Benchmarking for Boundary Element Methods

M. Kuhn* O. Steinbach†

Abstract

Many new algorithms for solving partial differential equations numerically have been proposed and implemented during the last years. This development causes the need for tools that allow an objective comparison of the new software packages and of the algorithms they are based on.

In this paper, we present a general framework for presenting both, problems and solutions of benchmark problems with special emphasis on Boundary Element Methods. Furthermore, first, rather academic problems are proposed.

1 Introduction

Many practical problems arising in science and engineering can often be formulated as partial differential equations (PDEs). Thus, the numerical solution of those equations is of great interest. Although the power of the computers which are available for such computations is growing almost day by day, the resources will always by restricted, either by memory or by time. That is why the application of new algorithms for solving the discrete systems arising from the PDEs is essential. During the last years many algorithms which are optimal or almost optimal with respect to numerical effort and required memory per unknown have been developed. Besides these theoretical results, it is even more important to compare the actual performance of the algorithms when the are applied to practical problems. For this purpose we introduce a framework for benchmarking which is to be a basis for a living process of creating problems and comparing results. Hereby, we will, in a first stage, concentrate on the mathematical aspects of such comparisons. Nevertheless, the problems should contain typical difficulties as they arise in practical applications. Later on, more realistic problems should be included which then also may demonstrate the robustness of certain algorithms.

The primary purpose of this benchmark is to obtain an objective comparison and quality measure of the various solution approaches using boundary element methods (BEM). It was established as a result of the priority research programme "Boundary Element Methods" supported by the German Research Foundation DFG. A first attempt of presenting benchmarks has been included in the final report of the priority research programme which will appear in [3].

*Institute of Mathematics, Johannes Kepler University Linz, Altenberger Str. 69, A-4040 Linz, Austria. (E-mail: kuhn@numa.uni-linz.ac.at)

†Mathematical Institute A, University of Stuttgart, Pfaffenwaldring 57, D-70569 Stuttgart, Germany. (E-mail: steinbach@mathematik.uni-stuttgart.de)

We invite everybody to create new benchmark problems and to submit solutions obtained by finite element methods (FEM), BEM or other discretization methods. These tests may give every group the opportunity to present their results and to underline the advantages and disadvantages of their algorithms. For this reason the participants are asked to submit a rather complete account of their computational results together with detailed information about the discretization and solution method used. The required information is described in this paper. As a result it should be possible to distinguish between "efficient" and "robust" and "less efficient" and "less robust" solution approaches. In this sense we follow ideas which have been discussed before, e.g. in [2], where benchmark problems for laminar flows are presented.

The paper is organized as follows. We start with some general remarks on boundary element methods in Section 2. In particular we state the most relevant questions with respect to the definition of benchmark problems. In Section 3, we introduce a hardware benchmark UniBench which simulates typical calls as the occur in any BEM software. Section 4 is devoted to the presentation of problems, a general frame is developed. In the following Section 5, the general form for presenting the solution is given. Section 6 contains first examples including solutions to one of the problems. Finally we give some concluding remarks in Section 7.

Addresses of FTP–sites and WWW–pages which contain the required sources as well as additional information can be found in the appendix.

2 BEM Benchmarking

Boundary Element Methods (BEM) have been used intensively for solving various types of problems, and there exist many software packages applying BEM. As a result of the priority research programme "Boundary Element Methods", a first collection of benchmark problems was formulated to allow a comparison of the different approaches and implementations using BEM.

Whenever BEM are to be applied the following components have to be chosen:

- Formulation:
  - direct or indirect formulation,
  - first or second kind integral equation,
  - symmetric formulation.
- Discretization:
  - Galerkin,
  - collocation.
- Generation of the discrete operators:
  - approximation and description of the boundary,
  - analytical integration,
  - numerical integration,

  - panel clustering methods,
  - wavelet-based methods.

