References

[1] F. Agakov, E. Bonilla, J. Cavazos, B. Franke, G. Fursin, M. F. P. O'Boyle, J. Thomson, M. Toussaint, and C. K. I. Williams. Using machine learning to focus iterative optimization. In *CGO '06: Proceedings of the Symposium on Code Generation and Optimization*, pages 295–305, 2006.

[2] L. Almagor, K. D. Cooper, A. Grosul, T. J. Harvey, S. W. Reeves, D. Subramanian, L. Torczon, and T. Waterman. Finding effective compilation sequences. In *Proceedings of the 2004 Conference on Languages, Compilers, and Tools for Embedded Systems*, pages 231–239, 2004.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA, 1995.

[4] S. M. Blackburn, R. Garner, C. Hoffmann, A. M. Khang, K. S. McKinley, R. Bentzur, A. Diwan, D. Feinberg, D. Frampton, S. Z. Guyer, M. Hirzel, A. Hosking, M. Jump, H. Lee, J. E. B. Moss, B. Moss, A. Phansalkar, D. Stefanović, T. VanDrunen, D. von Dincklage, and B. Wiedermann. The DaCapo benchmarks: Java benchmarking development and analysis. In *Proceedings of the 21st annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, OOPSLA '06, pages 169–190, 2006.

[5] J. Cavazos and M. F. P. O'Boyle. Method-specific dynamic compilation using logistic regression. In *Proceedings of the conference on Object-oriented programming systems, languages, and applications*, pages 229–240, 2006.

[6] K. Chow and Y. Wu. Feedback-directed selection and characterization of compiler optimizatons. Proc. 2nd Workshop on Feedback Directed Optimization, 1999.

[7] K. D. Cooper, P. J. Schielke, and D. Subramanian. Optimizing for reduced code space using genetic algorithms. In *Proceedings of the ACM SIGPLAN 1999 workshop on Languages, compilers, and tools for embedded systems*, pages 1–9, 1999.

[8] G. Fursin, Y. Kashnikov, A. Memon, Z. Chamski, O. Temam, M. Namolaru, E. Yom-Tov, B. Mendelson, A. Zaks, E. Courtois, F. Bodin, P. Barnard, E. Ashton, E. Bonilla, J. Thomson, C. Williams, and M. OBoyle. Milepost gcc: Machine learning enabled self-tuning compiler. *International Journal of Parallel Programming*, 39:296–327, 2011.

[9] A. Georges, D. Buytaert, and L. Eeckhout. Statistically rigorous java performance evaluation. In *Proceedings of the conference on Object-oriented programming systems and applications*, OOPSLA '07, pages 57–76, 2007.

[10] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[11] M. Haneda, P. M. W. Knijnenburg, and H. A. G. Wijshoff. Optimizing general purpose compiler optimization. In *Proceedings of the 2nd conference on Computing frontiers*, CF '05, pages 180–188, New York, NY, USA, 2005. ACM. ISBN 1-59593-019-1.

[12] K. Hoste and L. Eeckhout. Cole: compiler optimization level exploration. In *Proceedings of the 6th annual IEEE/ACM international symposium on Code generation and optimization*, CGO '08, pages 165–174, New York, NY, USA, 2008.

[13] http://www.oracle.com/technetwork/java/javase/memorymanagement-whitepaper 150215.pdf. Memory Management in the Java HotSpot Virtual Machine, April 2006.

[14] K. Ishizaki, M. Kawahito, T. Yasue, M. Takeuchi, T. Ogasawara, T. Suganuma, T. Onodera, H. Komatsu, and T. Nakatani. Design, implementation, and evaluation of optimizations in a just-in-time compiler. In *Proceedings of the ACM 1999 conference on Java Grande*, JAVA '99, pages 119–128, 1999.

[15] K. Ishizaki, M. Takeuchi, K. Kawachiya, T. Suganuma, O. Gohda, T. Inagaki, A. Koseki, K. Ogata, M. Kawahito, T. Yasue, T. Ogasawara, T. Onodera, H. Komatsu, and T. Nakatani. Effectiveness of cross-platform optimizations for a java just-in-time compiler. In *Proceedings of the 18th annual ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications*, pages 187–204, 2003.

[16] P. A. Kulkarni. JIT compilation policy for modern machines. In *Proceedings of the ACM international conference on Object oriented programming systems languages and applications*, pages 773–788, 2011.

[17] P. A. Kulkarni, D. B. Whalley, and G. S. Tyson. Evaluating heuristic optimization phase order search algorithms. In *CGO '07: Proceedings of the International Symposium on Code Generation and Optimization*, pages 157–169, 2007.

[18] P. A. Kulkarni, M. R. Jantz, and D. B. Whalley. Improving both the performance benefits and speed of optimization phase sequence searches. In *Proceedings of the ACM SIGPLAN/SIGBED 2010 conference on Languages, compilers, and tools for embedded systems*, LCTES '10, pages 95–104, 2010. ISBN 978-1-60558-953-4.

[19] S. Kulkarni and J. Cavazos. Mitigating the compiler optimization phase-ordering problem using machine learning. In *Proceedings of the ACM international conference on Object oriented programming systems languages and applications*, OOPSLA '12, pages 147–162. ACM, 2012.

[20] H. Lee, D. von Dincklage, A. Diwan, and J. E. B. Moss. Understanding the behavior of compiler optimizations. *Software Practice & Experience*, 36(8):835–844, July 2006.

[21] M. Paleczny, C. Vick, and C. Click. The Java hotspottm server compiler. In *Proceedings of the Symposium on JavaTM Virtual Machine Research and Technology Symposium*, pages 1–12, Berkeley, CA, USA, 2001. USENIX.

[22] Z. Pan and R. Eigenmann. PEAK: a fast and effective performance tuning system via compiler optimization orchestration. *ACM Trans. Program. Lang. Syst.*, 30:17:1–17:43, May 2008.

[23] R. Sanchez, J. Amaral, D. Szafron, M. Pirvu, and M. Stoodley. Using machines to learn method-specific compilation strategies. In *Code Generation and Optimization*, pages 257 –266, April 2011.

[24] SPEC98. Specjvm98 benchmarks. http://www.spec.org/jvm98/, 1998.

[25] S. Triantafyllis, M. Vachharajani, N. Vachharajani and D. I. August. Compiler optimization-space exploration. In *Proceedings of the International Symposium on Code Generation and Optimization*, pages 204–215. IEEE, 2003.