Parallel Solution of Contact Problems*

Z. Dostál † F. A. M. Gomes ‡ S. A. Santos‡

March 22, 2000

Abstract

An efficient non-overlapping domain decomposition algorithm of the Neumann-Neumann type for solving both coercive and semicoercive frictionless contact problems of elasticity has been recently presented. The method reduces, by the duality theory of convex programming, the discretized problem to a quadratic programming problem with simple bounds and equality constraints on the contact interface. This dual problem is further modified by means of orthogonal projectors to the natural coarse space, and the resulting problem is solved by an augmented Lagrangian type algorithm. The projectors guarantee an optimal rate of convergence for the solution of auxiliary linear problems by the conjugate gradients method. With this approach, it is possible to deal separately with each body or subdomain, so that the algorithm can be implemented in parallel. In this paper, an efficient parallel implementation of this method is presented, together with numerical experiments that indicate the high parallel scalability of the algorithm.

Key words: Parallel algorithms, domain decomposition, contact problems.

*Research supported by FAPESP 97-12676-4, CNPq, FINEP, FAEP-UNICAMP and by grants CAČR 201/97/0421 and 101/98/0535.

†Department of Applied Mathematics, VŠB-Technical University Ostrava, Ostrava, Czech Republic (zdenek.dostal@vsb.cz).

‡Department of Applied Mathematics, IMECC – UNICAMP, University of Campinas, CP 6065, 13081–970 Campinas SP, Brazil (chico@ime.unicamp.br, sandra@ime.unicamp.br).

1 Introduction.

Duality based domain decomposition methods proved to be practical and efficient tools for parallel solution of large elliptic boundary value problems [15, 16, 23]. Using this approach, a body is partitioned into non-overlapping subdomains, for each subdomain is defined an elliptic problem with Neumann boundary conditions on the subdomain interfaces, and intersubdomain field continuity is enforced via Lagrange multipliers. The Lagrange multipliers are evaluated by solving a relatively well conditioned dual problem of small size that may be efficiently solved by a suitable variant of the conjugate gradient algorithm. The first practical implementations by Farhat and Roux [15, 16] exploited the favorable distribution of the spectrum of the matrix of the smaller problem [22], known also as the dual Schur complement matrix, being an efficient algorithm only with a small number of subdomains. Later, they introduced a "natural coarse problem" whose solution was implemented by auxiliary projectors so that the resulting algorithm became optimal [17, 23]. Recently, the authors have shown how to use the "natural coarse grid" to the solution of a scalar variational inequality [11] and presented an efficient non-overlapping domain decomposition algorithm for solving both coercive and semicoercive frictionless contact problems of elasticity [12].

In this work, we focus on the computational implementation of parallel solution of contact problems. The parallelization is described, analysed and tested for a model problem.

This paper is organized as follows: in Section 2 we present the discretized problem formulation, from the primal to the modified dual by means of preconditioning with projectors. For completeness, in Section 3 we briefly describe the adopted quadratic programming algorithm, based on the augmented Lagrangian technique and adaptive precision control for solving auxiliary problems. In Section 4, the model problem is described, together with its domain decompositions. In Sections 5 and 6, we present a profile of the algorithm and the parallelization scheme, respectively. Numerical results are shown and discussed in Section 7. Finally, some conclusions and future perspectives are presented in Section 8.

2 Problem Formulation.

We consider a domain Ω defined by s homogeneous isotropic elastic bodies in contact, each one occupying, in a reference configuration, a subdomain $\Omega_p \subset I\!R^d, d = 2, 3$, with sufficiently smooth boundary. Imposing equilibrium conditions, after finite element discretization of $\Omega = \Omega_1 \cup \cdots \cup \Omega_s$, with a suitable numbering of nodes and assuming a secondary decomposition, we obtain the quadratic programming problem:

$$\min \quad \frac{1}{2}u^T K u - f^T u \quad \text{s.t.} \quad B_I u \le c \quad \text{and} \quad B_E u = 0, \tag{1}$$

where $K \in I\!R^{n\times n}$ is symmetric positive definite (or semidefinite) block diagonal (i.e. $K = \mathrm{diag}(K_1, \dots, K_s)$), $B_I \in I\!R^{m\times n}$ and $B_E \in I\!R^{\ell\times n}$ are full rank matrices, $f \in I\!R^n$ and $c \in I\!R^m$. The matrix B_I and the vector c describe the linearized incremental non-interpenetration conditions, whereas matrix B_E ensures continuity of the displacements across auxiliary interfaces. For more details, see [10, 12]. The vector f describes the nodal forces arising from the volume forces and/or some other imposed tractions. Typically n is large and m, ℓ are much smaller than n. The diagonal blocks K_p that correspond to subdomains Ω_p are positive definite or semidefinite sparse matrices. Moreover, we shall assume that the nodes of the discretization are numbered in such a way that K_p are banded matrices that can be effectively decomposed, possibly after some regularization, by means of the Cholesky factorization.

