

Parallel CFD at the UK-North Pilot Centre

D.R. Emerson, T. Franklin, P.K. Jimack, M.A. Leschziner

1. Introduction with Historical Perspective (Michael Leschziner)

The Parallel CFD Special Topic Group allied to the UK-North Pilot Centre was born in 1990 out a wish by industrial and academic colleagues to create a forum for sharing experience and a focus of mutually beneficial collaboration, with ERCOFTAC contributing a European perspective and a route to CEC funding. There was a feeling, especially among industrial partners, that established initiatives, based principally on Transputer technology and associated centres, failed to provide an environment within which parallel computing for CFD could be effectively exploited in an engineering context.

Much has changed in the few intervening years due to momentous developments in hardware technology and software tools. The Transputer has moved to the background, powerful machines based on much faster processors and massively parallel architectures have emerged, a range of harnesses and generic routines have been created to ease the pain of porting algorithms to MIND architectures, and new strategies based on virtual shared as well as real shared memory have entered the fray. These changes, coupled with the industrial emphasis on commercial objectives and the penetration of the package approach to CFD have conspired to cause, at least in the UK, a divergence of academic and industrial aspirations and, arising from this, a bifurcation of activities. On the one hand, industrial practitioners and software vendors have linked up with specialised groups - usually having strong expertise in informatics but little CFD background - to pursue highly targeted porting exercises involving particular codes and architectures. On the other hand, academic CFD groups have tended to engage, with the aid of government grants, in generic studies, either on their own or in collaboration with key laboratories funded by the Science and Engineering Research Council (SERC). The difficulties of establishing any rational links between the latter "open" and the former "closed" domains has, unfortunately, led to a marked diminution of industrial input into the STG's activities.

Until May 1993, the STG revolved around an axis connecting UMIST's Department of Mechanical Engineering and the University of Leeds' School of Computer Studies, as conveyed by Fig. 1. Direct links existed to the SERC Laboratory at Daresbury (30 km from Manchester) and to the Centre for Novel Computing at the Victoria University of Manchester. Between them, these core groups share a wide range of parallel architectures including a 64 node iPSC/860, a 32-node KSR-1, a 64-node Meiko and an 8-processor Alliant FX2808. Other links shown in Fig. 1, reflect participation in STG events and joint activities within SERC's Collaborative Computational Project «CCP12». The principal initiatives of the STG were two workshops, namely:

- *Porting Practical CFD Codes to Parallel Computers*, Manchester, March 1991
- *Domain Decomposition for CFD on Message Passing Machines*, March 1992

These attracted a total of 100 participants, and related reports were published in ERCOFTAC Bulletins 11 and 17.

In June 1993, The STG was "elevated" to the status of SIG, and the co-ordination centre moved from UMIST to the Daresbury Laboratory - the rationale for the latter being rooted in the fact that this Laboratory is specifically funded by SERC to act as a UK focus and coordinator of SERC-supported research in parallel computing.

While the SIG's core groups collaborate in certain areas, they are, on the whole, engaged in rather diverse sets of efforts, reflecting differing research emphases and types of owned parallel computers. It seems most appropriate, therefore, to convey the nature of these efforts in separate group-related sections.

2. Research on Parallel CFD at Daresbury Laboratory (David Emerson)

2.1 Background

The Daresbury Laboratory is situated near Warrington in the North West of England. It is the home of one of the United Kingdom's national parallel supercomputing facilities offering a peer reviewed service on an Intel iPSC/860 64 node, 16 MBytes/nodes, hypercube. This machine is accessed by both the scientific and engineering community to solve a wide range of problems including *ab initio* quantum chemistry and industrial applications involving chemically reacting flows. Daresbury Laboratory also has a number of complimentary and smaller scale parallel computing facilities including workstation clusters, in particular three IBM and four HP 735. These machines are connected via a number of media, including Ethernet, FDDI, and SOCC and are used to assess the viability of parallel workstation clusters in a wide range of computationally demanding applications.

