Naoki Kobayashi

List of Publications by Year
in descending order

Source: https://exaly.com/author-pdf/2651681/publications.pdf

Version: 2024-02-01

149	2,929	304368	301761
papers	citations	22 h-index	39 g-index

156	156	156	564
all docs	docs citations	times ranked	citing authors

#	Article	IF	Citations
1	Counterexample generation for program verification based on ownership refinement types. , 2021, , .		0
2	Symbolic Automatic Relations and Their Applications to SMT and CHC Solving. Lecture Notes in Computer Science, 2021, , 405-428.	1.0	2
3	Termination Analysis for the $$pi $$-Calculus by Reduction to Sequential Program Termination. Lecture Notes in Computer Science, 2021, , 265-284.	1.0	0
4	Toward Neural-Network-Guided Program Synthesis and Verification. Lecture Notes in Computer Science, 2021, , 236-260.	1.0	3
5	ICE-Based Refinement Type Discovery for Higher-Order Functional Programs. Journal of Automated Reasoning, 2020, 64, 1393-1418.	1.1	15
6	RustHorn: CHC-Based Verification for Rust Programs. Lecture Notes in Computer Science, 2020, , 484-514.	1.0	17
7	ConSORT: Context- and Flow-Sensitive Ownership Refinement Types for Imperative Programs. Lecture Notes in Computer Science, 2020, , 684-714.	1.0	11
8	Fold/Unfold Transformations for Fixpoint Logic. Lecture Notes in Computer Science, 2020, , 195-214.	1.0	6
9	A New Refinement Type System forÂAutomated $$u ext {HFL}_mathbb {Z}$$ Validity Checking. Lecture Notes in Computer Science, 2020, , 86-104.	1.0	5
10	Predicate Abstraction and CEGAR for $$u mathrm {HFL}_mathbb {Z}$$ Validity Checking. Lecture Notes in Computer Science, 2020, , 134-155.	1.0	5
11	Inclusion between the frontier language of a non-deterministic recursive program scheme and the Dyck language is undecidable. Theoretical Computer Science, 2019, 777, 409-416.	0.5	4
12	10 Years of the Higher-Order Model Checking Project (Extended Abstract). , 2019, , .		0
13	Reduction from branching-time property verification of higher-order programs to HFL validity checking. , 2019, , .		14
14	Combining higher-order model checking with refinement type inference. , 2019, , .		6
15	Temporal Verification of Programs via First-Order Fixpoint Logic. Lecture Notes in Computer Science, 2019, , 413-436.	1.0	16
16	A Temporal Logic for Higher-Order Functional Programs. Lecture Notes in Computer Science, 2019, , 437-458.	1.0	1
17	Holce: An ICE-Based Non-linear Horn Clause Solver. Lecture Notes in Computer Science, 2018, , 146-156.	1.0	19
18	Higher-Order Program Verification via HFL Model Checking. Lecture Notes in Computer Science, 2018, , 711-738.	1.0	20