Even though (1) is a standard convex quadratic programming problem, its formulation is not suitable for numerical solution. The reasons are that matrix K is typically ill conditioned, possibly singular and the feasible set is in general so complex that projections into it can hardly be effectivelly computed. Such difficulties may be essentially reduced by applying the duality theory of convex programming (e.g. [5, 6, 10]). Since the regular case has already been discussed [10] we shall assume that the matrix K has a nontrivial null space that defines the natural coarse grid ([23]).

The Lagrangian associated with problem (1) is

$$L(u, \lambda_I, \lambda_E) = \frac{1}{2}u^T K u - f^T u + \lambda_I^T (B_I u - c) + \lambda_E^T B_E u, \tag{2}$$

where λ_I and λ_E are the Lagrange multipliers associated with inequalities and equalities, respectively. Introducing notation

$$\lambda = \begin{bmatrix} \lambda_I \\ \lambda_E \end{bmatrix}, \quad B = \begin{bmatrix} B_I \\ B_E \end{bmatrix}, \quad \text{and} \quad \hat{c} = \begin{bmatrix} c \\ 0 \end{bmatrix},$$

we can write the Lagrangian briefly as

$$L(u,\lambda) = \frac{1}{2}u^T K u - f^T u + \lambda^T (Bu - \hat{c}).$$

Problem (1) is equivalent to the saddle point problem

$$\text{Find} \quad (\overline{u}, \overline{\lambda}) \quad \text{such that} \quad L(\overline{u}, \overline{\lambda}) = \sup_{\lambda_I \geq 0} \inf_u L(u,\lambda). \tag{3}$$

By eliminating u from (3) we obtain

$$\min \quad \Theta(\lambda) \quad \text{s.t.} \quad R^T(f - B^T\lambda) = 0 \quad \text{and} \quad \lambda_I \geq 0 \tag{4}$$

where

$$\Theta(\lambda) = \frac{1}{2}\lambda^T B K^\dagger B^T \lambda - \lambda^T (B K^\dagger f - \hat{c}), \tag{5}$$

R is a matrix whose columns span the null space of K and $K^\dagger$ denotes any matrix that satisfies $KK^\dagger K = K$. The essential fact is that the product of $K^\dagger$ by a vector should be effectivelly carried out (see e.g. [11, 14]). Once the solution λ of (4) is obtained, the vector u that solves (3) can be evaluated by an explicit formula (see [6, 10]).

The Hessian of Θ is, under reasonable assumptions, positive definite. Besides, it is closely related to that of the basic FETI method by Farhat and Roux [15, 16], so that its spectrum is relatively favorably distributed for application of the conjugate gradient method [22].

Even though problem (4) is much more suitable for computations than (1) and was used for efficient solution of contact problems [10], further improvement may be achieved by adapting the results of [17]. Let us denote $F = BK^\dagger B^T$, $\widetilde{d} = BK^\dagger f$, $\widetilde{G} = R^T B^T$, $\widetilde{e} = R^T f$ and let T denote a regular matrix that defines the orthonormalization of the rows of $\widetilde{G}$ so that matrix $G = T\widetilde{G}$ has orthogonal rows. After denoting $e = T\widetilde{e}$, problem (4) reads

$$\min \quad \frac{1}{2}\lambda^T F \lambda - \lambda^T \widetilde{d} \quad \text{s.t.} \quad G\lambda = e \quad \text{and} \quad \lambda_I \geq 0. \tag{6}$$

Next, the equality constraints may be homogeneized by means of an arbitrary $\overline{\lambda}$ that satisfies $G\overline{\lambda} = e$. Denoting $d = \widetilde{d} - F\overline{\lambda}$, the modified problem reads

$$\min \quad \frac{1}{2}\lambda^T F \lambda - \lambda^T d \quad \text{s.t.} \quad G\lambda = 0 \quad \text{and} \quad \lambda_I \geq -\overline{\lambda}_I. \tag{7}$$

