Optimizing Object-Oriented Programs

November 1, 2016

Handlíř Jaroslav, xhandl05@stud.fit.vutbr.cz
Švestka Jakub, xsvest05@stud.fit.vutbr.cz

Object-oriented programming languages bring programmers new opportunities for the development of programs that the existing non-object-oriented languages did not support. Encapsulation, class inheritance, polymorphism etc. allow to create programs or libraries that are very readable, easily customizable, and extensible.

Unfortunately this advantages increased demands on translation and reduced the performance of the resulting programs because of dynamic method lookup and a large number of sending messages between objects. It is acceptable cost for many applications, but for computationally sensitive programs such as mathematical or simulation libraries is any performance degradation a big problem.

Compilers use the same optimization techniques for object-oriented programming languages as for the optimization procedural programming languages, but during compilation of a code which uses class inheritance and dynamically-bound messages the compiler does not know which instance of the class (or subclass even override dynamic methods) a message is delivered.

Conventional optimization methods, such as e.g. line expansion, fail in these cases and optimization is not effective. Another problem is that the current computing machines can work more efficiently with previously known (constant) addresses than the addresses fetched from tables.

Thus arose new optimization techniques to support object-oriented languages. The most important of them is the replacement of dynamic method calls by a static call. This requires class hierarchy analysis in the code translation and use it to identify specific instances of classes which are the messages receivers. On this basis and other useful techniques, dynamic method call can be replaced by static method call and standard optimization techniques will deal with it.

References

[1] APPEL, Andrew W. a Jens. PALSBERG. *Modern compiler implementation in Java* [online]. 2nd ed. New York, NY, USA: Cambridge University Press, 2002 [cit. 2016-10-01]. ISBN 05-218-2060-X. Online: https://eden.dei.uc.pt/~amilcar/pdf/CompilerInJava.pdf

[2] DEAN, Jeffrey, David GROVE a Craig CHAMBERS, OLTHOFF, Walter. (ed.). *9th European Conference, Åarhus, Denmark, August 7-11, 1995 : proceedings: Optimization of Object-Oriented Programs Using Static Class Hierarchy Analysis* [online]. 1995-08-07. New York: Springer, c1995 [cit. 2016-10-09]. ISBN 35-406-0160-0. Online: http://web.cs.ucla.edu/~palsberg/tba/papers/dean-grove-chambers-ecoop95.pdf

[3] HÖLZLE, Urs a Ole AGESEN. Dynamic versus static optimization techniques for object-oriented languages. *Journal: Theory and Practice of Object Systems - Special issue: type systems* [online]. New York, NY, USA: John Wiley & Sons, Inc., 1995, 1995 (Volume 1 Issue 3), 167 - 188 [cit. 2016-10-17]. ISSN 1074-3227. Online: http://hoelzle.org/publications/tapos96.pdf