

Naoki Kobayashi

List of Publications by Year in descending order

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

Version: 2024-02-01

149
papers

2,929
citations

304743

22
h-index

302126

39
g-index

156
all docs

156
docs citations

156
times ranked

564
citing authors

#	ARTICLE	IF	CITATIONS
1	Linearity and the pi-calculus. ACM Transactions on Programming Languages and Systems, 1999, 21, 914-947.	2.1	167
2	Type-theoretic foundations for concurrent object-oriented programming. ACM SIGPLAN Notices, 1994, 29, 31-45.	0.2	118
3	A Type System Equivalent to the Modal Mu-Calculus Model Checking of Higher-Order Recursion Schemes. , 2009, , .		109
4	Linearity and the pi-calculus. , 1996, , .		103
5	A New Type System for Deadlock-Free Processes. Lecture Notes in Computer Science, 2006, , 233-247.	1.3	101
6	A generic type system for the Pi-calculus. Theoretical Computer Science, 2004, 311, 121-163.	0.9	93
7	Predicate abstraction and CEGAR for higher-order model checking. , 2011, , .		83
8	Type-based information flow analysis for the λ -calculus. Acta Informatica, 2005, 42, 291-347.	0.5	81
9	A partially deadlock-free typed process calculus. ACM Transactions on Programming Languages and Systems, 1998, 20, 436-482.	2.1	79
10	A Type System for Lock-Free Processes. Information and Computation, 2002, 177, 122-159.	0.7	73
11	Types and higher-order recursion schemes for verification of higher-order programs. , 2009, , .		73
12	Resource usage analysis. , 2002, , .		72
13	Quasi-linear types. , 1999, , .		61
14	A generic type system for the Pi-calculus. , 2001, , .		60
15	Environmental bisimulations for higher-order languages. ACM Transactions on Programming Languages and Systems, 2011, 33, 1-69.	2.1	56
16	Dependent type inference with interpolants. , 2009, , .		54
17	Model Checking Higher-Order Programs. Journal of the ACM, 2013, 60, 1-62.	2.2	54
18	Resource usage analysis. ACM Transactions on Programming Languages and Systems, 2005, 27, 264-313.	2.1	53

#	ARTICLE	IF	CITATIONS
19	A Type System for Lock-Free Processes. Information and Computation, 2002, 177, 122-159.	0.7	52
20	Environmental Bisimulations for Higher-Order Languages. , 2007, , .		52
21	Model-checking higher-order functions. , 2009, , .		52
22	Type Systems for Concurrent Programs. Lecture Notes in Computer Science, 2003, , 439-453.	1.3	46
23	Types and higher-order recursion schemes for verification of higher-order programs. ACM SIGPLAN Notices, 2009, 44, 416-428.	0.2	44
24	Higher-order multi-parameter tree transducers and recursion schemes for program verification. , 2010, , .		42
25	Predicate abstraction and CEGAR for higher-order model checking. ACM SIGPLAN Notices, 2011, 46, 222-233.	0.2	39
26	Type Reconstruction for Linear λ -Calculus with I/O Subtyping. Information and Computation, 2000, 161, 1-44.	0.7	32
27	Automating relatively complete verification of higher-order functional programs. , 2013, , .		32
28	Towards a scalable software model checker for higher-order programs. , 2013, , .		31
29	Automatic Termination Verification for Higher-Order Functional Programs. Lecture Notes in Computer Science, 2014, , 392-411.	1.3	30
30	ICE-Based Refinement Type Discovery for Higher-Order Functional Programs. Lecture Notes in Computer Science, 2018, , 365-384.	1.3	27
31	A Hybrid Approach to Online and Offline Partial Evaluation. Higher-Order and Symbolic Computation, 2001, 14, 101-142.	0.3	26
32	A hybrid type system for lock-freedom of mobile processes. ACM Transactions on Programming Languages and Systems, 2010, 32, 1-49.	2.1	26
33	Asynchronous communication model based on linear logic. Formal Aspects of Computing, 1995, 7, 113-149.	1.8	25
34	An Implicitly-Typed Deadlock-Free Process Calculus. Lecture Notes in Computer Science, 2000, , 489-504.	1.3	25
35	A Generalized Deadlock-Free Process Calculus. Electronic Notes in Theoretical Computer Science, 1998, 16, 225-247.	0.9	24
36	Static analysis of communication for asynchronous concurrent programming languages. Lecture Notes in Computer Science, 1995, , 225-242.	1.3	24