One of the primary functions of the laboratory is to provide computational support to the academic community. This is generally achieved through Collaborative Computational Projects (CCPs) which coordinate university based work in targeted scientific areas. In particular, the project associated with high performance computing in

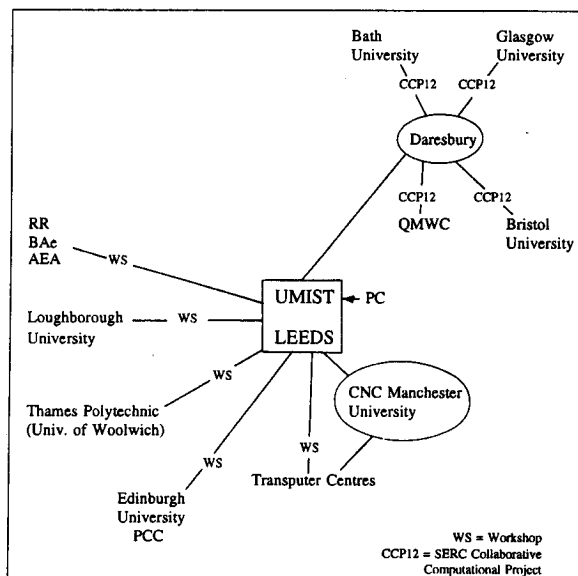


Fig. 1: Initial Structure of STG

engineering and parallel CFD is CCP12. This project is funded by a subcommittee of the Engineering Board of the Science and Engineering Research Council (SERC) to encourage the uptake of parallel and novel architecture computing by the engineering community. The following two sub-sections review two areas of work out of several in progress.

2.2 Turbulent Combustion Modelling

This is a collaborative project between Professor D. Bradley, Dr P.H. Gaskell and Dr Xiaojung Gu in the Department of Mechanical Engineering, University of Leeds, and CCP 12 at Daresbury Laboratory to develop a parallel implementation of a turbulent combustion code. The burning of fossil fuels will probably remain the predominant source of power for many decades, and it is essential that these resources are used efficiently. Furthermore, an understanding of the by-products produced from the burning of hydrocarbon fuels is also necessary to help reduce pollution from combustion products.

The numerical code is based on a slightly modified version of SIMPLE and employs a staggered grid approach with a range of discretisation schemes (e.g. QUICK, central differencing, power law etc.). A standard $k-\epsilon$ or full Reynolds stress turbulence model can be used. To date, computations have been restricted to steady state calculation for two-dimensional, axisymmetric combustors on relatively coarse grids. In reality, combustors are extremely complex and difficult to model and, in general, are three-dimensional and time dependent. Moreover, additional complexities are introduced through the coupling of the chemical rates and turbulence. Preliminary results with the parallel code [1], using the $k-\epsilon$ turbulence model, are encouraging. It is anticipated that further development of the code will enable calculations to be performed with the full Reynolds stress model on more refined grids, with the possible extension of the code to transient or three-dimensional problems.

2.3 External Aerodynamics: Parallelisation of FELISA

This is a collaborative project in conjunction with Dr J. Peraire from Imperial College of Science and Technology, London, and CCP 12 to investigate decomposition strategies for FELISA, a three-dimensional Euler solver on unstructured tetrahedral meshes. The numerical method employs a Galerkin finite-element approach for spatial discretisation and an explicit Runge-Kutta type time-marching scheme. One of the fundamental problems in the solution strategy is the decomposition of the computational mesh into individual subdomains performing equal amounts of work whilst minimising the communications between processors. The effectiveness of established partitioning algorithms, including Recursive Coordinate Bisection

(RCB), Recursive Graph Bisection (RGB), Recursive Spectral Bisection, and the MINCUT algorithm have been investigated at Daresbury Laboratory on a number of structural mechanics and CFD grids. A hybrid algorithm [2] which incorporates the best features of the RSB and MINCUT algorithms has been developed and implemented in FELISA. Preliminary results have been obtained for a computation employing 330,000 elements and Table 1 indicates the performance.

