Probabilistic Trace and Testing Semantics: The Importance of Being Coherent

Other titles in Foundations and Trends® in Programming Languages

Introduction to Neural Network Verification
Aws Albarghouthi
ISBN: 978-1-68083-910-4

Refinement Types: A Tutorial
Ranjit Jhala and Niki Vazou
ISBN: 978-1-68083-884-8

Shape Analysis
Bor-Yuh Evan Chang, Cezara Drăgoi, Roman Manevich, Noam Rinetzky and Xavier Rival
ISBN: 978-1-68083-732-2

Progress of Concurrent Objects
Hongjin Liang and Xinyu Feng
ISBN: 978-1-68083-672-1

QED at Large: A Survey of Engineering of Formally Verified Software
Talia Ringer, Karl Palmskog, Ilya Sergey, Milos Gligoric and Zachary Tatlock
ISBN: 978-1-68083-594-6

Reconciling Abstraction with High Performance: A MetaOCaml approach
Oleg Kiselyov
ISBN: 978-1-68083-436-9

Probabilistic Trace and Testing Semantics: The Importance of Being Coherent

Marco Bernardo
Università di Urbino
marco.bernardo@uniurb.it

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Programming Languages

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

M. Bernardo. *Probabilistic Trace and Testing Semantics: The Importance of Being Coherent.* Foundations and Trends® in Programming Languages, vol. 7, no. 4, pp. 244–332, 2022.

ISBN: 978-1-63828-075-0

Full text available at: http://dx.doi.org/10.1561/2500000056

Editorial Scope

Topics

Foundations and Trends® in Programming Languages publishes survey and tutorial articles in the following topics:

- Abstract Interpretation
- Compilation and Interpretation Techniques
- Domain Specific Languages
- Formal Semantics, including Lambda Calculi, Process Calculi, and Process Algebra
- Language Paradigms
- Mechanical Proof Checking
- Memory Management
- Partial Evaluation
- Program Logic
- Programming Language Implementation
- Programming Language Security
- Programming Languages for Concurrency
- Programming Languages for Parallelism
- Program Synthesis
- Program Transformations and Optimizations
- Program Verification
- Runtime Techniques for Programming Languages
- Software Model Checking
- Static and Dynamic Program Analysis
- Type Theory and Type Systems

Information for Librarians

Foundations and Trends® in Programming Languages, 2022, Volume 7, 4 issues. ISSN paper version 2325-1107. ISSN online version 2325-1131. Also available as a combined paper and online subscription.

Contents

Probabilistic Trace and Testing Semantics: The Importance of Being Coherent

Marco Bernardo

Università di Urbino, Italy; marco.bernardo@uniurb.it

ABSTRACT

It is well known that trace and testing semantics over nondeterministic and probabilistic processes are influenced by the class of schedulers used to resolve nondeterministic choices. In particular, it is the capability of suitably limiting the power of the considered schedulers that guarantees the validity of a number of desirable properties of those semantics. Among such properties we mention the fact of being coarser than bisimulation semantics, the fact of being a congruence with respect to typical process operators, and the fact of coinciding with the corresponding semantics when restricting to fully nondeterministic or fully probabilistic processes.

In this monograph, we recall various approaches against almighty schedulers appearing in the literature, we survey structure-preserving and structure-modifying resolutions of nondeterminism by providing a uniform definition for them, and we present an overview of behavioral equivalences for nondeterministic and probabilistic processes along with some anomalies affecting trace and testing semantics. We then introduce the notion of coherent resolution, which prevents a scheduler from selecting different continuations in

---

Marco Bernardo (2022), "Probabilistic Trace and Testing Semantics: The Importance of Being Coherent", Foundations and Trends® in Programming Languages Vol. 7, No. 4, pp 244-332. DOI: 10.1561/2500000056.

equivalent states of a process, so that the states to which they correspond in any resolution of the process have equivalent continuations too.

