Multi-Valued Reasoning about Reactive Systems

Other titles in Foundations and Trends® in Theoretical Computer Science

Quantified Derandomization: How to Find Water in the Ocean
Roei Tell
ISBN: 978-1-63828-092-7

Complexity Theory, Game Theory, and Economics: The Barbados Lectures
Tim Roughgarden
ISBN: 978-1-68083-654-7

Semialgebraic Proofs and Efficient Algorithm Design
Noah Fleming, Pravesh Kothari and Toniann Pitassi
ISBN: 978-1-68083-636-3

Higher-order Fourier Analysis and Applications
Hamed Hatami, Pooya Hatami and Shachar Lovett
ISBN: 978-1-68083-592-2

On Doubly-Efficient Interactive Proof Systems
Oded Goldreich
ISBN: 978-1-68083-424-6

Multi-Valued Reasoning about Reactive Systems

Orna Kupferman
The Hebrew University
orna@cs.huji.ac.il

the essence of knowledge
Boston — Delft

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

O. Kupferman. *Multi-Valued Reasoning about Reactive Systems*. Foundations and Trends® in Theoretical Computer Science, vol. 15, no. 2, pp. 126–228, 2022.

ISBN: 978-1-63828-139-9

Foundations and Trends® in Theoretical Computer Science
Volume 15, Issue 2, 2022

Editorial Board

Editorial Scope

Topics

Foundations and Trends® in Theoretical Computer Science publishes survey and tutorial articles in the following topics:

- Algorithmic game theory
- Computational algebra
- Computational aspects of combinatorics and graph theory
- Computational aspects of communication
- Computational biology
- Computational complexity
- Computational geometry
- Computational learning
- Computational Models and Complexity
- Computational Number Theory
- Cryptography and information security
- Data structures
- Database theory
- Design and analysis of algorithms
- Distributed computing
- Information retrieval
- Operations Research
- Parallel algorithms
- Quantum Computation
- Randomness in Computation

Information for Librarians

Foundations and Trends® in Theoretical Computer Science, 2022, Volume 15, 4 issues. ISSN paper version 1551-305X. ISSN online version 1551-3068. Also available as a combined paper and online subscription.

Contents

Multi-Valued Reasoning about Reactive Systems

Orna Kupferman

The Hebrew University, Israel; orna@cs.huji.ac.il

ABSTRACT

Traditional computer science is Boolean: a Turing machine accepts or rejects its input, and logic assertions are true or false. A primary use of logic in computer science has been the specification and verification of reactive systems. There, desired behaviors of systems are formally specified by temporal-logic formulas, and questions about systems and their behaviors are reduced to questions like satisfiability and model checking. While correctness is binary, many questions we want to ask about systems are multi-valued. The multi-valued setting arises directly in systems with quantitative aspects, for example systems with fuzzy assignments or stochastic dynamics, and arises also in Boolean systems, where it origins from the semantics of the specification formalism. In particular, beyond checking whether a system satisfies its specification, we may want to evaluate the quality in which the specification is satisfied. The term "quality" may refer to many aspects of the behavior: we may want to prioritize different satisfaction alternatives, refer to delays, costs, and many more. In recent years, we have seen a growing effort in the formal-method community to shift from Boolean specification formalisms to multi-valued ones.

Orna Kupferman (2022), "Multi-Valued Reasoning about Reactive Systems", Foundations and Trends® in Theoretical Computer Science: Vol. 15, No. 2, pp 126–228. DOI: 10.1561/0400000083.

The shift involves a development of multi-valued temporal logics as well as algorithms and tools for reasoning about such logics.

This survey describes the basics of specification and verification of reactive systems, and the automata-theoretic approach for them: by translating temporal-logic formulas to automata, one reduces questions like satisfiability and model checking to decision problems on automata, like nonemptiness and language containment.

