A Multithreaded Finite Element Algorithm with Element-by-Element Preconditioning

Michael C. Dracopoulos

Department of Mathematics, University of Athens, Greece

mdraco@math.uoa.gr

Key words: Finite elements, conjugate gradients, preconditioning, multithreaded computations, Graphics Processing Units (GPUs), multicore CPUs.

A natural approach to parallelizing the finite element method stems from the concept of the "element" as a building block in the whole methodology. Elements are treated as disjoint and decoupled in many stages of the finite element computations (namely the formation of element level characteristics, such as element stiffness, strains and stresses) which are therefore inherently parallel in nature. In addition, when an iterative solver is employed, the explicit formation of the global stiffness matrix can be avoided by inducing the element connectivity information to the solver vectors. An elegant way to achieve this is by grouping them into two types: "Force type" vectors for each element are those derived from the RHS vector and should add up to the global external "force" contribution for each node. "Displacement type" vectors, on the other hand, are those related to the solution vector and should guarantee the compatibility of the "displacements" around each node.

Under such an element distribution the conjugate gradient algorithm can be implemented with perfectly parallel matrix-vector multiplications and linked triad operations on the *element level*. The only communication involved is related to the two inner products and a single vector update among neighbouring element in order to maintain the "force/displacement type" distribution across all iterations. The bottleneck in the above procedure is the preconditioning of the *unassembled* stiffness matrix. Ideally, one would also like to use some kind of element level preconditioner that would follow this natural distribution. Apart from the trivial diagonal scaling, the options for such a preconditioner found in the literature are quite limited.

In this work, an SSOR type element-by-element preconditioner is proposed which is derived entirely from element stiffness matrices. Suitable modifications are enforced to ensure positive definiteness and to guarantee that its application conforms with the compatibility requirement for the "displacement-type" vectors. The preconditioner is analysed and evaluated in comparison with existing methods. The parallel implementation of the above finite element algorithm is based on multithreading which is particularly efficient on today's multicore CPUs and/or GPUs (Graphics Processing Units which can also be used for general purpose computations). This approach scales well as each thread can deal with as little as just a single element. Numerical results from structural analysis problems are given, tested on quad-core processors and NVIDIA GPUs.