Grigore Rosu

List of Publications by Citations

89 papers	2,789 citations	28 h-index	51 g-index
90 ext. papers	3,109 ext. citations	0.8 avg, IF	5.49 L-index

#	Paper	IF	Citations
89	An overview of the K semantic framework. *The Journal of Logic and Algebraic Programming*, **2010**, 79, 397-434		227
88	Synthesizing Monitors for Safety Properties. *Lecture Notes in Computer Science*, **2002**, 342-356	0.9	168
87	KEVM: A Complete Formal Semantics of the Ethereum Virtual Machine **2018**,		143
86	An overview of the MOP runtime verification framework. *International Journal on Software Tools for Technology Transfer*, **2012**, 14, 249-289	1.3	142
85	Monitoring Java Programs with Java PathExplorer. *Electronic Notes in Theoretical Computer Science*, **2001**, 55, 200-217	0.7	127
84	Rewriting-Based Techniques for Runtime Verification. *Automated Software Engineering*, **2005**, 12, 151-197	1.5	126
83	An Overview of the Runtime Verification Tool Java PathExplorer. *Formal Methods in System Design*, **2004**, 24, 189-215	1.4	125
82	Institution Morphisms. *Formal Aspects of Computing*, **2002**, 13, 274-307	1.2	118
81	Efficient monitoring of safety properties. *International Journal on Software Tools for Technology Transfer*, **2004**, 6, 158-173	1.3	106
80	Java-MOP: A Monitoring Oriented Programming Environment for Java. *Lecture Notes in Computer Science*, **2005**, 546-550	0.9	88
79	The rewriting logic semantics project. *Theoretical Computer Science*, **2007**, 373, 213-237	1.1	87
78	Efficient Monitoring of Languages. *Lecture Notes in Computer Science*, **2005**, 364-378	0.9	63
77	Mop. *ACM SIGPLAN Notices*, **2007**, 42, 569-588	0.2	58
76	Defining the undefinedness of C **2015**,		54
75	Towards Monitoring-Oriented Programming: A Paradigm Combining Specification and Implementation. *Electronic Notes in Theoretical Computer Science*, **2003**, 89, 108-127	0.7	53
74	KJS: a complete formal semantics of JavaScript **2015**,		51
73	K-Java **2015**,		50

72	A rewriting logic approach to operational semantics. *Information and Computation*, **2009**, 207, 305-340	0.8	46
71	Hardware Runtime Monitoring for Dependable COTS-Based Real-Time Embedded Systems **2008**,		46
70	Generating Optimal Monitors for Extended Regular Expressions. *Electronic Notes in Theoretical Computer Science*, **2003**, 89, 226-245	0.7	41
69	RV-Monitor: Efficient Parametric Runtime Verification with Simultaneous Properties. *Lecture Notes in Computer Science*, **2014**, 285-300	0.9	40
68	Mining parametric specifications **2011**,		39
67	JavaMOP: Efficient parametric runtime monitoring framework **2012**,		39
66	Detecting Errors in Multithreaded Programs by Generalized Predictive Analysis of Executions. *Lecture Notes in Computer Science*, **2005**, 211-226	0.9	39
65	Semantics-based program verifiers for all languages **2016**,		38
64	An executable formal semantics of C with applications. *ACM SIGPLAN Notices*, **2012**, 47, 533-544	0.2	33
63	All-Path Reachability Logic. *Lecture Notes in Computer Science*, **2014**, 425-440	0.9	29
62	Checking and Correcting Behaviors of Java Programs at Runtime with Java-MOP. *Electronic Notes in Theoretical Computer Science*, **2006**, 144, 3-20	0.7	28
61	One-Path Reachability Logic **2013**,		27
60	Matching logic **2011**,		25
59	Hiding more of hidden algebra. *Lecture Notes in Computer Science*, **1999**, 1704-1719	0.9	25
58	Matching Logic: An Alternative to Hoare/Floyd Logic. *Lecture Notes in Computer Science*, **2011**, 142-162	0.9	24
57	Weak inclusion systems. *Mathematical Structures in Computer Science*, **1997**, 7, 195-206	0.5	22
56	The rewriting logic semantics project: A progress report. *Information and Computation*, **2013**, 231, 38-69	0.8	21
55	Checking reachability using matching logic **2012**,		20

54	Experiments with Test Case Generation and Runtime Analysis. *Lecture Notes in Computer Science*, **2003**, 87-108	0.9	20
53	How good are the specs? a study of the bug-finding effectiveness of existing Java API specifications **2016**,		20
52	K-Maude: A Rewriting Based Tool for Semantics of Programming Languages. *Lecture Notes in Computer Science*, **2010**, 104-122	0.9	19
51	Efficient Monitoring of Parametric Context-Free Patterns **2008**,		18
50	Online Efficient Predictive Safety Analysis of Multithreaded Programs. *Lecture Notes in Computer Science*, **2004**, 123-138	0.9	18
49	A complete formal semantics of x86-64 user-level instruction set architecture **2019**,		17
48	Defining the undefinedness of C. *ACM SIGPLAN Notices*, **2015**, 50, 336-345	0.2	17
47	Testing Extended Regular Language Membership Incrementally by Rewriting. *Lecture Notes in Computer Science*, **2003**, 499-514	0.9	15
46	Composing Hidden Information Modules over Inclusive Institutions. *Lecture Notes in Computer Science*, **2004**, 96-123	0.9	14
45	KJS: a complete formal semantics of JavaScript. *ACM SIGPLAN Notices*, **2015**, 50, 346-356	0.2	12
44	Semantics-based program verifiers for all languages. *ACM SIGPLAN Notices*, **2016**, 51, 74-91	0.2	12
43	Generating Optimal Linear Temporal Logic Monitors by Coinduction. *Lecture Notes in Computer Science*, **2003**, 260-275	0.9	12
42	Checking reachability using matching logic. *ACM SIGPLAN Notices*, **2012**, 47, 555-574	0.2	11
41	A Rewrite Framework for Language Definitions and for Generation of Efficient Interpreters. *Electronic Notes in Theoretical Computer Science*, **2007**, 176, 215-231	0.7	11
40	Towards a Unified Theory of Operational and Axiomatic Semantics. *Lecture Notes in Computer Science*, **2012**, 351-363	0.9	11
39	From Hoare Logic to Matching Logic Reachability. *Lecture Notes in Computer Science*, **2012**, 387-402	0.9	11
38	(mathbb{K}) Framework Distilled. *Lecture Notes in Computer Science*, **2012**, 31-53	0.9	10
37	K Overview and SIMPLE Case Study. *Electronic Notes in Theoretical Computer Science*, **2014**, 304, 3-56	0.7	9

