Zhendong Su

List of Publications by Year
in descending order

Source: https://exaly.com/author-pdf/10207074/publications.pdf

Version: 2024-02-01

142 papers	7,337 citations	394421 19 h-index	265206 42 g-index
145 all docs	145 docs citations	145 times ranked	2480 citing authors

#	Article	IF	Citations
1	DECKARD: Scalable and Accurate Tree-Based Detection of Code Clones. Proceedings - International Conference on Software Engineering, 2007, , .	0.0	609
2	On the naturalness of software. , 2012, , .		345
3	The essence of command injection attacks in web applications. , 2006, , .		258
4	Static detection of cross-site scripting vulnerabilities. , 2008, , .		253
5	Guided, stochastic model-based GUI testing of Android apps. , 2017, , .		214
6	Scalable detection of semantic clones. , 2008, , .		208
7	Sound and precise analysis of web applications for injection vulnerabilities. , 2007, , .		193
8	Compiler validation via equivalence modulo inputs. , 2014, , .		193
9	FIREMAN: a toolkit for firewall modeling and analysis. , 2006, , .		168
10	On the naturalness of software. Communications of the ACM, 2016, 59, 122-131.	4.5	158
11	On the localness of software. , 2014, , .		156
12	A study of the uniqueness of source code. , 2010, , .		142
13	Context-based detection of clone-related bugs. , 2007, , .		139
14	Compiler validation via equivalence modulo inputs. ACM SIGPLAN Notices, 2014, 49, 216-226.	0.2	135
15	Partial online cycle elimination in inclusion constraint graphs. , 1998, , .		134
16	Javert. , 2008, , .		133
17	Automatic mining of functionally equivalent code fragments via random testing. , 2009, , .		120
18	Dynamic test input generation for web applications. , 2008, , .		114

#	Article	IF	Citations
19	On deriving unknown vulnerabilities from zero-day polymorphic and metamorphic worm exploits. , 2005, , .		108
20	Detecting code clones in binary executables. , 2009, , .		107
21	The essence of command injection attacks in web applications. ACM SIGPLAN Notices, 2006, 41, 372-382.	0.2	102
22	Coverage-directed differential testing of JVM implementations. , 2016, , .		91
23	Finding deep compiler bugs via guided stochastic program mutation. , 2015, , .		89
24	Finding compiler bugs via live code mutation. , 2016, , .		82
25	Practical GUI Testing of Android Applications Via Model Abstraction and Refinement. , 2019, , .		80
26	Context-aware statistical debugging. , 2007, , .		78
27	Large-scale analysis of framework-specific exceptions in Android apps. , 2018, , .		75
28	Automatic detection of floating-point exceptions. , 2013, , .		72
29	Online inference and enforcement of temporal properties. , 2010, , .		70
30	An Empirical Study on Real Bug Fixes. , 2015, , .		66
31	An empirical analysis of the co-evolution of schema and code in database applications. , 2013, , .		65
32	Has the bug really been fixed?. , 2010, , .		63
33	Toward understanding compiler bugs in GCC and LLVM. , 2016, , .		63
34	A Survey on Data-Flow Testing. ACM Computing Surveys, 2018, 50, 1-35.	23.0	62
35	SmartSynth. , 2013, , .		61
36	Perses. , 2018, , .		58

#	Article	IF	Citations
37	Sound and precise analysis of web applications for injection vulnerabilities. ACM SIGPLAN Notices, 2007, 42, 32-41.	0.2	56
38	Projection merging. , 2000, , .		56
39	Static checking of dynamically generated queries in database applications. ACM Transactions on Software Engineering and Methodology, 2007, 16, 14.	6.0	55
40	Symbolic mining of temporal specifications. , 2008, , .		54
41	Steering symbolic execution to less traveled paths. , 2013, , .		54
42	JDBC checker: a static analysis tool for SQL/JDBC applications. , 0, , .		53
43	Synthesizing method sequences for high-coverage testing. , 2011, , .		53
44	Skeletal program enumeration for rigorous compiler testing. , 2017, , .		52
45	Finding and analyzing compiler warning defects. , 2016, , .		50
46	Search, align, and repair: data-driven feedback generation for introductory programming exercises. , 2018, , .		49
47	Fast algorithms for Dyck-CFL-reachability with applications to alias analysis. , 2013, , .		47
48	Guided differential testing of certificate validation in SSL/TLS implementations. , 2015, , .		47
49	Structure-invariant testing for machine translation. , 2020, , .		47
50	Detecting nondeterministic payment bugs in Ethereum smart contracts. , 2019, 3, 1-29.		45
51	Finding deep compiler bugs via guided stochastic program mutation. ACM SIGPLAN Notices, 2015, 50, 386-399.	0.2	45
52	Detecting API documentation errors. , 2013, , .		44
53	Deep Differential Testing of JVM Implementations. , 2019, , .		42
54	Scalable and systematic detection of buggy inconsistencies in source code. , 2010, , .		41

