Conf-9503116--8

LBL-36944
UC-600

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

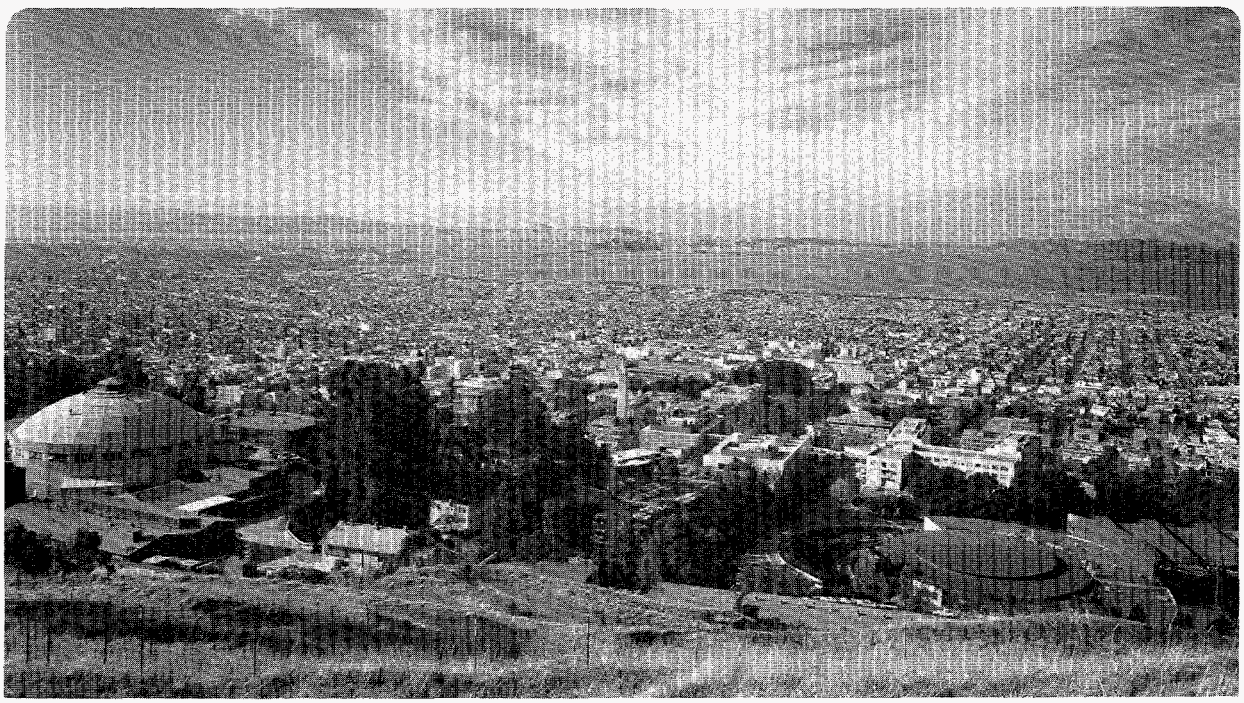
EARTH SCIENCES DIVISION

Presented at the TOUGH Workshop '95, Berkeley, CA,
March 20–22, 1995, and to be published in the Proceedings

On the Development of MP-TOUGH2

C.M. Oldenburg, R.L. Hinkins, G.J. Moridis, and K. Pruess

February 1995



Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

GH

LBL-36944
UC-600

On the Development of MP-TOUGH2

C.M. Oldenburg, R.L. Hinkins,* G.J. Moridis, and K. Pruess

Earth Sciences Division
*Information and Computing Sciences Division
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

February 1995

This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Geothermal Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

Presented at TOUGH Workshop '95
Lawrence Berkeley Laboratory
March 1995

LBL-36944

On the Development of MP-TOUGH2

C. M. Oldenburg, R. L. Hinkins*, G. J. Moridis, and K. Pruess

Earth Sciences Division
**Information and Computing Sciences Division*
Lawrence Berkeley Laboratory
University of California
Berkeley, CA 94720

