DeLICM – Undoing SSA Transformations for Polly

Michael Kruse

INRIA, France
michael.kruse@inria.fr

Abstract. The LLVM compiler and its intermediate representation are SSA-based. Most of its optimizations assume virtual registers and therefore the framework tries to convert as many load/store chains to virtual registers as possible. Unfortunately, this adds obstacles for polyhedral optimizers such as Polly. Virtual registers are 0-dimensional and as such often sequentialize otherwise parallel code. This is especially true for scalar optimizations that move code around such as Loop-Invariant Code Motion and Partial Reduction/Common Subexpression Elimination. Polly's current approach is to analyze the code before these scalar optimizations occur. We show that it is possible to undo the effects of these passes on the polyhedral level, which executing Polly later in LLVM pipeline where more SCoPs can be detected, without the additional scalar dependencies.

Keywords: Static Single Assignment, Polyhedral Compilation, Loop-Invariant Code Motion, Partial Redundancy Elimnation, LLVM, Polly