We show that coherency avoids anomalies related to the discriminating power, the compositionality, and the backward compatibility of probabilistic trace post-equivalence and pre-equivalence, which are variants of trace semantics. Moreover, we exhibit an alternative characterization of the former based on coherent trace distributions and an alternative characterization of the latter relying on coherent weighted trace sets. We finally extend the notion of coherent resolution by adding suitable transition decorations and prove that this ensures the insensitivity of probabilistic testing equivalence to the moment of occurrence of nondeterministic or probabilistic choices among identical actions, thus enhancing the backward compatibility of testing semantics.

1

Introduction

1.1 Probabilistic Behavioral Models and Relations

Quantitative models of computer, communication, and software systems combine, among others, functional and extra-functional aspects of system behavior. On the one hand, these models describe system operations and their execution order, possibly admitting nondeterminism in case of concurrency phenomena or to support implementation freedom. On the other hand, they include some information about the probabilities or the durations of activities and events in which the system is involved.

In the probabilistic setting, a particularly expressive model is given by probabilistic automata (Segala, 1995a) because they encompass as special cases fully nondeterministic models like labeled transition systems (Keller, 1976), fully probabilistic models like action-labeled variants of discrete-time Markov chains (Kemeny and Snell, 1960), and reactive probabilistic models like Markov decision processes (Derman, 1970). In a probabilistic automaton, which consists of states and transitions, the choice among the transitions departing from the current state is nondeterministic and can be influenced by the external environment, while the choice of the next state reached by the selected transition is probabilistic and is made internally by the process.

Behavioral relations (van Glabbeek, 2001; Jou and Smolka, 1990; Huynh and Tian, 1992; Baier *et al.*, 2005; Bernardo, 2007; Bernardo *et al.*, 2014b) play a fundamental role in the analysis of probabilistic models. They formalize observational mechanisms that permit relating models that, despite their different representations in the same mathematical domain, cannot be distinguished by external entities when abstracting from certain internal details. Moreover, they support system modeling and verification by providing a means to relate system descriptions expressed at different levels of abstraction, as well as to reduce the size of a system representation while preserving specific properties to be assessed later.

From the first comparative work (De Nicola, 1987) to the elaboration of the full spectrum (van Glabbeek, 2001), a number of equivalences have emerged over fully nondeterministic models, which range from the branching-time – i.e., (bi)simulation-based – endpoint (Park, 1981; Milner, 1989) to the linear-time – i.e., trace-based – endpoint (Brookes *et al.*, 1984) passing through testing relations (De Nicola and Hennessy, 1984). The spectrum becomes simpler when considering fully probabilistic models (Jou and Smolka, 1990; Huynh and Tian, 1992; Baier *et al.*, 2005; Bernardo, 2007), whereas as shown in Bernardo *et al.* (2014b), it is much more variegated in the case of models with nondeterminism and probabilities like probabilistic automata. The reason is that the probability of equivalence-specific events can be calculated only after removing nondeterminism. Examples of such events are the reachability via given actions of certain sets of equivalent states (bisimulation semantics), the execution of specific action sequences (trace semantics), and the passing of tests (testing semantics), with states/traces being possibly enriched with additional information.

In this monograph, we focus on trace and testing semantics for nondeterministic and probabilistic processes represented by simple probabilistic automata (Segala, 1995a).

A trace is a sequence of activities labeling a sequence of transitions performed by a process, thus abstracting from branching points in the process behavior. Several execution probabilities may be associated with the same trace, each corresponding to a different resolution of nondeterminism. Although the discriminating power of probabilistic trace equivalences depends on how nondeterminism is resolved, in general this power turns out to be excessive, which hampers the achievement of a number of desirable properties.

A test is formalized as a nondeterministic and probabilistic process extended with success states or success actions, which is run in parallel with the process under test thus resulting in an interaction or testing system. The probability of reaching success is not unique, but depends on the specific resolution of nondeterminism considered within the interaction system. Also in the testing approach, the resulting probabilistic behavioral equivalences tend to be overdiscriminating.

