Resolving Nondeterminism by Chance

Soumyajit Paul, David Purser, Sven Schewe, Qiyi Tang, Patrick Totzke, Di-De Yen

University of Liverpool
{soumyajit.paul, d.purser, s.schewe, qiyi.tang, totzke, d.d.yen}@liverpool.ac.uk

Here we present our work on stochastically resolvable automata that was published at CONCUR'25 [16]. An extended version can be found here [17].

Many successful verification techniques rely on automata representations for specifications that capture the languages of acceptable system traces. These automata models are typically nondeterministic: any given trace may give rise to multiple runs of the automaton, and is accepted if at least one of these runs is successful. This enables succinct representations but is also a major source of complexity due to costly intermediate determinisation steps.

It is therefore natural to put extra constraints on the extent to which an automaton allows nondeterministic choice to avoid determinisation. One way to do this, which has been extensively studied in recent years (cf. [11,8,13,2,5,1,19,6,7,14,15]), is to demand that choices can be resolved "on-the-fly": An automaton is *history-deterministic* (HD) if there is a strategy to select a continuation of a run given a next letter, so that if the overall word admits some accepting run then the constructed run is also accepting. This condition is strict in the sense that the resolver strategy must guarantee to produce an accepting run if one exists. History deterministic automata have been successfully used in reactive synthesis [11]. In some scenarios less strict guarantees may suffice, for example when model-checking Markov Decision Processes [10,9,3,4]. This motivates the study of automata that can be resolved in a weaker sense, namely, where the resolver strategy can randomise and is only required to succeed with a lower-bounded confidence. Similar concept in the qualitative setting has been studied in [12].

A *stochastic resolver* for some nondeterministic automaton $\mathcal{A}$ is a function that, for any finite run and input letter, gives a distribution over the possible transitions. This produces a probabilistic automaton $\mathcal{P}$ that assigns a probability of acceptance to every input word and, together with a threshold $\lambda > 0$, defines the language $\mathcal{L}(\mathcal{P}_{\geq\lambda})$ consisting of all words whose probability of acceptance is at least λ (see, e.g., [18]). Unless stated otherwise, we will consider *memoryless* stochastic resolvers, which base their decisions solely on the last state of the given run and the input letter. Fixing a memoryless resolver turns $\mathcal{A}$ into a probabilistic automaton $\mathcal{P}$ over the same set of states and where transitions that appear positively in $\mathcal{P}$ are also contained in $\mathcal{A}$. Consequently then, for every threshold $\lambda > 0$, the language $\mathcal{L}(\mathcal{P}_{\geq\lambda})$ is included in that of $\mathcal{A}$. We now ask which automata admit positional resolvers that also guarantee the inclusion in the other direction.

An automaton $\mathcal{A}$ called λ-*memoryless stochastically resolvable* (or simply λ-resolvable) if there exists a memoryless stochastic resolver with $\mathcal{L}(\mathcal{P}_{\geq\lambda}) = \mathcal{L}(\mathcal{A})$. An automaton is *positively memoryless stochastically resolvable* (or simply positively resolvable) if it is λ-resolvable for some $\lambda > 0$. By varying the threshold λ, stochastic resolvability defines a spectrum where 0-resolvable corresponds to unrestricted nondeterminism and, on finite words, 1-resolvable coincides with history-determinism. See Fig. 1 for distinguishing examples. Not all automata are stochastically resolvable and a key challenge is to recognize such automata. In this work, we primarily adress this challenge.

(a) NFA $\mathcal{A}$ is 1/2-resolvable. (b) NFA $\mathcal{B}$ is not positively resolvable.

Fig. 1: Two unambiguous NFA (all missing transitions implicitly go to a non-accepting sink). The one on the left is λ-resolvable for all $\lambda \leq 1/2$. The one on the right is not positively resolvable, because no matter the choice of transition probability, the probability of b^n tends to zero as n grows.

Our Results We focus on automata over finite words and consider the decision problems whether a given automaton is resolvable. We distinguish the two variants of this problem, asking whether the automaton is positively resolvable, or whether it is λ-resolvable for a given value of λ. Our results are as follows, see also Table 1 for a summary.