#	ARTICLE	IF	CITATIONS
37	A Practical Linear Time Algorithm for Trivial Automata Model Checking of Higher-Order Recursion Schemes. Lecture Notes in Computer Science, 2011, , 260-274.	1.3	24
38	Temporal verification of higher-order functional programs. , 2016, , .		22
39	Resource Usage Analysis for the λ -Calculus. Logical Methods in Computer Science, 2006, 2, .	0.4	21
40	Higher-Order Program Verification via HFL Model Checking. Lecture Notes in Computer Science, 2018, , 711-738.	1.3	20
41	Holce: An ICE-Based Non-linear Horn Clause Solver. Lecture Notes in Computer Science, 2018, , 146-156.	1.3	19
42	Type-theoretic foundations for concurrent object-oriented programing. , 1994, , .		18
43	Distributed concurrent linear logic programming. Theoretical Computer Science, 1999, 227, 185-220.	0.9	18
44	Higher-order multi-parameter tree transducers and recursion schemes for program verification. ACM SIGPLAN Notices, 2010, 45, 495-508.	0.2	18
45	Type-based useless variable elimination. , 1999, , .		17
46	Deadlock analysis of unbounded process networks. Information and Computation, 2017, 252, 48-70.	0.7	17
47	RustHorn: CHC-Based Verification for Rust Programs. Lecture Notes in Computer Science, 2020, , 484-514.	1.3	17
48	Model-Checking Higher-Order Programs with Recursive Types. Lecture Notes in Computer Science, 2013, , 431-450.	1.3	17
49	Complexity of Model Checking Recursion Schemes for Fragments of the Modal μ -Calculus. Logical Methods in Computer Science, 0, Volume 7, Issue 4, .	0.4	17
50	Toward Foundations of Concurrent Object-Oriented Programming Types and Language Design. Theory and Practice of Object Systems, 1995, 1, 243-268.	0.7	16
51	Temporal Verification of Programs via First-Order Fixpoint Logic. Lecture Notes in Computer Science, 2019, , 413-436.	1.3	16
52	ICE-Based Refinement Type Discovery for Higher-Order Functional Programs. Journal of Automated Reasoning, 2020, 64, 1393-1418.	1.4	15
53	Deadlock Analysis of Unbounded Process Networks. Lecture Notes in Computer Science, 2014, , 63-77.	1.3	15
54	Type-based analysis of communication for concurrent programming languages. Lecture Notes in Computer Science, 1997, , 187-201.	1.3	14

#	ARTICLE	IF	CITATIONS
55	Combining type-based analysis and model checking for finding counterexamples against non-interference. , 2006, , .		14
56	Reduction from branching-time property verification of higher-order programs to HFL validity checking. , 2019, , .		14
57	Type Systems for Concurrent Processes: From Deadlock-Freedom to Livelock-Freedom, Time-Boundedness. Lecture Notes in Computer Science, 2000, , 365-389.	1.3	14
58	Partial evaluation scheme for concurrent languages and its correctness. Lecture Notes in Computer Science, 1996, , 625-632.	1.3	14
59	Predicate Abstraction and CEGAR for Disproving Termination of Higher-Order Functional Programs. Lecture Notes in Computer Science, 2015, , 287-303.	1.3	14
60	Complexity of Model Checking Recursion Schemes for Fragments of the Modal Mu-Calculus. Lecture Notes in Computer Science, 2009, , 223-234.	1.3	14
61	Higher-order concurrent linear logic programming. Lecture Notes in Computer Science, 1995, , 137-166.	1.3	13
62	On the relationship between higher-order recursion schemes and higher-order fixpoint logic. , 2017, , .		13
63	Fractional Ownerships for Safe Memory Deallocation. Lecture Notes in Computer Science, 2009, , 128-143.	1.3	13
64	Untyped Recursion Schemes and Infinite Intersection Types. Lecture Notes in Computer Science, 2010, , 343-357.	1.3	13
65	Functional programs as compressed data. , 2012, , .		12
66	Functional programs as compressed data. Higher-Order and Symbolic Computation, 2012, 25, 39-84.	0.3	11
67	ConSORT: Context- and Flow-Sensitive Ownership Refinement Types for Imperative Programs. Lecture Notes in Computer Science, 2020, , 684-714.	1.3	11
68	Practical Alternating Parity Tree Automata Model Checking of Higher-Order Recursion Schemes. Lecture Notes in Computer Science, 2013, , 17-32.	1.3	11
69	Resource usage analysis for a functional language with exceptions. , 2006, , .		11
70	Automating relatively complete verification of higher-order functional programs. ACM SIGPLAN Notices, 2013, 48, 75-86.	0.2	11
71	A generic type system for the Pi-calculus. ACM SIGPLAN Notices, 2001, 36, 128-141.	0.2	10
72	Pumping by Typing. , 2013, , .		9

