Dynamically distributed and *p*-adaptive finite element simulation of soil–structure interaction based on multi-agent systems

Udo F. Meissner, Michael Mueller, Jochen Ruben*

Institute for Numerical Methods and Informatics in Civil Engineering, Darmstadt University of Technology, Darmstadt, Germany

Abstract

Complex engineering problems e.g. fluid–soil interaction for geotechnical problems, can be simulated using the finite-element method by considering the different phases in the foundation ground. The constitutive equations are described by the theory of porous media, in which the interaction between the involved phases is considered. The averaging of the phases over a representive elementary volume is carried out using the concept of volumetric content. The complexity and size of three-dimensional engineering problems in the field of fluid–soil interaction makes the simulation numerically very costly, especially when the results are to be improved by fully adaptive methods.

In this paper, we present a finite element formulation for the computation of a two-phase model for the fluid–soil interaction and the improvement of the results with the adaptive *p*-refinement as well as the parallel implementation in a distributed software system using mobile software agents.

An example of practical interest confirms the suitability and the new possibilities arising from this modern approach.

Keywords: FEM; Parallel computing; Software agents; *p*-adaptivity; Soil–fluid interaction

1. Finite element formulation for fluid–structure interaction

For geotechnical problems, porous media can be better described by the theory of mixtures rather than by classical soil mechanics. A porous medium is defined as a solid skeleton whose pores are filled with fluid and/or gas. The properties of such materials are characterized by the interaction of strains and stresses between the different components of the medium.

A finite element (FE) formulation for porous media has been presented [1,2] in which a binary system consisting of a solid skeleton whose pores are filled with an incompressible viscious fluid was assumed. The generalized balance equation (1) for a two-phase system with the phases α and β is given as follows:

$$\int_\Omega n_\alpha \frac{\partial}{\partial t}\langle e\rangle^\alpha d\Omega + \int_\Gamma \left(n_\alpha \langle e\boldsymbol{v} + \boldsymbol{j}^E\rangle^\alpha\right)\cdot \boldsymbol{n} d\Gamma \qquad \text{advection and diffusion}$$

$$+ \int_\Omega \langle e\,(\boldsymbol{v}-\boldsymbol{u})\cdot\boldsymbol{n}\rangle^{\Gamma_{\alpha\beta}} d\Omega \qquad \text{advection} \tag{1}$$

$$+ \int_\Omega \langle \mathbf{j}^E\cdot\boldsymbol{n}\rangle^{\Gamma_{\alpha\beta}} d\Omega \qquad \text{diffusion}$$

$$- \int_\Omega n_\alpha \langle \rho \boldsymbol{f}^E\rangle^\alpha d\Omega = 0 \qquad \text{sources and sinks}$$

This idealization, the different terms on the control

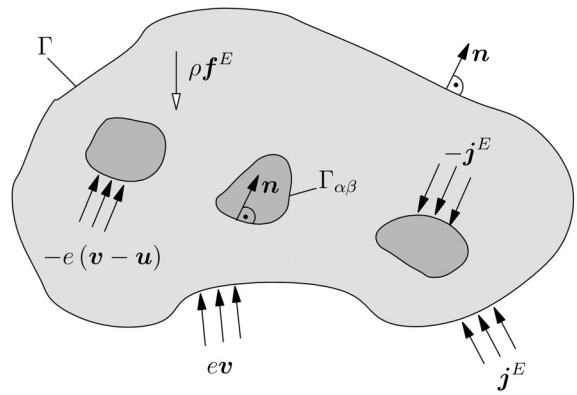

Fig. 1. Interaction between two phases.

* Corresponding author. Tel.: +49 6151 166675; Fax: +49 6151 165552; E-mail: ruben@iib.tu-darmstadt.de

Computational Fluid and Solid Mechanics 2005
K.J. Bathe (Editor)

volume and the interaction between the two phases are depicted in Fig.1, where e is the quantity to be balanced, n^{α} is the volumetric content of the phase α in the domain Ω with the boundary Γ, and

$$\langle \ldots \rangle = n_{\alpha} \langle \ldots \rangle^{\alpha} = \frac{1}{V} \int_{\Omega_{\alpha}} \ldots d\Omega_{\alpha} \tag{2}$$

being the volumetric average and

$$\langle \ldots \rangle^{\Gamma_{\alpha\beta}} = \frac{1}{V} \int_{\Gamma_{\alpha\beta}} \ldots d\Gamma_{\alpha\beta} \tag{3}$$

being the jump of a quantity on the surface $\Gamma_{\alpha\beta}$ between the phases α and β averaged over the volumetric content.