#	Article	IF	Citations
55	Validating SMT solvers via semantic fusion. , 2020, , .		40
56	Randomized stress-testing of link-time optimizers. , 2015, , .		39
57	Perturbing numerical calculations for statistical analysis of floating-point program (in)stability. , 2010, , .		38
58	Metamorphic object insertion for testing object detection systems. , 2020, , .		37
59	Detecting optimization bugs in database engines via non-optimizing reference engine construction. , 2020, , .		32
60	Back to the Future: A Framework for Automatic Malware Removal and System Repair. Proceedings of the Computer Security Applications Conference, 2006, , .	0.0	31
61	Blended, precise semantic program embeddings. , 2020, , .		31
62	ExecRecorder. , 2006, , .		30
63	Context-sensitive data-dependence analysis via linear conjunctive language reachability. , 2017, , .		30
64	Finding bugs in database systems via query partitioning. , 2020, 4, 1-30.		30
65	A general framework for benchmarking firewall optimization techniques. IEEE Transactions on Network and Service Management, 2008, 5, 227-238.	4.9	29
66	Partial online cycle elimination in inclusion constraint graphs. ACM SIGPLAN Notices, 1998, 33, 85-96.	0.2	29
67	A class of polynomially solvable range constraints for interval analysis without widenings. Theoretical Computer Science, 2005, 345, 122-138.	0.9	28
68	On the unusual effectiveness of type-aware operator mutations for testing SMT solvers. , 2020, 4, 1-25.		28
69	Exploring and exploiting the correlations between bug-inducing and bug-fixing commits. , 2019, , .		27
70	Benchmarking automated GUI testing for Android against real-world bugs. , 2021, , .		27
71	Exposing Library API Misuses Via Mutation Analysis. , 2019, , .		26
72	Temporal search. , 2006, , .		24

#	Article	IF	Citations
73	How test suites impact fault localisation starting from the size. IET Software, 2018, 12, 190-205.	2.1	24
74	A Genetic Algorithm for Detecting Significant Floating-Point Inaccuracies. , 2015, , .		23
75	XSat: A Fast Floating-Point Satisfiability Solver. Lecture Notes in Computer Science, 2016, , 187-209.	1.3	23
76	Detecting Energy Bugs in Android Apps Using Static Analysis. Lecture Notes in Computer Science, 2017, , 192-208.	1.3	23
77	Testing Machine Translation via Referential Transparency. , 2021, , .		23
78	Testing mined specifications. , 2012, , .		21
79	Detecting API documentation errors. ACM SIGPLAN Notices, 2013, 48, 803-816.	0.2	21
80	A toolkit for constructing type- and constraint-based program analyses. Lecture Notes in Computer Science, 1998, , 78-96.	1.3	20
81	Efficient subcubic alias analysis for C. , 2014, , .		20
82	Automatic detection of floating-point exceptions. ACM SIGPLAN Notices, 2013, 48, 549-560.	0.2	19
83	Client-Side Detection of XSS Worms by Monitoring Payload Propagation. Lecture Notes in Computer Science, 2009, , 539-554.	1.3	19
84	Stochastic Optimization of Program Obfuscation. , 2017, , .		18
85	Hunting for Bugs in Code Coverage Tools via Randomized Differential Testing. , 2019, , .		18
86	Understanding and finding system setting-related defects in Android apps. , 2021, , .		18
87	Abstracting runtime heaps for program understanding. IEEE Transactions on Software Engineering, 2013, 39, 774-786.	5.6	17
88	Automatic runtime recovery via error handler synthesis. , 2016, , .		17
89	Finding compiler bugs via live code mutation. ACM SIGPLAN Notices, 2016, 51, 849-863.	0.2	17
90	Combining Symbolic Execution and Model Checking for Data Flow Testing. , 2015, , .		16