We first describe the Boolean setting: temporal logics, and their applications in specification and verification. Since we care about on-going behaviors of non-terminating systems, the formalisms we study specify infinite computations, and we focus on the theoretical properties of automata on infinite words. The transition from finite to infinite words results in a beautiful mathematical model with much richer combinatorial properties. We then describe two multi-valued settings. The first is based on finite lattices and the second on arbitrary functions over $[0, 1]$. In both settings, the goal is to refine the Boolean correctness query to a quantitative-evaluation query. Accordingly, the formalisms we introduce are such that the satisfaction value of a temporal-logic formula in a model, or the membership value of a word in the language of an automaton, are multi valued, and classical decision problems become search problems.

1

Introduction

One of the main obstacles to the development of complex hardware and software systems lies in ensuring their correctness. *Temporal logics* are modal logics geared towards the description of the temporal ordering of events. In the early 1980s, temporal logics have been adopted as a powerful tool for specifying and verifying reactive systems [72], namely systems that interact with their environment and whose specification concerns the on-going interaction [38]. One of the most significant developments in this area is the discovery of algorithmic methods for verifying temporal logic properties of *finite-state* systems [17], [60], [74]. This derives its significance both from the fact that many synchronization and communication protocols can be modeled as finite-state systems, as well as from the great ease of use of fully algorithmic methods.

The idea is simple: a finite-state system that is defined with respect to a finite set AP of atomic propositions can be modeled by a finite *labeled state-transition graph*: the vertices of the graph correspond to configurations of the system, edges correspond to transitions between configurations, and each vertex is labeled by the assignment to the atomic propositions in AP that characterizes the corresponding configuration. Thus, verifying the correctness of a system with respect to a desired behavior, is reduced to checking that the finite graph that models the system satisfies a temporal-logic formula that specifies the behavior. Hence the name *model checking* for the verification methods derived from this viewpoint [18].

Finite *automata on infinite objects* were first introduced in the 1960s. Motivated by decision problems in mathematics and logic, Büchi, McNaughton, and Rabin developed a framework for reasoning about infinite words and infinite trees [11], [64], [75]. The framework has proved to be very powerful. Automata and their tight relation to second-order monadic logics were the key to the solution of several fundamental decision problems in mathematics and logic [76], [87]. Today, automata on infinite objects are used for specification and verification of finite-state systems. The fact the automata run on infinite objects makes them suitable for reasoning about *non-terminating* systems, which have infinite computations. Recall that we model a system over a set AP of atomic propositions by a graph whose vertices are labeled by assignments to AP. Each of the system's infinite computations induces an infinite word over the alphabet 2^{AP}, and the system itself induces a *language* of infinite words over this alphabet. This language can be defined by an automaton on infinite words. Similarly, a specification for the system, which describes all the allowed computations, can be viewed as a language of infinite words over 2^{AP}, and can be defined by an automaton. In the automata-theoretic approach to verification, we reduce questions about systems and their specifications to questions about automata. More specifically, questions such as satisfiability and model checking are reduced to questions such as non-emptiness and language containment [57], [90], [92].

The automata-theoretic approach for reasoning about systems and their specifications separates the logical and the combinatorial aspects of reasoning about systems. The translation of specifications to automata handles the logic and shifts all the combinatorial difficulties to automata-theoretic problems, yielding clean and asymptotically optimal algorithms, as well as better understanding of the complexity of the problems. Beyond leading to tight complexity bounds, automata have proven to be very helpful in practice. Automata-based methods have been implemented in both academic and industrial automated-verification tools (e.g., COSPAN [37], SPIN [39], ForSpec [85], and NuSMV [16]).

In recent years, researchers have considered extensions of the classical Boolean setting to a *multi-valued* one. One type of such extensions considers systems in which the atomic propositions are multi-valued. This includes systems in which the designer can give to the atomic propositions rich values, expressing, for example, energy consumption, waiting time, different levels of confidence, or inconsistent view-points [3], [6], [14], [40], [41]. The second type of such extensions considers systems in which the atomic propositions are possibly Boolean, yet the specification formalism itself includes multi-valued components. In particular, when considering the *quality* of a system, the different ways in which a specification may be satisfied induce different levels of quality, which should be reflected in the output of the verification procedure [8], [12], [20], [45].