- We introduce the quantitative notion of λ-memoryless stochastic resolvability, and show λ-resolvability induces a strict hierarchy of automata with varying parameter $\lambda \in (0, 1)$.
- We show that checking λ-resolvability is undecidable already for NFA, on finite words, and therefore also for automata on infinite words regardless of the accepting condition.
- We complement this by showing that both positive resolvability and λ-resolvability remain decidable for finitely-ambiguous automata.
- We present complexity upper and lower bounds for several well-studied variations of finitely-ambiguous automata (summarised in Table 1). In particular, checking positive resolvability is PSPACE-complete for finitely-ambiguous automata, NL-complete for unambiguous (1-ambiguous) automata, and in the polynomial hierarchy for (unrestricted) unary automata. Checking λ-resolvability is PSPACE-hard even for k-ambiguous automata (and coNP-hard over a unary alphabet).
- We show that our decidability results for finite-ambiguous automata, and undecidability results in the general case carry over to ω-regular automata.

		unambiguous	**finitely-ambiguous**	**general**
PR	unary	NL	coNP-hard $\quad \Sigma_2^P$	
	non-unary	NL-complete	PSPACE-complete	open
λ**R**	unary	PTIME	coNP-hard decidable	open
	non-unary	NL-hard PTIME	PSPACE-hard decidable	undecidable

Table 1: Deciding positive resolvability (PR) and λ-resolvability (λR) of NFAs where $\lambda \in (0, 1)$.

References

1. Bader Abu Radi and Orna Kupferman. Minimizing gfg transition-based automata. In *International Colloquium on Automata, Languages and Programming (ICALP)*, 2019. doi:10.4230/LIPIcs.ICALP.2019.100.
2. Marc Bagnol and Denis Kuperberg. Büchi Good-for-Games Automata Are Efficiently Recognizable. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 16:1–16:14, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.16.
3. Christel Baier, Luca de Alfaro, Vojtěch Forejt, and Marta Kwiatkowska. *Model Checking Probabilistic Systems*, pages 963–999. Springer International Publishing, 2018. doi:10.1007/978-3-319-10575-8_28.
4. Christel Baier, Holger Hermanns, and Joost-Pieter Katoen. *The 10,000 Facets of MDP Model Checking*, page 420–451. Springer-Verlag, 2022. doi:10.1007/978-3-319-91908-9_21.
5. Udi Boker and Karoliina Lehtinen. Good for games automata: From nondeterminism to alternation. In *International Conference on Concurrency Theory (CONCUR)*, volume 140, pages 19:1–19:16, 2019. doi:10.4230/LIPIcs.CONCUR.2019.19.
6. Udi Boker and Karoliina Lehtinen. History Determinism vs. Good for Gameness in Quantitative Automata. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 38:1–38:20, 2021. doi:10.4230/LIPIcs.FSTTCS.2021.38.
7. Udi Boker and Karoliina Lehtinen. Token games and history-deterministic quantitative automata. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 120–139. Springer International Publishing, 2022. doi:10.1007/978-3-030-99253-8_7.
8. Thomas Colcombet. The theory of stabilisation monoids and regular cost functions. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 139–150, 2009. doi:10.1007/978-3-642-02930-1_12.
9. Ernst Moritz Hahn, Guangyuan Li, Sven Schewe, Andrea Turrini, and Lijun Zhang. Lazy probabilistic model checking without determinisation. In *International Conference on Concurrency Theory (CONCUR)*, pages 354–367, 2015. doi:10.4230/LIPIcs.CONCUR.2015.354.
10. Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. Good-for-mdps automata for probabilistic analysis and

reinforcement learning. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 12078, pages 306–323. Springer, 2020. doi:10.1007/978-3-030-45190-5_17.

11. Thomas A. Henzinger and Nir Piterman. Solving games without determinization. In *Computer Science Logic (CSL)*, pages 395–410. Springer Berlin Heidelberg, 2006. doi:10.1007/11874683_26.
12. Thomas A. Henzinger, Aditya Prakash, and K. S. Thejaswini. Resolving nondeterminism with randomness. In *MFCS*, LIPIcs, pages 57:1–57:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025.
13. Denis Kuperberg and Michał Skrzypczak. On determinisation of good-for-games automata. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 299–310, 2015. doi:10.1007/978-3-662-47666-6_24.
14. Karoliina Lehtinen and Martin Zimmermann. Good-for-games ω-pushdown automata. *Log Meth Comput Sci*, 18, 2022. doi:10.1145/3373718.3394737.
15. Yong Li, Soumyajit Paul, Sven Schewe, and Qiyi Tang. Accelerating markov chain model checking: Good-for-games meets unambiguous automata. In *Computer Aided Verification (CAV)*, 2025.
16. Soumyajit Paul, David Purser, Sven Schewe, Qiyi Tang, Patrick Totzke, and Di-De Yen. Resolving nondeterminism by chance. In *CONCUR*, LIPIcs, pages 32:1–32:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025.
17. Soumyajit Paul, David Purser, Sven Schewe, Qiyi Tang, Patrick Totzke, and Di-De Yen. Resolving nondeterminism by chance. *CoRR*, abs/2504.10234, 2025.
18. Azaria Paz. *Introduction to Probabilistic Automata*. Academic Press, 2014.
19. Sven Schewe. Minimising good-for-games automata is NP-complete. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2020. doi:10.4230/LIPIcs.FSTTCS.2020.56.