Further improvement can be obtained based on the decomposition of the augmented Lagrangian for problem(7) by the orthogonal projectors $Q = G^T G$ and $P = I - Q$ on the image space of G^T and on the kernel of G, respectively. Indeed, since $P\lambda = \lambda$ for any feasible λ, problem (7) is equivalent to

$$\min \quad \frac{1}{2}\lambda^T PFP\lambda - \lambda^T Pd \quad \text{s.t.} \quad G\lambda = 0 \quad \text{and} \quad \lambda_I \geq -\overline{\lambda}_I \tag{8}$$

and the Hessian $H = PFP + \rho Q$ of the augmented Lagrangian

$$L(\lambda, \mu, \rho) = \frac{1}{2}\lambda^T (PFP + \rho Q)\lambda - \lambda^T Pd + \mu^T G\lambda \tag{9}$$

is decomposed by projectors P and Q whose image spaces are invariant subspaces of H. The analysis of Axelsson [2] and Dostál [8] together with results of the FETI method [17] provide ingredients to show that the rate of convergence for unconstrained minimization of the augmented Lagrangian (9) depends on neither the penalization parameter ρ nor the discretization parameter. In fact, provided the aspect ratios of both discretization and decomposition are close to one, the number of conjugate gradient iterations is bounded by the square root of the ratio between subdomain and mesh diameters (see [11, 12]).

3 Algorithm for Quadratic Programming with Equality Constraints and Simple Bounds.

Our development of an efficient algorithm for the solution of (8) is based on the observation that the solution of such problem may be reduced, by the augmented Lagrangian technique [4, 9], to the solution of a sequence of quadratic programming (QP) problems with simple bounds, and that the latter can be solved much more efficiently than more general QP problems due to the possibility of using projections and results on adaptive precision control in the active set strategy [3, 7, 18, 19, 20]. Here, for completeness, we briefly describe the QP algorithm proposed in [9], conveniently adjusted to problem (8).

To simplify our notation, let us denote $F_P = PFP$ so that the augmented Lagrangian for problem (8) and its gradient are given by

$$L(\lambda, \mu, \rho) = \frac{1}{2}\lambda^T F_P \lambda - \lambda^T Pd + \mu^T G\lambda + \frac{1}{2}\rho||Q\lambda||^2$$

and

$$g(\lambda, \mu, \rho) = F_P\lambda - Pd + G^T(\mu + \rho G\lambda),$$

respectively. The *projected gradient* $g^P = g^P(\lambda, \mu, \rho)$ of L at λ is then given componentwise by

$$g_i^P = g_i \text{ for } \lambda_i > -\overline{\lambda}_i \text{ or } i \notin I \text{ and } g_i^P = g_i^- \text{ for } \lambda_i = -\overline{\lambda}_i \text{ and } i \in I$$

with $g_i^- = \min(g_i, 0)$, where I is the set of indices of constrained entries of λ.

The algorithm that we describe here may be considered a variant of the one proposed by Conn, Gould and Toint [4] for identification of stationary points of more general problems. However, Algorithm 3.1 is modified to exploit the specific structure of our problem and get improved performance. The most important of such modifications consists in including the adaptive precision control of auxiliary problems in Step 1.

All the parameters that must be defined prior to the application of the algorithm are listed in Step 0, with typical values for our model problem given in section 7.

Algorithm 3.1. (Simple bounded variables and equality constraints)

Step 0. *Initialization of parameters.* Set $0 < \alpha < 1$ for equality precision update, $1 < \beta$ for penalty update, $\rho_0 > 0$ for initial penalty parameter, $\eta_0 > 0$ for initial equality precision, $M > 0$ for balancing ratio, $\varepsilon > 0$ for optimality precision, μ^0 for the Lagrangian multipliers and $k = 0$.

Step 1. Find λ^k so that $||g^P(\lambda^k, \mu^k, \rho_k)|| \leq M||G\lambda^k||$, by solving

$$\min \quad L(\lambda, \mu, \rho) \quad \text{s.t.} \quad \lambda_I \geq -\overline{\lambda}_I.$$

Step 2. If $||g^P(\lambda^k, \mu^k, \rho_k)|| \leq \varepsilon||d||$ and $||G\lambda^k|| \leq \varepsilon||f||$

then λ^k is the solution.

Step 3. If $||G\lambda^k|| \leq \eta_k$

Step 3a. then $\mu^{k+1} = \mu^k + \rho_k G\lambda^k$, $\rho_{k+1} = \rho_k$, $\eta_{k+1} = \alpha\eta_k$

Step 3b. else $\rho_{k+1} = \beta\rho_k$, $\eta_{k+1} = \eta_k$

end if.

