10 Years of the Higher-Order Model Checking Project (Extended Abstract)

Naoki Kobayashi
koba@is.s.u-tokyo.ac.jp
The University of Tokyo
Tokyo, Japan

ABSTRACT

We give an overview of the higher-order model checking project at the University of Tokyo. We provide references to the results obtained in the past 10 years, and explain what the project is now heading for.

CCS CONCEPTS

• **Theory of computation → Logic and verification**.

KEYWORDS

higher-order model checking, program verification

ACM Reference Format:

Naoki Kobayashi. 2019. 10 Years of the Higher-Order Model Checking Project (Extended Abstract). In *Principles and Practice of Programming Languages 2019 (PPDP '19), October 7–9, 2019, Porto, Portugal.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3354166.3354167

We summarize the higher-order model checking project at the University of Tokyo.[1] *Higher-order model checking* refers to two kinds of higher-order extensions of ordinary finite-state model checking [13]. One is *HORS model checking*, which is obtained by extending models (i.e., systems to be verified) to higher-order ones called *higher-order recursion schemes* (HORS) [15, 33]. It aims to check whether the (possibly infinite) tree generated by a given HORS satisfies a given tree property. The other higher-order model checking problem is *HFL model checking* [41], which is obtained by replacing the logic for specifying properties with higher-order modal fixpoint logic (HFL). HFL is a higher-order extension of the modal μ-calculus, and HFL model checking aims to check whether a given finite state system satisfies the property described by a given HFL formula.

Our project started about 10 years ago, following two key papers presented at POPL and PPDP in 2009 [16, 17] (see also [19], a revised version of the two papers). In the POPL paper [17], we have shown that program verification problems for higher-order functional programs can naturally be reduced to HORS model checking. In the PPDP paper [16], we have proposed the first practical HORS model checking algorithm, and constructed a HORS model checker TRecS, which runs reasonably fast for many inputs despite the k-EXPTIME completeness of the HORS model checking problem (where k is the highest type-theoretic order of the functions used in HORS). In prior to our work, Ong [33] has proved the decidability of HORS model checking in 2006. The algorithm described in his proof could not be used in practice, as it always suffers from the k-EXPTIME bottleneck. The combination of the results above [16, 17] suggested that a fully automated verification tool for functional programs can be constructed on top a HORS model checker, which led us to launch the project.

The project consisted of both theoretical and practical studies. On the practical side, we have constructed a fully automated verification tool MoCHi for a subset of OCaml [4, 25, 27, 28, 30, 36, 42]. MoCHi is based on a combination of the two results above with a technique of predicate abstraction inspired by earlier studies on software model checkers for C language [6, 7]. The development of MoCHi called for more efficient HORS model checking algorithms, which led us to develop various HORS model checking algorithms and tools [12, 18, 37, 39, 40]. Our state-of-the-art model checker HorSat2 [21] scales to HORS consisting of thousands of rewriting rules (though, of course, depending on the kinds of inputs). We have also studied an application of HORS model checking to data compression, where HORS is used as a compressed from of tree data [23, 38]. By using HORS model checking algorithms (and their extensions), one can manipulate the compressed data without decompression; this can be considered a higher-order extension of grammar-based data compression [35].

On the theoretical side, we have studied type-based characterizations of HORS model checking [17, 24]. All the HORS model checkers developed so far [10, 18, 29, 32, 34, 37] are based on the type-based characterizations, with the only exception of [11], which is based on collapsible pushdown automata. As a foundation for HORS model checking, we have also studied properties of higher-order grammars, such as pumping lemmas [1, 2, 20]. One of our current theoretical interests on HORS model checking is in finding theoretical justifications for why HORS model checking works in practice, despite the extremely high worst-case complexity. To this end, we have launched a sub-project to study the *average-case* complexity [9] of HORS model checking. As a first step in the project, we have studied the average-case length of β-reduction sequences of simply-typed λ-terms [3].

Since 2017, we have gradually been shifting our focus to the other notion of higher-order model checking: HFL model checking [5, 41]. HFL model checking has been introduced by Viswanathan and

[1]In this abstract, we focus on the results of our group in Tokyo. A number of other research groups have contributed to the field of higher-order model checking, not to mention the groups in Bordeaux, Oxford, Paris, and Warsaw.

PPDP '19, October 7–9, 2019, Porto, Portugal

ACM ISBN 978-1-4503-7249-7/19/10…$15.00
https://doi.org/10.1145/3354166.3354167

Viswanathan [41] in 2004, but somehow it has been drawing less attentions than HORS model checking. We have shown that there are mutual translations between HORS and HFL model checking [22], and that various program verification problems can be reduced to HFL model checking, even more naturally than to HORS model checking [26, 43]. We are now working to rebuild the whole verification infrastructure of MoCHi based on HFL model checking, as (i) it provides a more uniform approach to verification of infinite-data programs, and (ii) it naturally extends other popular approaches to automated program verification, such as CHC-based program verification [8, 14]. The first result in such direction is found in [31], albeit for first-order programs.

ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number JP23220001 and JP15H05706. The author thanks collaborators on this project, especially, Kazuyuki Asada, Atsushi Igarashi, Luke Ong, Ryosuke Sato, Ayumi Shinohara, Takeshi Tsukada, and Hiroshi Unno.

REFERENCES

[1] Kazuyuki Asada and Naoki Kobayashi. 2017. Pumping Lemma for Higher-order Languages. In *Proceedings of ICALP 2017 (LIPIcs)*, Vol. 80. 97:1–97:14.

[2] Kazuyuki Asada and Naoki Kobayashi. 2018. Lambda-Definable Order-3 Tree Functions are Well-Quasi-Ordered. In *Proceedings of FSTTCS 2018 (LIPIcs)*, Vol. 122. 14:1–14:15.

[3] Kazuyuki Asada, Naoki Kobayashi, Ryoma Sin'ya, and Takeshi Tsukada. 2019. Almost Every Simply Typed Lambda-Term Has a Long Beta-Reduction Sequence. *Logical Methods in Computer Science* Volume 15, Issue 1 (Feb. 2019).

[4] Kazuyuki Asada, Ryosuke Sato, and Naoki Kobayashi. 2017. Verifying relational properties of functional programs by first-order refinement. *Sci. Comput. Program.* 137 (2017), 2–62.

[5] Roland Axelsson, Martin Lange, and Rafal Somla. 2007. The Complexity of Model Checking Higher-Order Fixpoint Logic. *Logical Methods in Computer Science* 3, 2 (2007).

[6] Thomas Ball and Sriram K. Rajamani. 2002. The SLAM Project: Debugging System Software via Static Analysis. In *Proceedings of POPL*. ACM, 1–3.

[7] Dirk Beyer, Thomas A. Henzinger, Ranjit Jhala, and Rupak Majumdar. 2007. The software model checker Blast. *International Journal on Software Tools for Technology Transfer* 9, 5-6 (2007), 505–525.

[8] Nikolaj Bjørner, Arie Gurfinkel, Kenneth L. McMillan, and Andrey Rybalchenko. 2015. Horn Clause Solvers for Program Verification. In *Fields of Logic and Computation II - Essays Dedicated to Yuri Gurevich on the Occasion of His 75th Birthday (Lecture Notes in Computer Science)*, Vol. 9300. Springer, 24–51.

[9] Andrej Bogdanov and Luca Trevisan. 2006. Average-case Complexity. *Found. Trends Theor. Comput. Sci.* 2, 1 (Oct. 2006), 1–106.

[10] Christopher H. Broadbent, Arnaud Carayol, Matthew Hague, and Olivier Serre. 2012. A Saturation Method for Collapsible Pushdown Systems. In *Proceedings of ICALP 2012 (LNCS)*, Vol. 7392. Springer, 165–176.

[11] Christopher H. Broadbent, Arnaud Carayol, Matthew Hague, and Olivier Serre. 2013. C-SHORe: a collapsible approach to higher-order verification. In *Proceedings of ICFP'13*. ACM, 13–24.

[12] Christopher H. Broadbent and Naoki Kobayashi. 2013. Saturation-Based Model Checking of Higher-Order Recursion Schemes. In *Proceedings of CSL 2013 (LIPIcs)*, Vol. 23. 129–148.

[13] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. 1999. *Model Checking*. The MIT Press.

[14] Sergey Grebenshchikov, Nuno P. Lopes, Corneliu Popeea, and Andrey Rybalchenko. 2012. Synthesizing software verifiers from proof rules. In *Proceedings of PLDI '12*. 405–416.

[15] Teodor Knapik, Damian Niwinski, and Pawel Urzyczyn. 2002. Higher-Order Pushdown Trees Are Easy. In *FoSSaCS 2002 (LNCS)*, Vol. 2303. Springer, 205–222.

[16] Naoki Kobayashi. 2009. Model-Checking Higher-Order Functions. In *Proceedings of PPDP 2009*. ACM, 25–36.

[17] Naoki Kobayashi. 2009. Types and Higher-Order Recursion Schemes for Verification of Higher-Order Programs. In *Proceedings of POPL 2009*. ACM, 416–428.

[18] Naoki Kobayashi. 2011. A Practical Linear Time Algorithm for Trivial Automata Model Checking of Higher-Order Recursion Schemes. In *Proceedings of FoSSaCS 2011 (LNCS)*, Vol. 6604. Springer, 260–274.

[19] Naoki Kobayashi. 2013. Model Checking Higher-Order Programs. *J. ACM* 60, 3 (2013).

[20] Naoki Kobayashi. 2013. Pumping by Typing. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, LA, USA, June 25-28, 2013*. IEEE Computer Society, 398–407.

[21] Naoki Kobayashi. 2018. HorSat2: A Saturation-Based Higher-Order Model Checker for HORS. https://github.com/hopv/horsat2.

[22] Naoki Kobayashi, Étienne Lozes, and Florian Bruse. 2017. On the relationship between higher-order recursion schemes and higher-order fixpoint logic. In *Proceedings of POPL 2017*. 246–259.