At present, the code is far from optimised and there is certainly scope for a more detailed balance of the load and for hiding the communications costs using asynchronous message passing. The initial results are impressive and point to the power and cost effectiveness of parallel systems.

3. Parallel CFD at UMIST (Michael Leschziner)

3.1 Background

Research at UMIST focuses principally on turbulence modelling and computational studies of complex turbulent flows, both incompressible and compressible. Particular emphasis is put on the application of variants of second-moment closure to 3D internal flows, impinging jets, separated flows around aerofoils and 3D streamlined bodies, shock-induced separation, swirling flow, heat transfer, buoyancy-affected flows, and flows involving combustion and two-phase interaction. In this context, parallel computing is undertaken, with one exception, as part of studies directed towards resolving physical issues, i.e. as a means of enhancing throughput in the process of obtaining engineering statements.

The CFD group has its own Alliant FX2808, awarded by SERC, a 2-processor Stardent and will shortly take delivery of a 2-processor Cray-YMP-EL98. In addition, it has local access to a 64-T800 Meiko system, a 32-node KSR-1 and a number of vector computers. Parallel computing has been undertaken mainly on the Alliant and the Meiko machines, but there have also been collaborative efforts with the Novel Computing Group to exploit the KSR-1; that work will be reported in Section 4. Two particular areas of work reviewed briefly below relate to the Meiko and the Alliant machines.

3.2 Transonic Flow Modelling on the Meiko T800 Machine

Modelling shock/boundary-layer interaction with turbulence-transport closures has been one major area of activity over the past few years. One of three numerical procedures applied to transonic flow at UMIST is a cell-vertex/Lax-Wendroff time-marching scheme incorporating multigrid acceleration and second-moment closure. This procedure has been ported to the Meiko MIMD machine, and the present section gives some details on related work.

Machine	No. of Processors	Time (seconds)
CRAY YMP	1	183
iNTEL Ipsc/860	1	
"	2	803
"	4	460
"	8	209
"	16	114

Table 1:
Results for speedup for FELISA code using 330.000 elements (Daresbury Laboratory)

The parallel implementation is based on domain-decomposition [1,2]. For the example of a transonic flow over a channel bump, Fig. 2 shows the chosen processor topology, the associated sub-domains and the systolic loop along which global data is circulated. One feature of the present cell-vertex implementation is extensive data dependencies brought about by the large computational stencil which is required for an accurate approximation of diffusion and smoothing fluxes. The resulting inter-processor communication of local data is also identified in Fig. 2. Speed-up values and efficiencies for different processor configurations are given in Table 2 and Fig. 3. Recorded efficiency values are compared to "maximum" levels arising from the assumption of infinitely fast communication rates. A similar parallelisation strategy is adopted for aerofoil flows, though in that case particular techniques are needed to ensure that data transfer across the trailing edge "wake cut" arising from the C-mesh topology does not adversely affect the parallel efficiency.

3.3 Parallel Computing on the Alliant FX2808

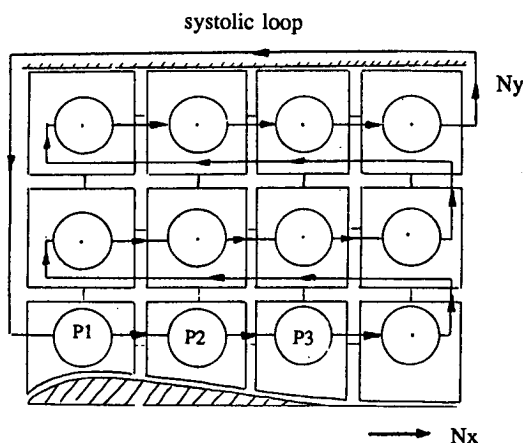


Fig. 2a: Transonic flow over channel bump. Processor topology and systolic loop (UMIST)