#	Article	IF	Citations
19	ICE-Based Refinement Type Discovery for Higher-Order Functional Programs. Lecture Notes in Computer Science, 2018, , 365-384.	1.0	27
20	Verifying relational properties of functional programs by first-order refinement. Science of Computer Programming, 2017, 137, 2-62.	1.5	9
21	Deadlock analysis of unbounded process networks. Information and Computation, 2017, 252, 48-70.	0.5	17
22	Verification of code generators via higher-order model checking. , 2017, , .		0
23	Modular Verification of Higher-Order Functional Programs. Lecture Notes in Computer Science, 2017, , 831-854.	1.0	2
24	On the relationship between higher-order recursion schemes and higher-order fixpoint logic. , 2017, , .		13
25	Almost Every Simply Typed $$lambda $$-Term Has a Long $$eta $$-Reduction Sequence. Lecture Notes in Computer Science, 2017, , 53-68.	1.0	1
26	On the relationship between higher-order recursion schemes and higher-order fixpoint logic. ACM SIGPLAN Notices, 2017, 52, 246-259.	0.2	2
27	Higher-Order Model Checking in Direct Style. Lecture Notes in Computer Science, 2016, , 295-313.	1.0	3
28	Temporal verification of higher-order functional programs. , 2016, , .		22
29	Temporal verification of higher-order functional programs. ACM SIGPLAN Notices, 2016, 51, 57-68.	0.2	5
30	Compact bit encoding schemes for simply-typed lambda-terms. , 2016, , .		1
31	Automatically disproving fair termination of higher-order functional programs. , 2016, , .		4
32	Verification of Higher-Order Concurrent Programs with Dynamic Resource Creation. Lecture Notes in Computer Science, 2016, , 335-353.	1.0	1
33	Equivalence-Based Abstraction Refinement for $$mu $$ HORS Model Checking. Lecture Notes in Computer Science, 2016, , 304-320.	1.0	1
34	Compact bit encoding schemes for simply-typed lambda-terms. ACM SIGPLAN Notices, 2016, 51, 146-157.	0.2	0
35	Automatically disproving fair termination of higher-order functional programs. ACM SIGPLAN Notices, 2016, 51, 243-255.	0.2	0
36	Verification of tree-processing programs via higher-order mode checking. Mathematical Structures in Computer Science, 2015, 25, 841-866.	0.5	0

#	Article	IF	Citations
37	Automata-Based Abstraction Refinement for µHORS Model Checking. , 2015, , .		3
38	Refinement Type Checking via Assertion Checking. Journal of Information Processing, 2015, 23, 827-834.	0.3	1
39	Verifying Relational Properties of Functional Programs by First-Order Refinement. , 2015, , .		9
40	Predicate Abstraction and CEGAR for Disproving Termination of Higher-Order Functional Programs. Lecture Notes in Computer Science, 2015, , 287-303.	1.0	14
41	Automata-Based Abstraction for Automated Verification of Higher-Order Tree-Processing Programs. Lecture Notes in Computer Science, 2015, , 295-312.	1.0	5
42	Decision Algorithms for Checking Definability of Order-2 Finitary PCF. Lecture Notes in Computer Science, 2015, , 313-331.	1.0	0
43	From Linear Types to Behavioural Types and Model Checking. Lecture Notes in Computer Science, 2014, , 128-143.	1.0	0
44	Efficient Algorithm and Coding for Higher-Order Compression. , 2014, , .		1
45	A ZDD-Based Efficient Higher-Order Model Checking Algorithm. Lecture Notes in Computer Science, 2014, , 354-371.	1.0	5
46	Unsafe Order-2 Tree Languages Are Context-Sensitive. Lecture Notes in Computer Science, 2014, , 149-163.	1.0	5
47	Complexity of Model-Checking Call-by-Value Programs. Lecture Notes in Computer Science, 2014, , 180-194.	1.0	3
48	Automatic Termination Verification for Higher-Order Functional Programs. Lecture Notes in Computer Science, 2014, , 392-411.	1.0	30
49	Pairwise Reachability Analysis for Higher Order Concurrent Programs by Higher-Order Model Checking. Lecture Notes in Computer Science, 2014, , 312-326.	1.0	5
50	Deadlock Analysis of Unbounded Process Networks. Lecture Notes in Computer Science, 2014, , 63-77.	1.0	15
51	Model Checking Higher-Order Programs. Journal of the ACM, 2013, 60, 1-62.	1.8	54
52	Towards a scalable software model checker for higher-order programs. , 2013, , .		31
53	Automating relatively complete verification of higher-order functional programs. , 2013, , .		32
54	Pumping by Typing. , 2013, , .		9