Applying a standard galerkin scheme, we obtain the equations for the discrete system, with the absolute displacement of the soil $\boldsymbol{u}$ and the relative displacement of the fluid to the soil $\boldsymbol{w}$. By introducing the relative quantity $\boldsymbol{w}$, the system of Eq. (4) becomes non-symmetric, which has a great impact on the numerical solving cost. Unfortunately, this step is absolutely essential for modelling real-world boundary conditions [3].

$$\begin{bmatrix} M_S & 0 \\ M_{SF} & M_F \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{u}} \\ \ddot{\boldsymbol{w}} \end{bmatrix} + \begin{bmatrix} 0 & -C_{FS} \\ 0 & C_F \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{u}} \\ \dot{\boldsymbol{w}} \end{bmatrix} + \begin{bmatrix} -K_S & -K_{FS} \\ -K_{SF} & -K_F \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R_S} \\ \boldsymbol{R_F} \end{bmatrix} \tag{4}$$

2. Improving the results by *p*-adaptivity

The accuracy of the FE solution of a differential equation depends on the characteristic element size and the polynomial order p of the shape functions. The order of the accuracy of the approximation can be stated as $O(h^p)$, leading to two essential possibilities to improve the accuracy of the approximation: on the one hand an adaptation of the grid size h (remeshing) and on the other hand an increasing of the polynomial degree p of the Taylor polynomials. The significant numerical advantages (especially for smooth solutions) of the p-refinement over the h-refinement have been described in many publications [4,5].

The p-refinement can be implemented in two ways. One possibility is to use Lagrange-shape functions of higher order for the elements that are to be refined, which leads to some problems regarding the compatibility between elements of different order. Another possibility that is especially advantageous in a fully adaptive context is to use hierarchical shape functions, offering the benefit of keeping the already computed polynomials of lower order. Furthermore, it is easy to implement the p-refinement in a way that guarantees the compatibility between adjacent elements by rising the order of one or more substructures e.g. edges, faces or bodies of an element.

In the latter case, usually hierarchical p-adaptivity on the basis of Legendre polynomials P with the maximum polynomial degree p for the displacement ansatz is applied [6]. Hierarchic shape functions offer the advantage of yielding system matrices with a smaller condition number when compared with classical shape functions. This can be improved further by orthonormalizing the Legendre polynomials. These orthonormalized polynomials can be derived directly from the Legendre polynomials and are given by:

$$\begin{aligned} P_1(\theta) &= \frac{1}{2}(1-\theta) \\ P_2(\theta) &= \frac{1}{2}(1+\theta) \\ P_i(\theta) &= \phi_{i-1}(\theta) \quad \text{for } i = 3,4,\ldots,p+1 \end{aligned} \tag{5}$$

with

$$\phi_j(\theta) = \sqrt{\frac{2j-1}{2}} \int_{-1}^{\theta} L_{j-1}(x)\, dx = \frac{1}{\sqrt{4j-2}} (L_j(\theta) - L_{j-2}(\theta)) \quad \text{for } j = 2,3,\ldots \tag{6}$$

and

$$L_j(\theta) = \frac{1}{j! \cdot 2^j} \frac{d^j}{d\theta^j} \left[(\theta^2 - 1)^j \right] \quad \text{for } j = 0,1,\ldots \tag{7}$$

where $L_j(\theta)$ are the Legendre polynomials of the degree j and $P_1(\theta)$ and $P_2(\theta)$ are the linear nodal shape functions. All higher polynomials and bubble modes have the value 0 at the nodes because of their orthogonal nature.

The ansatz spaces for two- and three-dimensional elements are computed by the tensor product of the basic polynomials given in Eqs (5) and (6) for all dimensions, respectively. For polynomials of order $p > 1$, the tensor product leads to displacements not of the nodes but of the edges, faces or bodies.

Some two-dimensional polynomials are depicted in Fig. 2.

3. Parallelization

The p-refinement is ideally suited for parallelization on a distributed memory architecture for various reasons. On this kind of distributed architecture, computers have to communicate for solving the resulting equation system and one aims to minimize this communication.

