Algorithm Engineering Lab Assignment 5

Brian Zahoransky (brian.zahoransky@uni-jena.de)

February 22, 2021

1. What is CMake?

CMake is a script-language that allows creating build-files, for instance, make-scripts. CMake-scripts can be used cross-platform and cross-compiler. It provides functions for building (cmake), testing (ctest) and packing (cpack). The cmake-command creates the mentioned build-file. Ctest generates a test-program. Cpack can be used, for example, to build a .deb-package. The command-based language allows one command per line, all its variables are strings, and simple mathematical operations are available anyway. Furthermore, for-loops and if-clauses are usable as well.

2. What role do targets play in CMake?

Targets are the objects which will be built. For instance, they are executable files or libraries. In the lecture, they were compared with objects in object-orientated programming. Targets can be created by invoking the constructors "add_executable" or "add_library". The User can specify the properties of targets. Discussing properties in the lecture, they were associated with variables of an object in object-orientated programming.

3. How would you proceed to optimize code?

The in the lecture explained procedure seems reasonable. However, another option is defining an optimization circle instead of a program. Therefore, modifying a proven problem-solving strategy seems to be a plausible approach. The following describes how Polya's method could be adapted to the optimization process. **Polya's method** is defined by four steps:

First, understanding the problem. At the beginning of a project, it could mean researching. Had someone solved the problem before? Can the task be implemented efficiently? How complex is the problem? Later on, in advanced cycles, figuring out, what the bottlenecks are, will be one of the main tasks.

Second, elaborate a plan. Even implementing a naive approach for a task that seems to be trivial, it has turned out that planning is essential. In further cycles, based on a correct working program, the goal is to analyze which optimization techniques can be applied.

Third, carry out the plan. During the first circle, one could write a program-skeleton or a naive program. Later, one will implement the planned optimizations.

Fourth, evaluating the result. Relevant questions for the evaluation step are: Is my Code fast enough for my purposes? Am I able to/ Is it even possible to speed up my program? Was the speed increased as expected? Afterwards, the cycle can be executed again.