#	Article	IF	Citations
55	Practical Alternating Parity Tree Automata Model Checking of Higher-Order Recursion Schemes. Lecture Notes in Computer Science, 2013, , 17-32.	1.0	11
56	Model-Checking Higher-Order Programs with Recursive Types. Lecture Notes in Computer Science, 2013, , 431-450.	1.0	17
57	Automating relatively complete verification of higher-order functional programs. ACM SIGPLAN Notices, 2013, 48, 75-86.	0.2	11
58	Functional programs as compressed data. , 2012, , .		12
59	Functional programs as compressed data. Higher-Order and Symbolic Computation, 2012, 25, 39-84.	0.3	11
60	Exact Flow Analysis by Higher-Order Model Checking. Lecture Notes in Computer Science, 2012, , 275-289.	1.0	9
61	Program Certification by Higher-Order Model Checking. Lecture Notes in Computer Science, 2012, , 9-10.	1.0	0
62	Environmental bisimulations for higher-order languages. ACM Transactions on Programming Languages and Systems, 2011, 33, 1-69.	1.7	56
63	Ordered Types for Stream Processing of Tree-Structured Data. Journal of Information Processing, 2011, 19, 74-87.	0.3	0
64	Predicate abstraction and CEGAR for higher-order model checking. , 2011, , .		83
65	Higher-Order Model Checking: From Theory to Practice. , 2011, , .		7
66	Predicate abstraction and CEGAR for higher-order model checking. ACM SIGPLAN Notices, 2011, 46, 222-233.	0.2	39
67	A Practical Linear Time Algorithm for Trivial Automata Model Checking of Higher-Order Recursion Schemes. Lecture Notes in Computer Science, 2011, , 260-274.	1.0	24
68	Type-Based Automated Verification of Authenticity in Asymmetric Cryptographic Protocols. Lecture Notes in Computer Science, 2011, , 75-89.	1.0	2
69	Higher-order multi-parameter tree transducers and recursion schemes for program verification. , 2010, , .		42
70	A hybrid type system for lock-freedom of mobile processes. ACM Transactions on Programming Languages and Systems, 2010, 32, 1-49.	1.7	26
71	Untyped Recursion Schemes and Infinite Intersection Types. Lecture Notes in Computer Science, 2010, , 343-357.	1.0	13
72	Verification of Tree-Processing Programs via Higher-Order Model Checking. Lecture Notes in Computer Science, 2010, , 312-327.	1.0	7

#	Article	IF	Citations
73	Higher-order multi-parameter tree transducers and recursion schemes for program verification. ACM SIGPLAN Notices, 2010, 45, 495-508.	0.2	18
74	Types and higher-order recursion schemes for verification of higher-order programs. , 2009, , .		73
75	Model-checking higher-order functions. , 2009, , .		52
76	Types and higher-order recursion schemes for verification of higher-order programs. ACM SIGPLAN Notices, 2009, 44, 416-428.	0.2	44
77	Undecidable equivalences for basic parallel processes. Information and Computation, 2009, 207, 812-829.	0.5	4
78	Dependent type inference with interpolants. , 2009, , .		54
79	A Type System Equivalent to the Modal Mu-Calculus Model Checking of Higher-Order Recursion Schemes. , 2009, , .		109
80	Fractional Ownerships for Safe Memory Deallocation. Lecture Notes in Computer Science, 2009, , 128-143.	1.0	13
81	Type-Based Automated Verification of Authenticity in Cryptographic Protocols. Lecture Notes in Computer Science, 2009, , 222-236.	1.0	2
82	Higher-Order Program Verification and Language-Based Security. Lecture Notes in Computer Science, 2009, , 17-23.	1.0	2
83	Complexity of Model Checking Recursion Schemes for Fragments of the Modal Mu-Calculus. Lecture Notes in Computer Science, 2009, , 223-234.	1.0	14
84	Types and Recursion Schemes for Higher-Order Program Verification. Lecture Notes in Computer Science, 2009, , 2-3.	1.0	4
85	A New Type System for JVM Lock Primitives. New Generation Computing, 2008, 26, 125-170.	2.5	1
86	A Coq Library for Verification of Concurrent Programs. Electronic Notes in Theoretical Computer Science, 2008, 199, 17-32.	0.9	8
87	Translation of tree-processing programs into stream-processing programs based on ordered linear type. Journal of Functional Programming, 2008, 18, 333-371.	0.5	2
88	A Hybrid Type System for Lock-Freedom of Mobile Processes. Lecture Notes in Computer Science, 2008, , 80-93.	1.0	6
89	On-Demand Refinement of Dependent Types. , 2008, , 81-96.		5
90	Tree Automata for Non-linear Arithmetic. Lecture Notes in Computer Science, 2008, , 291-305.	1.0	1