ABSTRACT

We are developing MP-TOUGH2 for exploiting massively parallel computers. The goals of this effort are to (1) create a data-parallel subsurface transport code for solving larger problems than currently practical on workstations, (2) write portable code that can take advantage of scalability to run on machines with more processors, and (3) minimize the necessity for additional validation and verification of the resulting code. The initial strategy we have followed is to focus on optimizing the generic and time-consuming task of linear equation solution while leaving the bulk of TOUGH2 unmodified. In so doing, we have implemented a massively parallel direct solver (MPDS) that takes advantage of the banded structure of TOUGH2 Jacobian matrices. We have compared timings of the iterative conjugate gradient solvers DSLUBC, DSLUCS, and DSLUGM written in Fortran77 for the front end with the MPDS which uses the data parallel unit. The MPDS shows good performance relative to the iterative conjugate gradient solvers on our free-convection test problem. The robust direct solution provided by MPDS can be used to (1) check on the veracity of a given iterative conjugate gradient solution, or (2) be used on certain problems where iterative solvers fail to converge.

INTRODUCTION

One of the current challenges of scientific computing is the development of simulation codes that exploit massively parallel computers. Massively parallel computers have more than 1000 processors and are generally designed as single-instruction multiple data (SIMD) machines. Such computers can produce results faster and more efficiently using data-parallel methods supported by languages such as Fortran 90, where the same operations can be performed simultaneously on large arrays. Furthermore, Fortran 90 code is flexible because the same source code can be ported to other massively parallel computers with little or no modification. Given adequate memory resources, the maximum size of a tractable simulation problem scales with the number of processors on the available machine.

The promise of power and flexibility that data-parallel coding and massively parallel computers can theoretically bring to bear on simulation work must be tempered by the reality of the significant amounts of time and resources required to develop new simulation codes or rewrite existing codes. In the meantime, serial workstations and personal computers will become yet faster and cheaper, decreasing the potential return on investment in recoding. In addition, simulation codes such as TOUGH2 have received years of use and validation which have led to wide acceptance among users. New and rewritten code needs to be verified and validated and there is always considerable and justifiable resistance among users (and their customers) to use new codes.

The compelling potential of data-parallel coding coupled with the reality of scarce resources and uncertain return on investment has guided our initial development efforts. The strategy we have chosen to begin development of MP-TOUGH2, a massively-parallel version of TOUGH2, involves identifying time-consuming generic operations in TOUGH2 and exploiting optimized code to perform these tasks. The majority of computational effort in moderate- and large-size TOUGH2 problems goes to the solution of the system of linear equations involving a large, sparse, banded Jacobian matrix at each Newtonian iteration. This generic and time-consuming task is therefore the focus of our initial efforts.

In this report, we discuss our implementation of a massively parallel direct solver (MPDS) for the linear equation solution in TOUGH2. We use the 4096-processor MasPar MP-2204 at Lawrence Berkeley Laboratory, which consists of a front-end DEC Alpha 3000/300x and a 64 x 64 processor element array, also called the data-parallel unit (DPU). In keeping with the goal of minimizing recoding, we implemented MPDS as Fortran90 code that uses the DPU but is called from TOUGH2 compiled in Fortran77 on the front end. We compare timings of a two-dimensional natural convection test problem solved with MPDS running on the DPU and the same problem solved with iterative conjugate gradient solvers (Moridis and Pruess, 1995) running on the front end. We will show that the direct solver MPDS is competitive with the iterative conjugate gradient solvers on the test problem.