#	Article	IF	Citations
91	Skeletal program enumeration for rigorous compiler testing. ACM SIGPLAN Notices, 2017, 52, 347-361.	0.2	16
92	Detecting floating-point errors via atomic conditions. , 2020, 4, 1-27.		16
93	The first-order theory of subtyping constraints. , 2002, , .		15
94	Machine translation testing via pathological invariance. , 2020, , .		15
95	Reusing debugging knowledge via trace-based bug search. , 2012, , .		14
96	Finding and understanding bugs in software model checkers. , 2019, , .		14
97	Search, align, and repair: data-driven feedback generation for introductory programming exercises. ACM SIGPLAN Notices, 2018, 53, 481-495.	0.2	14
98	Automatic detection of unsafe component loadings. , 2010, , .		13
99	Coverage-directed differential testing of JVM implementations. ACM SIGPLAN Notices, 2016, 51, 85-99.	0.2	13
100	Generative type-aware mutation for testing SMT solvers. , 2021, 5, 1-19.		13
101	A Class of Polynomially Solvable Range Constraints for Interval Analysis without Widenings and Narrowings. Lecture Notes in Computer Science, 2004, , 280-295.	1.3	12
102	Automated backward error analysis for numerical code. , 2015, , .		12
103	Fast algorithms for Dyck-CFL-reachability with applications to alias analysis. ACM SIGPLAN Notices, 2013, 48, 435-446.	0.2	11
104	Symbolic verification of message passing interface programs. , 2020, , .		11
105	Detecting races in relay ladder logic programs. Lecture Notes in Computer Science, 1998, , 184-200.	1.3	10
106	Scalable and systematic detection of buggy inconsistencies in source code. ACM SIGPLAN Notices, 2010, 45, 175-190.	0.2	10
107	Understanding the syntactic rule usage in java. Journal of Systems and Software, 2017, 123, 160-172.	4.5	10
108	Symbolic verification of regular properties. , 2018, , .		10

#	Article	IF	Citations
109	Context-sensitive data-dependence analysis via linear conjunctive language reachability. ACM SIGPLAN Notices, 2017, 52, 344-358.	0.2	10
110	Bezoar: Automated virtual machine-based full-system recovery from control-flow hijacking attacks. , 2008, , .		9
111	Achieving high coverage for floating-point code via unconstrained programming. , 2017, , .		9
112	Temporal search. Operating Systems Review (ACM), 2006, 40, 25-36.	1.9	8
113	Efficient subcubic alias analysis for C. ACM SIGPLAN Notices, 2014, 49, 829-845.	0.2	8
114	Finding missed optimizations through the lens of dead code elimination. , 2022, , .		8
115	Calling-to-reference context translation via constraint-guided CFL-reachability. , 2018, , .		7
116	Effective floating-point analysis via weak-distance minimization. , 2019, , .		7
117	Achieving high coverage for floating-point code via unconstrained programming. ACM SIGPLAN Notices, 2017, 52, 306-319.	0.2	7
118	Modeling High-Level Behavior Patterns for Precise Similarity Analysis of Software. , 2011, , .		6
119	Liberating the programmer with prorogued programming. , 2012, , .		6
120	Server interface descriptions for automated testing of JavaScript web applications. , 2013, , .		6
121	Capturing and Exploiting IDE Interactions. , 2014, , .		6
122	Global Optimization of Numerical Programs Via Prioritized Stochastic Algebraic Transformations. , 2019, , .		6
123	Static Detection of Unsafe Component Loadings. Lecture Notes in Computer Science, 2012, , 122-143.	1.3	6
124	Detecting races in Relay Ladder Logic programs. International Journal on Software Tools for Technology Transfer, 2000, 3, 93-105.	1.9	5
125	Profile-guided program simplification for effective testing and analysis. , 2008, , .		5
126	Automatic Detection of Unsafe Dynamic Component Loadings. IEEE Transactions on Software Engineering, 2012, 38, 293-313.	5.6	5

#	Article	IF	Citations
127	Temporal search. ACM SIGPLAN Notices, 2006, 41, 25-36.	0.2	5
128	Student Adoption and Perceptions of a Web Integrated Development Environment. , 2020, , .		5
129	Feature Omission Vulnerabilities: Thwarting Signature Generation for Polymorphic Worms. , 2007, , .		4
130	Temporal search. Computer Architecture News, 2006, 34, 25-36.	2.5	3
131	Static Validation of C Preprocessor Macros. , 2009, , .		3
132	BQL. , 2011, , .		3
133	Automated coverage-driven testing: combining symbolic execution and model checking. Science China Information Sciences, 2016, 59, 1.	4.3	3
134	Guided, Deep Testing of X.509 Certificate Validation via Coverage Transfer Graphs. , 2020, , .		2
135	Automated backward error analysis for numerical code. ACM SIGPLAN Notices, 2015, 50, 639-654.	0.2	2
136	Fast linear programming through transprecision computing on small and sparse data. , 2020, 4, 1-28.		2
137	Putting Trojans on the Horns of a Dilemma: Redundancy for Information Theft Detection. Lecture Notes in Computer Science, 2009, , 244-262.	1.3	2
138	Detecting and analyzing insecure component usage. , 2012, , .		1
139	Reusing debugging knowledge via trace-based bug search. ACM SIGPLAN Notices, 2012, 47, 927-942.	0.2	1
140	Building white-box abstractions by program refinement. , 2016, , .		0
141	Entailment with Conditional Equality Constraints. Lecture Notes in Computer Science, 2001, , 170-189.	1.3	0
142	Coverage-Directed Differential Testing of X.509 Certificate Validation in SSL/TLS Implementations. ACM Transactions on Software Engineering and Methodology, 0, , .	6.0	0