The Complexity of Constant Propagation

Markus Müller-Olm and Oliver Rüthing

Universität Dortmund, Baroper Str. 301, D-44221 Dortmund, Germany
{mmo,ruething}@ls5.cs.uni-dortmund.de
http://sunshine.cs.uni-dortmund.de/

Abstract. *Constant propagation* (*CP*) is one of the most widely used optimisations in practice (cf. [2]). Intuitively, it addresses the problem of statically detecting whether a variable evaluates to a constant at runtime. Unfortunately, as proved by Reif and Lewis [3], CP is in general undecidable even if the interpretation of branches is completely ignored. On the other hand, it is certainly decidable in more restricted settings, like on loop-free programs (cf. [1]). In this talk, we explore the complexity of CP for a three-dimensional taxonomy. The first dimension is given by the general nature of the problem which essentially falls into a *must-*, and a *may-* variant of CP where the latter one can be further distinguished into a *single-value* and a *multiple-value* variant. The second dimension distinguishes between arbitrary and loop-free programs. Finally, the third dimension captures the underlying signature, where we consider signatures without operators (*copy constants*), with only linear expressions of the form $x := a\,y + b$ (*linear constants*), with operators $+, -$, with operators $+, -, *$, and the standard signature.
We present some new results, in particular some exact complexity classifications, and discuss open problems, i.e., scenarios where only upper or lower bounds are known. As a general message we observe that detecting may-constants is significantly harder than detecting their must-counterparts.

References

1. J. Knoop and O. Rüthing. Constant propagation on the value graph: Simple constants and beyond. In *Proc. 9^{th} Conference on Compiler Construction* (*CC*), Lecture Notes in Computer Science 1781, pages 94 – 109, Berlin, Germany, 2000. Springer-Verlag.
2. S. S. Muchnick. *Advanced Compiler Design & Implementation*. Morgan Kaufmann, San Francisco, CA, 1997.
3. J. H. Reif and R. Lewis. Symbolic evaluation and the gobal value graph. In *Conf. Record of the 4^{th} ACM Symposium on the Principles of Programming Languages* (*POPL*), pages 104 – 118, Los Angeles, CA, 1977.