Trinder / Michaelson / Pena

Implementation of Functional Languages

15th International Workshop, IFL 2003, Edinburgh, UK, September 8-11, 2003. Revised Papers

Functional programminghas a long history,reaching back through early reali- tions in languages like LISP to foundational theories of computing, in particular ?-calculus and recursive function theory. In turn, functional programming has had wide in?uence in computing, both through developments within the dis- pline, such as formal semantics, polymorphic type checking, lazy evaluation and structural proof, and as a practical embodiment of formalized approaches, such as speci?cation, transformation and partial application. One of the engaging features of functional programming is precisely the crossover between theory and practice. In particular, it is regarded as ess- tial that all aspects of functional programming are appropriately formalized, especially the speci? cation and implementation of functional languages. Thus, specialist functional programming events like the International Workshop on the Implementation of Functional Languages (IFL) attract contributions where strong use is made of syntactic, semantic and meta-mathematical formalisms to motivate, justify and underpin very practical software systems. IFL grew out of smaller workshops aimed at practitioners wrestling with the nuts and bolts of making concrete implementations of highly abstract l- guages. Functional programming has always been bedeviled by an unwarranted reputation for slowand ine?cient implementations. IFL is one venue where such problemsaretackledheadon,alwaysusing formaltechniques to justify practical implementations.

Functional programminghas a long history,reaching back through early reali- tions in languages like LISP to foundational theories of computing, in particular ?-calculus and recursive function theory. In turn, functional programming has had wide in?uence in computing, both through developments within the dis- pline, such as formal semantics, polymorphic type checking, lazy evaluation and structural proof, and as a practical embodiment of formalized approaches, such as speci?cation, transformation and partial application. One of the engaging features of functional programming is precisely the crossover between theory and practice. In particular, it is regarded as ess- tial that all aspects of functional programming are appropriately formalized, especially the speci? cation and implementation of functional languages. Thus, specialist functional programming events like the International Workshop on the Implementation of Functional Languages (IFL) attract contributions where strong use is made of syntactic, semantic and meta-mathematical formalisms to motivate, justify and underpin very practical software systems. IFL grew out of smaller workshops aimed at practitioners wrestling with the nuts and bolts of making concrete implementations of highly abstract l- guages. Functional programming has always been bedeviled by an unwarranted reputation for slowand ine?cient implementations. IFL is one venue where such problemsaretackledheadon,alwaysusing formaltechniques to justify practical implementations.

53,49 €
49,99 € (zzgl. MwSt.)

Lieferfrist: bis zu 10 Tage

Artikelnummer: 9783540237273
Medium: Buch
ISBN: 978-3-540-23727-3
Verlag: Springer
Erscheinungstermin: 29.11.2004
Sprache(n): Englisch
Auflage: 1. Auflage 2004
Serie: Lecture Notes in Computer Science
Produktform: Kartoniert
Gewicht: 312 g
Seiten: 190
Format (B x H): 155 x 235 mm

Kundenservice Verlag Dr. Otto Schmidt KG
Neumannstraße 10, 40235 Düsseldorf | kundenservice@otto-schmidt.de | 0221-93738-997 (Inland)