LINEAR EQUATION SOLUTION WITH MPDS

TOUGH2 generates a large and sparse banded Jacobian matrix at each Newtonian iteration which defines a set of linear equations to be solved for the changes in primary variables. Specifically, the equations

$$R_n^{(\kappa)k+1}(x_{i,p+1}) = R_n^{(\kappa)k+1}(x_{i,p}) + \sum_i \left.\frac{\partial R_n^{(\kappa)k+1}}{\partial x_i}\right|_p (x_{i,p+1} - x_{i,p}) \doteq 0 \qquad (1)$$

are solved where $R_n^{(\kappa)}$ is the residual for conservation of component κ in gridblock n, k is the time step index, x_i are the primary variables, and p is the index for the Newtonian iterations (Pruess, 1987; 1991). In a problem with NEL gridblocks and NEQ equations per gridblock, the Jacobian matrix $\{\partial R_n^{(\kappa)k+1}/\partial x_i\}$ is of order N = NEL x NEQ. It is this linear equation solution that consumes the majority of computation time for any moderate- or large-size TOUGH2 simulation.

We have implemented a banded direct solver provided to us by MasPar that takes advantage of library routines optimized for the DPU on the MP-2204. The MPDS is written in Fortran90 and uses the DPU. Meanwhile MP-TOUGH2 is compiled in Fortran77 to run on the DecAlpha 3000/300x front end, calling the Fortran90 MPDS for linear equation solution. In this way, neither TOUGH2 nor the iterative conjugate gradient solvers has been modified except for adding the option of calling the MPDS.

For simplicity in discussing the MPDS, we write the linear matrix equation (1) in the idealized form Ax = b, where A is the Jacobian matrix of order N, x is the vector containing the changes in primary variables, and b is the vector of residuals. The MPDS works by decomposing A into L(ower) and U(pper) where A = L U and then solving L (U x) = b. This decomposition and solution can be applied to any general matrix A. The Jacobian matrix in TOUGH2 is sparse and banded, with its non-zero elements near the

main diagonal, i.e. a(i,j) = 0 for abs(i - j) > m, where the bandwidth is 2m + 1. One can take advantage of this structure and modify the matrix storage scheme to achieve greater efficiency on a serial machine if m < N/3 (Dongarra et al., 1979). On a parallel SIMD machine with distributed memory, we can get parallel computation from the linear algebra library routines (MasPar Mathematics Library, 1993) by modifying the band storage scheme so that the non-zeroes in the band are submatrices of order N_S, where N_S is called the slab width. Each submatrix is distributed across the processors with one matrix element per processor. We choose N_S = 64 to optimize the mapping of the submatrices to the MP-2204 which has processor elements laid out on a 64 x 64 grid. The slab width parameter is generally chosen to match the processor array on the target machine, thus ensuring scalability for larger problems running on larger machines. The solve time depends on problem-dependent parameters including the matrix order and the bandwidth as well as the submatrix size.

RESULTS

The test problem used in this study is the Elder pure thermal convection problem (Elder, 1967). For generality, we use the TOUGH2 equation of state module EOS3 for water, air, and heat, although this particular problem is a single-phase flow problem. In prior work we used EOS7 and adapted the problem to pure solutal convection with a salt source at the top (Oldenburg and Pruess, 1994). Whether solutal or thermal convection is considered, the large maximum density change (20%) makes this a strongly coupled flow problem. Elder's actual laboratory apparatus was a Hele-Shaw cell 5 cm by 20 cm, but we use a two-dimensional porous medium with physical dimensions and parameters scaled for similarity to Elder's work. These parameters are presented in Table 1. Note that we have used heat capacity of zero for the matrix rock for similarity to pure solutal convection. The domain, boundary conditions, and coarsest discretization (60 x 32) for the problem are presented in Fig. 1. The flow and temperature fields at t = 2 yrs. are shown in Fig. 2. For a more complete discussion and analysis of the Elder free convection problems, see Oldenburg and Pruess (1994, 1995).

We have used 3 discretizations for the calculation time comparison of the various solvers. These were 60 x 32 gridblocks, 84 x 40 gridblocks, and 128 x 64 gridblocks in the Y- and Z-directions, respectively; NX is always 1 in this two-dimensional problem. With three equations per gridblock, these discretizations give rise to Jacobian matrices of order (N) equal to 5760, 10080, and 24576, respectively. Non-zero elements of the Jacobian matrix occur in bands of width 197, 245, and 389, for the three discretizations.