- Boundary conditions
  - approximation or interpolation,
  - exact.

- Trial functions:
  - h–methods,
  - p–methods,
  - h–p–methods.

- Method of refinement:
  - uniform refinement,
  - adaptive refinement (includes the choice of an error estimator).

- Solver:
  - direct solver,
  - iterative solver,
  - choice of a preconditioner.

- Method of parallelization:
  - sequential method (no parallelization),
  - parallized sequential solver,
  - Domain Decomposition.

Both, the collection of the components and the choices given are not complete. For a good performance of the whole algorithm each single component is as important as the right combination of all components. Furthermore, the point of view for evaluating certain components may change as the surrounding conditions, e.g. computers which are available, change.

In the following, the problems and the way both, solutions and problems are presented underlines the special interest in Boundary Element Methods.

3 The Hardware Benchmark

3.1 The Code UniBench

In order to allow a fair comparison of CPU time between the different results we supply a special code by M. Maischak [1] (C source and F77 source) which is to be compiled using the same options as for the scientific code. As a result the benchmark code gives two numbers relating to special BEM-matrix calls and a call of a solver. These numbers

should help to interpret the pure MFlop-rate of a computer. If you submit any Benchmark results, please report also the numbers generated by UniBench for comparison.

In particular, the program UniBench compares the performance of different computers using a first kind boundary element equation. For testing the scalar-performance we use the time for assembling the Galerkin matrix. For testing the vector-performance we use the time for solving the Galerkin-system by a CG-scheme without any preconditioning. An example of the standard output of UniBench looks as follows.

```
#####################################################
#                                                   #
#               Results of UniBench                 #
#                                                   #
# Degrees of Freedom      :                  512    #
#---------------------------------------------------#
# Mat-Time                :                1.058    #
# LGS-Time                :                2.602    #
# Iterations              :                  91    #
#---------------------------------------------------#
# Relative Error          :    0.4835573252E-10    #
# E-Norm (x*l)            :    0.2128244285E+01    #
#                                                   #
#####################################################
```

The two BEM-rates (Mat-Time and LGS-Time) obtained by UniBench are closely related to typical calls of a BEM-code:

Mat-Time corresponds to the matrix generation. Thus, function calls which are typical for the BE-matrix generation are tested.

LGS-Time corresponds to the solver of the linear system. Thus, real/double operations which are typical for any direct or iterative solver are tested.

System	Mat-Time	LGS-Time	Iterations
SUN ULTRA 1	1.06	2.60	91
SGI IndySC (R 4400)	2.81	4.72	88
SUN SS4	2.83	7.62	91
SUN SS20	2.89	4.63	83
XPlorer (Parsytec, 1 proc.)	3.23	3.50	89

Table 1: Results of UniBench .

You will also get the number of iterations performed and the energy norm of the solution. The latter should be about 2.1282442853. Table 1 shows some typical results.

3.2 Using UniBench

After untaring the file *bench.tar* one can start UniBench by typing *runbench.* Then, first, a script called 'config' tries to determine the name of the Fortran compiler, it chooses system

dependent options for compiling the code and checks which libraries are available. If the NAG-library is available the subroutine x05baf is used for time-measuring. Otherwise 'config' looks for the function ETIME of the Berkeley-extensions which are available of most systems. If this also fails you have to write yourself a timer-routine which then is called from the routine in *adsys.f.* After this, *unibench.f* will be compiled and invoked with a problem size of 512 degrees of freedom.

To start UniBench via the run-script is very comfortable and turns out to be very useful for testing hardware components. Following our experience, it worked on all single-processor machines, whereas on parallel machines it was necessary to modify the routine for time measuring in *adsys.f.*

In our case it is desired to compile UniBench using the same compiler options as for the scientific code. That is why we ask the participants to modify the script *config* appropriately, in particular to define the parameters **OPT** and **OPTS** in line 252f manually according to the choice of the user. Alternatively, UniBench could be compiled separately, as it is necessary for parallel computers in any case.