Step 4. Increase k and return to Step 1.

The implementation of Step 1 may be carried out by means of any algorithm for quadratic minimization with simple bounds (e.g. [3, 7, 18, 19, 20]). The unique solution $\widetilde{\lambda} = \widetilde{\lambda}(\mu, \rho)$ of this auxiliary problem satisfies the Karush-Kuhn-Tucker conditions $g^P(\widetilde{\lambda}, \mu, \rho) = 0$.

Salient features of this algorithm are that it deals completely separately with each type of constraint and that it accepts inexact solutions of the auxiliary box constrained problems in Step 1. Algorithm 3.1 has been proved to converge for any set of parameters that satisfy the relations prescribed at Step 0 (see [9]). Moreover, the penalty parameter is uniformly bounded and the asymptotic rate of convergence is the same as for the algorithm with exact solution of auxiliary quadratic programming problems (i.e. $M = 0$).

4 A Model Problem and its Domain Decomposition.

We consider the model problem that comes from the finite difference discretization of the following continuous problem

$$\begin{aligned} &\text{Minimize} \quad q(u_1, u_2) = \sum_{i=1}^{2} \left(\int_{\Omega_i} |\nabla u_i|^2 d\Omega - \int_{\Omega_i} f u_i d\Omega \right) \\ &\text{subject to} \quad u_1(0, y) \equiv 0 \text{ and } u_1(1, y) \leq u_2(1, y) \text{ for } y \in [0, 1], \end{aligned}$$

where $\Omega_1 = (0,1) \times (0,1)$, $\Omega_2 = (1,2) \times (0,1)$, $f(x,y) = -5$ for $(x,y) \in (0,1) \times [0.75, 1)$, $f(x,y) = 0$ for $(x,y) \in (0,1) \times (0, 0.75)$, $f(x,y) = -1$ for $(x,y) \in (1,2) \times (0, 0.25)$ and $f(x,y) = 0$ for $(x,y) \in (1,2) \times (0.25, 1)$.

The solution $u \equiv (u_1, u_2)$ of the model problem may be interpreted as the displacement of two membranes under the traction f, as shown in Figure 1. The left membrane is fixed on the left and the left edge of the right membrane is not allowed to penetrate below the edge of the left membrane. Moreover, both membranes are stretched by normalized horizontal forces. This problem is semicoercive due to the lack of Dirichlet data on the boundary of Ω_2, but the solution is unique because the right membrane is pressed down. More details about this model problem, including some other results, may be found in [11, 12].

The model problem was discretized by regular grids defined by the stepsize $h = 1/n$ with $n + 1$ nodes in each direction per subdomain Ω_i, $i = 1, 2$.

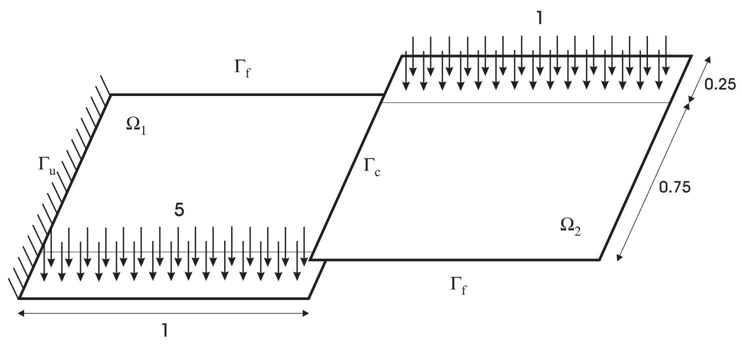

Figure 1: Model problem.

Each subdomain Ω_i was decomposed into $n_x \times n_y$ identical rectangles with dimensions $H_x = 1/n_x$ and $H_y = 1/n_y$. According to the values of n_y, we may have a decomposition into *strips* ($n_y = 1$) or into a *chessboard* pattern ($n_y > 1$).

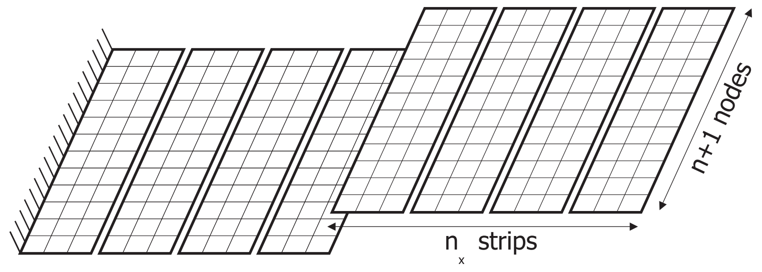

Figure 2: Decomposition into strips.