Given the many ways one can time multi-processor computations, we chose a simple method that we felt would make sense to users experienced with serial workstations. The timings presented are calculation times, as obtained by the TOUGH2 subroutine SECOND, divided by the number of linear equation solutions performed. Thus the time is an average time for one Newtonian iteration including the small overhead from other subroutines such as MULTI and EOS, among others. Timings of the solution of the Elder problem with three different two-dimensional discretizations are presented in Fig. 3.

We see in Fig. 3 that the MPDS running on the DPU is faster than the conjugate gradient solver DSLUGM running on the DecAlpha 3000/300x front end for all three discretizations. The performance of DSLUGM on this test problem degrades sharply as the order of the Jacobian matrix increases. The MPDS is about 30% slower than the conjugate gradient solvers DSLUBC and DSLUCS for the largest example problem. It is

important to emphasize that MPDS is a direct solver and is inherently more robust than iterative solvers. As such, it will be valuable as a tool to check upon the veracity of a given conjugate gradient solution, and may prove indispensible for certain difficult problems where the iterative conjugate gradient solvers may converge poorly.

Among the limitations of the MPDS on the MP-2204 are the memory restrictions of 64 kbytes per processor, and a 64 x 64 processor array. This memory limit affects calculation speed based upon both the bandwidth and the order of the matrix for a given problem. Another limitation is the relatively slow double-precision floating point operation on the MasPar processors. Communication between memory and processors is another factor affecting speed in parellel computing. For example, each processor element on the MP-2204 has its own local memory and sharing data between processors or between processors and the front end requires communication that can take considerable time. Nevertheless, the maximum size problem MPDS can handle will scale with the machine. MPDS will require no recoding to run larger problems on machines with more processors.

CONCLUSIONS

Guided by the considerable potential benefits and risks of developing a massively parallel subsurface simulation code, we have initiated development of a massively parallel version of TOUGH2 called MP-TOUGH2. We implemented the massively parallel direct solver MPDS to perform the generic and time-consuming task of linear equation solution in MP-TOUGH2. In this initial code, the essentially unmodified TOUGH2 is compiled in Fortran77 and runs on the front end. This code calls the MPDS which is written and compiled in Fortran90 and uses the DPU. We tested the MPDS on a strongly coupled two-dimensional pure thermal convection problem. Comparison of calculation times shows the MPDS to be competitive with the iterative conjugate gradient solvers on the front end. Because MPDS is a direct solver, it will be useful for establishing correct solutions and verifying conjugate gradient solver solutions. In addition, it may find application in solving problems where the iterative solvers coverge slowly, or fail. Future work will focus on testing and improving the MPDS as well as targeting other TOUGH2 tasks where data-parallelism can be exploited.

Acknowledgments

We thank the MasPar Computer Corporation for their generous contributions and continued support, and Frank Hale and Stefan Finsterle for reviews. This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Geothermal Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

Nomenclature

C	heat capacity	$J\,kg^{-1}\,°C^{-1}$
g	gravitational acceleration	$m\,s^{-2}$
k	permeability	m^2
K	thermal conductivity	$J\,s^{-1}\,m^{-1}\,°C^{-1}$
N_S	slab width	
R	residual	kg

t	time	years
T	temperature	°C
X	X-coordinate	m
Y	Y-coordinate	m
Z	Z-coordinate (positive upward)	m

Greek symbols

μ	dynamic viscosity	kg m^{-1} s^{-1}
ϕ	porosity	-
ρ	density	kg m^{-3}

Subscripts and superscripts

k	time step index
n	grid block index
p	Newtonian iteration index
κ	mass components

REFERENCES

Dongarra, J.J., C.B. Moler, J.R. Bunch, and G.W. Stewart, *LINPACK Users' Guide*, pg. 2.1, Society of Industrial and Applied Mathematics (SIAM), Philadelphia, 1979.