4 Form for Presenting a Problem

4.1 Preliminary Remarks

Every benchmark problem which is presented should have some specific goal, that is, it should act as a test for *single* components of the algorithm or for the behaviour of the algorithm with respect to *single* 'bad' parameters. Then, the problem should concentrate on this special goal, i.e., it should not contain additional difficulties. For example, if the efficiency of the generation of the matrices is to be documented, boundary conditions and the underlying geometry should be as simple as possible to allow as many competitors as possible. However, computation times should be related to the accuracy of the numerical solution in any case, e.g. by the simple computable L^2–norm of the error for academic test problems where the analytical solution is known. In order to compare different approaches, the task has to involve the computation of components of the solution. If possible, the quantities which are to be computed should be *single* numbers. The computation of these numbers should be as easy as possible, except the case that the computation itself is a goal of the problem. Additionally, for each problem case studies for certain parameters may be proposed if they are not a primary part of the problem.

Afterall, the benchmark problems should give answers to questions like:

- How accurate are the results of the algorithm ?
- What is the numerical effort to get this accuracy ?
- What is the numerical effort to get a specified accuracy ?
- How robust is the algorithm with respect to 'bad' parameters ?

In particular the discretization parameter h is one of those 'bad' parameters. The dependence of the performance of the solver on h is included in the presentation of the results by default.

In contrast to academic problems, it would be very interesting, in particular for testing the robustness of algorithms, to consider problems where the solution or the special

behaviour of it is known to the presenter of the problem only. Then, this person could initiate a special competition for all groups (except his own !) which are interested in presenting their results.

The evaluation of the manpower which is necessary for implementing the algorithm is beyond the scope of this benchmarking.

4.2 Components of a Presentation

The presentation of problems should follow the fixed form given in Subsections 4.2.2–4.2.6 below.

The quantities of Point 4.2.4 must be defined clearly, since they are the only basis for a real comparison of the different algorithms. For example, a comparison of the efficiency of the generation of the BEM matrices, will require a final error check of the results, since a cheap numerical integration may be fast but is inefficient with respect to the error of the solution. The remarks of point 4.2.6 should involve a table which is to be used for presenting the results.

The tex-file containing the general form can be accessed via FTP or WWW (see appendix).

4.2.1 Problem and Author

1. Name of the problem:
2. Address of the author:

4.2.2 Specific Goal

State the specific goal of the problem.

4.2.3 Formal Description

1. Geometry:
2. Partial differential equation:
3. Boundary conditions:

4.2.4 Quantities to be computed

State the quantities which have to be computed to allow a comparison of the algorithms. These values should finally allow to evaluate the quality of the algorithm with respect to the goal formulated above.

4.2.5 Parameter Tests

State further tests by manipulating certain parameters of the problem, as, e.g., the wave-number of the Helmholtz equation.

4.2.6 Extensions and Remarks for Presenting the Results

Propose a form of a table which is to be used for presenting the results and, if necessary, state extensions for presenting the results which are related to the goal of the problem.

5 Form for Presenting the Solution of a Problem

5.1 Preliminary Remarks

First of all we state some general rules for the presentation.

- The software must be available, i.e., we suggest to check the results during personal visits between groups.
- If an iterative solver is used, the iterative process should start from zero values.
- The finest spatial mesh h_1 can be chosen by the user.
- The convergence criteria for the iterative method can be chosen by the user.
- If possible the calculations should be performed on a workstation. For all computers used, the theoretical peak–performance as well as the two BEM-rates (Mat-Time and LGS-Time) should be provided. The values should be obtained with the same compiler options as used for the scientific BEM solver.
- Beside the benchmark results, a description of the solution methods should be given. To facilitate comparison, the presentation should be adapted to the form given below.
- For parallel algorithms additional results as propose in point5.2.5 should be provided. For this purpose, results of calculations using, at least, two different numbers of processors should be provided.