This survey studies the automata-theoretic approach for reasoning about systems and their specifications, with a focus on its extension to multi-valued settings. We start with the Boolean setting: in Section 2, which is based on [51], we introduce *Linear Temporal Logic* (LTL) [71], [72], demonstrate its use in specifying on-going behaviors of reactive systems, and study its theoretical properties. Essentially, LTL extends propositional logic by *temporal operators* like G ("always") and F ("eventually"). For example, the LTL formula $G(req \rightarrow F(grant \vee ack)$ states that every request is eventually granted or acknowledged. Section 2 continues with *Büchi automata*. We introduce them, study their theoretical properties, and describe the automata-theoretic approach to reasoning about LTL specifications.

In Sections 3 and 4 we describe extensions of the Boolean setting to two types of extensions to the multi-valued setting. In Section 3, which is based on [52], we study the first extension, where the atomic propositions with respect to which the system is defined take values from a finite lattice. A *lattice* is a partially-ordered set $\mathcal{L} = \langle A, \leq \rangle$ in which every two elements ℓ and ℓ' have a least upper bound (ℓ *join* ℓ', denoted $\ell \vee \ell'$) and a greatest lower bound (ℓ *meet* ℓ', denoted $\ell \wedge \ell'$). Finite lattices capture several useful quantitative settings. Of special practical interest are two classes of lattices: (1) *Fully-ordered lattices*, where $\mathcal{L} = \langle\{0,\ldots,n-1\},\leq\rangle$, for an integer $n \geq 0$ and the usual "less than or equal" order. In this lattice, the operators $\vee$ and $\wedge$ correspond to max and min, respectively. Fully-ordered lattices are sometimes useful as is (for example, when modeling uncertainty or priorities [5], [6]), and sometimes thanks to the fact that real values can often be approximated by finitely many linearly ordered classes. (2) *Power-set lattices*, where $\mathcal{L} = \langle 2^X, \subseteq\rangle$, for a finite set X, and the containment order. In this lattice, the operators $\vee$ and $\wedge$ correspond to union and intersection, respectively. The power-set lattice models a wide range of partially-ordered values. For example, in a setting with inconsistent viewpoints, we have a set of agents, each with a different viewpoint of the system, and the truth value of a signal or a formula indicates the set of agents according to whose viewpoint the signal or the formula are true [23]. As another example, in a peer-to-peer network, one can refer to the different attributes of the communication channels by assigning with them subsets of attributes.

We introduce *Lattice Linear Temporal Logic* (LLTL), where atomic propositions and formulas take values from a finite lattice. An LLTL formula in which the atomic propositions take values from a lattice $\mathcal{L}$ maps computations to a value in $\mathcal{L}$. For example, when the atomic propositions take values from the fully-ordered lattice $\langle\{0,\ldots,n-1\},\leq\rangle$, then the satisfaction value of the LLTL formula $G(req \rightarrow F(grant \vee ack)$ is the maximal value $v \in \{0,\ldots,n-1\}$ such that every request of value greater than $(n-1)-v$ is eventually followed by a grant or an acknowlegement of value at least v. Then, when the atomic propositions take values from the partially-ordered lattice $\langle X, \subseteq\rangle$, for a set X of agents, then the satisfaction value of the formula is the set $S \subseteq X$ of exactly all agents x such that every request that is viewed by x is eventually followed by a grant or an acknowlegement that are viewed by x. Since the satisfaction value of LLTL formulas is an element in the lattice, questions like LLTL satisfiability and model checking become *search*, rather than decision, problems. Section 3 also introduces and studies *lattice automata*. Each lattice automaton is defined with respect to a lattice $\mathcal{L}$, and it maps words to values in $\mathcal{L}$. The Boolean setting can be viewed as a special case of the lattice setting, for the Boolean lattice $\langle\{0,1\},\leq\rangle$. We study the theoretical properties of LLTL and lattice automata, and describe an automata-theoretic approach to reasoning about LLTL specifications.