1.2 Struggling Against Demonic Schedulers

Nondeterminism is resolved by resorting to *policies*, according to the terminology of Bellman (1957), or *schedulers*, according to the terminology of Vardi (1985). They establish which is the next transition or combination of transitions to be executed, possibly based on the sequence of states traversed so far.

The problem with almighty schedulers yielding a demonic view of nondeterminism is well known for both trace and testing semantics. In the case of a process given by the parallel composition of several subprocesses, or in a testing scenario where a process is composed in parallel with a test, schedulers come into play *after* the various components have been assembled together. As a consequence, schedulers can solve both choices local to the individual components and choices arising from their interleaving execution. In other words, this *centralized* approach enables any scheduler to make decisions in one component on the basis of those made in other components, especially in the case of history-dependent schedulers (Vardi, 1985).

To cope with the aforementioned information leakage, the idea of *distributed* scheduling was proposed in de Alfaro *et al.* (2001), which is akin to partial-information policies (de Alfaro, 1999). Given a number of modules, i.e., of variable-based versions of automata, that interact *synchronously* by updating all variables during every round, for each module there are several schedulers. One of them chooses the initial values and the updated values for the module external variables; for each atom, intended as a cluster of variables of the module, a further scheduler chooses the initial values and the updated values for the private and interface variables controlled by that atom. Compose-and-schedule is thus replaced by schedule-and-compose.

Distributed scheduling was then applied in Cheung *et al.* (2006) to the *asynchronous* model of switched probabilistic input/output automata. Following the terminology of van Glabbeek *et al.* (1995), given a reactive interpretation to input actions and a generative interpretation to output actions, an input scheduler and an output scheduler are considered for each automaton occurring in a system. A token passing mechanism among the automata eliminates global choices by ensuring that a single automaton at a time can select a generative output action, to which the other automata can respond with reactive input actions having the same name.

Both de Alfaro *et al.* (2001) and Cheung *et al.* (2006) guarantee the compositionality of the probabilistic trace-distribution equivalence of Segala (1995b), which is *not* a congruence with respect to parallel composition under centralized scheduling. As shown in Lynch *et al.* (2003), the coarsest congruence contained in that linear-time equivalence turns out to be a variant of the simulation equivalence of Segala and Lynch (1994), which is a branching-time equivalence.

Distributed scheduling was further studied in Giro and D'Argenio (2007; 2009) for interleaved probabilistic input/output automata, a variant of switched ones in which an interleaving scheduler replaces the token passing mechanism. The examined problem was the attainment of the extremal probabilities of satisfying reachability properties under different classes of distributed schedulers (memoryless vs. history-dependent, deterministic vs. randomized), knowing that in the centralized case those probabilities are obtained when using memoryless deterministic schedulers (Bianco and de Alfaro, 1995).

The overwhelming power of schedulers already shows up in the *memoryless* case, i.e., when neglecting the path followed to reach the current state. Under memoryless schedulers, a different definition of probabilistic trace equivalence allows compositionality to be recovered without resorting to distributed scheduling.

In the probabilistic trace-distribution equivalence of Segala (1995b), for each resolution of either process there must exist a resolution of the other process such that the two resolutions are *fully matching*, in the sense that, for every trace, both resolutions feature the same probability of executing that trace. This is called probabilistic trace *post*-equivalence as the quantification over traces occurs *after* the quantifications over resolutions, which is a source of overdiscrimination.

In Bernardo *et al.* (2014a) it was proposed to exchange the order of those quantifications, which avoids hardly justifiable process distinctions and regains compositionality. Given an arbitrary trace, for each resolution of either process there must exist a resolution of the other process such that both of them exhibit the same probability of executing that trace. In this case, resolutions are *partially matching*, as a resolution of either process can be matched by different resolutions of the other process with respect to different traces. The resulting relation is called probabilistic trace *pre*-equivalence because the quantification over traces occurs *before* the quantifications over resolutions.