#	Article	IF	Citations
91	Linear Declassification. , 2008, , 224-238.		1
92	Substructural Type Systems for Program Analysis. , 2008, , 14-14.		1
93	Environmental Bisimulations for Higher-Order Languages. , 2007, , .		52
94	Type-Based Analysis of Deadlock for a Concurrent Calculus with Interrupts. Lecture Notes in Computer Science, 2007, , 490-504.	1.0	6
95	Undecidability of 2-Label BPP Equivalences and Behavioral Type Systems for the Ï€-Calculus. Lecture Notes in Computer Science, 2007, , 740-751.	1.0	5
96	On the Complexity of Termination Inference for Processes. , 2007, , 140-155.		9
97	Logical Bisimulations and Functional Languages. , 2007, , 364-379.		3
98	Type-Based Verification of Correspondence Assertions for Communication Protocols. , 2007, , 191-205.		7
99	Combining type-based analysis and model checking for finding counterexamples against non-interference. , 2006, , .		14
100	A New Type System for Deadlock-Free Processes. Lecture Notes in Computer Science, 2006, , 233-247.	1.0	101
101	Resource usage analysis for a functional language with exceptions. , 2006, , .		11
102	Resource Usage Analysis for the Ï€-Calculus. Logical Methods in Computer Science, 2006, 2, .	0.4	21
103	Extension of Type-Based Approach to Generation of Stream-Processing Programs by Automatic Insertion of Buffering Primitives. Lecture Notes in Computer Science, 2006, , 98-114.	1.0	2
104	Verification of Concurrent Programs Using the Coq Proof Assistant: A Case Study. IPSJ Digital Courier, 2005, 1, 117-127.	0.3	3
105	Type-based information flow analysis for the Ï€-calculus. Acta Informatica, 2005, 42, 291-347.	0.5	81
106	Partial Order Reduction for Verification of Spatial Properties of Pi-Calculus Processes. Electronic Notes in Theoretical Computer Science, 2005, 128, 151-168.	0.9	3
107	Resource usage analysis. ACM Transactions on Programming Languages and Systems, 2005, 27, 264-313.	1.7	53
108	Resource Usage Analysis for the Ï€-Calculus. Lecture Notes in Computer Science, 2005, , 298-312.	1.0	6

#	Article	IF	Citations
109	A generic type system for the Pi-calculus. Theoretical Computer Science, 2004, 311, 121-163.	0.5	93
110	Translation of Tree-Processing Programs into Stream-Processing Programs Based on Ordered Linear Type. Lecture Notes in Computer Science, 2004, , 41-56.	1.0	4
111	Type Systems for Concurrent Programs. Lecture Notes in Computer Science, 2003, , 439-453.	1.0	46
112	AnZenMail: A Secure and Certified E-mail System. Lecture Notes in Computer Science, 2003, , 201-216.	1.0	4
113	Formalization and Verification of a Mail Server in Coq. Lecture Notes in Computer Science, 2003, , 217-233.	1.0	7
114	Useless-Code Elimination and Program Slicing for the Pi-Calculus. Lecture Notes in Computer Science, 2003, , 55-72.	1.0	2
115	Time regions and effects for resource usage analysis. , 2003, , .		5
116	Time regions and effects for resource usage analysis. ACM SIGPLAN Notices, 2003, 38, 50-61.	0.2	4
117	Resource usage analysis. , 2002, , .		72
118	Resource usage analysis. ACM SIGPLAN Notices, 2002, 37, 331-342.	0.2	8
119	A Type System for Lock-Free Processes. Information and Computation, 2002, 177, 122-159.	0.5	73
120	A Type System for Lock-Free Processes. Information and Computation, 2002, 177, 122-159.	0.5	52
121	A new type system for JVM lock primitives. , 2002, , .		7
122	Type-Based Useless-Variable Elimination. Higher-Order and Symbolic Computation, 2001, 14, 221-260.	0.3	5
123	A Hybrid Approach to Online and Offline Partial Evaluation. Higher-Order and Symbolic Computation, 2001, 14, 101-142.	0.3	26
124	A generic type system for the Pi-calculus. , 2001, , .		60
125	A generic type system for the Pi-calculus. ACM SIGPLAN Notices, 2001, 36, 128-141.	0.2	10
126	Type Reconstruction for Linear Ï€-Calculus with I/O Subtyping. Information and Computation, 2000, 161, 1-44.	0.5	32

