Hanspeter MÃ¶ssenbÃ¶ck

List of Publications by Year
in descending order

133	1,514	933447	794594
papers	citations	10 h-index	19 g-index

138	138	138	401
all docs	docs citations	times ranked	citing authors

#	Article	IF	Citations
1	Design of the Java HotSpotâ„¢ client compiler for Java 6. Transactions on Architecture and Code Optimization, 2008, 5, 1-32.	2.0	121
2	An intermediate representation for speculative optimizations in a dynamic compiler. , 2013, , .		82
3	Partial Escape Analysis and Scalar Replacement for Java. , 2014, , .		60
4	Single-pass generation of static single-assignment form for structured languages. ACM Transactions on Programming Languages and Systems, 1994, 16, 1684-1698.	2.1	55
5	Optimized interval splitting in a linear scan register allocator. , 2005, , .		51
6	Escape analysis in the context of dynamic compilation and deoptimization. , 2005, , .		41
7	An object storage model for the truffle language implementation framework. , 2014, , .		41
8	Cross-language compiler benchmarking: are we fast yet?. , 2016, , .		35
9	A Comprehensive Solution for Deterministic Replay Debugging of SoftPLC Applications. IEEE Transactions on Industrial Informatics, 2011, 7, 641-651.	11.3	32
10	Bringing low-level languages to the JVM: efficient execution of LLVM IR on Truffle. , 2016, , .		30
11	High-performance cross-language interoperability in a multi-language runtime. , 2015, , .		30
12	Array bounds check elimination for the Java HotSpot#8482; client compiler. , 2007, , .		27
13	A Comprehensive Java Benchmark Study on Memory and Garbage Collection Behavior of DaCapo, DaCapo Scala, and SPECjvm2008. , 2017, , .		26
14	Dominance-based duplication simulation (DBDS): code duplication to enable compiler optimizations. , 2018, , .		26
15	The taming of the shrew. , 2014, , .		25
16	Accurate and Efficient Object Tracing for Java Applications. , 2015, , .		25
17	An experimental study of the influence of dynamic compiler optimizations on Scala performance. , 2013, , .		24
18	Dynamically composing languages in a modular way: supporting C extensions for dynamic languages. , 2015, , .		23

#	Article	IF	Citations
19	Trace-based compilation for the Java HotSpot virtual machine. , 2011, , .		22
20	Partial Escape Analysis and Scalar Replacement for Java. , 2014, , .		22
21	Speculation without regret. , 2014, , .		19
22	TruffleC. , 2014, , .		19
23	Automatic feedback-directed object inlining in the java hotspotâ„¢ virtual machine. , 2007, , .		18
24	Run-Time Support for Optimizations Based on Escape Analysis. , 2007, , .		18
25	Efficient and thread-safe objects for dynamically-typed languages. , 2016, , .		18
26	A Component Plug-In Architecture for the .NET Platform. Lecture Notes in Computer Science, 2006, , 287-305.	1.3	16
27	Linear Scan Register Allocation in the Context of SSA Form and Register Constraints. Lecture Notes in Computer Science, 2002, , 229-246.	1.3	16
28	Automatic array inlining in java virtual machines. , 2008, , .		15
29	Memory-safe Execution of C on a Java VM. , 2015, , .		15
30	Efficient and deterministic record & replay for actor languages. , 2018, , .		15
31	Cross-Language Interoperability in a Multi-Language Runtime. ACM Transactions on Programming Languages and Systems, 2018, 40, 1-43.	2.1	15
32	Lazy continuations for Java virtual machines. , 2009, , .		14
33	Compilation queuing and graph caching for dynamic compilers. , 2012, , .		14
34	Efficient Rebuilding of Large Java Heaps from Event Traces. , 2015, , .		13
35	A Study of Concurrency Bugs and Advanced Development Support for Actor-based Programs. Lecture Notes in Computer Science, 2018, , 155-185.	1.3	12
36	High-performance cross-language interoperability in a multi-language runtime. ACM SIGPLAN Notices, 2016, 51, 78-90.	0.2	12