Elder, J.W., Transient convection in a porous medium, *J. Fluid Mech.*, *27*(3), 609–623, 1967.

MasPar Mathematics Library, *Document Part Number 9302-5001*, MasPar Computer Corporation, 749 North Mary Ave., Sunnyvale CA 94086, 1993.

Moridis, G.J., K. Pruess, T2CG1, A package of preconditioned conjugate gradient solvers for the TOUGH2 family of codes, Lawrence Berkeley *Laboratory Report, LBL-36235*, Berkeley, California, March 1995.

Oldenburg, C.M. and Pruess, K., Numerical simulation of coupled flow and transport with TOUGH2: A verifictation study, *Lawrence Berkeley Laboratory Report*, *LBL-35273*, Berkeley, California, October 1994).

Oldenburg, C.M. and Pruess, K., Dispersive transport dynamics in a strongly coupled groundwater-brine flow problem, *Water Res. Res, 31*(2), 289–302, 1995.

Pruess, K., TOUGH User's Guide, Nuclear Regulatory Commission, Report NUREG/CR-4645, June 1987 (also *Lawrence Berkeley Laboratory Report*, *LBL-20700*, Berkeley, California, June 1987).

Pruess, K., TOUGH2 - A General Purpose Numerical Simulator for Multiphase Fluid and Heat Flow, *Lawrence Berkeley Laboratory Report*, *LBL-29400*, Berkeley, California, May 1991).

Table 1. Parameters for the Elder (1967) pure thermal convection problem.

symbol	quantity	value	units
ϕ	porosity	.1	–
k	permeability	1.21×10^{-10}	m^2
μ	viscosity ($T = 20°C$)	1.0×10^{-3}	Pa s
K	thermal conductivity	1.49	$J\,kg^{-1}\,s^{-1}\,m^{-1}$
C	heat capacity of rock	0.	$J\,kg^{-1}\,°C^{-1}$
g	gravity	9.81	$m\,s^{-2}$
T_0	initial temperature	12.	°C

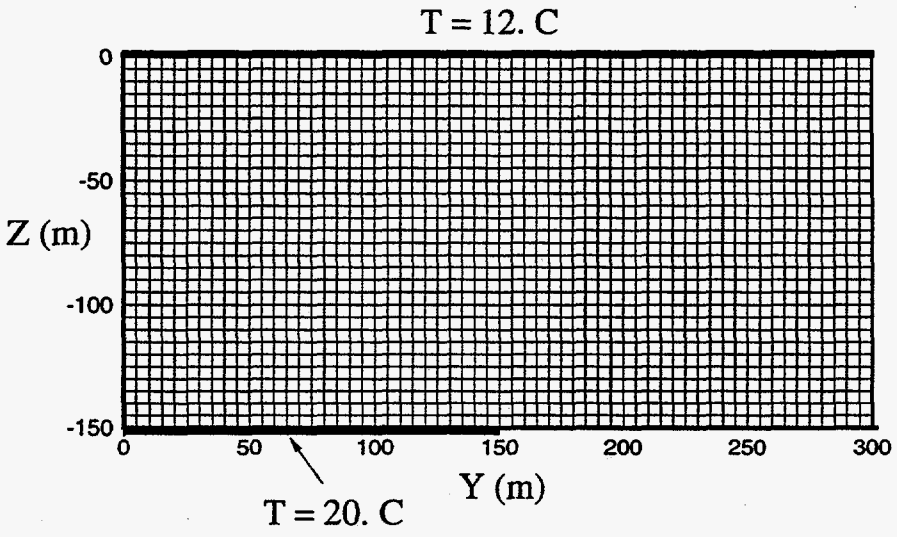


Figure 1. Domain and boundary conditions for the Elder pure thermal convection problem with the 60 x 32 gridblock mesh.

