Kevin Hammond

List of Publications by Year
in descending order

65	1,386	687363 13	395702 33
papers	citations	h-index	g-index

71	71	71	445
all docs	docs citations	times ranked	citing authors

#	Article	IF	Citations
1	Report on the programming language Haskell. ACM SIGPLAN Notices, 1992, 27, 1-164.	0.2	539
2	Type classes in Haskell. ACM Transactions on Programming Languages and Systems, 1996, 18, 109-138.	2.1	142
3	Static determination of quantitative resource usage for higher-order programs. , 2010, , .		74
4	Benchmarking implementations of functional languages with â€˜Pseudoknotâ€™, a float-intensive benchmark. Journal of Functional Programming, 1996, 6, 621-655.	0.8	51
5	Inferring Cost Equations for Recursive, Polymorphic and Higher-Order Functional Programs. Lecture Notes in Computer Science, 2004, , 86-101.	1.3	50
6	Hume: A Domain-Specific Language for Real-Time Embedded Systems. Lecture Notes in Computer Science, 2003, , 37-56.	1.3	38
7	â€œCarbon Creditsâ€•for Resource-Bounded Computations Using Amortised Analysis. Lecture Notes in Computer Science, 2009, , 354-369.	1.3	37
8	Engineering parallel symbolic programs in GPH. Concurrency and Computation: Practice and Experience, 1999, 11, 701-752.	0.5	27
9	AUTOMATIC SKELETONS IN TEMPLATE HASKELL. Parallel Processing Letters, 2003, 13, 413-424.	0.6	24
10	A verified staged interpreter is a verified compiler. , 2006, , .		24
11	Automatic amortised analysis of dynamic memory allocation for lazy functional programs. , 2012, , .		24
12	The ParaPhrase Project: Parallel Patterns for Adaptive Heterogeneous Multicore Systems. Lecture Notes in Computer Science, 2013, , 218-236.	1.3	24
13	Cost-Directed Refactoring for Parallel Erlang Programs. International Journal of Parallel Programming, 2014, 42, 564-582.	1.5	23
14	Scrapping your inefficient engine. , 2010, , .		18
15	Discovering parallel pattern candidates in Erlang. , 2014, , .		15
16	Type-Based Cost Analysis for Lazy Functional Languages. Journal of Automated Reasoning, 2017, 59, 87-120.	1.4	14
17	A Dependently Typed Framework for Static Analysis of Program Execution Costs. Lecture Notes in Computer Science, 2006, , 74-90.	1.3	13
18	Static determination of quantitative resource usage for higher-order programs. ACM SIGPLAN Notices, 2010, 45, 223-236.	0.2	13

#	Article	IF	Citations
19	Type classes in Haskell. Lecture Notes in Computer Science, 1994, , 241-256.	1.3	13
20	ParaForming: Forming Parallel Haskell Programs Using Novel Refactoring Techniques. Lecture Notes in Computer Science, 2012, , 82-97.	1.3	13
21	Type-Based Allocation Analysis for Co-recursion in Lazy Functional Languages. Lecture Notes in Computer Science, 2015, , 787-811.	1.3	12
22	Granularity-Aware Work-Stealing for Computationally-Uniform Grids. , 2010, , .		11
23	Domain Specific Languages (DSLs) for Network Protocols (Position Paper). , 2009, , .		10
24	Agricultural Reform: More Efficient Farming Using Advanced Parallel Refactoring Tools. , 2014, , .		10
25	SymGrid-Par: Designing a Framework for Executing Computational Algebra Systems on Computational Grids. Lecture Notes in Computer Science, 2007, , 617-624.	1.3	10
26	A language-independent parallel refactoring framework. , 2012, , .		9
27	Why Parallel Functional Programming Matters: Panel Statement. Lecture Notes in Computer Science, 2011, , 201-205.	1.3	9
28	Resource-Safe Systems Programming with Embedded Domain Specific Languages. Lecture Notes in Computer Science, 2012, , 242-257.	1.3	9
29	Paraphrasing: Generating Parallel Programs Using Refactoring. Lecture Notes in Computer Science, 2013, , 237-256.	1.3	9
30	HaskSkel: Algorithmic Skeletons in Haskell. Lecture Notes in Computer Science, 2000, , 181-198.	1.3	8
31	Towards formally verifiable resource bounds for real-time embedded systems. ACM SIGBED Review, 2006, 3, 27-36.	1.8	8
32	Correct-by-Construction Concurrency: Using Dependent Types to Verify Implementations of Effectful Resource Usage Protocols. Fundamenta Informaticae, 2010, 102, 145-176.	0.4	8
33	Finding parallel functional pearls: Automatic parallel recursion scheme detection in Haskell functions via anti-unification. Future Generation Computer Systems, 2018, 79, 669-686.	7.5	7
34	Generic Access to Web and Grid-based Symbolic Computing Services: the SymGrid-Services Framework. , 2007, , .		6
35	Scrapping your inefficient engine. ACM SIGPLAN Notices, 2010, 45, 297-308.	0.2	6
36	Automatic amortised analysis of dynamic memory allocation for lazy functional programs. ACM SIGPLAN Notices, 2012, 47, 165-176.	0.2	6