5 A Profile of the Algorithm.

Applying duality theory to (1) greatly reduces the dimension of the problem. In fact, it can be shown that the dimension of the dual problem is $O(nn_x)$, while the primal dimension is $O(n^2)$.

Since the size of the dual problem may be still considerable large, a very efficient way to carry out step 1 of Algorithm 3.1 consists in applying a conjugate gradient type method to solve the bound-constrained quadratic problem using a modified lumped preconditioner in the form $C^{-1} = PBKB^TP + (1/\rho)G^TG$ to accelerate the convergence. In this case, the product of $F = BK^{\dagger}B^T$ by a vector has to be computed at least once per iteration. Observing that F includes $K^{\dagger}$, the generalized inverse of the primal stiffness matrix, one can conclude that this product might dominate the overall time of Algorithm 3.1.

Fortunately, the product of $K^{\dagger}$ by a vector can be efficiently performed in parallel, since this matrix is block-diagonal, with $2n_x$ blocks for the decomposition into strips and $2n_xn_y$ blocks if the chessboard decomposition is used. Moreover, for our semicoercive model problem, $K^{\dagger}$ contains only two different banded blocks, so storing this matrix is not a main concern even if a distributed parallel environment is used.

Besides the product of $K^{\dagger}$ by a vector, other relevant steps of the computation are

- The Cholesky decomposition of the two distinct diagonal blocks of K, used to compute $K^{\dagger}$ times a vector.
- The generation of matrix $G = T\widetilde{G}$. In our implementation, this matrix is obtained from the thin QR decomposition of a "condensed" version of $\widetilde{G}^T$ (see [13]).
- The product of $F = BK^{\dagger}B^T$ by a vector.
- The products of G and G^T by a vector.

To obtain an efficient implementation of Algorithm 3.1, it is important to minimize the time spent on generating matrices $K^{\dagger}$ and G, since this generation involves matrix decompositions that are difficult to parallelize.

For the model problem, the percentage of the total time that is spent in computing each of the steps described above is presented in Table 1. The chessboard decomposition was defined by setting $n_y = n_x$. Missing results correspond either to problems that are too large for the available memory or problems so small that the results provided by the profiler were not reliable.

Table 1 indicates that, for the decomposition into strips, the generation of $K^{\dagger}$ is much more expensive than the generation of G, so this last product may be neglected. Besides, the problem can be solved more efficiently in

Table 1.
Percentage of the time spent by each routine.

n	n_x	decomposition into strips					chessboard decomposition				
		chol $K^\dagger$	*qr* $\widetilde{G}^T$	Fv	Gv, G^Tv	*other*	*chol* $K^\dagger$	*qr* $\widetilde{G}^T$	Fv	Gv, G^Tv	*other*
32	2	15.6	0.0	82.9	0.2	1.3	–	–	–	–	–
	4	4.5	0.5	89.5	2.2	3.3	–	–	–	–	–
	8	2.1	1.0	91.2	2.6	3.1	–	–	–	–	–
64	2	27.2	0.1	72.3	0.1	0.3	5.2	0.2	93.0	0.8	0.8
	4	12.8	0.0	86.2	0.5	0.5	0.6	0.4	90.9	4.7	3.4
	8	4.8	0.1	93.1	0.9	1.1	0.1	14.5	63.5	14.7	7.2
	16	1.4	0.0	95.6	1.3	1.7	0.0	85.6	7.3	5.1	2.0
	32	0.3	0.0	94.3	2.1	3.3	0.0	98.9	0.4	0.6	0.1
128	2	30.8	0.0	69.1	0.0	0.1	5.2	0.2	93.0	0.8	0.8
	4	13.7	0.0	86.0	0.1	0.2	0.8	0.1	96.8	1.4	0.9
	8	5.4	0.0	94.0	0.3	0.3	0.1	2.0	89.5	5.0	3.4
	16	1.8	0.0	97.4	0.4	0.4	0.0	64.6	25.6	6.4	3.4
	32	0.5	0.0	97.8	0.7	1.0	0.0	98.2	0.8	0.8	0.2
256	4	16.2	0.0	83.1	0.0	0.7	0.4	0.1	98.4	0.8	0.3
	8	6.8	0.0	93.1	0.0	0.1	0.2	0.4	96.7	1.7	1.0
	16	2.5	0.0	97.1	0.1	0.3	0.0	27.0	65.9	4.4	2.7
	32	0.6	0.0	98.9	0.3	0.2	0.0	95.6	3.0	1.1	0.3
512	4	–	–	–	–	–	2.3	0.0	97.4	0.1	0.2
	8	–	–	–	–	–	0.5	0.0	98.7	0.7	0.1
	16	–	–	–	–	–	0.0	4.3	92.7	1.8	1.2
1024	16	–	–	–	–	–	0.4	0.1	98.4	0.8	0.3

