Preface

Probabilistic programs

Probabilistic programs describe recipes for inferring statistical conclusions from a complex mixture of uncertain data and real-world observations. They can represent probabilistic graphical models far beyond the capabilities of Bayesian networks and are expected to have a major impact on machine intelligence. Probabilistic programs are ubiquitous. They steer autonomous robots and self-driving cars, are key to describe security mechanisms, naturally code up randomised algorithms for solving NP-hard or even unsolvable problems, and are rapidly encroaching on AI. Probabilistic programming aims to make probabilistic modelling and machine learning accessible to the programmer.

What is this book all about?

Probabilistic programs, though typically relatively small in size, are hard to grasp, let alone check automatically. Elementary questions are notoriously hard – even the most elementary question "does a program halt with probability one? – is "more undecidable" than the halting problem. This book is about the theoretical foundations of probabilistic programming. It is primarily concerned with fundamental questions such as the following: What is Bayesian probability theory? What is the precise mathematical meaning of probabilistic programs? How show almost-sure termination? How determine the (possibly infinite) expected runtime of probabilistic programs? How can two similar programs be compared? It covers several analysis techniques on probabilistic programs such as abstract interpretation, algebraic reasoning and determining concentration measures.

These chapters are complemented with chapters on the formal definition of concrete probabilistic programming languages and some possible applications of the use of probabilistic programs.

How to read this volume?

The volume consists of five parts: semantics, verification, logic, security, and programming languages.

Semantics.
The first part on semantics consists of four chapters on different aspects of the formal semantics of probabilistic programming languages. Dahlqvist *et al.* start off in Chapter 1 by presenting an operational and denotational semantics of an imperative language with discrete and continuous distributions. The chapter by Staton presents a compositional measure-theoretic semantics for a first-order probabilistic language with arbitrary soft conditioning. Chapter 3 by Huang *et al.* studies semantics with a focus on computability. Motivated by the tension between the discrete and the continuous in probabilistic modelling, type-2 computable distributions are introduced as the elementary mathematical objects to give semantics to probabilistic languages. Dal Lago's Chapter 4 completes the semantics part by treating a probabilistic version of the λ-calculus, the backbone language for functional programming languages.

Verification.
The second part on formal verification starts with a chapter by Barthe and Hsu on the use of couplings, a well-known concept in probability theory for verifying probabilistic programs. Kaminski *et al.* present a weakest pre-condition calculus in the style of Dijkstra for determining the expected run-time of a probabilistic program. This can be used to determine whether a program needs infinitely many steps to terminate with probability one. Chapter 7 by Chatterjee *et al.* presents various techniques based on supermartingales to decide the almost-sure termination of a probabilistic program, i.e., does a program terminate with probability one on all possible inputs? The verification part ends with a chapter by Sankaranarayanan about the quantitative analysis of probabilistic programs by means of concentration of measure inequalities. This includes Chernoff–Hoeffding and Bernstein inequalities.

Logic.
The third part focuses on logic and consists of two chapters. In Chapter 9, Jacobs and Zanasi present an insightful new perspective on fundamental aspects of probability theory which are relevant for probabilistic programming. Their chapter introduces a category-theoretic formalization of Bayesian reasoning, and in particular a string-diagram-friendly one. This contribution is complemented by the chapter by Bacci *et al.* which surveys some recent contributions on extending the classical

Birkhoff/Lawvere theory of equational reasoning to the quantitative setting. In that setting, compared entities are not necessarily equal but rather treated by a notion of distance.

Security.
Part four is concerned with security, an important application field in which probabilities are pivotal. Chapter 11 by Calderon *et al.* collects together results on probabilistic abstract interpretation and applies them to probabilistic programming in the context of security. Chapter 12 by Gibbons *et al.* presents an embedded domain-specific language in Haskell to compute hyper-distributions induced by programs. This is used to compute the amount of leakage of a program by measuring variations on post-distributions that include Shannon entropy and Bayes' risk (that is, the maximum information an attacker can learn in a single run).

Programming languages.
The final part of this volume is concerned with three concrete probabilistic programming languages: Luck, Tabular, and Rely. Chapter 13 describes Luck, proposed by Lampropoulos *et al.*, a language for test generation and a framework for property-based testing of functional programs. Luck combines local instantiation of unknown variables and global constraint solving to make test generation more efficient than existing approaches. Gordon *et al.* introduce Tabular, a domain-specific programming language designed to express probabilistic models and to perform probabilistic inference over relational data. Tabular can be used from Microsoft Excel or as stand-alone software. Chapter 14 presents the syntax, semantics and type system of Tabular and shows how it can be used to design probabilistic models and to perform probabilistic inference. Chapter 15, the last chapter of this volume, by Carbin and Misailovic, presents Rely, a programming language that enables reasoning about the probability that a program produces the correct result when executed on unreliable hardware.

How this volume emerged

This book consists of 15 contributed chapters and a preface. The idea for this volume emerged at the first summer school on Foundations of Programming and Software Systems held in Braga, Portugal, May–June 2017. This biennial school series is supported by EATCS (European Association for Theoretical Computer Science), ETAPS (European Conference on Theory and Practice of Software), ACM SIGPLAN (Special Interest Group on Programming Languages) and ACM SIGLOG (Special Interest Group on Logic and Computation). It was felt that there is no comprehensive book on the theoretical foundations of probabilistic programming languages. We sincerely hope that this volume contributes to filling this gap. Enjoy reading!

Acknowledgements

Many people have helped us to make this volume possible. First and foremost, we like to thank all authors for their fine contributions and for their patience in this book process. All chapters have been subject to peer reviews. We thank the reviewers Alejandro Aguirre, Krishnendu Chatterjee, Ugo Dal Lago, Thomas Ehrhard, Claudia Faggian, Marco Gaboardi, Francesco Gavazzo, Jeremy Gibbons, Andrew D. Gordon, Ichiro Hasuo, Jan Hoffmann, Justin Hsu, Bart Jacobs, Benjamin Lucien Kaminski, Pasquale Malacaria, Radu Mardare, Christoph Matheja, Annabelle McIver, Sasa Misailovic, Petr Novotný, Prakash Panangaden, Corina Pasareanu, Alejandro Russo, Sriram Sankaranarayanan, Steffen Smolka, and Marcin Szymczak for their thorough reviewing work.

We thank Luis Barbosa, Catarina Fernandes and Renato Neves for their invaluable efforts in the local organisation of the FoPPS 2017 summer school. We thank Joshua Moerman for his efforts in the editing process. David Tranah and Anna Scriven from Cambridge University Press are thanked for their support during this process. We thank Hanna Schraffenberger for designing the cover of the book. Finally, we are grateful for the financial support from the ERC (AdG FRAPPANT and StG ProFoundNet) and the MPI on Security and Privacy which enabled to publish this volume under gold open access.