Due to the fact that the resulting system matrices have a very small condition number, very few iterations

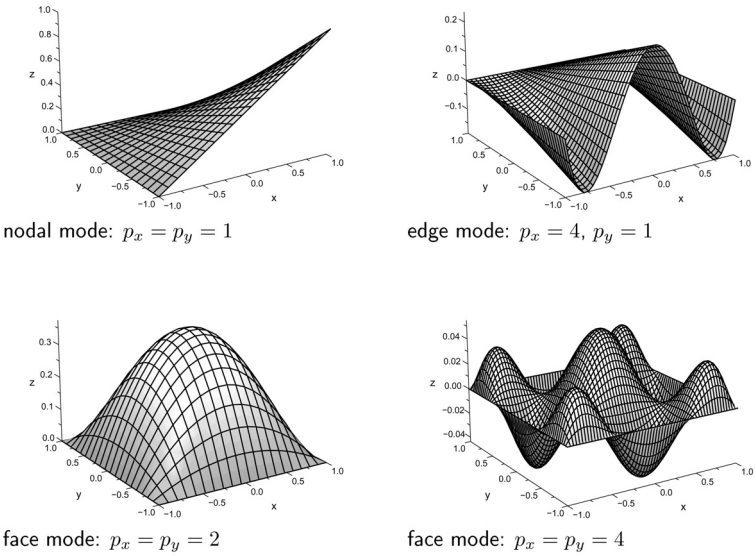

Fig. 2. Examples of two-dimensional hierarchical shape functions.

during the iterative solving process have to be carried out. Using a non-overlapping domain decomposition, at least one vector has to be exchanged during each iteration. Due to the non-symmetrical system matrices resulting from the relative fluid displacement, simple solvers such as *CG* and *PCG* cannot be applied, and more complex solvers such as the preconditioned *STABBiCGSTAB*, which requires four communications per iteration, have to be used. This fact underlines the importance of well-conditioned matrices and thus the suitability of the *p*-refinement.

It can be observed that with higher polynomial degrees, the numerical cost shifts away considerably from the solving of the equation system towards the computation of the element stiffness matrices because of the expensive numerical integration. This step, however, does not involve any communication, making it a perfect candidate for parallelization.

In recent years, great efforts have been made to develop parallel solving techniques using homogeneous workstation clusters in local area networks (LAN). Today, the use of the mobile software agent technology is a consequent step towards a new parallel finite element technique using the Internet with heterogeneous computer resources. The heterogeneity refers to different operating systems as well as to very different performances of the participating computers.

The main focus of the current research is to evaluate the software agent technology in a network of heterogeneous workstations, e.g. a LAN or the Internet, by using the more efficient *p*-version for soil–fluid interaction problems that require substantial computations in local subdomains.

Software agents possess various characteristics, such as autonomy, reactivity and proactivity [7]. A software agent can be seen as an artificial robot that operates within a software environment. The environment includes operating systems, software applications, databases, networks and computer domains. By making decisions and acting independently in their environments, software agents reduce human workload by interacting with their users only when it is time to deliver results.

For numerical simulations, available software agent systems offer many advantages, e.g. an efficient communication layer and mobility combined with a very flexible way of distributing the software code since in many systems the agent carries its own code and state and reloads it transparently over the network. The agent-oriented paradigm as an extension to the object-oriented paradigm (OOP) usually leads to loosely coupled components and offers an optimal support for heterogeneous resources.

4. Application example

The efficiency and performance of the presented techniques are demonstrated by a typical geotechnical system. Fig. 3 shows a discretized combined pile–raft foundation. The left-hand illustration shows the discretization resulting in 1500 brick elements with 11 000 degrees of freedom when using linear shape functions.

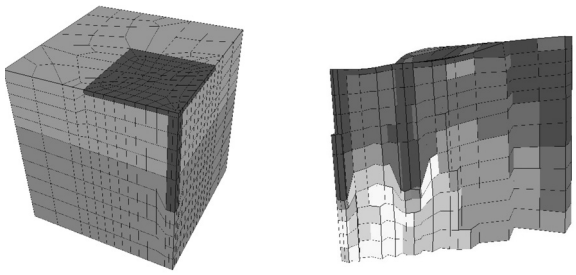

Fig. 3. Discretization and distribution of pressure under a combined pile–raft foundation.

The right-hand illustration displays the distribution of the pore pressure. The system consists of an impervious concrete slab and five concrete piles that found into a sustainable soil layer.

During the *p*-adaptive simulation, choosing a maximum polynomial degree of six, mainly the polynomial degree of the elements under the foot of the piles was increased. The degrees of freedom rose to about 100 000, with the error in the energy norm falling below 1%.

The simulation was carried out with four computation agents on standard Intel PCs ranging from 1 GHz up to 2.4 GHz, running both Linux and Windows operating systems. The efficiency of the parallelization can be described by the characteristic cost of the calculation. Additionally, the extra costs of the communication between the different processors while solving the problem have to be taken into account [8].