parallel if the number of strips is large, since the dimension of the diagonal blocks of $K^\dagger$ is inversely proportional to the number of strips. Thus, for a reasonably large value of r, the Cholesky decomposition of $K^\dagger$ is almost inexpensive and only the product of $BK^\dagger B^T$ by a vector needs to be performed in parallel.

For the chessboard decomposition, the time spent computing the QR decomposition of $\widetilde{G}^T$ increases exponentially as we increase the number of subdomains, since the number of columns of $\widetilde{G}^T$ is proportional to $n_x n_y$. In this case, some care must be taken in order to prevent the QR decomposition from dominating the overall time spent by the algorithm because this decomposition cannot be efficiently parallelized. Fortunately, Table 1 suggests that this can be accomplished by keeping ratios n_x/n and n_y/n small. For the model problem, supposing that $n_x = n_y$, this means that the relation

$n_x \leq \sqrt{n/2}$ must hold. On the other hand, if n_x and n_y are too small, the decomposition of $K^\dagger$ may reduce the efficiency of the parallelization, so it is important to choose these parameters very carefully.

6 The Parallel Scheme.

For the model problem, a careful choice of n_x and n_y may ensure that more than 95 percent of the total time of the algorithm will be spent on computing the product of $BK^\dagger B^T$ by a vector.

If the decomposition into strips is used, almost all of the remaining time is spent in computing the Cholesky factors of matrix $K^\dagger$, which means that no other part of the algorithm can be efficiently parallelized. For the chessboard decomposition, though, it is worth considering computing in parallel the product of the entire Hessian $H = PFP + \rho Q$ by a vector, as the products of $Q = G^T G$ or $P = I - Q$ by a vector are also easily performed in parallel.

In this paper, however, we will illustrate the parallel solution of the model problem using only the decomposition into strips. Therefore, we restrict our attention to the computation of product

$$y = BK^\dagger B^T v. \tag{10}$$

Product (10) can be decomposed into three parts. First, v is "expanded" and stored in a vector z with the same dimension as K. Then $K^\dagger z$ is obtained using the Cholesky factors previously computed. Finally, the resulting vector is compressed to fit in y.

The way this product is computed in parallel depends on the computational model used. In our code, the SPMD (single program, multiple data) model was adopted, which means that the same program is executed by all of the n_{proc} processors. Besides, MPI was chosen as the communication library.

Since vector v is available to all of the processors, each one can pick one part of the vector, expand it, compute the effect of the corresponding diagonal blocks of $K^\dagger$ on it and compress the resulting vector into y. At the end of this procedure, each processor stores a small portion of y, so it is necessary to gather all these parts up and distribute y to all of the processors in order to resume the algorithm. Fortunately, this is the only communication point of the entire algorithm and can be efficiently implemented using routine `MPI_AllGatherV` from MPI. However, some care must be taken when gathering y, since some of the $(2n_x - 1)n$ elements of this vector belong to the interface of two strips and receive contributions from different blocks of $K^{\dagger}Bv$. To circumvent this problem, a vector with $(2n_x + n_{proc} - 2)n$ elements is used to store all of the n_{proc} parts in which y was divided. After being distributed, this larger vector is compressed by each processor and y is finally generated.

7 Numerical Results.

To evaluate the behavior of our parallel algorithm, a FORTRAN code was written. All of the tests were performed on a SGI Origin 2000 shared memory computer, with 4 processors, using MPICH, a portable implementation of MPI developed jointly by the Argonne National Laboratory and the Mississippi State University.

The model problem was solved for a variety of values of n and n_x in order to test experimentally the dependence of the rate of convergence on the discretization parameter. The bound constrained quadratic solver described in [3] was used to compute Step 1. The numerical data used were $\alpha = 0.1$, $\beta = 10$, $\rho_0 = 10^4$, $\eta_0 = 0.1$, $M = 10^4$, $\varepsilon = 10^{-5}$ and $\mu^0 = 0$.