On the other hand, the probabilistic testing equivalences of Yi and Larsen (1992), Jonsson and Yi (1995), and Segala (1996) are *not* backward compatible with testing equivalences for simpler processes such as fully nondeterministic ones (De Nicola and Hennessy, 1984) and fully probabilistic ones (Cleaveland *et al.*, 1999).

Indeed, in Jonsson and Yi (2002) and Deng *et al.* (2008) it was shown that those equivalences can be characterized in terms of branching-time, simulation-like relations, which is consistent with the fact that they are *not* insensitive to the moment of occurrence of nondeterministic or probabilistic choices among identical actions. In addition to centralized scheduling, this is a consequence of a special instance of the *copying capability* (Abramsky, 1987), which shows up in the presence of a nondeterministic choice in either component that synchronizes with a probabilistic choice in the other, thus creating copies of a state possessing several outgoing transitions where different decisions can be made.

Under centralized scheduling, in Georgievska and Andova (2012) additional labels were used so that the same decision is made by schedulers in distinct copies of the same state of a testing system, which weakens the discriminating power of the probabilistic testing equivalences of

Yi and Larsen (1992), Jonsson and Yi (1995), and Segala (1996). An analogous weakening result under the same class of schedulers was obtained in Bernardo *et al.* (2014a) by means of a different definition of probabilistic testing equivalence, in which success probabilities are compared in a trace-by-trace fashion rather than cumulatively. Instead of the overall success probability, the probability of reaching success is examined separately for each possible trace.

1.3 Coherent Resolutions of Nondeterminism

Being a congruence with respect to parallel composition, which is ensured by distributed scheduling (de Alfaro *et al.*, 2001; Cheung *et al.*, 2006) as well as partially matching resolutions (Bernardo *et al.*, 2014a), is not the only desirable property of probabilistic trace equivalences. In addition to compositionality with respect to other typical process operators, it is necessary to address the inclusion of the probabilistic bisimilarity of Segala and Lynch (1994) together with the backward compatibility with respect to trace equivalences over less expressive models, such as fully nondeterministic processes (Brookes *et al.*, 1984) and fully probabilistic processes (Jou and Smolka, 1990).

We will see that the validity of the aforementioned properties of trace semantics, as well as the possibility of enhancing the backward compatibility of testing semantics, critically depend on the capability of limiting the freedom of schedulers and can be achieved if we restrict ourselves to *coherent resolutions* of nondeterminism. Similar to Georgievska and Andova (2012), the basic idea is that schedulers cannot select different continuations in states of a process that are equivalent to each other, so that the states to which they correspond in any resolution of the process also have equivalent continuations.

As a preliminary step towards the study of the impact of resolution coherency on the discriminating power, on the compositionality, and on the backward compatibility of probabilistic trace and testing equivalences, we will provide a uniform way of defining the resolutions induced by different subclasses of centralized, memoryless schedulers. In particular, we formalize any resolution as a fully probabilistic automaton, which we equip with a *correspondence function* from the acyclic state space of the resolution to the possibly cyclic state space of the original automaton. This technique was introduced for the first time in Jonsson *et al.* (1994) for deterministic schedulers.

We divide resolutions into *structure preserving* and *structure modifying*, depending on whether they respect or alter the structure of the automaton from which they are obtained. A structure-preserving resolution is produced by a *deterministic scheduler*, which selects at the current state one of the transitions departing from that state or no transitions at all. A structure-modifying resolution is derived via a *randomized scheduler* (Segala, 1995a), which probabilistically combines the transitions departing from the current state, or an *interpolating scheduler* (Deng *et al.*, 2007), which splits the current state into copies, each having at most one outgoing transition and whose probabilities sum up to the probability of the original state.

We will then present a number of anomalies affecting the probabilistic trace equivalences of Segala (1995b) and Bernardo *et al.* (2014a), mostly arising under deterministic schedulers. More precisely, we show that they do not contain probabilistic bisimilarity, are not congruences with respect to action prefix, and are not backward compatible with their versions for fully probabilistic models. The reason is that schedulers have the freedom to make *different* decisions in *equivalent* states occurring in the target distribution of a transition, with these decisions not necessarily replicable in equivalent distributions of distinct automata. This is especially true for deterministic schedulers, as the resolutions they induce must be structure preserving.

