Shan Lu

List of Publications by Year
in descending order

100	4,362	15	26
papers	citations	h-index (567281)	g-index (552781)
103	103	103	1172
all docs	docs citations	times ranked	citing authors

#	Article	IF	Citations
1	Learning from mistakes. , 2008, , .		562
2	AVIO. , 2006, , .		294
3	CTrigger. , 2009, , .		242
4	PRES. , 2009, , .		221
5	Have things changed now?. , 2006, , .		201
6	Understanding and detecting real-world performance bugs. , 2012, , .		201
7	MUVI. , 2007, , .		152
8	Automated atomicity-violation fixing. , 2011, , .		135
9	SafeMem: Exploiting ECC-Memory for Detecting Memory Leaks and Memory Corruption During Production Runs. , 0, , .		94
10	Instrumentation and sampling strategies for cooperative concurrency bug isolation. , 2010, , .		91
11	ConMem. , 2010, , .		87
12	ConSeq. , 2011, , .		85
13	Understanding and detecting real-world performance bugs. ACM SIGPLAN Notices, 2012, 47, 77-88.	0.2	83
14	Learning from mistakes. ACM SIGPLAN Notices, 2008, 43, 329-339.	0.2	80
15	AccMon: Automatically Detecting Memory-Related Bugs via Program Counter-Based Invariants. , 0, , .		77
16	TaxDC. , 2016, , .		72
17	Do I use the wrong definition?. , 2010, , .		67
18	Toddler: Detecting performance problems via similar memory-access patterns. , 2013, , .		61

#	Article	IF	Citations
19	A study of interleaving coverage criteria. , 2007, , .		58
20	CARAMEL: Detecting and Fixing Performance Problems That Have Non-Intrusive Fixes. , 2015, , .		53
21	AVIO: Detecting Atomicity Violations via Access-Interleaving Invariants. IEEE Micro, 2007, 27, 26-35.	1.8	52
22	Statistical debugging for real-world performance problems. , 2014, , .		46
23	Automated atomicity-violation fixing. ACM SIGPLAN Notices, 2011, 46, 389-400.	0.2	45
24	How <i>not</i> to structure your database-backed web applications. , 2018, , .		44
25	AutoTap: Synthesizing and Repairing Trigger-Action Programs Using LTL Properties. , 2019, , .		43
26	A Study of Linux File System Evolution. ACM Transactions on Storage, 2014, 10, 1-32.	2.1	41
27	Efficient scalable thread-safety-violation detection. , 2019, , .		41
28	Interruptible tasks. , 2015, , .		40
29	Sweeper. , 2007, , .		37
30	Performance Diagnosis for Inefficient Loops. , 2017, , .		37
31	ConAir. , 2013, , .		35
32	Learning from mistakes. Computer Architecture News, 2008, 36, 329-339.	2.5	32
33	Learning from mistakes. Operating Systems Review (ACM), 2008, 42, 329-339.	1.9	32
34	Production-run software failure diagnosis via hardware performance counters. , 2013, , .		31
35	MUVI. Operating Systems Review (ACM), 2007, 41, 103-116.	1.9	28
36	What change history tells us about thread synchronization. , 2015, , .		28

#	Article	IF	Citations
37	DCatch. , 2017, , .		28
38	Understanding and Auto-Adjusting Performance-Sensitive Configurations. , 2018, , .		28
39	What bugs cause production cloud incidents?. , 2019, , .		28
40	CTrigger. ACM SIGPLAN Notices, 2009, 44, 25-36.	0.2	27
41	Finding Atomicity-Violation Bugs through Unserializable Interleaving Testing. IEEE Transactions on Software Engineering, 2012, 38, 844-860.	5.6	27
42	ConMem. ACM SIGPLAN Notices, 2010, 45, 179-192.	0.2	26
43	Understanding and generating high quality patches for concurrency bugs. , 2016, , .		23
44	Skyway. , 2018, , .		23
45	Leveraging the short-term memory of hardware to diagnose production-run software failures. , 2014, , .		20
46	Understanding and automatically detecting conflicting interactions between smart home IoT applications. , 2020, , .		20
47	AVIO. Operating Systems Review (ACM), 2006, 40, 37-48.	1.9	19
48	Efficient concurrency-bug detection across inputs. , 2013, , .		19
49	AVIO. ACM SIGPLAN Notices, 2006, 41, 37-48.	0.2	18
50	Instrumentation and sampling strategies for cooperative concurrency bug isolation. ACM SIGPLAN Notices, 2010, 45, 241-255.	0.2	17
51	Applying transactional memory to concurrency bugs. , 2012, , .		17
52	PathExpander: Architectural Support for Increasing the Path Coverage of Dynamic Bug Detection. Microarchitecture (MICRO), Proceedings of the Annual International Symposium on, 2006, , .	0.0	16
53	Pcatch. , 2018, , .		16
54	FCatch. , 2018, , .		16