In Section 4, which is based on [4], we study the second extension of the Boolean setting, where specifications describe the quality of computations. We introduce and study the linear temporal logic LTL[$\mathcal{F}$], which extends LTL with an arbitrary set $\mathcal{F}$ of functions over $[0,1]$. Using the functions in $\mathcal{F}$, a specifier can formally and easily prioritize the different ways of satisfaction. The logic LTL[$\mathcal{F}$] is really a family of logics, each parameterized by a set $\mathcal{F} \subseteq \{f : [0,1]^k \to [0,1] \,:\, k \in \mathbb{N}\}$ of functions (of arbitrary arity) over $[0,1]$. For example, $\mathcal{F}$ may contain the $\min\{x,y\}$, $\max\{x,y\}$, and $1-x$ functions, which are the standard quantitative analogues of the $\wedge$, $\vee$, and $\neg$ operators. The novelty of LTL[$\mathcal{F}$] is the ability to manipulate values by arbitrary functions. For example, $\mathcal{F}$ may contain the quantitative operator $\triangledown_\lambda$, for $\lambda \in [0,1]$, that tunes down the quality of a sub-specification. Formally, the quality of the satisfaction of the specification $\triangledown_\lambda \varphi$ is the multiplication of the quality of the satisfaction of φ by λ. For example, the satisfaction value of the LTL[$\mathcal{F}$] formula $G(\mathit{req} \to F(\mathit{grant} \vee \triangledown_{\frac{3}{4}} \mathit{ack})$ is 1 when all requests are eventually granted, is $\frac{3}{4}$ when all requests are eventually granted or acknowledged yet some are only acknowledged, and is 0 when some requests are neither granted nor acknowledged.

For an automata-theoretic approach to LTL[$\mathcal{F}$], it seems natural to translate formulas to *weighted automata* [22], [67]. Such automata map input words to values from a semi-ring. In particular, they can map computations to values in $[0,1]$. Weighted automata, however, are complicated, and many problems become undecidable for them (e.g., the universality problem – [2], [50]). We show that it is possible to bound the number of possible satisfaction values of LTL[$\mathcal{F}$] formulas, and use this bound in order to translate LTL[$\mathcal{F}$] formulas to Boolean automata. From a technical point of view, the big challenge in our setting is to maintain the simplicity and the complexity of the algorithms for LTL, even though the number of possible values is exponential.

References

[1] L. de Alfaro, M. Faella, and M. Stoelinga, "Linear and branching metrics for quantitative transition systems," in *Proc. 31st Int. Colloq. on Automata, Languages, and Programming*, pp. 97–109, 2004.

[2] S. Almagor, U. Boker, and O. Kupferman, "What's decidable about weighted automata?" In *9th Int. Symp. on Automated Technology for Verification and Analysis*, ser. Lecture Notes in Computer Science, vol. 6996, pp. 482–491, Springer, 2011.

[3] S. Almagor, U. Boker, and O. Kupferman, "Formalizing and reasoning about quality," in *Proc. 40th Int. Colloq. on Automata, Languages, and Programming*, ser. Lecture Notes in Computer Science, vol. 7966, pp. 15–27, Springer, 2013.

[4] S. Almagor, U. Boker, and O. Kupferman, "Formalizing and reasoning about quality," *Journal of the ACM*, vol. 63, no. 3, 2016, 24:1–24:56.

[5] S. Almagor and O. Kupferman, "Latticed-LTL synthesis in the presence of noisy inputs," *Discrete Event Dynamic Systems*, vol. 27, no. 3, 2017, pp. 547–572.

[6] R. Alur, A. Kanade, and G. Weiss, "Ranking automata and games for prioritized requirements," in *Proc. 20th Int. Conf. on Computer Aided Verification*, ser. Lecture Notes in Computer Science, vol. 5123, pp. 240–253, Springer, 2008.

[7] S. Arora and B. Barak, *Computational Complexity - A Modern Approach.* Cambridge University Press, 2009.

[8] R. Bloem, K. Chatterjee, T. Henzinger, and B. Jobstmann, "Better quality in synthesis through quantitative objectives," in *Proc. 21st Int. Conf. on Computer Aided Verification*, ser. Lecture Notes in Computer Science, vol. 5643, pp. 140–156, Springer, 2009.

[9] S. Breuers, C. Löding, and J. Olschewski, "Improved Ramsey-based Büchi complementation," in *Proc. 15th Int. Conf. on Foundations of Software Science and Computation Structures*, ser. Lecture Notes in Computer Science, vol. 7213, pp. 150–164, Springer, 2012.