5.2 Components of the Presentation

As the presentation of problems, the presentation of results should follow a predefined form. The tex-file containing this form can be accessed via FTP or WWW (see appendix).

5.2.1 Problem and Author

1. Name of the problem:
2. Address of the author:

5.2.2 Hardware and Compiler

1. Hardware:
   (a) DEC/SUN/IBM/...
   (b) xx MFlop peak performance

(c) xx MB RAM, xx MB used, xx KB Cache
(d) Mat–Time: xx
(e) LGS–Time: xx
(f) # Iterations: xx

2. Compiler:
   (a) CC++/F77/F90/...
   (b) Options: -xx -xx

5.2.3 Solution Components

1. Equation:

   State the BE-equation being used.

2. Discretization:
   (a) Approximation of the boundary.
   (b) Discretization of the boundary.
   (c) Test and trial functions.
   (d) Integration scheme for the operators.

3. Approximation of the Boundary Conditions:

   State how the boundary conditions have been implemented.

4. Solver:
   (a) Short description of the solver.
   (b) Stopping criterion for iterative solvers.

5. Postprocessing:

   State how the required quantities, e.g. the L^2–error, have been computed.

5.2.4 Results

In order to allow the evaluation of the efficiency of the algorithm the results should be given for different (at least) three (subsequent) mesh sizes related to the discretization parameter h_1, h_2, h_3. We ask at least for the following quantities:

1. Number of unknowns, on $\partial\Omega$ and overall if these numbers are different as in DD–BE or even FE methods.
2. Number of iterations for iterative solvers.
3. CPU time for generating the matrices and for solving the discrete system.
4. Computed quantities according to the problem.

A predefined table (latex) for presenting the results should be supplied by the presenter of the problem.

5.2.5 Efficiency (parallel algorithms)

State results of the previous section for (at least) two different numbers of processors $p_1, p_2, \ldots, p_M$ and compute the scale-up

$$S_c^{(k,l)} := \frac{T_k^{(s)}}{T_l^{(s)}} * \frac{N_l^{(s)}}{N_k^{(s)}} \quad \text{for} \quad N_k^{(s)} \approx N_l^{(s)}, \quad (1 \le k \le l \le M),$$

and the scaled efficiency

$$E_s^{(k,l)} := \frac{T_k^{(e)}}{T_l^{(e)}} * \frac{N_l^{(e)}}{N_k^{(e)}} * \frac{p_k}{p_l} \quad \text{for} \quad \frac{N_k^{(e)}}{p_k} \approx \frac{N_l^{(e)}}{p_l}, \quad (1 \le k \le l \le M),$$

where $T_i^{(s,e)}$ is the CPU-time on p_i processors required for generating and solving the discrete problem with $N_i^{(s,e)}$ unknowns, where the superscripts (e, s) stand for the (different) experiments for measuring $S_c^{(*)}$ and $E_s^{(*)}$, respectively.

	$i = 1$	$i = 2$	$i = 3$
p_i $N_i^{(s)}$ $T_i^{(s)}$			
S_c S_c	1.0 –	$S_c^{(1,2)}$ 1.0	$S_c^{(1,3)}$ $S_c^{(2,3)}$
p_i $N_i^{(e)}$ $T_i^{(e)}$			
E_s E_s	1.0 –	$E_s^{(1,2)}$ 1.0	$E_s^{(1,3)}$ $E_s^{(2,3)}$

Thus, for parallel algorithms the table given above should be completed for at least $i = 1, 2$.

6 Academic Problems

6.1 Problem 1

6.1.1 Problem and Author

1. Name of the problem: 2-d Potential Dirichlet Problem.
2. Address of the author: O. Steinbach, University of Stuttgart, Mathematical Institute A, Pfaffenwaldring 57, D-70569 Stuttgart.