#	ARTICLE	IF	CITATIONS
73	Verifying Relational Properties of Functional Programs by First-Order Refinement. , 2015, , .		9
74	Verifying relational properties of functional programs by first-order refinement. Science of Computer Programming, 2017, 137, 2-62.	1.9	9
75	On the Complexity of Termination Inference for Processes. , 2007, , 140-155.		9
76	Exact Flow Analysis by Higher-Order Model Checking. Lecture Notes in Computer Science, 2012, , 275-289.	1.3	9
77	Resource usage analysis. ACM SIGPLAN Notices, 2002, 37, 331-342.	0.2	8
78	A Coq Library for Verification of Concurrent Programs. Electronic Notes in Theoretical Computer Science, 2008, 199, 17-32.	0.9	8
79	Higher-Order Model Checking: From Theory to Practice. , 2011, , .		7
80	Formalization and Verification of a Mail Server in Coq. Lecture Notes in Computer Science, 2003, , 217-233.	1.3	7
81	Verification of Tree-Processing Programs via Higher-Order Model Checking. Lecture Notes in Computer Science, 2010, , 312-327.	1.3	7
82	A new type system for JVM lock primitives. , 2002, , .		7
83	Type-Based Verification of Correspondence Assertions for Communication Protocols. , 2007, , 191-205.		7
84	Combining higher-order model checking with refinement type inference. , 2019, , .		6
85	Resource Usage Analysis for the λ -Calculus. Lecture Notes in Computer Science, 2005, , 298-312.	1.3	6
86	Fold/Unfold Transformations for Fixpoint Logic. Lecture Notes in Computer Science, 2020, , 195-214.	1.3	6
87	A Hybrid Type System for Lock-Freedom of Mobile Processes. Lecture Notes in Computer Science, 2008, , 80-93.	1.3	6
88	Type-Based Analysis of Deadlock for a Concurrent Calculus with Interrupts. Lecture Notes in Computer Science, 2007, , 490-504.	1.3	6
89	Type-Based Useless-Variable Elimination. Higher-Order and Symbolic Computation, 2001, 14, 221-260.	0.3	5
90	A New Refinement Type System for Automated $\text{HFL}_{\text{ext}}\{\mathbb{Z}\}$ Validity Checking. Lecture Notes in Computer Science, 2020, , 86-104.	1.3	5

#	ARTICLE	IF	CITATIONS
91	Predicate Abstraction and CEGAR for $\mathbb{HFL}_{\mathbb{Z}}$ Validity Checking. Lecture Notes in Computer Science, 2020, , 134-155.	1.3	5
92	A ZDD-Based Efficient Higher-Order Model Checking Algorithm. Lecture Notes in Computer Science, 2014, , 354-371.	1.3	5
93	Automata-Based Abstraction for Automated Verification of Higher-Order Tree-Processing Programs. Lecture Notes in Computer Science, 2015, , 295-312.	1.3	5
94	Undecidability of 2-Label BPP Equivalences and Behavioral Type Systems for the λ -Calculus. Lecture Notes in Computer Science, 2007, , 740-751.	1.3	5
95	On-Demand Refinement of Dependent Types. , 2008, , 81-96.		5
96	Unsafe Order-2 Tree Languages Are Context-Sensitive. Lecture Notes in Computer Science, 2014, , 149-163.	1.3	5
97	Pairwise Reachability Analysis for Higher Order Concurrent Programs by Higher-Order Model Checking. Lecture Notes in Computer Science, 2014, , 312-326.	1.3	5
98	Temporal verification of higher-order functional programs. ACM SIGPLAN Notices, 2016, 51, 57-68.	0.2	5
99	Time regions and effects for resource usage analysis. , 2003, , .		5
100	Undecidable equivalences for basic parallel processes. Information and Computation, 2009, 207, 812-829.	0.7	4
101	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.	0.9	4
102	AnZenMail: A Secure and Certified E-mail System. Lecture Notes in Computer Science, 2003, , 201-216.	1.3	4
103	Automatically disproving fair termination of higher-order functional programs. , 2016, , .		4
104	Time regions and effects for resource usage analysis. ACM SIGPLAN Notices, 2003, 38, 50-61.	0.2	4
105	Translation of Tree-Processing Programs into Stream-Processing Programs Based on Ordered Linear Type. Lecture Notes in Computer Science, 2004, , 41-56.	1.3	4
106	Types and Recursion Schemes for Higher-Order Program Verification. Lecture Notes in Computer Science, 2009, , 2-3.	1.3	4
107	Online-and-offline partial evaluation (extended abstract). , 1999, , .		3
108	Online-and-offline partial evaluation (extended abstract). ACM SIGPLAN Notices, 1999, 34, 12-21.	0.2	3

