Creating Coarse-Grained Parallelism

Tomáš Hanus, xhanus11@stud.fit.vutbr.cz
Jakub Tutko, xtutko00@stud.fit.vutbr.cz
November 7, 2016

Parallelization of the sequential code can be done by different approaches based on various granularity. Parallelism that has no minimum granularity is called fine-grained. It is useful in vector machines, such as VLIW and super-scalar processors. The key to achieve high performance also on multiple asynchronous processors with shared global memory is to use coarse-grained parallelism and its creating is also the main aim of this presentation. While the first mentioned is designed to be primarily used in inner-most loops, the coarse-grained works well with outer loops.

Parallelism can be explained as creating threads on multiple processors, executing some code in parallel for a period of time with occasional synchronization and at the end synchronizing with barriers. The main challenge while creating coarse-grained parallelism is to package it with granularity large enough to pay for the overhead of parallelism synchronization and initiation.

Mechanism to create a solution to this problem can be presented in three contexts. In single loops transformations are used to eliminate carried dependencies in loops. Perfect loop nests context combines techniques from single loops to develop a general algorithm for uncovering parallel loops. Imperfectly nested loops technique presents an idea of state when it is clear that each statement will do better in a different outer loop. Imperfect loops context considers code generation strategies in which maximal loop distribution is tried first and then multilevel loop fusion is used to increase granularity. Presentation will be focused on single loops transformations, including privatization, alignment, and replication.

References

[1] Allen, R., Kennedy, K. *: Optimizing Compilers for Modern Architectures*, Morgan Kaufmann, 2002.