[10] G. Bruns and P. Godefroid, "Model checking with multi-valued logics," in *Proc. 31st Int. Colloq. on Automata, Languages, and Programming*, ser. Lecture Notes in Computer Science, vol. 3142, pp. 281–293, 2004.

[11] J. Büchi, "On a decision method in restricted second order arithmetic," in *Proc. Int. Congress on Logic, Method, and Philosophy of Science. 1960*, pp. 1–12, Stanford University Press, 1962.

[12] P. Cerný, K. Chatterjee, T. Henzinger, A. Radhakrishna, and R. Singh, "Quantitative synthesis for concurrent programs," in *Proc. 23rd Int. Conf. on Computer Aided Verification*, pp. 243–259, 2011.

[13] A. Chandra, D. Kozen, and L. Stockmeyer, "Alternation," *Journal of the Association for Computing Machinery*, vol. 28, no. 1, 1981, pp. 114–133.

[14] K. Chatterjee, L. Doyen, and T. Henzinger, "Quantative languages," in *Proc. 17th Annual Conf. of the European Association for Computer Science Logic*, pp. 385–400, 2008.

[15] Y. Choueka, "Theories of automata on ω-tapes: A simplified approach," *Journal of Computer and Systems Science*, vol. 8, 1974, pp. 117–141.

[16] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri, "NuSMV: A new symbolic model checker," *Software Tools for Technology Transfer*, vol. 2, no. 4, 2000, pp. 410–425.

[17] E. Clarke, E. Emerson, and A. Sistla, “Automatic verification of finite-state concurrent systems using temporal logic specifications,” *ACM Transactions on Programming Languagues and Systems*, vol. 8, no. 2, 1986, pp. 244–263.

[18] E. Clarke, O. Grumberg, D. Kroening, D. Peled, and H. Veith, *Model Checking, Second Edition*. MIT Press, 2018.

[19] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1990.

[20] D. Spinellis, *Code Quality: The Open Source Perspective*. Addison-Wesley Professional, 2006.

[21] A. Donzé, O. Maler, E. Bartocci, D. Nickovic, R. Grosu, and S. Smolka, “On temporal logic and signal processing,” in *10th Int. Symp. on Automated Technology for Verification and Analysis*, pp. 92–106, Springer, 2012.

[22] M. Droste, W. Kuich, and H. V. (eds.), *Handbook of Weighted Automata*. Springer, 2009.

[23] S. Easterbrook and M. Chechik, “A framework for multi-valued reasoning over inconsistent viewpoints,” in *Proc. 23rd Int. Conf. on Software Engineering*, pp. 411–420, IEEE Computer Society Press, 2001.

[24] E. Emerson and C. Jutla, “The complexity of tree automata and logics of programs,” in *Proc. 29th IEEE Symp. on Foundations of Computer Science*, pp. 328–337, 1988.

[25] E. Emerson and C. Jutla, “Tree automata, μ-calculus and determinacy,” in *Proc. 32nd IEEE Symp. on Foundations of Computer Science*, pp. 368–377, 1991.

[26] E. Emerson and C.-L. Lei, “Modalities for model checking: Branching time logic strikes back,” in *Proc. 12th ACM Symp. on Principles of Programming Languages*, pp. 84–96, 1985.

[27] E. Emerson and C.-L. Lei, “Temporal model checking under generalized fairness constraints,” in *Proc. 18th Hawaii Int. Conf. on System Sciences*, Western Periodicals Company, 1985.

[28] M. Faella, A. Legay, and M. Stoelinga, “Model checking quantitative linear time logic,” *Electr. Notes Theor. Comput. Sci.*, vol. 220, no. 3, 2008, pp. 61–77.

[29] E. Filiot, R. Gentilini, and J. Raskin, “Finite-valued weighted automata,” in *Proc. 34th Conf. on Foundations of Software Technology and Theoretical Computer Science*, pp. 133–145, 2014.

[30] S. Fogarty, O. Kupferman, M. Vardi, and T. Wilke, “Unifying Büchi complementation constructions,” in *Proc. 20th Annual Conf. of the European Association for Computer Science Logic*, pp. 248–263, 2011.