#	ARTICLE	IF	CITATIONS
109	Verification of Concurrent Programs Using the Coq Proof Assistant: A Case Study. IPSJ Digital Courier, 2005, 1, 117-127.	0.3	3
110	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
111	Automata-Based Abstraction Refinement for π -HORS Model Checking. , 2015, , .		3
112	Higher-Order Model Checking in Direct Style. Lecture Notes in Computer Science, 2016, , 295-313.	1.3	3
113	Complexity of Model-Checking Call-by-Value Programs. Lecture Notes in Computer Science, 2014, , 180-194.	1.3	3
114	Toward Neural-Network-Guided Program Synthesis and Verification. Lecture Notes in Computer Science, 2021, , 236-260.	1.3	3
115	Logical Bisimulations and Functional Languages. , 2007, , 364-379.		3
116	Translation of tree-processing programs into stream-processing programs based on ordered linear type. Journal of Functional Programming, 2008, 18, 333-371.	0.8	2
117	Modular Verification of Higher-Order Functional Programs. Lecture Notes in Computer Science, 2017, , 831-854.	1.3	2
118	Useless-Code Elimination and Program Slicing for the Pi-Calculus. Lecture Notes in Computer Science, 2003, , 55-72.	1.3	2
119	Symbolic Automatic Relations and Their Applications to SMT and CHC Solving. Lecture Notes in Computer Science, 2021, , 405-428.	1.3	2
120	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.	1.3	2
121	Type-Based Automated Verification of Authenticity in Cryptographic Protocols. Lecture Notes in Computer Science, 2009, , 222-236.	1.3	2
122	Higher-Order Program Verification and Language-Based Security. Lecture Notes in Computer Science, 2009, , 17-23.	1.3	2
123	Type-Based Automated Verification of Authenticity in Asymmetric Cryptographic Protocols. Lecture Notes in Computer Science, 2011, , 75-89.	1.3	2
124	On the relationship between higher-order recursion schemes and higher-order fixpoint logic. ACM SIGPLAN Notices, 2017, 52, 246-259.	0.2	2
125	Asynchronous communication model based on linear logic. Lecture Notes in Computer Science, 1993, , 331-336.	1.3	1
126	A New Type System for JVM Lock Primitives. New Generation Computing, 2008, 26, 125-170.	3.3	1

#	ARTICLE	IF	CITATIONS
127	Efficient Algorithm and Coding for Higher-Order Compression. , 2014, , .		1
128	Refinement Type Checking via Assertion Checking. Journal of Information Processing, 2015, 23, 827-834.	0.4	1
129	An Overview of the HFL Model Checking Project. Electronic Proceedings in Theoretical Computer Science, EPTCS, 0, 344, 1-12.	0.8	1
130	A Temporal Logic for Higher-Order Functional Programs. Lecture Notes in Computer Science, 2019, , 437-458.	1.3	1
131	Almost Every Simply Typed λ -Term Has a Long η -Reduction Sequence. Lecture Notes in Computer Science, 2017, , 53-68.	1.3	1
132	Compact bit encoding schemes for simply-typed lambda-terms. , 2016, , .		1
133	Verification of Higher-Order Concurrent Programs with Dynamic Resource Creation. Lecture Notes in Computer Science, 2016, , 335-353.	1.3	1
134	Equivalence-Based Abstraction Refinement for μ HORS Model Checking. Lecture Notes in Computer Science, 2016, , 304-320.	1.3	1
135	Tree Automata for Non-linear Arithmetic. Lecture Notes in Computer Science, 2008, , 291-305.	1.3	1
136	Linear Declassification. , 2008, , 224-238.		1
137	Substructural Type Systems for Program Analysis. , 2008, , 14-14.		1
138	Ordered Types for Stream Processing of Tree-Structured Data. Journal of Information Processing, 2011, 19, 74-87.	0.4	0
139	From Linear Types to Behavioural Types and Model Checking. Lecture Notes in Computer Science, 2014, , 128-143.	1.3	0
140	Verification of tree-processing programs via higher-order mode checking. Mathematical Structures in Computer Science, 2015, 25, 841-866.	0.6	0
141	Verification of code generators via higher-order model checking. , 2017, , .		0
142	10 Years of the Higher-Order Model Checking Project (Extended Abstract). , 2019, , .		0
143	Counterexample generation for program verification based on ownership refinement types. , 2021, , .		0
144	Termination Analysis for the π -Calculus by Reduction to Sequential Program Termination. Lecture Notes in Computer Science, 2021, , 265-284.	1.3	0

#	ARTICLE	IF	CITATIONS
145	Program Certification by Higher-Order Model Checking. Lecture Notes in Computer Science, 2012, , 9-10.	1.3	0
146	Distributed and Concurrent Objects Based on Linear Logic. , 1999, , 399-400.		0
147	Decision Algorithms for Checking Definability of Order-2 Finitary PCF. Lecture Notes in Computer Science, 2015, , 313-331.	1.3	0
148	Compact bit encoding schemes for simply-typed lambda-terms. ACM SIGPLAN Notices, 2016, 51, 146-157.	0.2	0
149	Automatically disproving fair termination of higher-order functional programs. ACM SIGPLAN Notices, 2016, 51, 243-255.	0.2	0