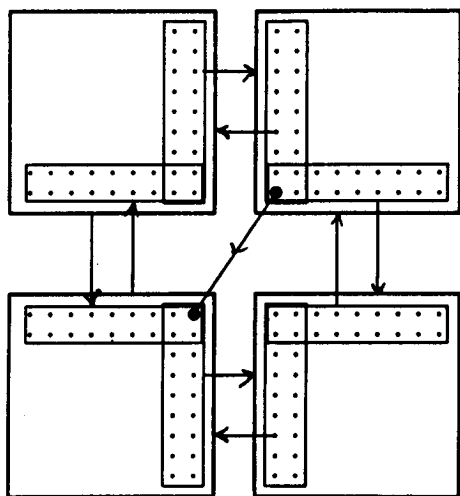


Fig. 2b: Interprocessor communication using cell-vertex scheme (UMIST)

The Alliant is a shared-memory machine. UMIST's configuration consists of 2 modules of 4 i860 processors, each containing a 8KB data cache and 4KB instruction cache. However, only 4 processors can operate as a concurrent cluster at any one time. A global cache of 1MB and a 64 MB main memory support the cluster.

Efforts with this machine have been directed towards achieving high speed-up levels with a wide variety of codes modelling complex 2D and 3D turbulent flows, both compressible and incompressible. Moreover, multigrid and unstructured-grid algorithms for turbulent transonic flows have been implemented.

To achieve high parallel efficiencies, some code restructuring has been undertaken in all cases. This involved removal of conditionals, branching and control statements; unrolling and splitting *do* loops, removal of data dependencies in recursive routines and the use of so-called concurrent subroutine calls in *do* loops instead of nesting. In most applications, even the most complex 3D flows in bends and transonic jets, speed-up values of order 3.5 have been achieved on a 4 processor cluster.

One simple generic application which ties up with the work reported in section 3.2 is a domain-decomposed implementation of an elliptic-flow algorithm on the 4-processor cluster. This technique is of particular relevance in the context of MIMD machines, but can also be implemented on a shared-memory architecture. Here too, sub-domains are assigned to processors. This can be done, for example, within a concurrently called subroutine embedded in a *do* loop:

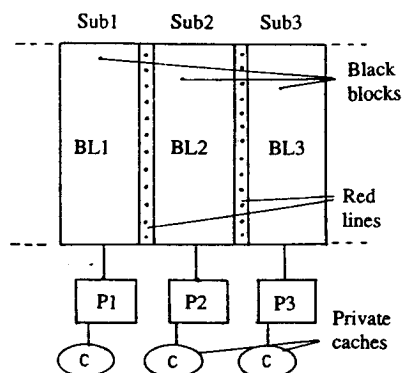
```
do i=1, numproc
  sub (i1,i2)
end do
```

Although data dependencies between sub-domains can be accommodated in a shared-memory environment, their removal is the key to high parallel-efficiency levels, in which case sub-domain data can be allowed to reside on the chip cache while access to the global memory can be minimised. If, for example LSOR is used to solve the discretised conservation equations on a 2D structured grid, removal of data dependencies can be achieved by use of a red-black strategy - or more precisely, a black-blocks / red-lines strategy, as indicated below.

```
cvd encall
do ip=1,numproc
  i1=index (ip,1)   >>>
  i2=index (ip,2)
  call Blocksolve (i1,i2)
end do

cvd encall
do ired=1, numproc
  ii=index (ired,1)
  call Redlines (ii)
end do
```

where the first line is an instruction initiating the



No. Procs	Speed-Up	Efficiency	
		Maximum	Observed
2 x 2	3.8	94.8	91.7
3 x 2	5.6	93.8	90.0
4 x 2	7.5	93.3	88.5
6 x 2	11.3	94.1	86.4
4 x 4	14.3	89.3	81.7
6 x 4	21.4	89.1	77.6
8 x 4	28.1	87.8	72.6

