinitely often some state in F") holds with positive probability. This problem is a generalization of the non-emptiness problem for *probabilistic Büchi automata* (PBA) with the probable semantics [3] which is known to be undecidable [2].

However, some qualitative model checking problems for POMDPs are decidable. Examples for decidable verification problems for POMDPs are the question whether there exists an observation-based scheduler such that an invariance $\Box F$ ("always F") holds with positive probability [16], or whether the maximal probability under all observation-based schedulers for a reachability condition $\Diamond F$ ("eventually F") or a repeated reachability condition $\Box\Diamond F$ ("infinitely often F") is 1 [2], see also [10,11]. The algorithms for such qualitative model checking problems for POMDPs rely on variants of the powerset construction that has been introduced for incomplete-information games [29].

Besides the observation that ranging over the full class of schedulers can yield too pessimistic worst-case probabilities, also several other techniques suffer from the power of general schedulers.

In the context of *partial order reduction* for MDPs, it has been noticed in [4,14] that the criteria that are known to be sound for non-probabilistic systems are not sufficient to preserve extremal probabilities for ω-regular path events. In this setting, the problem is that the commutativity of independent probabilistic actions is a local property that does not carry over to a global setting. Consider again the MDP in Example 2. Action α is independent from both β and γ, as α accesses just variable y, while β and γ operate on x, without any reference to y. Indeed if we consider the parallel execution of α and β (or α and γ) in isolation, then the order of α and β (or α and γ) is irrelevant for the probabilities of the final outcome. But the commutativity of α and β resp. α and γ does not carry over to the nondeterministic choice between β and γ (process P_1).

– The scheduler which first chooses α and then β if $x{=}0 \wedge y = 1$ and γ if $x{=}0 \wedge y = 2$ yields probability 1 for a final outcome where $x = y \in \{1,2\}$ hold.

– For the schedulers that first choose one of the actions β or γ and then perform α, an outcome where $x = y \in \{1, 2\}$ hold is obtained with probability $\frac{1}{2}$.

This observation causes some care for the design of partial order reduction techniques for MDPs and either requires an extra condition [4,14] or to restrict the class of schedulers for the worst-case analysis [17].

Another problem that arises from the power of the full class of schedulers is the lack of compositionality of trace distribution equivalence in MDP-like models [30]. This problem can be avoided by introducing some appropriate concept of distributed scheduling [12]. But as the series of undecidability results for POMDPs shows, the price one has to pay when switching from the full class of schedulers to more realistic ones is the loss of model checking algorithms for the quantitative analysis against infinite-horizon path events, and partly also verification algorithms for qualitative properties.

Nevertheless, further restrictions on the scheduler types might be possible to overcome the limitations due to undecidability results.

References

1. Andres, M., Palamidessi, C., van Rossum, P., Sokolova, A.: Information hiding in probabilistic concurrent systems. In: Proc. of the 7th International Conference on Quantitative Evaluation of SysTems (QEST 2010). IEEE Computer Society Press, Los Alamitos (to appear 2010)
2. Baier, C., Bertrand, N., Grösser, M.: On decision problems for probabilistic Büchi automata. In: Amadio, R.M. (ed.) FOSSACS 2008. LNCS, vol. 4962, pp. 287–301. Springer, Heidelberg (2008)
3. Baier, C., Grösser, M.: Recognizing ω-regular languages with probabilistic automata. In: Proc. of the 20th IEEE Symposium on Logic in Computer Science (LICS 2005), pp. 137–146. IEEE Computer Society Press, Los Alamitos (2005)
4. Baier, C., Grösser, M., Ciesinski, F.: Partial order reduction for probabilistic systems. In: Proc. of the First International Conference on Quantitative Evaluation of SysTems (QEST), pp. 230–239. IEEE Computer Society Press, Los Alamitos (2004)
5. Baier, C., Größer, M., Ciesinski, F.: Model checking linear time properties of probabilistic systems. In: Handbook of Weighted Automata, pp. 519–570 (2009)
6. Baier, C., Größer, M., Ciesinski, F.: Quantitative analysis under fairness constraints. In: Liu, Z., Ravn, A.P. (eds.) ATVA 2009. LNCS, vol. 5799, pp. 135–150. Springer, Heidelberg (2009)
7. Baier, C., Kwiatkowska, M.: Model checking for a probabilistic branching time logic with fairness. Distributed Computing 11 (1998)
8. Bianco, A., de Alfaro, L.: Model checking of probabilistic and non-deterministic systems. In: Thiagarajan, P.S. (ed.) FSTTCS 1995. LNCS, vol. 1026, pp. 499–513. Springer, Heidelberg (1995)
9. Cassandra, A.R.: A survey of POMDP applications. Presented at the AAAI Fall Symposium (1998), `http://pomdp.org/pomdp/papers/applications.pdf`
10. Chadha, R., Sistla, P., Viswanathan, M.: Power of randomization in automata on infinite strings. In: Bravetti, M., Zavattaro, G. (eds.) CONCUR 2009. LNCS, vol. 5710, pp. 229–243. Springer, Heidelberg (2009)