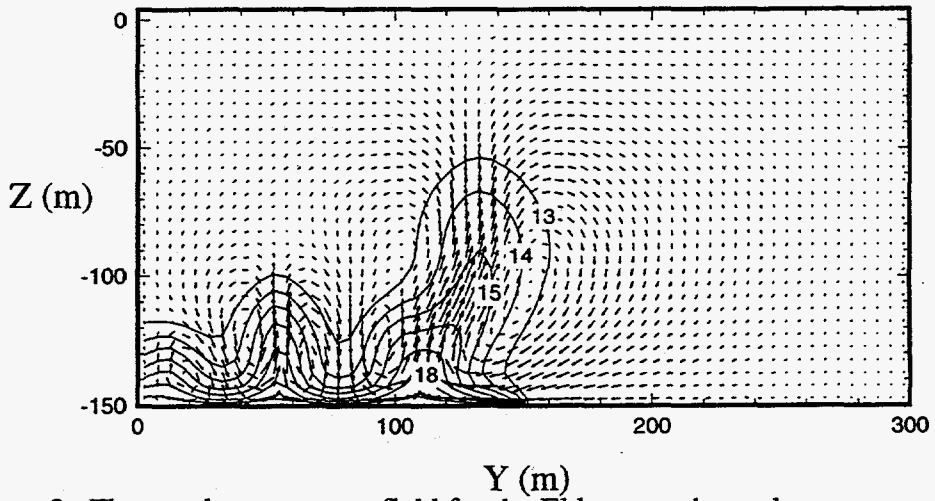


Figure 2. Flow and temperature field for the Elder pure thermal convection problem at $t = 2$ yrs.

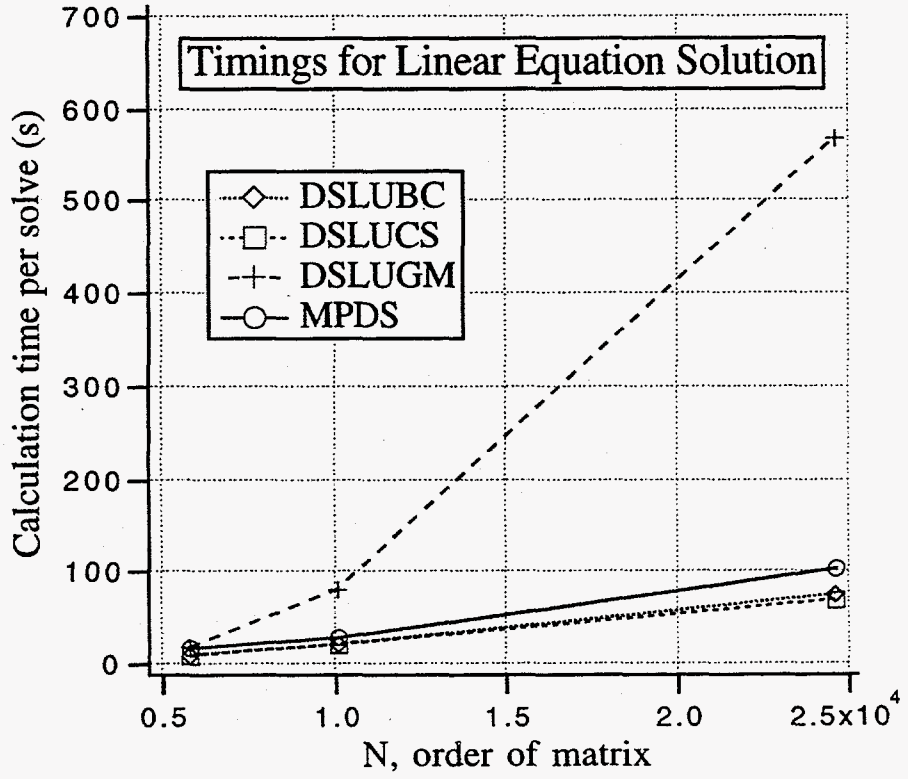


Figure 3. Calculation time plotted against N, the order of the Jacobian matrix, for three conjugate gradient solvers (DSLUBC, DSLUCS, DSLUGM) and for the massively parallel direct solver (MPDS).