Register allocation

Michal Hornický, Jan Velecký
{xhorni14,xvelec07}@stud.fit.vutbr.cz

November 9, 2018

Abstract

Register allocation is an important part of the compilation process, where it is decided which variable resides which register. Due to high access latency of memory compared to the latency of registers, decent register allocation has also massive impact on the performance of generated code, especially for register architectures.

In intermediate representation, there is an unlimited amount of temporary variables, while the number of physical registers is limited. To bridge this gap, an algorithm for register allocation is used. The goal of the algorithm is to effectively allocate registers to store individual variables, for the duration of variables' use, and to decide which variables will be spilled – stored in a memory and loaded into a register only for a moment of an use. Finding out which variables are used at the point of the program or later on – i.e. are live – is called liveness analysis.

The problem of register allocation can be represented as an interference graph of nodes as variables and edges as simultaneously live connected variables. Most commonly used algorithms are based on the graph coloring problem, which is NP-complete. We usually give up searching for the optimal solution as a typical graph is not colorable anyway. Many heuristics exists for the problem, most clever of them use information from intermediate representation.