Analysis of automatic program parallelization based on bytecode

There are many algorithms for automatic parallelization and this work explores the possible application of these algorithms to programs based on their bytecode or similar intermediate code. All these algorithms require the identification of independent code segments, because if two parts of code do not interfere with one another then they can be run in parallel without any danger of data corruption. Dependence testing is an extremely complicated problem and in general application, it is not algorithmically solvable. However, independences can be discovered in special cases and then they can be used as a basis for application of automatic parallelization, like the use of vector instructions. The first step is function inlining that allows the compiler to analyze the code more precisely, without unnecessary dependences caused by unknown functions. Next, it is necessary to identify all control flow constructs, like loops, and after that the compiler can attempt to locate dependences between the statements or instructions. Parallelization can be achieved only if the analysis discovered some independent parts in the code. This work is accompanied by an implementation of function inlining and code analysis for the .NET framework.