Jouannaud

Functional Programming Languages and Computer Architecture

Proceedings, Nancy, France, September 16-19, 1985

Miranda: A non-strict functional language with polymorphic types.- Data flow graph optimization in if1.- Strictness analysis ¿ a practical approach.- The categorical abstract machine.- High order programming in extended FP.- Secd-m: a virtual machine for applicative programming.- Cobweb ¿ A combinator reduction architecture.- How to replace failure by a list of successes a method for exception handling, backtracking, and pattern matching in lazy functional languages.- Lazy memo-functions.- An architecture for fast data movement in the FFP machine.- An architecture that efficiently updates associative aggregates in applicative programming languages.- Lambda lifting: Transforming programs to recursive equations.- Optimizing almost-tail-recursive prolog programs.- Designing regular array architectures using higher order functions.- $$v\mathcal{F}\mathcal{P}$$: An environment for the multi-level specification, analysis, and synthesis of hardware algorithms.- A distributed garbage collection algorithm.- Cyclic reference counting for combinator machines.- Design for a multiprocessing heap with on-board reference counting.- A functional language and modular architecture for scientific computing.- Practical polymorphism.- Program verification in a logical theory of constructions.- Transforming recursive programs for execution on parallel machines.- Compiling pattern matching.- Serial combinators: "optimal" grains of parallelism.- The G-machine: A fast, graph-reduction evaluator.

Miranda: A non-strict functional language with polymorphic types.- Data flow graph optimization in if1.- Strictness analysis ¿ a practical approach.- The categorical abstract machine.- High order programming in extended FP.- Secd-m: a virtual machine for applicative programming.- Cobweb ¿ A combinator reduction architecture.- How to replace failure by a list of successes a method for exception handling, backtracking, and pattern matching in lazy functional languages.- Lazy memo-functions.- An architecture for fast data movement in the FFP machine.- An architecture that efficiently updates associative aggregates in applicative programming languages.- Lambda lifting: Transforming programs to recursive equations.- Optimizing almost-tail-recursive prolog programs.- Designing regular array architectures using higher order functions.- $$v\mathcal{F}\mathcal{P}$$: An environment for the multi-level specification, analysis, and synthesis of hardware algorithms.- A distributed garbage collection algorithm.- Cyclic reference counting for combinator machines.- Design for a multiprocessing heap with on-board reference counting.- A functional language and modular architecture for scientific computing.- Practical polymorphism.- Program verification in a logical theory of constructions.- Transforming recursive programs for execution on parallel machines.- Compiling pattern matching.- Serial combinators: "optimal" grains of parallelism.- The G-machine: A fast, graph-reduction evaluator.

53,49 €
49,99 € (zzgl. MwSt.)

Lieferfrist: bis zu 10 Tage

Artikelnummer: 9783540159759
Medium: Buch
ISBN: 978-3-540-15975-9
Verlag: Springer
Erscheinungstermin: 01.09.1985
Sprache(n): Englisch
Auflage: 1. Auflage 1985
Serie: Lecture Notes in Computer Science
Produktform: Kartoniert
Gewicht: 1310 g
Seiten: 416
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)