Final Report

Parallel Computing for Computational Mechanics

Johannes Grafen 380149
johannes.grafen@rwth-aachen.de

Abstract: Lorem ipsum dolor sit amet, consectetuer sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut lre et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

1 Introduction

The parallelisation of code for the simulation of complex mechanical systems is a key-topic in modern computational mechanics. Constantly increasing computing power of high-performance clusters allow for the simulation of large and complex systems. This requires an efficient parallelisation approach, where different parallelisation paradigms can be of use.

In this report the parallelisation of a finite element code for the simulation of the stationary temperature distribution on a two-dimensional disk is investigated. For the shared memory approach the OpenMP library is used, where as the distributed memory approach is assessed by using a one-sided parallelisation approach with MPI. The different parallelisation approaches are discussed and compared for different number of CPUs. This report is structured as follows. In the second chapter *Theory and Methods* are presented. The *Implementation and Validation* of the presented problem is discussed in chapter 3. The results of the optimisation of the runtime of the code is illustrated in chapter 4 *Results* and the assessment of the different optimisation techniques is discussed in chapter 5 *Discussion*. Finally, this report is closed with a *Conclusion* in chapter 6.

serial optimisation is performed by

2 Theory and Methods

3 Implementation and Validation

Parallelisation with OpenMP

Parallelisation with MPI

4 Results

5 Discussion

6 Conclusion

Lorem ipsum dolor sit amet, consectetuer sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut lre et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

References