The parallel algorithm attained the same precision as the sequential one. Moreover, both performed the same number of iterations and matrix-vector products. Figure 3 exhibits a typical solution for the strips decomposition.

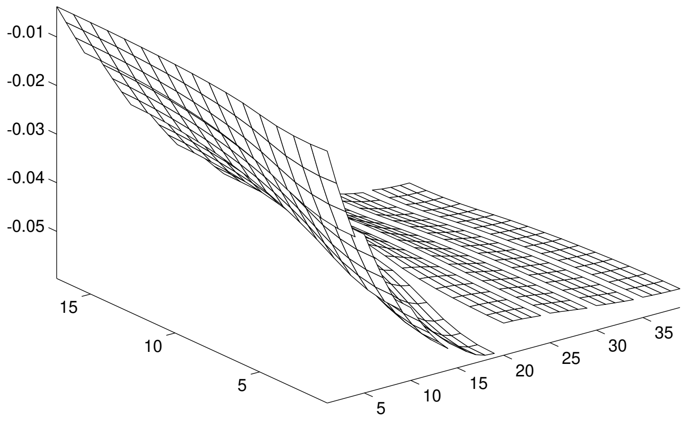

Figure 3: Typical solution.

The performance of the parallel algorithm is shown in Table 2, where S_p and S_a denote, respectively, the "predicted" and the speedup actually obtained in the experiments. For predicted speedup we mean the speedup that could be obtained if the time spent on computing (10) using n_{proc} processors was the time spent by one processor divided by n_{proc}, i.e. in the absence of communication costs. We also include the ratio S_a/S_p as a measure of the efficiency of our parallel implementation.

Table 2.
Parallel performance of the algorithm.

n	n_x	$n_{proc}=4$			$n_{proc}=2$		
		S_a	S_p	S_a/S_p	S_a	S_p	S_a/S_p
64	4	2.43	2.83	0.86	1.62	1.76	0.92
	8	2.79	3.31	0.84	1.75	1.87	0.93
	16	2.86	3.53	0.81	1.78	1.92	0.93
	32	2.63	3.42	0.77	1.71	1.89	0.90
128	4	2.82	2.82	1.00	1.75	1.75	1.00
	8	3.28	3.39	0.97	1.88	1.89	1.00
	16	3.36	3.71	0.91	1.87	1.95	0.96
	32	3.30	3.75	0.88	1.87	1.96	0.96
256	4	2.46	2.65	0.93	1.45	1.71	0.85
	8	2.84	3.31	0.86	1.57	1.87	0.84
	16	3.11	3.68	0.85	1.65	1.94	0.85
	32	3.21	3.87	0.83	1.83	1.98	0.93

Since a vector with $(2n_x+n_{proc}-2)n$ components need to be distributed to all of the processors, it should be expected that the efficiency decays as n_{proc} grows. However, for small problems, (10) is computed so fast that the time spent on communication becomes more significant, as can be seen in Table 2 for $n = 64$.

The predicted speedup values obtained show that, for each n, the parallel scheme is very efficient for appropriate choices of n_x. The figures for the actual speedup confirm the effectiveness of the algorithm.

Naturally, as the number of processors is increased, other routines than the product (10) need also to be implemented in parallel in order to improve efficiency to a better extent.

8 Final Remarks.

In this work, we described the computational implementation of a parallel code for solving contact problems. A profile of the algorithm was presented for a model problem with two membranes, using two domain decompositions (strips and chessboard pattern). This profile suggests that an efficient parallel scheme can be obtained. Numerical results that confirm the effectiveness of our implementation were also provided for the decomposition into strips.

Among the possible improvements on the algorithm, it is worth mentioning that, for the strips decomposition, better results could be obtained treating the diagonal blocks $K^\dagger$ as general sparse matrices, instead of storing them using a band format as we currently do. With this new approach, we could apply the minimum degree algorithm to permute the columns of $K^\dagger$ and reduce the number of nonzero elements in the resulting Cholesky factors.

A 3D contact problem with Signorini type of contact conditions was solved by the sequential version of the algorithm [12]. Future work includes extending the parallel scheme to this more realistic problem motivated by mining engineering, and also to the solution of 2D contact problems with Coulomb friction.

Acknowledgements. We would like to acknowledge professor Patricio Letelier, from UNICAMP, who kindly made his parallel computer available for our numerical experiments.

References

[1] O. Axelsson, A class of iterative methods for finite element equations. *Comp. Meth. in Appl. Mech. and Engng.* 9 (1976) 127-137.