#	Article	IF	Citations
37	Automatically deriving cost models for structured parallel processes using hylomorphisms. Future Generation Computer Systems, 2018, 79, 653-668.	7.5	6
38	Improving Granularity in Parallel Functional Programs: A Graphical Winnowing System for Haskell. Workshops in Computing, 1995, , 111-126.	0.4	6
39	The Missing Link! A New Skeleton for Evolutionary Multi-agent Systems in Erlang. International Journal of Parallel Programming, 2018, 46, 4-22.	1.5	5
40	Towards resource-certified software. , 2007, , .		4
41	Low-pain, high-gain multicore programming in Haskell. , 2008, , .		4
42	Making a packet: Cost-effective communication for a parallel graph reducer. Lecture Notes in Computer Science, 1997, , 184-199.	1.3	4
43	SymGrid: A Framework for Symbolic Computations on the Grid. , 2007, , .		3
44	Mapping parallel programs to heterogeneous CPU/GPU architectures using a Monte Carlo Tree Search. , 2013, , .		3
45	HPCâ€GAP: engineering a 21stâ€century highâ€performance computer algebra system. Concurrency Computation Practice and Experience, 2016, 28, 3606-3636.	2.2	3
46	Towards semi-automatic data-type translation for parallelism in Erlang. , 2016, , .		3
47	Extending the â€œOpen-Closed Principleâ€• to Automated Algorithm Configuration. Evolutionary Computation, 2019, 27, 173-193.	3.0	3
48	Farms, pipes, streams and reforestation: reasoning about structured parallel processes using types and hylomorphisms. ACM SIGPLAN Notices, 2016, 51, 4-17.	0.2	3
49	Mind Your Outcomes: The Î”QSD Paradigm for Quality-Centric Systems Development and Its Application to a Blockchain Case Study. Computers, 2022, 11, 45.	3.3	3
50	Using application information to drive adaptive grid middleware scheduling decisions. , 2008, , .		2
51	Comparing and Optimising Parallel Haskell Implementations for Multicore Machines. , 2009, , .		2
52	PAEAN: Portable and scalable runtime support for parallel Haskell dialects. Journal of Functional Programming, 2016, 26, .	0.8	2
53	How to be a Successful Thief. Lecture Notes in Computer Science, 2013, , 114-125.	1.3	2
54	Improving Your CASH Flow: The Computer Algebra SHell. Lecture Notes in Computer Science, 2011, , 169-184.	1.3	1

#	Article	IF	Citations
55	Naira: A parallel 2Haskell compiler. Lecture Notes in Computer Science, 1998, , 214-230.	1.3	1
56	The Peter Landin prize. Higher-Order and Symbolic Computation, 2009, 22, 305-312.	0.3	0
57	GUEST EDITORS NOTE: HIGH-LEVEL PROGRAMMING FOR HETEROGENEOUS AND HIERARCHICAL PARALLEL SYSTEMS. Parallel Processing Letters, 2012, 22, 1202002.	0.6	0
58	Kindergarten cop: dynamic nursery resizing for GHC. , 2016, , .		0
59	In search of a map: using program slicing to discover potential parallelism in recursive functions. , 2017, , .		0
60	Special issue on Parallel and distributed computing based on the functional programming paradigm. Concurrency Computation Practice and Experience, 2018, 30, e4842.	2.2	0
61	Learning-Based Dynamic Pinning of Parallelized Applications in Many-Core Systems. , 2019, , .		0
62	Refactoring for introducing and tuning parallelism for heterogeneous multicore machines in Erlang. Concurrency Computation Practice and Experience, 2021, 33, e5420.	2.2	0
63	Repeating History: Execution Replay for Parallel Haskell Programs. Lecture Notes in Computer Science, 2013, , 231-246.	1.3	0
64	Timing Properties and Correctness for Structured Parallel Programs on x86-64 Multicores. Lecture Notes in Computer Science, 2016, , 101-125.	1.3	0
65	Flexible Formality Practical Experience with Agile Formal Methods. Lecture Notes in Computer Science, 2020, , 94-120.	1.3	0