#	Article	IF	Citations
37	Array bounds check elimination in the context of deoptimization. Science of Computer Programming, 2009, 74, 279-295.	1.9	11
38	Monacoâ€”A domain-specific language solution for reactive process control programming with hierarchical components. Computer Languages, Systems and Structures, 2013, 39, 67-94.	1.4	11
39	An efficient native function interface for Java. , 2013, , .		11
40	Efficient Memory Traces with Full Pointer Information. , 2016, , .		11
41	An Analysis of x86-64 Inline Assembly in C Programs. , 2018, , .		11
42	A generator for production quality compilers. Lecture Notes in Computer Science, 1991, , 42-55.	1.3	11
43	Visualization of Program Dependence Graphs. , 2008, , 193-196.		10
44	Sulong, and Thanks for All the Bugs. , 2018, , .		10
45	Designing a framework by stepwise generalization. Lecture Notes in Computer Science, 1995, , 479-498.	1.3	10
46	Active Text for Structuring and Understanding Source Code. Software - Practice and Experience, 1996, 26, 833-850.	3.6	9
47	Context-sensitive trace inlining for Java. Computer Languages, Systems and Structures, 2013, 39, 123-141.	1.4	9
48	A concurrency-agnostic protocol for multi-paradigm concurrent debugging tools. , 2017, , .		9
49	Multi-language dynamic taint analysis in a polyglot virtual machine. , 2020, , .		9
50	Automatic feedback-directed object fusing. Transactions on Architecture and Code Optimization, 2010, 7, 1-35.	2.0	8
51	Efficient and Viable Handling of Large Object Traces. , 2016, , .		8
52	User-defined Classification and Multi-level Grouping of Objects in Memory Monitoring. , 2018, , .		8
53	Understanding GCC builtins to develop better tools. , 2019, , .		8
54	Analyzing Data Structure Growth Over Time to Facilitate Memory Leak Detection. , 2019, , .		8

#	Article	IF	Citations
55	Efficient Tracing and Versatile Analysis of Lock Contention in Java Applications on the Virtual Machine Level. , 2016, , .		8
56	User-centered Offline Analysis of Memory Monitoring Data. , 2017, , .		8
57	Improving aspect-oriented programming with dynamic code evolution in an enhanced Java virtual machine. , 2010, , .		7
58	Evaluation of trace inlining heuristics for Java. , 2012, , .		7
59	Lightweight Java Profiling with Partial Safepoints and Incremental Stack Tracing. , 2015, , .		7
60	Trace-based Register Allocation in a JIT Compiler. , 2016, , .		7
61	Lenient Execution of C on a Java Virtual Machine. , 2017, , .		7
62	Utilizing object reference graphs and garbage collection roots to detect memory leaks in offline memory monitoring. , 2018, , .		7
63	Parallelization of dynamic languages: synchronizing built-in collections. , 2018, 2, 1-30.		7
64	Applications of enhanced dynamic code evolution for Java in GUI development and dynamic aspect-oriented programming. , 2010, , .		6
65	Adding genericity to a plug-in framework. , 2010, , .		6
66	An efficient approach for accessing C data structures from JavaScript. , 2014, , .		6
67	Fast Java profiling with scheduling-aware stack fragment sampling and asynchronous analysis. , 2014, , .		6
68	Techniques and applications for guest-language safepoints. , 2015, , .		6
69	Sulong - execution of LLVM-based languages on the JVM. , 2016, , .		6
70	Trace Register Allocation Policies. , 2017, , .		6
71	Fast-path loop unrolling of non-counted loops to enable subsequent compiler optimizations. , 2018, , .		6
72	Introspection for C and its Applications to Library Robustness. The Art Science and Engineering of Programming, 2017, 2, .	0.5	6

#	ARTICLE	IF	CITATIONS
73	Rule-Based Composition Behaviors in Dynamic Plug-In Systems. , 2010, , .		5
74	Efficient and accurate stack trace sampling in the Java hotspot virtual machine. , 2014, , .		5
75	Java-to-JavaScript translation via structured control flow reconstruction of compiler IR. , 2015, , .		5
76	KÃ³Î¼Ï€oÏ¸. , 2017, , .		5
77	Towards efficient, multi-language dynamic taint analysis. , 2019, , .		5
78	Zero-overhead exception handling using metaprogramming. Lecture Notes in Computer Science, 1997, , 423-431.	1.3	5
79	A cost model for a graph-based intermediate-representation in a dynamic compiler. , 2018, , .		5
80	AntTracks TrendViz. , 2019, , .		5
81	Memory Cities: Visualizing Heap Memory Evolution Using the Software City Metaphor. , 2020, , .		5
82	Deterministic Replay Debugging of IEC 61131-3 SoftPLC programs. , 2010, , .		4
83	Trace transitioning and exception handling in a trace-based JIT compiler for java. Transactions on Architecture and Code Optimization, 2014, 11, 1-26.	2.0	4
84	Efficient Sampling-based Lock Contention Profiling for Java. , 2017, , .		4
85	Applying Optimizations for Dynamically-typed Languages to Java. , 2017, , .		4
86	Can we Predict Performance Events with Time Series Data from Monitoring Multiple Systems?. , 2019, , .		4
87	Where has all my memory gone?. , 2013, , .		4
88	CompGen: generation of fast JIT compilers in a multi-language VM. , 2021, , .		4
89	The Domain-Specific Language Monaco and its Visual Interactive Programming Environment. , 2007, , .		3
90	Testing the composability of plug-and-play components: A method for unit testing of dynamically composed applications. , 2010, , .		3

