# Accelerating Object-Sensitive Pointer Analysis by Exploiting Object Containment and Reachability (Artifact)

**Dongjie He**
University of New South Wales, Sydney, Australia

**Jingbo Lu**
University of New South Wales, Sydney, Australia

**Yaoqing Gao**
Huawei, Toronto, Canada

**Jingling Xue**
University of New South Wales, Sydney, Australia

## Abstract

Object-sensitive pointer analysis for an object-oriented program can be accelerated if context-sensitivity can be selectively applied to some precision-critical variables/objects in the program. Existing pre-analyses, which are performed to make such selections, either preserve precision but achieve limited speedups by reasoning about all the possible value flows in the program conservatively or achieve greater speedups but sacrifice precision (often unduly) by examining only some but not all the value flows in the program heuristically. In this paper, we introduce a new approach, named Turner, that represents a sweet spot between the two existing ones, as it is designed to enable object-sensitive pointer analysis to run significantly faster than the former approach and achieve significantly better precision than the latter approach. Turner is simple, lightweight yet effective due to two novel aspects in its design. First, we exploit a key observation that some precision-uncritical objects can be approximated based on the object-containment relationship pre-established (by applying Andersen's analysis). This approximation introduces a small degree yet the only source of imprecision into Turner. Second, leveraging this initial approximation, we introduce a simple DFA to reason about object reachability for a method intra-procedurally from its entry to its exit along all the possible value flows established by its statements to finalize its precision-critical variables/objects identified. We have validated Turner with an implementation in Soot against the state of the art using a set of 12 popular Java benchmarks and applications.

**2012 ACM Subject Classification** Theory of Computation → Program Analysis

**Keywords and phrases** Object-Sensitive Pointer Analysis, CFL Reachability, Object Containment

**Digital Object Identifier** 10.4230/DARTS.7.2.12

**Acknowledgements** We thank the reviewers for their constructive comments. This work is supported by an ARC DP grant DP180104069 and a UNSW-Huawei research grant (YBN2019105002).

**Related Article** Dongjie He, Jingbo Lu, Yaoqing Gao, and Jingling Xue, "Accelerating Object-Sensitive Pointer Analysis by Exploiting Object Containment and Reachability", in 35th European Conference on Object-Oriented Programming (ECOOP 2021), LIPIcs, Vol. 194, pp. 16:1–16:31, 2021.
https://doi.org/10.4230/LIPIcs.ECOOP.2021.16

**Related Conference** 35th European Conference on Object-Oriented Programming (ECOOP 2021), July 12–16, 2021, Aarhus, Denmark (Virtual Conference)

## 1 Scope

This artifact contains a Java pointer analysis framework which is built on top of a context-insensitive Anderson's pointer analysis, Spark [2], and its object-sensitive version developed by ourselves. We have implemented Turner, Eagle [4] and integrated Zipper's lattest version (`b83b038`[1]) in this framework.

The artifact could be used to reproduce all figures, tables and raw data used in our paper. It supports the following claims of the paper: (1) Turner can accelerate $k$`OBJ` significantly with only negligible precision loss. (2) Turner-guided $k$`OBJ` is significantly faster than Eagle-guided $k$`OBJ`. (3) Turner-guided $k$`OBJ` is significantly precise than Zipper-guided $k$`OBJ`. (4) Turner, as a pre-analysis, is significantly faster than Eagle and Zipper [3] as it runs linearly in terms of the number of statements in a program.

## 2 Content

The artifact package includes:

- a Docker image, which contains
  - an executable jar file with Turner, Eagle and Zipper packaged,
  - benchmarks (including 9 benchmarks from DaCaPo2006 [1] and 3 Java applications),
  - a Java library (i.e., `JRE1.6.0_45`),
  - the scripts for running all experiments and extracting results,
  - the PDF of the artifact manual, and
  - the PDF of the paper (i.e., *Accelerating Object-Sensitive Pointer Analysis by Exploiting Object Containment and Reachability*).
- the PDF of the artifact manual, and
- a license file.

## 3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). In addition, the artifact is also available at: `https://hub.docker.com/r/hdjay2013/turner_artifact`.

## 4 Tested platforms

We have carried out all the experiments on an Intel(R) Xeon(R) CPU E5-2637 3.5GHz machine with 512GB of RAM. The underline operating system is Ubuntu 20.04. The time budget used for running each object-sensitive pointer analysis on a program is set as 24 hours.

## 5 License

The artifact is available under license GPL v3.

## 6 MD5 sum of the artifact

edbb0b5d6eeefa4a17bdf94d1a83ea99

[1] `https://github.com/silverbullettt/zipper`

## 7 Size of the artifact

0.61 GiB

## References


1 Stephen M. Blackburn, Robin Garner, Chris Hoffmann, Asjad M. Khang, Kathryn S. McKinley, Rotem Bentzur, Amer Diwan, Daniel Feinberg, Daniel Frampton, Samuel Z. Guyer, Martin Hirzel, Antony Hosking, Maria Jump, Han Lee, J. Eliot B. Moss, Aashish Phansalkar, Darko Stefanović, Thomas VanDrunen, Daniel von Dincklage, and Ben Wiedermann. The DaCapobenchmarks: Java benchmarking development and analysis. In *Proceedings of the 21st annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, pages 169–190, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1167515.1167488.

2 Ondřej Lhoták and Laurie Hendren. Scaling Java points-to analysis using spark. In *International Conference on Compiler Construction*, pages 153–169, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. doi:10.5555/1765931.1765948.

3 Yue Li, Tian Tan, Anders Møller, and Yannis Smaragdakis. Precision-guided context sensitivity for pointer analysis. *Proceedings of the ACM on Programming Languages*, 2(OOPSLA):1–29, 2018. doi:10.1145/3276511.

4 Jingbo Lu and Jingling Xue. Precision-preserving yet fast object-sensitive pointer analysis with partial context sensitivity. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–29, 2019. doi:10.1145/3360574.