6.1.2 Specific Goal

This academic benchmark problem is to compare two specific components of a boundary element code, i.e. the discretization of the boundary integral operators and the solution process to solve the corresponding algebraic system of linear equations. For this we define

a two–dimensional Dirichlet problem for the Laplacian and ask for a boundary element method to get the remaining Cauchy data, i.e. the flux. The analytical solution is known so that we can compute the error in some norm.

6.1.3 Formal Description

1. Geometry: L–shape domain
$$\Omega = [-0.25, 0.25]^2 / [-0.25, 0]^2$$
2. Partial differential equation: Laplace equation
$$\Delta u(x) \;=\; 0 \quad \text{for } x \in \Omega$$
3. Boundary conditions: Dirichlet boundary conditions
$$u(x) \;=\; \ln|x - x^*| \quad \text{with } x^* = (-0.1, -0.1)$$

6.1.4 Quantities to be computed

Besides the standard values (time for generating and solving the discrete system) we ask for

1. L^2–error of the numerical solution, while the exact solution of the problem is given by
$$t(x) \;=\; \frac{(n(x), x - x^*)}{|x - x^*|^2}\,.$$
Then the error can be computed by
$$||t - t_h||_{L^2(\Gamma)} \;=\; \left(\int_\Gamma |t(x) - t_h(x)|^2 \, ds_x \right)^2,$$
e.g. by a numerical integration scheme.

The values should be provided for (at least) three (subsequent) mesh sizes related to the discretization parameter h_1, h_2, h_3, where $N = 512$ (number of unknowns on $\partial\Omega$) should be included.

6.1.5 Parameter Tests

No further tests.

6.1.6 Extensions and Remarks for Presenting the Results

$N = 512$ (number of unknowns on $\partial\Omega$) should be included in the presentation. The following table should be used for presenting the results.

	System	Solver		
N	sec	Iter	sec	L^2–error

6.2 Problem 2

6.2.1 Problem and Author

1. Name of the problem: Linear Elasticity in 2D (plain strain)

2. Address of the author: O. Steinbach, University of Stuttgart, Mathematical Institute A, Pfaffenwaldirng 57, D–70569 Stuttgart

