Implementation of a Graph Coloring Register Allocator for the Graal Compiler

Master/Bachelor thesis for ...
Matr.Nr.: ...
E-Mail: ...

Graal [1] is an effort to create a new just-in-time compiler for Java that is itself written in Java. It is based on a port of the HotSpot client compiler from C++ to Java.

Currently, Graal uses a Linear Scan Register Allocator (LSRA). Because LSRA only requires a single linear pass to allocate registers it is fast and therefore especially suited for just-in-time compilation. The downside of this approach is that the result is not optimal.

Register Allocation via Graph Coloring is considered to find a better solution but requires polynomial time. Thus, it is mainly used for static compilation. Nevertheless it is interesting to see how big the gap really is.

The goal of this thesis is to implement and test a Graph Coloring Register Allocator for Graal and to compare both, the compile-time overhead as well as the run-time improvement to our current Linear Scan Allocator.

The scope of this thesis is as follows:

- Test the implementation on a variety of non-trivial Java programs.
- Compare the implementation to the currently used Linear Scan allocation (compile time, run time).

Optional goals are:

- Apply the improvements proposed by Briggs [3] to the implementation.
- Identify the situations where LSRA is outperformed by graph coloring.

The work's progress should be discussed with the supervisor at least every 2 weeks. Please note the guidelines of the Institute for System Software when preparing the written thesis.

Supervisor: Dipl.-Ing. Josef Eisl, Dr. Roland Schatz