Table 2:
Transonic flow over channel bump speed-up and efficiency data on Meiko (UMIST)

NumProcs	Meiko Time[s]	% Efficiency	Alliant Time[s]	% Efficiency
1	2142.4	100	263.1	100
2	1097.4	97	141.3	93
3	760.8	93	100.6	87
4	594.3	90	81.0	81

Table 3:
Lid-driven cavity 51x51 grid - efficiency data of Alliant compared to Meiko (UMIST)

activation of concurrency within the *do* loops and each subroutine call executes the solution of tri-diagonal matrices along *i*-lines.

For the example of a lid-driven cavity flow computed with 52x52 grid, Table 3 gives values of total parallel efficiency achieved with upto 4 processors in comparison with values attained on the Meiko MIMD machine.

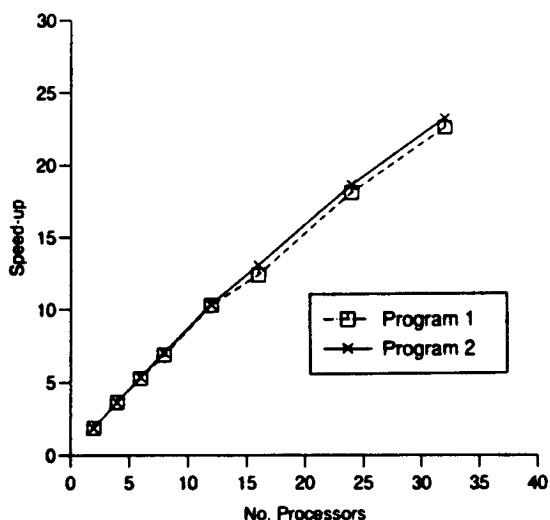


Fig. 3:
Speedup for transonic flow over bump (UMIST)

4. Parallel CFD at the Centre for Novel Computing (Tom Franklin)

4.1 Background

The Centre for Novel Computing (CNC) was set up in November 1990 with foundation funding provided by the Novel Architecture Computing Committee (NACC) of the SERC. This award reflected a recognition of the problems most users of high performance parallel systems were facing in effectively exploiting the potential power of these machines.

The aim of the CNC is to make high performance parallel computing accessible to all users who have a need for the power it can offer. In particular, the CNC aims to help make progress in tackling realistic engineering problems. Currently, many users are forced to alter their applications to fit a specific machine. In some cases the whole problem has necessarily been redefined in response to the computer's limitations. The CNC helps users to implement their problems on suitable architectures, striving to achieve maximum benefit with minimum alteration to the basic algorithms.

The CNC has 11 research staff, 9 research students, and the active involvement of eight academic staff drawn from the Departments of Computer Science and Mathematics at the University of Manchester. The Centre also has close links with similar centres at Edinburgh and Southampton.

The CNC covers many aspects of parallel and novel computing. It collaborates with around thirty groups of scientists and engineers in a variety of disciplines including CFD, meteorology, astronomy, environmental modelling and database applications. A cornerstone of the CNC's work is a KSR1-32 supercomputer. Apart from the

collaborative research noted above, independent research is being pursued by the Centre in scheduling and minimal operating systems, and automated parallelism through more sophisticated compilation methods.

The CNC has been working, principally in collaboration with the Engineering Department of the University of Manchester and UMIST's Mechanical Engineering Department, on a variety of CFD applications, ranging from the supersonic flows to monitoring human ventilation. Two efforts are briefly summarised below, following a brief description of the KSR1.

4.2 The KSR1

In November 1991 the CNC acquired a KSR1-32 supercomputer, the second to be delivered world-wide, and the first outside the USA; this is shortly to be upgraded to a 64 processor system. The KSR1 is the first computer to offer an architecture which is both scalable and offers the user the ease of a shared memory programming model.

