

"In this book the author will make you have second thoughts about the possibility and desirability of compiling away pattern matching ... It is amazing that the dynamic pattern calculus is syntactically almost as simple as the pure lambda-calculus, yet it is much more expressive."
Eugenio Moggi, University of Genoa

Barry Jay

Jay



Pattern Calculus

Jay Pattern Calculus

The pattern calculus is a new foundation for computation, in which the expressive power of functions and of data structures are combined within pattern-matching functions. The best existing foundations focus on either functions, as in the lambda-calculus, or on data structures, as in Turing machines, or on compromises involving both, as in object-orientation. By contrast, a small typed pattern calculus is able to support all the main programming styles, including functional, imperative, object-oriented and query-based styles, and there is evidence that it can support a language for Web services, able to exploit data structures about which almost nothing is known.

The book is divided into three parts, on terms, types and programs, and contains many new results. Part I introduces static and then dynamic pattern calculus. The former supports path polymorphic functions, able to traverse arbitrary paths through data structures. It also shows how Lisp is more than just lambda-calculus. The dynamic calculus allows any term to be a pattern, so that patterns can be discovered, combined and simplified on the fly. Part II supports a family of type systems for pattern calculi that build on novel typings of lambda-calculus. The type system for query calculus allows database queries to be applied uniformly to arbitrary data structures, while still guaranteeing that evaluation terminates. Subtyping and type parameters combine to produce type inequalities, whose solutions provide an expressive account of object-orientation. Also, typing can be made implicit, which simplifies implementation. Part III realises all these ideas in a new programming language, bondi, where the various programming styles can be combined, including algebraic data types and object-oriented classes.

This book is useful for researchers with an interest in the foundations of computing, programming language design, the integration of existing programming styles, or the development of new programming styles. The reader is supported throughout with proofs and examples, both in text and in bondi, the source code of which is freely available online.

ISBN 978-3-540-89184-0



9 783540 891840

 springer.com

Pattern Calculus

Computing with
Functions and Structures

 Springer