Such anomalies can be avoided by employing coherent resolutions in the definition of probabilistic trace equivalences. If several states in the target distribution of a transition are equivalent, then the states to which they correspond in a resolution must be equivalent as well. The coherency constraints can be formalized by reasoning on *coherent trace distributions*, i.e., suitable families of sets of traces weighted with their execution probabilities in a given resolution.

In the case of testing semantics, coherency will be accompanied by additional transition decorations, so that the same decisions are made by schedulers in distinct copies of the same state of a process or a test occurring in a choice within the testing system. This is similar to the technique employed in Georgievska and Andova (2012) for processes in which branchings based on actions, nondeterminism, and probabilities alternate, with the remarkable difference that our decoration procedure turns out to be much simpler.

The resulting probabilistic testing equivalence retrieves insensitivity to the moment of occurrence of nondeterministic or probabilistic choices among identical actions, thus enhancing backward compatibility with respect to Yi and Larsen (1992), Jonsson and Yi (1995), and Segala (1996). Consistent with the ready-trace semantics characterization of Georgievska and Andova (2012), a counterexample inspired by failure semantics for fully nondeterministic processes shows that complete backward compatibility cannot be achieved in the presence of certain synchronizations among external choices, a fact that has nothing to do with coherency.

1.4 Alternative Characterizations

In a fully nondeterministic setting, two processes are trace equivalent if, and only if, for each trace α, both processes can perform α or neither can. An immediate alternative characterization is that two trace equivalent processes possess the same trace set (Brookes *et al.*, 1984), where this set can be viewed as the language accepted by the automata underlying those processes. Likewise, two fully probabilistic processes are trace equivalent if and only if, for each trace α, both processes can perform α with the same probability, which amounts to possessing the same set of traces each weighted with its execution probability (Jou and Smolka, 1990), i.e., the same probabilistic language. In either case, process equivalence reduces to (possibly weighted) trace set equality.

Straightforward characterizations of that form are not possible in the case of nondeterministic and probabilistic processes because (i) traces can have different execution probabilities in different coherent resolutions, and (ii) trace semantics can be defined according to different approaches leading to probabilistic trace post-/pre-equivalences. This motivates the investigation of alternative characterizations for the two aforementioned equivalences under coherent resolutions arising from centralized, memoryless schedulers. We will see that the coherency-based variant of the probabilistic trace post-equivalence of Segala (1995b) can be characterized in terms of the coherent trace distributions used for defining the coherency constraints. In contrast, since it treats traces individually without keeping track of the resolutions in which they can be executed, the coherency-based variant of the probabilistic trace pre-equivalence of Bernardo *et al.* (2014a) can be characterized by something weaker, which is constituted by coherent weighted trace sets.

1.5 Outline

This work is an extended, revised, and integrated version of Bernardo (2019a; 2020a; 2020b), which is organized as follows. In Section 2 we recall the simple probabilistic automaton model and its specializations to fully nondeterministic and fully probabilistic models. In Section 3 we survey different ways of resolving nondeterminism in the aforementioned model, which preserve or modify the model structure, and provide a uniform manner of defining all of them. In Section 4 we present an overview of different approaches to probabilistic behavioral equivalences and then recall the formal definitions of probabilistic bisimulation equivalence, probabilistic trace post-/pre-equivalences, and probabilistic testing equivalence. In Section 5 we illustrate three anomalies of the two probabilistic trace equivalences related to their discriminating power, their compositionality, and their backward compatibility. In Section 6 we show how to avoid those anomalies by resorting to coherent resolutions, which are formulated in terms of coherency constraints based on coherent trace distributions. In Section 7 we develop alternative characterizations of the coherency-based variants of the two probabilistic trace equivalences, respectively relying on coherent trace distributions and coherent weighted trace sets, and use them to express some considerations about congruence with respect to parallel composition. In Section 8 we illustrate that the backward compatibility of probabilistic testing equivalence is only partial due to the sensitivity to the moment of occurrence of nondeterministic or probabilistic choices among identical actions. In Section 9 we show how to enhance compatibility through the combined use of coherent resolutions and suitable transition decorations. Finally, in Section 10 we provide some concluding remarks.