[23] Naoki Kobayashi, Kazutaka Matsuda, Ayumi Shinohara, and Kazuya Yaguchi. 2013. Functional Programs as Compressed Data. *Higher-Order and Symbolic Computation* (2013).

[24] Naoki Kobayashi and C.-H. Luke Ong. 2009. A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In *Proceedings of LICS 2009*. IEEE Computer Society Press, 179–188.

[25] Naoki Kobayashi, Ryosuke Sato, and Hiroshi Unno. 2011. Predicate Abstraction and CEGAR for Higher-Order Model Checking. ACM, 222–233.

[26] Naoki Kobayashi, Takeshi Tsukada, and Keiichi Watanabe. 2018. Higher-Order Program Verification via HFL Model Checking. In *Proceedings of ESOP 2018 (Lecture Notes in Computer Science)*, Vol. 10801. Springer, 711–738.

[27] Takuya Kuwahara, Ryosuke Sato, Hiroshi Unno, and Naoki Kobayashi. 2015. Predicate Abstraction and CEGAR for Disproving Termination of Higher-Order Functional Programs. In *Proceedings of CAV 2015 (Lecture Notes in Computer Science)*, Vol. 9207. Springer, 287–303.

[28] Takuya Kuwahara, Tachio Terauchi, Hiroshi Unno, and Naoki Kobayashi. 2014. Automatic Termination Verification for Higher-Order Functional Programs. In *Proceedings of ESOP 2014 (Lecture Notes in Computer Science)*, Vol. 8410. Springer, 392–411.

[29] M. M. Lester, R. P. Neatherway, C.-H. Luke Ong, and S. J. Ramsay. 2011. Model checking liveness properties of higher-order functional programs. In *Proceedings of ML Workshop 2011*.

[30] Akihiro Murase, Tachio Terauchi, Naoki Kobayashi, Ryosuke Sato, and Hiroshi Unno. 2016. Temporal Verification of Higher-Order Functional Programs. In *Proceedings of POPL 2016*. 57–68.

[31] Atsushi Igarashi Naoki Kobayashi, Takeshi Nishikawa and Hiroshi Unno. 2019. Temporal Verification of Programs via First-Order Fixpoint Logic. In *Proceedings of SAS 2019*.

[32] Robin P. Neatherway, Steven James Ramsay, and C.-H. Luke Ong. 2012. A traversal-based algorithm for higher-order model checking. In *Proceedings of ICFP '12*. 353–364.

[33] C.-H. Luke Ong. 2006. On Model-Checking Trees Generated by Higher-Order Recursion Schemes. In *LICS 2006*. IEEE Computer Society Press, 81–90.

[34] Steven Ramsay, Robin Neatherway, and C.-H. Luke Ong. 2014. An Abstraction Refinement Approach to Higher-Order Model Checking. In *Proceedings of POPL 2014*. ACM, 61–72.

[35] Wojciech Rytter. 2004. Grammar Compression, LZ-Encodings, and String Algorithms with Implicit Input. In *ICALP'04 (LNCS)*, Vol. 3142. Springer, 15–27.

[36] Ryosuke Sato, Hiroshi Unno, and Naoki Kobayashi. 2013. Towards a scalable software model checker for higher-order programs. In *Proceedings of PEPM 2013*. ACM, 53–62.

[37] Ryota Suzuki, Koichi Fujima, Naoki Kobayashi, and Takeshi Tsukada. 2017. Streett Automata Model Checking of Higher-Order Recursion Schemes. In *Proceedings of FSCD 2017 (LIPIcs)*, Vol. 84. 32:1–32:18.

[38] Kotaro Takeda, Naoki Kobayashi, Kazuya Yaguchi, and Ayumi Shinohara. 2016. Compact bit encoding schemes for simply-typed lambda-terms. In *Proceedings of ICFP 2016*. 146–157.

[39] Taku Terao and Naoki Kobayashi. 2014. A ZDD-Based Efficient Higher-Order Model Checking Algorithm. In *Proceedings of APLAS 2014 (Lecture Notes in Computer Science)*, Vol. 8858. Springer, 354–371.

[40] Taku Terao, Takeshi Tsukada, and Naoki Kobayashi. 2016. Higher-Order Model Checking in Direct Style. In *Proceedings of APLAS 2016 (Lecture Notes in Computer Science)*, Vol. 10017. 295–313.

[41] M. Viswanathan and R. Viswanathan. 2004. A Higher Order Modal Fixed Point Logic. In *CONCUR (Lecture Notes in Computer Science)*, Vol. 3170. Springer, 512–528.

[42] Keiichi Watanabe, Ryosuke Sato, Takeshi Tsukada, and Naoki Kobayashi. 2016. Automatically disproving fair termination of higher-order functional programs. In *Proceedings of ICFP 2016*. ACM, 243–255.

[43] Keiichi Watanabe, Takeshi Tsukada, Hiroki Oshikawa, and Naoki Kobayashi. 2019. Reduction from branching-time property verification of higher-order programs to HFL validity checking. In *Proceedings of PEPM 2019*. 22–34.