PARALLEL NUMERICAL ANALYSIS OF THE SHIELD TUNNELLING PROCESS ON DISTRIBUTED MEMORY SYSTEMS

* Janosch Stascheit and Günter Meschke

Ruhr University Bochum, Institute for Structural Mechanics
Universitätsstr. 150, D-44780 Bochum, Germany
janosch.stascheit@rub.de, guenther.meschke@rub.de

Key Words: parallel computing, mechanised tunnelling, distributed memory, Finite Element Method.

Abstract. A numerical simulation software [1] is presented that allows for the analysis of the shield supported tunnel advance by means of the Finite Element Method. The modular Finite Element framework KRATOS [2] has been enhanced by elements, constitutive laws, and solving strategies that address the particular requirements of a process-oriented numerical analysis of mechanised tunnelling.

The simulation software has been parallelised for distributed memory systems by adapting the parallelisation capabilities of KRATOS to the specific problems associated with the simulation of shield tunnelling. The contribution discusses the implementation of a surface-to-surface contact algorithm and the de- and reactivation of elements and boundary conditions into the distributed memory version of KRATOS. Furthermore, the choice and the tuning of parameters of a feasible iterative parallel solver is addressed.

The numerical performance of the parallelised simulation software is demonstrated by means of a full-size simulation of a shield supported tunnel excavation in fully saturated soft soil.

1 INTRODUCTION

A process-oriented numerical simulation of mechanised tunnelling by means of the finite element software can be a valuable tool not only in the design phase of tunnel projects but also during construction. By means of a sophisticated numerical model the ground behaviour, the interactions between ground and shield, and the influence of process parameters such as the heading face support pressure, the tail void grouting pressure, or the jacking forces can be assessed. Simulations that are run simultaneous to the construction process further allow for an investigation of possible hazards and respective mitigation measures.

Based on KRATOS, a simulation software has been developed that is capable of representing all important interaction mechanisms and can perform a process-oriented simulation of the construction process in mechanised tunnelling. This complex model, however, comes at the cost of high computational effort and leads to long runtimes. A simulation concomitant to the construction process is therefore infeasible if the computation times are not decreased below the
real construction time. In the attempt to reduce the runtimes significantly, a parallelisation concept has been applied to the simulation software such that both shared and distributed memory computers can be used to accelerate the simulation software. While large scale shared memory computers are very costly and yet fight with deficiencies in parallel efficiency with respect to the memory bandwidth, it was decided to concentrate the development effort on distributed memory computers. Here, a domain decomposition concept is applied to the tunnelling simulation software that accounts for the specific needs of the model. Here, in particular the handling of contact between the shield and the ground as well as the de- and reactivation of elements representing the excavated soil and the lining and tail void grouting have been major challenges.

2 PARALLELISATION CONCEPT

The simulation software for shield tunnelling problems utilises the built-in parallelisation concept of KRATOS for distributed memory systems. In principle, a classical domain decomposition scheme is applied to distribute the Finite Element mesh among several processors. However, in order to allow for maximum flexibility in the choice of the actual solution method to be applied, a twofold decomposition concept is applied. On the one hand side, the complete mesh is decomposed into non-overlapping subdomains right from the beginning of the execution. Here, each thread reads a part of the mesh that has been decomposed previously by means of the ParMETIS library. This decomposition is then used to perform the assembling of the global system of linear equations including all operation that have to be made on the elements and conditions.

In order to store the global system matrix and the system vectors, the Epetra interface of Trilinos [3] is used. The Trilinos library offers a large variety of interfaces to different parallel linear solvers. Matrices stored in the Epetra format appear to the caller as a monolithic resource of data while in fact they are stored in a distributed manner. Depending on the linear solver of choice, this virtually monolithic matrix can be decomposed a second time, in non-overlapping or overlapping subdomains. Here, the best decomposition for the respective linear solver can be obtained as needed.

The contact between the shield and the surrounding ground is in the simulation model handled by a surface-to-surface contact algorithm according to [4]. This requires yet for a global search over the complete model, a procedure that may cause a tremendous communication overhead if performed on a distributed system. However, since the involved surfaces cover only a small number of nodes compared to the complete model, a dedicated subdomain is created for all nodes and surfaces that belong to the potential contact zones. The whole solution of the contact problem is then performed in this dedicated thread while the rest of the model is distributed among the remaining threads.

3 EXAMPLES

To investigate the parallel performance of the simulation software in the first place a simple benchmark has been computed on a computer cluster using different numbers of threads. The model features a cube structure that is meshed with three different levels of mesh refinement. Two linear solvers, the distributed SuperLU solver [508] and a Multi-Level preconditioned GMRES solver have been used.

Figure 1 shows the speedup and the parallel efficiency of the assembling process and of the different linear solvers, respectively. It can be seen that the assembling process exhibits a nearly perfect speedup. The iterative GMRES solver still provides a fair speedup for larger numbers
of degrees of freedom whereas the direct SuperLU solver leads to a rather poor parallel performance.

The capabilities of the parallelised simulation software has further been tested by means of a real-size tunnelling problem. Without reference to a specific project an excavation in water bearing soil has been computed on a distributed memory computer cluster. Here, the same model was computed using different numbers of threads in order to investigate both the coherence of the simulation results and the parallel performance.

Figure 2 shows the model setup for this example. In the left image the domain decomposition is visualised for a simulation with 16 threads. The right image shows a contour plot of vertical displacements after the shield has passed the simulation domain.

Figure 3 shows the parallel performance of the assembling and solving processes for the tunnel
example. It can be seen that not only the assembling process shows a very good speedup but also the solving process exhibits remarkably better performance than it does in the simple benchmark example.

4 CONCLUDING REMARKS

A parallelisation concept has been presented that allows for the simulation of shield tunnelling problems on distributed memory computers. Here, it has been shown that the parallel performance is at a good level not only for benchmark problems but also for realistic models of a tunnel excavation thus leading to a parallel efficiency that is still more than 80 percent at 16 threads.

REFERENCES