References

Abramsky, S. (1987). "Observational Equivalence as a Testing Equivalence". *Theoretical Computer Science*. 53: 225–241.

Baier, C., J.-P. Katoen, H. Hermanns, and V. Wolf. (2005). "Comparative Branching-Time Semantics for Markov Chains". *Information and Computation*. 200: 149–214.

Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press.

Bernardo, M. (2007). "A Survey of Markovian Behavioral Equivalences". In: *Formal Methods for Performance Evaluation*. Vol. 4486. *LNCS*. Springer. 180–219.

Bernardo, M. (2018). "ULTraS at Work: Compositionality Metaresults for Bisimulation and Trace Semantics". *Journal of Logical and Algebraic Methods in Programming*. 94: 150–182.

Bernardo, M. (2019a). "Coherent Resolutions of Nondeterminism". In: *Proc. of the 16th European Performance Engineering Workshop (EPEW 2019)*. Vol. 12039. *LNCS*. Springer. 16–32.

Bernardo, M. (2019b). "Genesis and Evolution of ULTraS: Metamodel, Metaequivalences, Metaresults". In: *Models, Languages, and Tools for Concurrent and Distributed Programming*. Vol. 11665. *LNCS*. Springer. 92–111.

Bernardo, M. (2020a). "Alternative Characterizations of Probabilistic Trace Equivalences on Coherent Resolutions of Nondeterminism". In: *Proc. of the 17th Int. Conf. on the Quantitative Evaluation of Systems (QEST 2020)*. Vol. 12289. *LNCS*. Springer. 35–53.

Bernardo, M. (2020b). "Extending Backward Compatibility of Probabilistic Testing via Coherent Resolutions". In: *Proc. of the 21st Italian Conf. on Theoretical Computer Science (ICTCS 2020)*. Vol. 2756. CEUR-WS. 208–222.

Bernardo, M., R. De Nicola, and M. Loreti. (2014a). "Revisiting Trace and Testing Equivalences for Nondeterministic and Probabilistic Processes". *Logical Methods in Computer Science*. 10(1:16): 1–42.

Bernardo, M., R. De Nicola, and M. Loreti. (2014b). "Relating Strong Behavioral Equivalences for Processes with Nondeterminism and Probabilities". *Theoretical Computer Science*. 546: 63–92.

Bernardo, M., R. De Nicola, and M. Loreti. (2015). "Revisiting Bisimilarity and its Modal Logic for Nondeterministic and Probabilistic Processes". *Acta Informatica*. 52: 61–106.

Bernardo, M., D. Sangiorgi, and V. Vignudelli. (2014c). "On the Discriminating Power of Testing Equivalences for Reactive Probabilistic Systems: Results and Open Problems". In: *Proc. of the 11th Int. Conf. on the Quantitative Evaluation of Systems (QEST 2014)*. Vol. 8657. *LNCS*. Springer. 281–296.

Bianco, A. and L. de Alfaro. (1995). "Model Checking of Probabilistic and Nondeterministic Systems". In: *Proc. of the 15th Int. Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 1995)*. Vol. 1026. *LNCS*. Springer. 499–513.

Bonchi, F., A. Sokolova, and V. Vignudelli. (2019). "The Theory of Traces for Systems with Nondeterminism and Probability". In: *Proc. of the 34th ACM/IEEE Symp. on Logic in Computer Science (LICS 2019)*. IEEE-CS Press. (19:62)1–14.