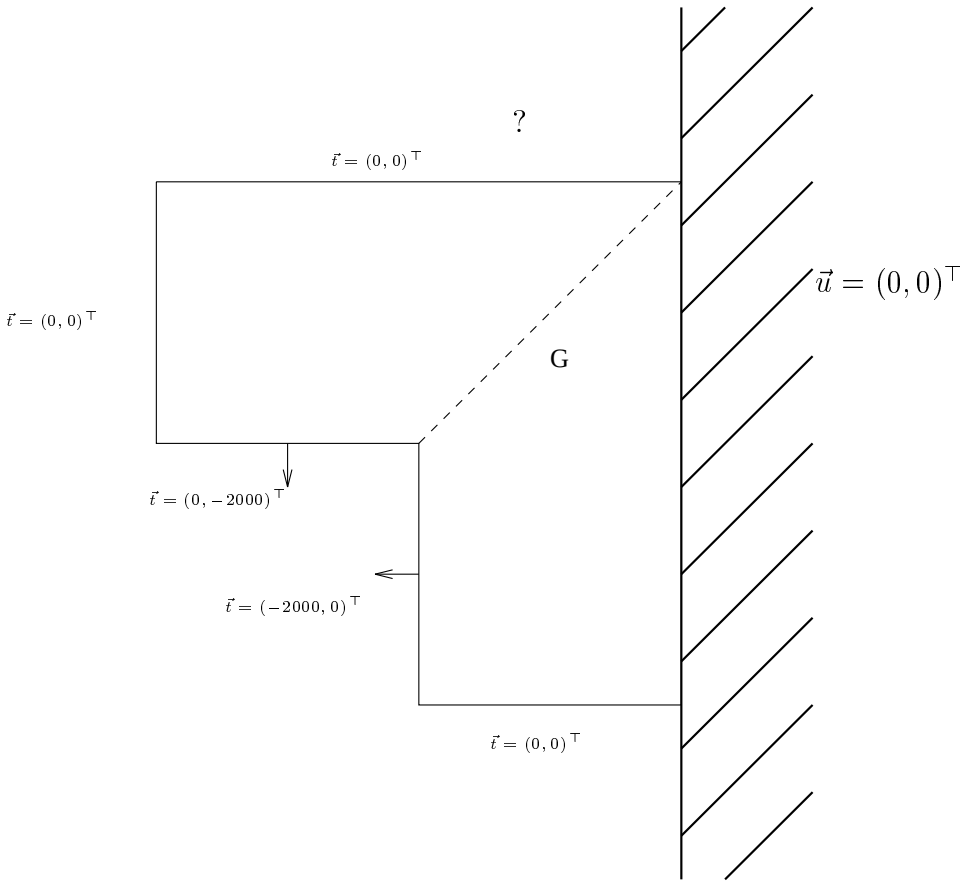
6.2.2 Specific goal

This rather academic problem is to check boundary element methods for mixed boundary value problems in 2d-linear elasticity (plain strain). We are interested in the formulation of related boundary integral equations and their discretization as well as in the solution of the resulting linear system. Finally we will check the dependence on Poissons ratio for $\nu \to \frac{1}{2}$.

6.2.3 Formal Description

1. Geometry: L–shape domain

$$\Omega = [-0.25, 0.25]^2 / [-0.25, 0]^2$$



2. Partial differential equation: Linear Elasticity (plain strain)

$$\mu \Delta u(x) + (\lambda + \mu) \text{grad div}\, u(x) = 0 \quad \text{for } x \in \Omega$$

3. Material parameter:

$$\nu = 0.3,\ E = 200000 \ \Rightarrow\ \lambda = 115384.6,\ \mu = 76923.1$$

4. Boundary conditions: as indicated in the sketch

6.2.4 Quantities to be computed

- Approximate solutions for the remaining Cauchy data.
- Approximate solutions for the displacements u and the main stress σ along G.

6.2.5 Parameter Tests

Perform the computations for Poissons ratio

$$\nu_1 = 0.3, \quad \nu_2 = 0.4, \quad \nu_3 = 0.49\,.$$

6.2.6 Extensions and Remarks for Presenting the Results

- The numerical solutions should be given graphically with respect to a parameter representation of $\partial\Omega$ and G, respectively.
- The solution in $x^* = (0.01, 0.01)^*$ should be given in a table together with the computing times for the generation of the discrete system and its solution.

Discretization		Matrix	Solver	
N	$h_{\max}/h_{\min}$	sec	Iter	sec

	Displacements		Main stresses	
N	$u_1(x^*)$	$u_2(x^*)$	$\sigma_1(x^*)$	$\sigma_2(x^*)$

6.3 Problem 3

6.3.1 Problem and Author

1. Name of the problem: Linear Elasticity in 3D
2. Address of the author: O. Steinbach, University of Stuttgart, Mathematical Institute A, Pfaffenwaldirng 57, D–70569 Stuttgart

