Shibo Xing

April 11th, 2020

CS2210

Dr. Xulong Tang

*Lazy Code Motion Review*

The paper aims to improve the code motion algorithm, which is developed in order to reduce the re-computation of the variables in program. The old code motion moves computation early even when there is no run-time gain from doing so. This phenomenon causes enormous register pressure. The new code motion proposed by this paper, which the authors name with the term ‘lazy code motion’, only computes variables at the latest time possible. With a series of measures they can guarantee that placements of auxiliary variable are all ‘safe placements’, of which the definition is given by them in the paper.

The paper adds two new predicates to the original code motion control flow graph, D-Safe and Earliest, which could guarantee no overheads would be generated by code motion itself and no new variables would be instantiated because of this motion. This measure is a part of the measure to keep the code motion ‘safe’, as described by the paper.

The paper then analyzes the placement of the auxiliary variable, which it says could be strictly restricted to the node entries. All nodes branched from parents have siblings will be preceded by a synthetic node. This uniform operation simplifies the subsequent analysis and is essential in assuring safe placements.

The paper presents more lemmas that explain why delayed computation is more economic than the algorithms presented by the previous study. This paper’s core conclusion is that uni-directional flow graph should place computations as early as necessary but as late as possible, which guarantees computational and lifetime optimality, respectively. The paragraphs are formatted nicely, since the two parts of the conclusion are fully analyzed and presented in two corresponding sections. I still have a hard time understanding most of its lemmas and comprehend perhaps only the key concepts. A lot more time is needed for this paper if I truely want to understand the conclusion.