


On Synthesis of Distributed Monitors

Anca Muscholl 

LaBRI, University of Bordeaux, France

Abstract

This talk addresses the synthesis problem of distributed monitors for concurrency properties.

2012 ACM Subject Classification Theory of computation → Distributed computing models; Theory of computation → Verification by model checking

Keywords and phrases Distributed synthesis, monitoring

Digital Object Identifier 10.4230/LIPIcs.MFCS.2025.5

Category Invited Talk

Multithreading programming as an essential aspect of modern software and infrastructure demanding scalability and speed. Such programs involve multiple threads of execution running simultaneously, which communicate and coordinate with each other. This introduces complexities in ensuring that the program behaves as expected, since the interactions between threads can be difficult to predict. As a result, identifying potential errors in these programs is very challenging, especially because bugs depend on interaction order and are hard to reproduce.

Runtime monitoring is a lightweight method that offers several advantages over the traditional verification of concurrent programs. It is a dynamic process that occurs during the program's execution, allowing it to catch errors in real-time. This is particularly useful in a concurrent setting, because common errors like races or serializability violations may only manifest under specific timing conditions that can be hard to predict or replicate in a controlled testing or verification environment.

This talk addresses the synthesis problem of distributed monitors for concurrency properties. One of the few, relevant theoretical frameworks for this problem is rooted in the theory of Mazurkiewicz traces [16]. This is one of several prominent models of concurrency such as Petri nets, Hoare's CSP, Milner's CCS, and Winskel's event structures. The simplicity of this theory made it highly popular in a number of areas in computer science, including programming languages and distributed computing [9, 8, 10, 3, 14, 12, 2, 5]. The most fundamental, and deepest, result of this theory, is due to Zielonka [19]. Zielonka's theorem shows how to translate a sequential program, into a distributed one. The sequential program can describe the concurrency error that we want to track. For example, a race condition can be described as the occurrence of two subsequent operations of two different threads that try to modify the same piece of data.

Zielonka's theorem is a highly non-trivial result and all known constructions are combinatorially very complex [19, 4, 17, 7, 18, 11, 1]. The talk's general aim is to provide an insight into the general construction, discuss relevant simpler cases [6, 15, 13] and remaining open research venues.

References

- 1 Bharat Adsul, Paul Gastin, Shantanu Kulkarni, and Pascal Weil. An expressively complete local past propositional dynamic logic over Mazurkiewicz traces and its applications. In Paweł Sobocinski, Ugo Dal Lago, and Javier Esparza, editors, *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2024, Tallinn, Estonia, July 8-11, 2024*, pages 2:1–2:13. ACM, 2024. doi:10.1145/3661814.3662110.



© Anca Muscholl;

licensed under Creative Commons License CC-BY 4.0

50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025).

Editors: Paweł Gawrychowski, Filip Mazowiecki, and Michał Skrzypczak; Article No. 5; pp. 5:1–5:3