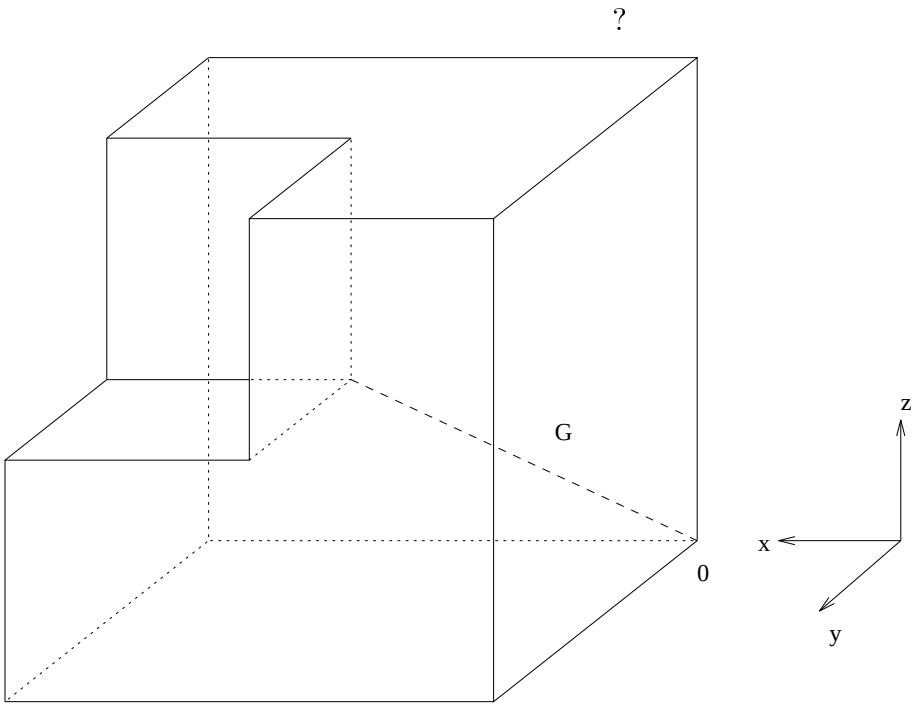
6.3.2 Specific goal

This rather academic problem is to check boundary element methods for mixed boundary value problems in 3d-linear elasticity. We are interested in the formulation of related boundary integral equations and their discretization as well as in the solution of the resulting linear system. Finally we will check the dependence on Poissons ratio for $\nu \to \frac{1}{2}$.

6.3.3 Formal Description

1. Geometry: The domain Ω is given by

$$\Omega \ = \ (0,2)^3 \backslash (1,2)^3$$



2. Partial differential equation:

$$\mu \Delta u(x) + (\lambda + \mu) \text{grad div}\, u(x) \ = \ 0 \quad \text{for} x \in \Omega$$

3. Material parameter:

$$\nu = 0.3, E = 200000 \quad \Rightarrow \quad \lambda = 115384.6, \mu = 76923.1$$

4. Boundary conditions: For the faces characterized by

   - $z = 0$: $t_1 = 0$, $t_2 = 0$, $t_3 = 0$.
   - $z = 2$: $t_1 = 0$, $t_2 = 0$, $u_3 = 0$.
   - $x = 0$: $t_1 = -1000$, $t_2 = 0$, $t_3 = 0$.
   - $y = 2$: $t_1 = 0$, $u_2 = 0$, $t_3 = 0$.

- $y = 0$: $t_1 = 0$, $t_2 = 0$, $t_3 = 0$.
- $x = 2$: $u_1 = 0$, $t_2 = 0$, $t_3 = 0$.

For the faces of the Fichera edge:

- $u_1 = 0$, $u_2 = 0$, $u_3 = 0$.

6.3.4 Quantities to be computed

- Approximate solutions for the remaining Cauchy data.
- Approximate solutions for the displacements u and the main stresses σ along G.

6.3.5 Parameter Tests

Perform the computations for Poissons ratio

$$\nu_1 = 0.3, \quad \nu_2 = 0.4, \quad \nu_3 = 0.49\,.$$

6.3.6 Extensions and Remarks for Presenting the Results

- The numerical solutions should be given graphically with respect to a parameter representation of G. respectively.
- The solution in $x^* = (0.01, 0.01, 0.01)^*$ should be given in a table together with the computing times for the generation of the discrete system and its solution.