Brookes, S. D., C. A. R. Hoare, and A. W. Roscoe. (1984). "A Theory of Communicating Sequential Processes". *Journal of the ACM*. 31: 560–599.

Cheung, L., N. A. Lynch, R. Segala, and F. Vaandrager. (2006). "Switched PIOA: Parallel Composition via Distributed Scheduling". *Theoretical Computer Science*. 365: 83–108.

Cleaveland, R., Z. Dayar, S. A. Smolka, and S. Yuen. (1999). "Testing Preorders for Probabilistic Processes". *Information and Computation*. 154: 93–148.

de Alfaro, L. (1999). "The Verification of Probabilistic Systems under Memoryless Partial-Information Policies is Hard". In: *Proc. of the 2nd Int. Workshop on Probabilistic Methods in Verification (PROBMIV 1999)*. University of Birmingham, Technical Report CSR-99-9. 19–32.

de Alfaro, L., T. A. Henzinger, and R. Jhala. (2001). "Compositional Methods for Probabilistic Systems". In: *Proc. of the 12th Int. Conf. on Concurrency Theory (CONCUR 2001)*. Vol. 2154. *LNCS*. Springer. 351–365.

De Nicola, R. (1987). "Extensional Equivalences for Transition Systems". *Acta Informatica*. 24: 211–237.

De Nicola, R. and M. Hennessy. (1984). "Testing Equivalences for Processes". *Theoretical Computer Science*. 34: 83–133.

Deng, Y., R. J. van Glabbeek, M. Hennessy, and C. Morgan. (2008). "Characterising Testing Preorders for Finite Probabilistic Processes". *Logical Methods in Computer Science*. 4(4:4): 1–33.

Deng, Y., R. J. van Glabbeek, C. Morgan, and C. Zhang. (2007). "Scalar Outcomes Suffice for Finitary Probabilistic Testing". In: *Proc. of the 16th European Symp. on Programming (ESOP 2007)*. Vol. 4421. *LNCS*. Springer. 363–378.

Derman, C. (1970). *Finite State Markovian Decision Processes*. Academic Press.

Georgievska, S. and S. Andova. (2012). "Probabilistic May/Must Testing: Retaining Probabilities by Restricted Schedulers". *Formal Aspects of Computing*. 24: 727–748.

Giro, S. and P. R. D'Argenio. (2007). "Quantitative Model Checking Revisited: Neither Decidable nor Approximable". In: *Proc. of the 5th Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS 2007)*. Vol. 4763. *LNCS*. Springer. 179–194.

Giro, S. and P. R. D'Argenio. (2009). "On the Expressive Power of Schedulers in Distributed Probabilistic Systems". In: *Proc. of the 7th Int. Workshop on Quantitative Aspects of Programming Languages (QAPL 2009)*. Vol. 253(3). *ENTCS*. Elsevier. 45–71.

Hansson, H. and B. Jonsson. (1990). "A Calculus for Communicating Systems with Time and Probabilities". In: *Proc. of the 11th IEEE Real-Time Systems Symp. (RTSS 1990)*. IEEE-CS Press. 278–287.

Huynh, D. T. and L. Tian. (1992). "On Some Equivalence Relations for Probabilistic Processes". *Fundamenta Informaticae*. 17: 211–234.

Jifeng, H., K. Seidel, and A. McIver. (1997). "Probabilistic Models for the Guarded Command Language". *Science of Computer Programming*. 28: 171–192.

Jonsson, B., C. Ho-Stuart, and W. Yi. (1994). "Testing and Refinement for Nondeterministic and Probabilistic Processes". In: *Proc. of the 3rd Int. Symp. on Formal Techniques in Real Time and Fault Tolerant Systems (FTRTFT 1994)*. Vol. 863. *LNCS*. Springer. 418–430.

Jonsson, B. and W. Yi. (1995). "Compositional Testing Preorders for Probabilistic Processes". In: *Proc. of the 10th IEEE Symp. on Logic in Computer Science (LICS 1995)*. IEEE-CS Press. 431–441.