Efficiency is a very clear performance indicator of parallel computation methods. It expresses de facto the degree to which the performance potential of the parallel processors is used. The efficiency E is defined by the speed-up S

$$S(n) = \frac{T_{seq}}{T_n} \tag{8}$$

$$E(n) = \frac{\frac{T_{seq}}{T_n}}{n} = \frac{S(n)}{n} = \frac{\frac{T_{seq}}{n}}{T_n} \tag{9}$$

where n is the number of processors, T_n is the computer time on a specific parallel processor and T_{seq} is the time on a single sequential machine.

Fig. 4 shows a combined speed-up and efficiency diagram for the calculation of the system in Fig. 3. These are the first results evaluating the application of mobile software agents for numerical simulations. As shown, the speed-up using four computation agents in a heterogeneous environment is close to three, which corresponds to an efficiency of nearly 70%.

It is important to mention that only a small part of the loss of performance results from the communication between the agents but a large part results from the fact that different machines have been used, resulting in a simulation that is not perfectly balanced. Ongoing and future work will improve the dynamics of the implemented system for load balancing between the computation agents with respect to the performance of their host processors. The presented system can be a great chance to leverage the computational power that is available nearly within every engineering company.

References

[1] Meissner UF, Mueller M, Ruben J. *p*-Adaptive FEM simulation of fluid flow in porous media based on agent technology. In: Proceedings of the 15th International Conference on Computational Methods in Water Resources (CMWR XV), Vol. 2, CT Miller, MW Farthing, WC Gray, editors, Aachen, Germany, Shaker Verlag, Chapel Hill, NC: 2004; Oxford: Elsevier 1091–1102.

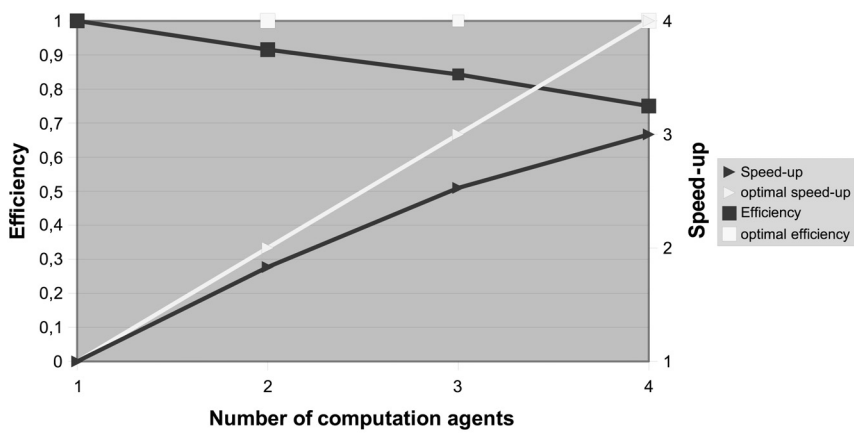

Fig. 4. Speed-up and efficiency.

[2] Terlinden I, Meissner UF. Simulation of fluid flow in porous media with object oriented techniques. In: (Bathe, KJ ed.). Proceedings of the 1st MIT Conference on Computational Fluid and Solid Mechanics, Vol. 2 Cambridge, MA: Elsevier Science, 2001; pp. 985–988.
[3] Müller M, Ruben J. Adaptive dynamisch verteilte baugrund-tragwerk-simulation auf der grundlage der theorie poröser medien. In: Forum Bauinformatik 2004, JZ Emmermann, S Geller, editors, 2004.
[4] Düster A. High Order Finite Elements for Three-Dimensional, Thin-Walled Nonlinear Continua. Bauwesen: Shaker Verlag, 2002.
[5] Müller M, Ruben J. Aspekte der agentenbasierten adaptiven fe-simulation. In: A Kaapke, K Wulf, editors; Forum Bauinformatik 2003 – Junge Wissenschaftler forschen, Aachen, Germany, Shaker Verlag, 2003; pp. 10–20.
[6] Promwungkwa A. Data structure and error estimation for an adaptive *p*-version finite element method in 2-D and 3-D solids. PhD thesis, Mechanical Engineering Department, Virginia Tech, 1998.
[7] Wooldridge M. An Introduction to Multiagent Systems. Chichester, UK: John Wiley & Sons, 2002.
[8] Lämmer L. Parallelisierung von Anwendungen der Finite-Element-Methode im Bauingenieurwesen. Institute for Numerical Methods and Informatives in Civil Engineering, Darmstadt University of Engineering, Habilitation, thesis, 1997.