36	Axiomatizability in inclusive equational logics. *Mathematical Structures in Computer Science*, **2002**, 12, 541-563	0.5	9
35	The Rewriting Logic Semantics Project: A Progress Report. *Lecture Notes in Computer Science*, **2011**, 1-37	0.9	9
34	From Rewriting Logic, to Programming Language Semantics, to Program Verification. *Lecture Notes in Computer Science*, **2015**, 598-616	0.9	8
33	Online efficient predictive safety analysis of multithreaded programs. *International Journal on Software Tools for Technology Transfer*, **2006**, 8, 248-260	1.3	8
32	Behavioral abstraction is hiding information. *Theoretical Computer Science*, **2004**, 327, 197-221	1.1	8
31	RV-Match: Practical Semantics-Based Program Analysis. *Lecture Notes in Computer Science*, **2016**, 447-453	0.9	8
30	Language definitions as rewrite theories. *Journal of Logical and Algebraic Methods in Programming*, **2016**, 85, 98-120	1	7
29	GPredict: Generic Predictive Concurrency Analysis **2015**,		7
28	Garbage collection for monitoring parametric properties. *ACM SIGPLAN Notices*, **2011**, 46, 415-424	0.2	7
27	IELE: A Rigorously Designed Language and Tool Ecosystem for the Blockchain. *Lecture Notes in Computer Science*, **2019**, 593-610	0.9	7
26	Runtime Verification - 17 Years Later. *Lecture Notes in Computer Science*, **2018**, 3-17	0.9	7
25	Term-generic logic. *Theoretical Computer Science*, **2015**, 577, 1-24	1.1	6
24	The . *Electronic Notes in Theoretical Computer Science*, **2014**, 304, 57-80	0.7	6
23	Efficient parametric runtime verification with deterministic string rewriting **2013**,		6
22	Executing Formal Semantics with the (mathbb K) Tool. *Lecture Notes in Computer Science*, **2012**, 267-271	0.9	6
21	Evolution-Aware Monitoring-Oriented Programming **2015**,		5
20	A general approach to define binders using matching logic **2020**, 4, 1-32		5
19	Towards a Trustworthy Semantics-Based Language Framework via Proof Generation. *Lecture Notes in Computer Science*, **2021**, 477-499	0.9	5

18	A Language-Independent Approach to Smart Contract Verification. *Lecture Notes in Computer Science*, **2018**, 405-413	0.9	5
17	Language Definitions as Rewrite Theories. *Lecture Notes in Computer Science*, **2014**, 97-112	0.9	4
16	Finite-Trace Linear Temporal Logic: Coinductive Completeness. *Lecture Notes in Computer Science*, **2016**, 333-350	0.9	4
15	Program Verification by Coinduction. *Lecture Notes in Computer Science*, **2018**, 589-618	0.9	4
14	Finite-trace linear temporal logic: coinductive completeness. *Formal Methods in System Design*, **2018**, 53, 138-163	1.4	4
13	**2019**,		3
12	An instrumentation technique for online analysis of multithreaded programs. *Concurrency Computation Practice and Experience*, **2007**, 19, 311-325	1.4	3
11	Certifying and Synthesizing Membership Equational Proofs. *Lecture Notes in Computer Science*, **2003**, 359-380	0.9	3
10	A Truly Concurrent Semantics for the (mathbb{K}) Framework Based on Graph Transformations. *Lecture Notes in Computer Science*, **2012**, 294-310	0.9	3
9	Introduction to the special issue on runtime verification. *Formal Methods in System Design*, **2012**, 41, 233-235	1.4	2
8	(mathbb {K})A Semantic Framework for Programming Languages and Formal Analysis. *Lecture Notes in Computer Science*, **2020**, 122-158	0.9	2
7	A Theoretical Foundation for Programming Languages Aggregation. *Lecture Notes in Computer Science*, **2015**, 30-47	0.9	2
6	How effective are existing Java API specifications for finding bugs during runtime verification?. *Automated Software Engineering*, **2019**, 26, 795-837	1.5	1
5	Towards Behavioral Maude: Behavioral Membership Equational Logic. *Electronic Notes in Theoretical Computer Science*, **2002**, 65, 197-253	0.7	0
4	Abstract Semantics for . *Electronic Notes in Theoretical Computer Science*, **2014**, 304, 127-149	0.7	
3	The (mathbb {K}) Vision for the Future of Programming Language Design and Analysis. *Lecture Notes in Computer Science*, **2021**, 3-9	0.9	
2	Towards a (mathbb {K})ool Future. *Lecture Notes in Computer Science*, **2016**, 325-343	0.9	
1	Making Maude Definitions More Interactive. *Lecture Notes in Computer Science*, **2012**, 83-98	0.9	