[31] E. Friedgut, O. Kupferman, and M. Vardi, “Büchi complementation made tighter,” *Int. J. Found. Comput. Sci.*, vol. 17, no. 4, 2006, pp. 851–868.

[32] C. Fritz, “Constructing Büchi automata from linear temporal logic using simulation relations for alternating Büchi automata,” in *Proc. 8th Int. Conf. on Implementation and Application of Automata*, ser. Lecture Notes in Computer Science, pp. 35–48, Springer, 2003.

[33] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi, “On the temporal analysis of fairness,” in *Proc. 7th ACM Symp. on Principles of Programming Languages*, pp. 163–173, 1980.

[34] P. Gastin and D. Oddoux, “Fast LTL to Büchi automata translation,” in *Proc. 13th Int. Conf. on Computer Aided Verification*, ser. Lecture Notes in Computer Science, vol. 2102, pp. 53–65, Springer, 2001.

[35] R. Gerth, D. Peled, M. Vardi, and P. Wolper, “Simple on-the-fly automatic verification of linear temporal logic,” in *Protocol Specification, Testing, and Verification*, P. Dembiski and M. Sredniawa, Eds., pp. 3–18, Chapman & Hall, 1995.

[36] D. Giannakopoulou and F. Lerda, “From states to transitions: Improving translation of LTL formulae to Büchi automata,” in *Proc. 22nd International Conference on Formal Techniques for Networked and Distributed Systems*, ser. Lecture Notes in Computer Science, vol. 2529, pp. 308–326, Springer, 2002.

[37] R. Hardin, Z. Har’el, and R. Kurshan, “COSPAN,” in *Proc. 8th Int. Conf. on Computer Aided Verification*, ser. Lecture Notes in Computer Science, vol. 1102, pp. 423–427, Springer, 1996.

[38] D. Harel and A. Pnueli, "On the development of reactive systems," in *Logics and Models of Concurrent Systems*, ser. NATO Advanced Summer Institutes, K. Apt, Ed., vol. F-13, Springer, 1985, pp. 477–498.

[39] G. Holzmann, "The model checker SPIN," *IEEE Transactions on Software Engineering*, vol. 23, no. 5, 1997, pp. 279–295.

[40] M. Huth and S. Pradhan, "Consistent partial model checking," *Electr. Notes Theor. Comput. Sci.*, vol. 73, 2004, pp. 45–85.

[41] IEEE, *IEEE standard multivalue logic system for VHDL model interoperability (Std_logic_1164)*, 1993.

[42] N. Immerman, "Nondeterministic space is closed under complement," *Information and Computation*, vol. 17, 1988, pp. 935–938.

[43] D. Kähler and T. Wilke, "Complementation, disambiguation, and determinization of Büchi automata unified," in *Proc. 35th Int. Colloq. on Automata, Languages, and Programming*, ser. Lecture Notes in Computer Science, vol. 5126, pp. 724–735, Springer, 2008.

[44] J. Kamp, "Tense logic and the theory of order," Ph.D. dissertation, UCLA, 1968.

[45] S. Kans, *Metrics and Models in Software Quality Engineering.* Addison-Wesley Longman Publishing Co., 2002.

[46] D. Kirsten and S. Lombardy, "Deciding unambiguity and sequentiality of polynomially ambiguous min-plus automata," in *26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, February 26-28, 2009, Freiburg, Germany, Proceedings*, pp. 589–600, 2009.

[47] N. Klarlund, "Progress measures for complementation of ω-automata with applications to temporal logic," in *Proc. 32nd IEEE Symp. on Foundations of Computer Science*, pp. 358–367, 1991.

[48] J. Kretínský and J. Esparza, "Deterministic automata for the (f, g)-fragment of LTL," in *Proc. 24th Int. Conf. on Computer Aided Verification*, ser. Lecture Notes in Computer Science, vol. 7358, pp. 7–22, Springer, 2012.