Discretization		Matrix	Solver	
N	$h_{\max}/h_{\min}$	sec	Iter	sec

N	Displacements			Main stresses		
	$u_1(x^*)$	$u_2(x^*)$	$u_3(x^*)$	$\sigma_1(x^*)$	$\sigma_2(x^*)$	$\sigma_3(x^*)$

6.4 Results for Problem 1

6.4.1 Problem and Author

1. Name of the problem: 2-d Potential Dirichlet Problem.
2. Address of the author: O. Steinbach, University of Stuttgart, Mathematical Institute A, Pfaffenwaldring 57, D-70569 Stuttgart.

6.4.2 Hardware and Compiler

1. Hardware:
   (a) SUN
   (b) xx MFlop peak performance
   (c) xx MB RAM, xx MB used
   (d) Mat–Time: 2.88
   (e) LGS–Time: 13.34
   (f) Iterations: 83
2. Compiler:
   (a) F77
   (b) Options: -u -c

6.4.3 Solution Components

1. Equation:

   Direct boundary integral equation

   $$(Vt)(x) = (\frac{1}{2}I + K)u(x)$$

   with the single layer potential V and the double layer potential K.

2. Discretization:
   (a) Polygonal approximation of the boundary curve.
   (b) Uniform discretization of the boundary.
   (c) Piecewise linear trial functions for the flux, discontinuous at the corners.
   (d) Galerkin with full analytic integration scheme for the operators for a polygonal approximation of the boundary curve.
3. Approximation of the Boundary Conditions:

   Interpolation with piecewise linear splines.

4. Solver:
   (a) CG preconditioned by the discrete hypersingular integral operator.
   (b) Relative accuracy $\varepsilon = 10^{-8}$.

6.4.4 Results

1. Number of unknowns $N = 64, \ldots, 1024$ on $\partial\Omega$.
2. Number of iterations (Iter) for the cg-solver.
3. CPU time in seconds (sec).
4. L^2–error.

	System	Solver		
N	sec	Iter	sec	L^2–error
64	0.50	19	0.08	7.60 –2
128	1.90	20	0.17	2.39 –2
256	7.68	19	0.55	7.82 –3
512	30.16	19	2.54	2.65 –3
1024	120.82	20	8.51	9.14 –4

7 Conclusions

We have presented a general framework for benchmarking in Boundary Element Methods. The problems are well suited for comparing existing software packages, including both, commercial and scientific codes. On the other hand, the problems collected here can serve as first test problems for new codes which are under development.

Once again, everybody is invited to participate – either by presenting problems or by joining the competition. The success of such a benchmark, depends on the feedback of numerous participants ! It is a big chance to present the results of research to a wider public and to underline the advantage of modern discretization and solving techniques.

Additional problems, e.g. for the 3d-Helmholtz/Laplace equation, will be presented very soon.

References

[1] M. Maischak. UniBench – a boundary element hardware benchmarking code. F77-code, University of Hannover, Institute of Applied Mathematics, 1996.

[2] M. Schäfer and S. Turek. Benchmark computations of laminar flow around a cylinder. Preprint 96-03 (SFB 359), IWR, University of Heidelberg, 1996.

[3] W. Wendland, editor. *Boundary Element Methods 1989–1995*, Heidelberg, 1996. Reports from the Final Conference of the Priority Research Programme Boundary Element Methods 1989–1995, Stuttgart, October 1995, Springer Verlag.

A How to Get Files and News

- The FTP-server can be reached by
  ftp.mathematik.uni-stuttgart.de
  using the account ftp and password your e-mail. Change to the directory
  cd pub/projects/rem_benchmark
  and see the file README for more informations.

- There is a WWW-page by
  http://www.mathematik.uni-stuttgart.de/mathA/lst6/lehrsta6.html

- There is a mailing list. To become a member just mail to
  steinbach@mathematik.uni-stuttgart.de

- By postal mail:
  O. Steinbach, Mathematical Institute A, University of Stuttgart, Pfaffenwaldring 57, D-70569 Stuttgart, Germany.