#	Article	IF	Citations
55	Do I use the wrong definition?. ACM SIGPLAN Notices, 2010, 45, 160-174.	0.2	15
56	Statically inferring performance properties of software configurations. , 2020, , .		15
57	Understanding and Detecting Software Upgrade Failures in Distributed Systems. , 2021, , .		15
58	Statistical debugging for real-world performance problems. ACM SIGPLAN Notices, 2014, 49, 561-578.	0.2	14
59	TaxDC. ACM SIGPLAN Notices, 2016, 51, 517-530.	0.2	14
60	Visualizing Differences to Improve End-User Understanding of Trigger-Action Programs. , 2020, , .		14
61	Gerenuk. , 2019, , .		13
62	A study of interleaving coverage criteria. , 2007, , .		12
63	CTrigger. Computer Architecture News, 2009, 37, 25-36.	2.5	12
64	Leveraging parallelism for multi-dimensional packetclassification on software routers. Performance Evaluation Review, 2010, 38, 227-238.	0.6	12
65	ConSeq. ACM SIGPLAN Notices, 2011, 46, 251-264.	0.2	12
66	ConMem. ACM Transactions on Software Engineering and Methodology, 2013, 22, 1-33.	6.0	12
67	View-Centric Performance Optimization for Database-Backed Web Applications. , 2019, , .		12
68	Sweeper. Operating Systems Review (ACM), 2007, 41, 115-128.	1.9	11
69	ConSeq. Computer Architecture News, 2011, 39, 251-264.	2.5	11
70	AI: a lightweight system for tolerating concurrency bugs. , 2014, , .		10
71	DFix: automatically fixing timing bugs in distributed systems. , 2019, , .		10
72	Detecting Concurrency Bugs from the Perspectives of Synchronization Intentions. IEEE Transactions on Parallel and Distributed Systems, 2012, 23, 1060-1072.	5.6	9

#	Article	IF	Citations
73	Low-overhead and fully automated statistical debugging with abstraction refinement. , 2016, , .		9
74	Efficient concurrency-bug detection across inputs. ACM SIGPLAN Notices, 2013, 48, 785-802.	0.2	8
75	Fixing, preventing, and recovering from concurrency bugs. Science China Information Sciences, 2015, 58, 1-18.	4.3	8
76	A Lightweight System for Detecting and Tolerating Concurrency Bugs. IEEE Transactions on Software Engineering, 2016, 42, 899-917.	5.6	7
77	DCatch. Computer Architecture News, 2017, 45, 677-691.	2.5	7
78	AVIO. Computer Architecture News, 2006, 34, 37-48.	2.5	5
79	ConMem. Computer Architecture News, 2010, 38, 179-192.	2.5	5
80	TaxDC. Computer Architecture News, 2016, 44, 517-530.	2.5	5
81	Applying transactional memory to concurrency bugs. ACM SIGPLAN Notices, 2012, 47, 211-222.	0.2	4
82	ConSeq. ACM SIGPLAN Notices, 2012, 47, 251.	0.2	4
83	ConAir. Computer Architecture News, 2013, 41, 113-126.	2.5	4
84	DCatch. ACM SIGPLAN Notices, 2017, 52, 677-691.	0.2	3
85	SherLock: unsupervised synchronization-operation inference. , 2021, , .		3
86	Production-run software failure diagnosis via hardware performance counters. ACM SIGPLAN Notices, 2013, 48, 101-112.	0.2	3
87	DCatch. Operating Systems Review (ACM), 2017, 51, 677-691.	1.9	2
88	Hytrace. , 2017, , .		2
89	Automated atomicity-violation fixing. ACM SIGPLAN Notices, 2012, 47, 389.	0.2	2
90	Validating Library Usage Interactively. Lecture Notes in Computer Science, 2013, , 796-812.	1.3	2

#	Article	IF	Citations
91	Leveraging the short-term memory of hardware to diagnose production-run software failures. ACM SIGPLAN Notices, 2014, 49, 207-222.	0.2	2
92	TaxDC. Operating Systems Review (ACM), 2016, 50, 517-530.	1.9	2
93	Leveraging the short-term memory of hardware to diagnose production-run software failures. Computer Architecture News, 2014, 42, 207-222.	2.5	1
94	Production-run software failure diagnosis via hardware performance counters. Computer Architecture News, 2013, 41, 101-112.	2.5	1
95	ConAir. ACM SIGPLAN Notices, 2013, 48, 113-126.	0.2	1
96	Toward More Efficient Statistical Debugging with Abstraction Refinement. ACM Transactions on Software Engineering and Methodology, 2023, 32, 1-38.	6.0	1
97	Applying transactional memory to concurrency bugs. Computer Architecture News, 2012, 40, 211-222.	2.5	0
98	Roundtable: Research Opportunities and Challenges for Large-Scale Software Systems. Journal of Computer Science and Technology, 2016, 31, 851-860.	1.5	0
99	Applying Transactional Memory for Concurrency-Bug Failure Recovery in Production Runs. IEEE Transactions on Parallel and Distributed Systems, 2019, 30, 990-1006.	5.6	0
100	Low-overhead and fully automated statistical debugging with abstraction refinement. ACM SIGPLAN Notices, 2016, 51, 881-896.	0.2	0