#	Article	IF	Citations
91	Adding genericity to a plug-in framework. ACM SIGPLAN Notices, 2011, 46, 93-102.	0.2	3
92	Sampling-based Steal Time Accounting under Hardware Virtualization. , 2015, , .		3
93	Context-Aware Failure-Oblivious Computing as a Means of Preventing Buffer Overflows. Lecture Notes in Computer Science, 2018, , 376-390.	1.3	3
94	Efficient and thread-safe objects for dynamically-typed languages. ACM SIGPLAN Notices, 2016, 51, 642-659.	0.2	3
95	An Analysis of x86-64 Inline Assembly in C Programs. ACM SIGPLAN Notices, 2018, 53, 84-99.	0.2	3
96	A convenient way to incorporate semantic actions in two-pass compiling schemes. Software - Practice and Experience, 1988, 18, 691-700.	3.6	2
97	Optimized strings for the Java HotSpot#8482; virtual machine. , 2008, , .		2
98	Deriving code coverage information from profiling data recorded for a trace-based just-in-time compiler. , 2013, , .		2
99	Composing user-specific web applications from distributed plug-ins. Computer Science - Research and Development, 2013, 28, 85-105.	2.7	2
100	Efficient dynamic analysis of the synchronization performance of Java applications. , 2015, , .		2
101	Cross-language compiler benchmarking: are we fast yet?. ACM SIGPLAN Notices, 2017, 52, 120-131.	0.2	2
102	A principled approach towards debugging communicating event-loops. , 2017, , .		2
103	Using Crash Frequency Analysis to Identify Error-Prone Software Technologies in Multi-System Monitoring. , 2018, , .		2
104	Using machine learning to predict the code size impact of duplication heuristics in a dynamic compiler. , 2021, , .		2
105	Low-overhead multi-language dynamic taint analysis on managed runtimes through speculative optimization. , 2021, , .		2
106	trcview: Interactive Architecture Agnostic Execution Trace Analysis. , 2020, , .		2
107	Treating statement sequences as block objects. ACM SIGPLAN Notices, 1992, 27, 83-86.	0.2	2
108	Supporting on-stack replacement in unstructured languages by loop reconstruction and extraction. , 2019, , .		2

#	Article	IF	Citations
109	SymJEx: symbolic execution on the GraalVM. , 2020, , .		2
110	The Oberon-2 Reflection Model and its Applications. Lecture Notes in Computer Science, 1999, , 40-53.	1.3	1
111	Supporting Model Maintenance in Component-based Product Lines. , 2012, , .		1
112	Debugging native extensions of dynamic languages. , 2018, , .		1
113	Dynamic fitness functions for genetic improvement in compilers and interpreters. , 2018, , .		1
114	A Framework for Preprocessing Multivariate, Topology-Aware Time Series and Event Data in a Multi-System Environment. , 2019, , .		1
115	Trends in object-oriented programming. ACM Computing Surveys, 1996, 28, 160.	23.0	1
116	Java-to-JavaScript translation via structured control flow reconstruction of compiler IR. ACM SIGPLAN Notices, 2016, 51, 91-103.	0.2	1
117	Asynchronous snapshots of actor systems for latency-sensitive applications. , 2019, , .		1
118	Object-oriented programmingâ€"what for?. Journal of Microcomputer Applications, 1991, 14, 217-228.	0.1	0
119	Compact and efficient strings for Java. Science of Computer Programming, 2010, 75, 1077-1094.	1.9	0
120	Optimized memory management for class metadata in a JVM. , 2011, , .		0
121	Composition mechanisms classified by their contributor provision characteristics. , 2012, , .		0
122	DuckTracks. , 2017, , .		0
123	Parallel trace register allocation. , 2018, , .		0
124	Sulong, and thanks for all the fish. , 2018, , .		0
125	Dominance-based duplication simulation (DBDS): code duplication to enable compiler optimizations. , 2018, , .		0
126	Architecture-agnostic dynamic type recovery. , 2021, , .		0

#	Article	IF	Citations
127	Data Mappings in the Model-View-Controller Pattern. Lecture Notes in Computer Science, 2004, , 121-132.	1.3	0
128	Automatic Object Colocation Based on Read Barriers. Lecture Notes in Computer Science, 2006, , 326-345.	1.3	0
129	Applications of enhanced dynamic code evolution for Java in GUI development and dynamic aspect-oriented programming. ACM SIGPLAN Notices, 2011, 46, 123-126.	0.2	0
130	Safe and atomic run-time code evolution for Java and its application to dynamic AOP. ACM SIGPLAN Notices, 2011, 46, 825-844.	0.2	0
131	A concurrency-agnostic protocol for multi-paradigm concurrent debugging tools. ACM SIGPLAN Notices, 2017, 52, 3-14.	0.2	0
132	Sulong, and Thanks for All the Bugs. ACM SIGPLAN Notices, 2018, 53, 377-391.	0.2	0
133	Promoting Talents for Computer Science. , 2019, , .		0