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- 2 Benedikt Bollig, Cinzia Di Giusto, Alain Finkel, Laetitia Laversa, Étienne Lozes, and Amrita Suresh. A unifying framework for deciding synchronizability. In *32nd International Conference on Concurrency Theory, CONCUR 2021, August 24-27, 2021, Virtual Conference*, volume 203 of *LIPICs*, pages 14:1–14:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.CONCUR.2021.14.
- 3 Ahmed Bouajjani, Anca Muscholl, and Tayssir Touili. Permutation rewriting and algorithmic verification. *Inf. Comput.*, 205(2):199–224, 2007. doi:10.1016/J.IC.2005.11.007.
- 4 Robert Cori, Yves Métivier, and Wiesław Zielonka. Asynchronous mappings and asynchronous cellular automata. *Inf. Comput.*, 106(2):159–202, 1993. doi:10.1006/INCO.1993.1052.
- 5 Romain Delpy, Anca Muscholl, and Grégoire Sutre. An automata-based approach for synchronizable mailbox communication. In *35th International Conference on Concurrency Theory, CONCUR 2024, September 9-13, 2024, Calgary, Canada*, volume 311 of *LIPICs*, pages 22:1–22:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICs.CONCUR.2024.22.
- 6 Volker Diekert and Anca Muscholl. A note on Métivier’s construction of asynchronous automata for triangulated graphs. *Fundam. Informaticae*, 25(3):241–246, 1996. doi:10.3233/FI-1996-253402.
- 7 Volker Diekert and Grzegorz Rozenberg, editors. *The Book of Traces*. World Scientific, 1995. doi:10.1142/2563.
- 8 Azadeh Farzan, Dominik Klumpp, and Andreas Podelski. Stratified commutativity in verification algorithms for concurrent programs. *Proc. ACM Program. Lang.*, 7(POPL):1426–1453, 2023. doi:10.1145/3571242.
- 9 Azadeh Farzan and P. Madhusudan. The complexity of predicting atomicity violations. In Stefan Kowalewski and Anna Philippou, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 15th International Conference, TACAS 2009, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2009, York, UK, March 22-29, 2009. Proceedings*, volume 5505 of *Lecture Notes in Computer Science*, pages 155–169. Springer, 2009. doi:10.1007/978-3-642-00768-2_14.
- 10 Azadeh Farzan and Umang Mathur. Coarser equivalences for causal concurrency. *Proc. ACM Program. Lang.*, 8(POPL):911–941, 2024. doi:10.1145/3632873.
- 11 Blaise Genest, Hugo Gimbert, Anca Muscholl, and Igor Walukiewicz. Optimal Zielonka-type construction of deterministic asynchronous automata. In *Automata, Languages and Programming, ICALP 2010*, volume 6199 of *Lecture Notes in Computer Science*, pages 52–63. Springer, 2010. doi:10.1007/978-3-642-14162-1_5.
- 12 Blaise Genest, Dietrich Kuske, and Anca Muscholl. On communicating automata with bounded channels. *Fundamenta Informaticae*, 80(1-3):147–167, 2007. URL: <http://content.iospress.com/articles/fundamenta-informaticae/fi80-1-3-09>.
- 13 Daniel Hausmann, Mathieu Lehaut, and Nir Piterman. Distribution of reconfiguration languages maintaining tree-like communication topology. In *Automated Technology for Verification and Analysis - 22nd International Symposium, ATVA 2024, Kyoto, Japan, October 21-24, 2024, Proceedings*, *Lecture Notes in Computer Science*. Springer, 2024. doi:10.1007/978-3-031-78709-6_8.
- 14 Jesper G. Henriksen, Madhavan Mukund, K. Narayan Kumar, Milind A. Sohoni, and P. S. Thiagarajan. A theory of regular MSC languages. *Information and Computation*, 202(1):1–38, 2005. doi:10.1016/J.IC.2004.08.004.
- 15 Siddharth Krishna and Anca Muscholl. A quadratic construction for Zielonka automata with acyclic communication structure. *Theor. Comput. Sci.*, 503:109–114, 2013. doi:10.1016/j.tcs.2013.07.015.
- 16 Antoni Mazurkiewicz. Concurrent program schemes and their interpretations. *DAIMI Report Series*, (78), 1977.
- 17 Madhavan Mukund and Milind A. Sohoni. Keeping track of the latest gossip in a distributed system. *Distributed Comput.*, 10(3):137–148, 1997. doi:10.1007/S004460050031.

- 18 Alin Stefanescu, Javier Esparza, and Anca Muscholl. Synthesis of distributed algorithms using asynchronous automata. In Roberto M. Amadio and Denis Lugiez, editors, *CONCUR 2003 - Concurrency Theory, 14th International Conference, Marseille, France, September 3-5, 2003, Proceedings*, volume 2761 of *Lecture Notes in Computer Science*, pages 27–41. Springer, 2003. doi:10.1007/978-3-540-45187-7_2.
- 19 Wieslaw Zielonka. Notes on finite asynchronous automata. *RAIRO Theor. Informatics Appl.*, 21(2):99–135, 1987. doi:10.1051/ita/1987210200991.