[2] O. Axelsson, *Iterative Solution Methods*, Cambridge Univ. Press, Cambridge, 1995.

[3] R. H. Bielschowsky, A. Friedlander, F. A. M. Gomes, J. M. Martínez, M. Raydan, An adaptive algorithm for bound constrained quadratic minimization, *Investigación Operativa* vol.7, no.1-2 (1997) 67-102.

[4] A. R. Conn, N. I. M.Gould, Ph. L. Toint, A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds, *SIAM J. Num. Anal.* 28 (1991) 545-572.

[5] Z. Dostál, Duality based domain decomposition with inexact subproblem solver for contact problems. Contact Mechanics II, eds. M.H. Alibiadi, C. Alessandri, Wessex Inst. of Technology, Southampton (1995) 461-468.

[6] Z. Dostál, Duality based domain decomposition with proportioning for the solution of free boundary problems, *J. Comp. Appl. Math.* 63 (1995) 203-208.

[7] Z. Dostál, Box constrained quadratic programming with proportioning and projections, *SIAM J. Opt.* 7 (1997) 871-887.

[8] Z. Dostál, On preconditioning and penalized matrices, *Num. Lin. Alg. Appl.* 6 (1999) 109-114.

[9] Z. Dostál, A. Friedlander, S. A. Santos, Augmented Lagrangians with adaptive precision control for quadratic programming with simple bounds and equality constraints, Technical Report RP 74/96, IMECC-UNICAMP, University of Campinas, October 1996.

[10] Z. Dostál, A. Friedlander, S. A. Santos, Solution of coercive and semicoercive contact problems by FETI domain decomposition, *Contemporary Math.* 218 (1998) 82-93.

[11] Z. Dostál, F. A. M. Gomes, S. A. Santos, Duality-based domain decomposition with natural coarse-space for variational inequalities, Technical Report RP63/98, IMECC - Unicamp, State University of Campinas, October 1998.

[12] Z. Dostál, F. A. M. Gomes, S. A. Santos, Solution of Contact Problems by FETI Domain Decomposition with Natural Coarse Space Projections, Technical Report RP64/98, IMECC - Unicamp, State University of Campinas, October 1998. To appear in *Comp. Meth. in Appl. Math. and Eng.*

[13] Z. Dostál, F. A. M. Gomes, S. A. Santos, On the implementation of an algorithm for the numerical solution of contact problems. In preparation.

[14] C. Farhat, M. Gérardin, On the general solution by a direct method of a large scale singular system of linear equations: application to the analysis of floating structures, *Int. J. Num. Met. Eng.* 41 (1997) 675-696.

[15] C. Farhat, F.-X. Roux, An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems, *SIAM J. Sci. Stat. Comput.* 13 (1992) 379-396.

[16] C. Farhat, P. Chen, F.-X. Roux, The dual Schur complement method with well posed local Neumann problems, *SIAM J. Sci. Stat. Comput.* 14 (1993) 752-759.

[17] C. Farhat, J. Mandel, F.-X. Roux, Optimal convergence properties of the FETI domain decomposition method, *Comput. Methods Appl. Mech. Eng.* 115 (1994) 365-385.

[18] A. Friedlander, J. M. Martínez, On the maximization of concave quadratic function with box constraints, *SIAM J. Opt.* 4 (1994) 177-192.

[19] A. Friedlander, J. M. Martínez, M. Raydan, A new method for large scale box constrained quadratic minimization problems, *Optimization Methods and Software* 5 (1995) 57-74.

[20] A. Friedlander, J. M. Martínez, S. A. Santos, A new trust region algorithm for bound constrained minimization, *Applied Mathematics & Optimization* 30 (1994) 235-266.

[21] R. Glowinski, P. Le Tallec, *Augmented Lagrangians and Operator Splitting Methods*, SIAM, Philadelphia 1989.

[22] F.-X. Roux, Spectral analysis of interface operator, Proceedings of the 5th Int. Symp. on Domain Decomposition Methods for Partial Differential Equations, ed. D. E. Keyes et. al., SIAM, Philadelphia, (1992) 73-90

[23] F.-X. Roux, C. Farhat, Parallel Implementation of Direct Solution Strategies for the Coarse Grid Solvers in 2-level FETI Method, *Contemporary Math.* 218 (1998) 158-173.

[24] J. S. Simo, J. A. Laursen, An augmented Lagrangian treatment of contact problems involving friction, *Comp.& Structures* 42 (1992) 97-116.