Jonsson, B. and W. Yi. (2002). "Testing Preorders for Probabilistic Processes Can Be Characterized by Simulations". *Theoretical Computer Science*. 282: 33–51.

Jou, C.-C. and S. A. Smolka. (1990). "Equivalences, Congruences, and Complete Axiomatizations for Probabilistic Processes". In: *Proc. of the 1st Int. Conf. on Concurrency Theory (CONCUR 1990)*. Vol. 458. *LNCS*. Springer. 367–383.

Keller, R. M. (1976). "Formal Verification of Parallel Programs". *Communications of the ACM*. 19: 371–384.

Kemeny, J. G. and J. L. Snell. (1960). *Finite Markov Chains*. Van Nostrand.

Larsen, K. G. and A. Skou. (1991). "Bisimulation Through Probabilistic Testing". *Information and Computation*. 94: 1–28.

Lynch, N. A., R. Segala, and F. Vaandrager. (2003). "Compositionality for Probabilistic Automata". In: *Proc. of the 14th Int. Conf. on Concurrency Theory (CONCUR 2003)*. Vol. 2761. *LNCS*. Springer. 208–221.

Milner, R. (1989). *Communication and Concurrency*. Prentice Hall.

Park, D. (1981). "Concurrency and Automata on Infinite Sequences". In: *Proc. of the 5th GI Conf. on Theoretical Computer Science*. Vol. 104. *LNCS*. Springer. 167–183.

Rabin, M. O. (1963). "Probabilistic Automata". *Information and Control*. 6: 230–245.

Segala, R. (1995a). *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD Thesis.

Segala, R. (1995b). "A Compositional Trace-Based Semantics for Probabilistic Automata". In: *Proc. of the 6th Int. Conf. on Concurrency Theory (CONCUR 1995)*. Vol. 962. *LNCS*. Springer. 234–248.

Segala, R. (1996). "Testing Probabilistic Automata". In: *Proc. of the 7th Int. Conf. on Concurrency Theory (CONCUR 1996)*. Vol. 1119. *LNCS*. Springer. 299–314.

Segala, R. and N. A. Lynch. (1994). "Probabilistic Simulations for Probabilistic Processes". In: *Proc. of the 5th Int. Conf. on Concurrency Theory (CONCUR 1994)*. Vol. 836. *LNCS*. Springer. 481–496.

Sokolova, A. and E. P. de Vink. (2004). "Probabilistic Automata: System Types, Parallel Composition and Comparison". In: *Validation of Stochastic Systems*. Vol. 2925. *LNCS*. Springer. 1–43.

Song, L., L. Zhang, J. C. Godskesen, and F. Nielson. (2013). "Bisimulations Meet PCTL Equivalences for Probabilistic Automata". *Logical Methods in Computer Science*. 9(2:7): 1–34.

Tracol, M., J. Desharnais, and A. Zhioua. (2011). "Computing Distances Between Probabilistic Automata". In: *Proc. of the 9th Int. Workshop on Quantitative Aspects of Programming Languages (QAPL 2011)*. Vol. 57. *EPTCS*. 148–162.

van Glabbeek, R. J. (2001). "The Linear Time – Branching Time Spectrum I". In: *Handbook of Process Algebra*. Elsevier. 3–99.

van Glabbeek, R. J., S. A. Smolka, and B. Steffen. (1995). "Reactive, Generative and Stratified Models of Probabilistic Processes". *Information and Computation*. 121: 59–80.

Vardi, M. Y. (1985). "Automatic Verification of Probabilistic Concurrent Finite-State Programs". In: *Proc. of the 26th IEEE Symp. on Foundations of Computer Science (FOCS 1985)*. IEEE-CS Press. 327–338.

Yi, W. and K. G. Larsen. (1992). "Testing Probabilistic and Nondeterministic Processes". In: *Proc. of the 12th Int. Symp. on Protocol Specification, Testing and Verification (PSTV 1992)*. North-Holland. 47–61.