Parallel Incomplete Factorization of 3D NC FEM Elliptic Systems

Yavor Vutov

We consider the model elliptic boundary value problem:

$$\begin{aligned} Lu \equiv -\nabla \cdot (a(x)\nabla u(x)) &= f(x) \text{ in } \Omega, \\ u &= 0 \text{ on } \Gamma_D, \\ (a(x)\nabla u(x)) \cdot n &= 0 \text{ on } \Gamma_N, \end{aligned}$$

where $\Omega = [0,1]^3 \subset \mathbb{R}^3$, $\Gamma_D \cup \Gamma_N = \partial\Omega$ and $a(x)$ is a symmetric and positive definite coefficient matrix. The problem is discretized using rotated trilinear finite elements. The resulting linear algebraic system is assumed to be large. The stiffness matrix A is symmetric and positive definite. This implies the use of iterative solution methods. The preconditioned conjugate gradient method (PCG) is known to be the best one for such systems. Modified incomplete Cholesky (MIC(0)) factorization on an approximation B of A is used as a preconditioner. B has a special block structure, which allows a stable factorization and efficient parallel implementation. Let us note, that the considered non-conforming FEM and MIC(0) factorization are robust for problems with possible jumps of the coefficients. The auxiliary matrix B is constructed element-by-element. It is important, that A and B are spectral equivalent, and the relative condition number $\kappa(B^{-1}A)$ is uniformly bounded. The numerical tests confirm robustness of the point-wise MIC(0) preconditioner.

The implementation of PCG method includes solution of systems with the preconditioner $C_{MIC(0)}(B) = (X - L)X^{-1}(X - L^T)$. Here $-L$ is lower triangular part of B and X is diagonal. The factors $(X - L)$ and $(X - L^T)$ have diagonal blocks which are diagonal matrices. These blocks correspond to mesh (nodal) lines subject to a proper node numbering. The proposed construction allows the solution of systems with the matrices $(X - L)$ and $(X - L^T)$ to be performed fully parallel within each of these blocks.

Estimates for the parallel times, speed-ups and efficiencies are derived. The algorithm is implemented in C++ using the Message Passing Interface (MPI) library. Experiments on several distributed computing systems are presented. The results agree well with the theoretical analysis. It is shown that computations and communications can be overlapped in time, thus improving the parallel efficiency.