#	Article	IF	Citations
127	An Implicitly-Typed Deadlock-Free Process Calculus. Lecture Notes in Computer Science, 2000, , 489-504.	1.0	25
128	Type Systems for Concurrent Processes: From Deadlock-Freedom to Livelock-Freedom, Time-Boundedness. Lecture Notes in Computer Science, 2000, , 365-389.	1.0	14
129	Quasi-linear types. , 1999, , .		61
130	Online-and-offline partial evaluation (extended abstract). , 1999, , .		3
131	Type-based useless variable elimination. , 1999, , .		17
132	Linearity and the pi-calculus. ACM Transactions on Programming Languages and Systems, 1999, 21, 914-947.	1.7	167
133	Online-and-offline partial evaluation (extended abstract). ACM SIGPLAN Notices, 1999, 34, 12-21.	0.2	3
134	Distributed concurrent linear logic programming. Theoretical Computer Science, 1999, 227, 185-220.	0.5	18
135	Distributed and Concurrent Objects Based on Linear Logic. , 1999, , 399-400.		0
136	A Generalized Deadlock-Free Process Calculus. Electronic Notes in Theoretical Computer Science, 1998, 16, 225-247.	0.9	24
137	A partially deadlock-free typed process calculus. ACM Transactions on Programming Languages and Systems, 1998, 20, 436-482.	1.7	79
138	Type-based analysis of communication for concurrent programming languages. Lecture Notes in Computer Science, 1997, , 187-201.	1.0	14
139	Linearity and the pi-calculus. , 1996, , .		103
140	Partial evaluation scheme for concurrent languages and its correctness. Lecture Notes in Computer Science, 1996, , 625-632.	1.0	14
141	Asynchronous communication model based on linear logic. Formal Aspects of Computing, 1995, 7, 113-149.	1.4	25
142	Toward Foundations of Concurrent Objectâ€Oriented Programmingâ€Types and Language Design. Theory and Practice of Object Systems, 1995, 1, 243-268.	0.8	16
143	Higher-order concurrent linear logic programming. Lecture Notes in Computer Science, 1995, , 137-166.	1.0	13
144	Static analysis of communication for asynchronous concurrent programming languages. Lecture Notes in Computer Science, 1995, , 225-242.	1.0	24

#	Article	IF	Citations
145	Type-theoretic foundations for concurrent object-oriented programing. ACM SIGPLAN Notices, 1994, 29, 31-45.	0.2	118
146	Type-theoretic foundations for concurrent object-oriented programing. , 1994, , .		18
147	Asynchronous communication model based on linear logic. Lecture Notes in Computer Science, 1993, , 331-336.	1.0	1
148	An Overview of the HFL Model Checking Project. Electronic Proceedings in Theoretical Computer Science, EPTCS, 0, 344, 1-12.	0.8	1
149	Complexity of Model Checking Recursion Schemes for Fragments of the Modal Mu-Calculus. Logical Methods in Computer Science, 0, Volume 7, Issue 4, .	0.4	17