[49] S. Krishnan, A. Puri, and R. Brayton, "Deterministic ω-automata vis-a-vis deterministic Büchi automata," in *Algorithms and Computations*, ser. Lecture Notes in Computer Science, vol. 834, pp. 378–386, Springer, 1994.

[50] D. Krob, "The equality problem for rational series with multiplicities in the tropical semiring is undecidable," *International Journal of Algebra and Computation*, vol. 4, no. 3, 1994, pp. 405–425.

[51] O. Kupferman, "Automata theory and model checking," in *Handbook of Model Checking*, Springer, 2018, pp. 107–151.

[52] O. Kupferman and Y. Lustig, "Lattice automata," in *Proc. 8th Int. Conf. on Verification, Model Checking, and Abstract Interpretation*, ser. Lecture Notes in Computer Science, vol. 4349, pp. 199–213, Springer, 2007.

[53] O. Kupferman and M. Vardi, "Weak alternating automata are not that weak," *ACM Transactions on Computational Logic*, vol. 2, no. 2, 2001, pp. 408–429.

[54] O. Kupferman and M. Vardi, "Complementation constructions for nondeterministic automata on infinite words," in *Proc. 11th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, ser. Lecture Notes in Computer Science, vol. 3440, pp. 206–221, Springer, 2005.

[55] O. Kupferman, M. Vardi, and P. Wolper, "An automata-theoretic approach to branching-time model checking," *Journal of the ACM*, vol. 47, no. 2, 2000, pp. 312–360.

[56] R. Kurshan, "Complementing deterministic Büchi automata in polynomial time," *Journal of Computer and Systems Science*, vol. 35, 1987, pp. 59–71.

[57] R. Kurshan, *Computer Aided Verification of Coordinating Processes*. Princeton Univ. Press, 1994.

[58] L. Landweber, "Decision problems for ω–automata," *Mathematical Systems Theory*, vol. 3, 1969, pp. 376–384.

[59] F. Laroussinie and P. Schnoebelen, "A hierarchy of temporal logics with past," *Theoretical Computer Science*, vol. 148, no. 2, 1995, pp. 303–324.

[60] O. Lichtenstein and A. Pnueli, "Checking that finite state concurrent programs satisfy their linear specification," in *Proc. 12th ACM Symp. on Principles of Programming Languages*, pp. 97–107, 1985.

[61] O. Lichtenstein, A. Pnueli, and L. Zuck, "The glory of the past," in *Logics of Programs*, ser. Lecture Notes in Computer Science, vol. 193, pp. 196–218, Springer, 1985.

[62] C. Löding, "Optimal bounds for the transformation of ω-automata," in *Proc. 19th Conf. on Foundations of Software Technology and Theoretical Computer Science*, ser. Lecture Notes in Computer Science, vol. 1738, pp. 97–109, 1999.

[63] Z. Manna and A. Pnueli, *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer, 1992.

[64] R. McNaughton, "Testing and generating infinite sequences by a finite automaton," *Information and Control*, vol. 9, 1966, pp. 521–530.

[65] A. Meyer and L. Stockmeyer, "The equivalence problem for regular expressions with squaring requires exponential space," in *Proc. 13th IEEE Symp. on Switching and Automata Theory*, pp. 125–129, 1972.

[66] M. Michel, "Complementation is more difficult with automata on infinite words," 1988.

[67] M. Mohri, "Finite-state transducers in language and speech processing," *Computational Linguistics*, vol. 23, no. 2, 1997, pp. 269–311.

[68] S. Moon, K. Lee, and D. Lee, "Fuzzy branching temporal logic," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 34, no. 2, 2004, pp. 1045–1055.

[69] D. Muller and P. Schupp, "Alternating automata on infinite trees," in *Automata on Infinite Words*, ser. Lecture Notes in Computer Science, vol. 192, pp. 100–107, Springer, 1985.

[70] N. Piterman, "From nondeterministic Büchi and Streett automata to deterministic parity automata," *Logical Methods in Computer Science*, vol. 3, no. 3, 2007, p. 5.