11. Chatterjee, K., Doyen, L., Henzinger, T.: Qualitative analysis of partially-observable markov decision processes. In: Proc. Mathematical Foundation of Computer Science. LNCS, Springer, Heidelberg (2010)
12. Cheung, L., Lynch, N., Segala, R., Vaandrager, F.: Switched PIOA: Parallel composition via distributed scheduling. Theoretical Computer Science 365(1-2), 83–108 (2006)
13. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. Journal of the ACM 42(4), 857–907 (1995)
14. D'Argenio, P.R., Niebert, P.: Partial order reduction on concurrent probabilistic programs. In: Proc. of the First International Conference on Quantitative Evaluation of SysTems (QEST), pp. 240–249. IEEE Computer Society Press, Los Alamitos (2004)
15. de Alfaro, L.: Formal Verification of Probabilistic Systems. PhD thesis, Stanford University, Department of Computer Science (1997)
16. de Alfaro, L.: The verification of probabilistic systems under memoryless partial-information policies is hard. In: Proc. of the 2nd International Workshop on Probabilistic Methods in Verification (ProbMiV 1999), pp. 19–32. Birmingham University (1999), Research Report CSR-99-9
17. Giro, S., D'Argenio, P., María Ferrer Fioriti, L.: Partial order reduction for probabilistic systems: A revision for distributed schedulers. In: Bravetti, M., Zavattaro, G. (eds.) CONCUR 2009. LNCS, vol. 5710, pp. 338–353. Springer, Heidelberg (2009)
18. Giro, S., D'Argenio, P.R.: Quantitative model checking revisited: neither decidable nor approximable. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 179–194. Springer, Heidelberg (2007)
19. Grädel, E., Thomas, W., Wilke, T. (eds.): Automata, Logics, and Infinite Games: A Guide to Current Research. LNCS, vol. 2500. Springer, Heidelberg (2002)
20. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic symbolic model checking with PRISM: A hybrid approach. International Journal on Software Tools for Technology Transfer (STTT) 6(2), 128–142 (2004)
21. Littman, M.: Algorithms for Sequential Decision Making. PhD thesis, Brown University, Department of Computer Science (1996)
22. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. Artificial Intelligence 147(1-2), 5–34 (2003)
23. Monahan, G.: A survey of partially observable Markov decision processes: Theory, models and algorithms. Management Science 28(1), 1–16 (1982)
24. Mundhenk, M., Goldsmith, J., Lusena, C., Allender, E.: Complexity of finite-horizon Markov decision process problems. Journal of the ACM 47(4), 681–720 (2000)
25. Papadimitriou, C., Tsitsiklis, J.: The complexity of Markov decision processes. Mathematics of Operations Research 12(3) (1987)
26. Paz, A.: Introduction to probabilistic automata. Academic Press Inc., London (1971)
27. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley and Sons, Chichester (1994)
28. Rabin, M.O.: Probabilistic automata. Information and Control 6(3), 230–245 (1963)

29. Reif, J.H.: The complexity of two-player games of incomplete information. Journal of Computer System Sciences 29(2), 274–301 (1984)
30. Segala, R.: Modeling and Verification of Randomized Distributed Real-Time Systems. PhD thesis, Massachusetts Institute of Technology (1995)
31. Thomas, W.: Languages, automata and logic. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 3, pp. 389–455. Springer, New York (1997)
32. Vardi, M.Y.: Automatic verification of probabilistic concurrent finite-state programs. In: Proc. of the 26th Symposium on Foundations of Computer Science (FOCS), pp. 327–338. IEEE Computer Society Press, Los Alamitos (1985)
33. Vardi, M.Y., Wolper, P.: An automata-theoretic approach to automatic program verification. In: Proc. of the 1st IEEE Symposium on Logic in Computer Science (LICS), pp. 332–345. IEEE Computer Society Press, Los Alamitos (1986)