ELSEVIER

Comput. Methods Appl. Mech. Engrg. 184 (2000) 143–144

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Preface

Have you ever wondered why the F117 stealth aircraft has flat, angular surfaces and looks like a flying diamond, while the B-2 stealth aicraft has aerodynamic rounded surfaces? Simply because the F117 was designed around 1975, when mainframes did not have a sufficiently large memory and a sufficiently powerful CPU to allow three-dimensional designs, or rounded shapes, and therefore when the radar cross section of an aircraft configuration could be accurately calculated only if it was in two dimensions. On the other hand, the B-2 was designed later using three-dimensional computations on vector supercomputers capable of hundreds of megaflops. This is perhaps the most dramatic example of the impact of high-performance computing on engineering design, but not the only one. After the unfortunate accident of Challenger, thousands of CRAY X-MP CPU hours have contributed to the redesign of the Solid Rocket Booster. Today, high-performance computing platforms are very visible in the automotive industry and play an important role in structural integrity, noise, vibration, and harshness studies.

Nevertheless, the use of computers in engineering applications has traditionally lagged hardware advances. Software tools have to be provided and their functionality accepted by the user community at large before an application base emerges. Vector supercomputers made their debut in government laboratories devoted to complex physics calculations. Only a decade later they became widely used in industrial computing. Today, high-performance workstations are routinely used by large corporations, and personal computers are used by thousands of engineers for tasks that challenged a mainframe two decades ago. The first generation of commercial parallel computers that emerged in the mid 1980s were thought of as exotic machines. They were essentially devoid of software. The latest distributed shared memory "boxes" are gaining acceptance in industry, and popularizing parallel processing in computational engineering. A few computational mechanics commercial software have even embraced some form of parallel processing. However, the full incorporation of parallel processors in the mainstream of large-scale computations – which has proved to be more challenging than that of vector supercomputers – continues to face many obstacles that center on algorithm scalability, software portability, and economic pressure.

With the advent of commercial parallel computers, domain decomposition has gained momentum because it is the most natural route for scalable parallel processing. For many practitioners, domain decomposition is a paradigm for designing a portable parallel software architecture based on the concept of mesh partitioning. However, domain decomposition is not only a computing strategy suitable for high-performance parallel computing, but also the basis of many partial differential equations solvers. The earliest method of this kind is believed to be the alternating method of H.A. Schwarz, introduced more than 120 years ago. Initially, this method was not intended as a numerical algorithm, but as a means to prove the existence of the solution of the Dirichlet problem for Laplace's equation on regions with non-smooth boundaries. Today, domain decomposition methods are primarily used to construct *parallel scalable* preconditioners that can be used for accelerating the convergence of iterative methods such as the conjugate gradient and generalized minimal residual algorithms, when applied to the solution of elliptic, parabolic, and hyperbolic partial differential equations discretized by finite elements, finite differences, and spectral methods.

The present issue includes sixteen papers on the subject of domain decomposition and parallel processing. They address various fields of applications, namely computational structural mechanics (Brown et al. and Farhat et al.), advection–diffusion (Achdou et al.), computational fluid dynamics (Sarkis and Koobus, Evangelinos et al., Carre et al. and Rifai et al.), fluid/structure interaction (Glowinski et al.), wave propagation and acoustic scattering (Collino et al. and Farhat et al.), and subsurface porous media

PII: S0045-7825(99)00226-1

(Wheeler and Yotov). Three papers consider adaptive solution strategies (Griebel and Zumbusch, Teresco et al. and Leyland and Richter), and one paper focuses exclusively on the corresponding dynamic load balancing problem (Hendrickson and Devine). The parallelization of direct solvers is also considered in at least one paper (Amestoy et al.). These papers are organised in four groups: the first one describes different extensions of domain decomposition algorithms to new fields of applications, the second group covers elliptic and structural problems, the third group deals with CFD applications, and the last two papers present parallelisation tools for load balancing issues and for direct solvers.

We hope that because the papers presented here are diverse in a number of ways, this special issue of CMAME will generate a strong interest among all of researchers, software developers, and computational engineers.

Charbel Farhat
Patrick Le Tallec