[71] N. Piterman and A. Pnueli, "Temporal logic and fair discrete systems," in *Handbook of Model Checking.* Springer, 2018, pp. 27–73.
[72] A. Pnueli, "The temporal semantics of concurrent programs," *Theoretical Computer Science*, vol. 13, 1981, pp. 45–60.
[73] A. Pnueli and A. Zaks, "On the merits of temporal testers," in *25 Years of Model Checking*, ser. Lecture Notes in Computer Science, vol. 5000, pp. 172–195, Springer, 2008.
[74] J. Queille and J. Sifakis, "Specification and verification of concurrent systems in Cesar," in *Proc. 8th ACM Symp. on Principles of Programming Languages*, ser. Lecture Notes in Computer Science, vol. 137, pp. 337–351, Springer, 1982.
[75] M. Rabin, "Decidability of second order theories and automata on infinite trees," *Transaction of the AMS*, vol. 141, 1969, pp. 1–35.
[76] M. Rabin, "Decidable theories," in *Handbook of Mathematical Logic*, J. Barwise, Ed., Amsterdam: North-Holland, 1977, pp. 595–629.
[77] M. Rabin and D. Scott, "Finite automata and their decision problems," *IBM Journal of Research and Development*, vol. 3, 1959, pp. 115–125.
[78] S. Safra, "On the complexity of ω-automata," in *Proc. 29th IEEE Symp. on Foundations of Computer Science*, pp. 319–327, 1988.
[79] S. Schewe, "Büchi complementation made tight," in *Proc. 26th Symp. on Theoretical Aspects of Computer Science*, ser. LIPIcs, vol. 3, pp. 661–672, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009.
[80] S. Schewe, "Beyond Hyper-Minimisation—Minimising DBAs and DPAs is NP-Complete," in *Proc. 30th Conf. on Foundations of Software Technology and Theoretical Computer Science*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 8, pp. 400–411, 2010.
[81] C. E. Shannon, "The synthesis of two terminal switching circuits," *BELL-SYST-TECH*, vol. 28, no. 1, 1949, pp. 59–98.
[82] A. Sistla and E. Clarke, "The complexity of propositional linear temporal logic," *Journal of the ACM*, vol. 32, 1985, pp. 733–749.

[83] A. Sistla, M. Vardi, and P. Wolper, "The complementation problem for Büchi automata with applications to temporal logic," *Theoretical Computer Science*, vol. 49, 1987, pp. 217–237.

[84] F. Somenzi and R. Bloem., "Efficient Büchi automata from LTL formulae," in *Proc. 12th Int. Conf. on Computer Aided Verification*, ser. Lecture Notes in Computer Science, vol. 1855, pp. 248–263, Springer, 2000.

[85] Synopsys, "Assertion-based verification," 2003, URL: www.open-vera.com.

[86] R. Tarjan, "Depth first search and linear graph algorithms," *SIAM Journal of Computing*, vol. 1(2), 1972, pp. 146–160.

[87] W. Thomas, "Automata on infinite objects," *Handbook of Theoretical Computer Science*, J. V. Leeuwen, Ed., 1990, pp. 133–191.

[88] M. Vardi and P. Wolper, "An automata-theoretic approach to automatic program verification," in *Proc. 1st IEEE Symp. on Logic in Computer Science*, pp. 332–344, 1986.

[89] M. Vardi and P. Wolper, "Automata-theoretic techniques for modal logics of programs," *Journal of Computer and Systems Science*, vol. 32, no. 2, 1986, pp. 182–221.

[90] M. Vardi and P. Wolper, "Reasoning about infinite computations," *Information and Computation*, vol. 115, no. 1, 1994, pp. 1–37.

[91] P. Wolper, "Temporal logic can be more expressive," in *Proc. 22nd IEEE Symp. on Foundations of Computer Science*, pp. 340–348, 1981.

[92] P. Wolper, M. Vardi, and A. Sistla, "Reasoning about infinite computation paths," in *Proc. 24th IEEE Symp. on Foundations of Computer Science*, pp. 185–194, 1983.

[93] Q. Yan, "Lower bounds for complementation of ω-automata via the full automata technique," in *Proc. 33rd Int. Colloq. on Automata, Languages, and Programming*, ser. Lecture Notes in Computer Science, vol. 4052, pp. 589–600, Springer, 2006.