Historically, when users have bought a parallel computer, they have had to choose between shared memory computers, which are easier to program, but not scalable above around 30 processors, or distributed memory computers, which can be scaled to thousands of processors but are extremely difficult to program. The KSR1 offers the scalability of distributed memory computers, with the ease of programming of shared memory computers. To achieve this KSR have implemented an «ALLCACHE» system.

At the heart of the system is the KSR1 «Cell», which includes a full 64 bit 20 MHz (40 Mflop/s peak) IEEE conformant superscalar processor, 32 Mbytes of "cache" memory, 0.5 Mbytes of "sub-cache", and the ALLCACHE memory management hardware. The Cell carries twelve full custom CMOS chips, of six different types. The cells are linked by a hierarchy of rings. Up to 32 cells may reside on a «level 0» ring, and up to 34 level 0 rings can be connected by a «level 1» ring. The level 0 ring has a bandwidth of 1 GByte/s, and the level 1 ring of 1-4 GByte/s. The ALLCACHE memory management hardware implements a sequentially consistent 40 bit single virtual address space between all the cells in the system. Thus, the KSR1 has a physically distributed memory which is seen by the programmer as a single, coherent address space. The programmer does not need to know where the data resides, and the data migrates to the processor as it is needed. This

is all handled by the ALLCACHE hardware, and is the main breakthrough in the KSR architecture.

4.3 Laminar Flow Control in Aerodynamics

One of the first applications on which the CNC worked, in collaboration with Professor Ian Poll and Mark Gallagher of the Department of Engineering at Manchester University, was Laminar Flow Control (LFC) in external aerodynamics. LFC is becoming a credible approach to increasing the fuel efficiency of aircraft. This may be exemplified by the fact that, in January 1993, a major UK aero engine company conducted real flight tests of a nacelle (engine casing) whose shape had been determined by the application of LFC. In these tests, laminar flow conditions were achieved over 60% of the nacelle, giving a 3% overall saving in specific fuel consumption (total saving in fuel) over the previous shape. To put this reduction into perspective, it has been estimated that for a typical aircraft, a 3% reduction in specific fuel consumption will give a reduction in operating cost of \$300,000 per large aircraft per annum, and it has been noted that a 4% difference in specific fuel consumption will enable an aircraft to have a decisive competitive advantage in the civil-aviation market. LFC has an even greater impact in reducing skin friction for submarines and long distance pipelines as frictional drag is, in these cases, a much greater proportion of total drag than for aircraft operation.

LFC is based on delaying the onset of turbulence. Boundary Layer Stability theory can be used to model the turbulent transition point and features causing it. Flow properties such as velocity and pressure are split into steady components and small oscillatory disturbance terms. The latter are modelled by the Orr-Sommerfeld equation.

In the study undertaken at the CNC [6,7], flow over an aerofoil was modelled within a 2-D incompressible environment, in which case the corresponding Orr-Sommerfeld equation reduces to a 4th order boundary value problem. When the amplification rate of the instability wave solution of the Orr-Sommerfeld equation (N-factor) exceeds a critical value, turbulence is held to occur. Starting with an initial approximation, the N-factor is solved by the combination of a shooting method and a Newton Raphson search technique. The judicious choice of the initial approximation guarantees convergence of the iterative method. The user gives an initial approximate solution for the lowest frequency and position (station) closest to the front of the aerofoil; the algorithm then automatically generates approximations for each subsequent frequency specified at that station. The solution for the lowest frequency is then used as an initial guess for the next station.

The specific objective pursued by the CNC was to reduce the interactive run time of low resolution searches to a low level without significant algorithmic changes. The methods used could then be applied to larger problems as well. The sequential code was written in standard Fortran 77. It compiled and ran on a single cell of the KSR1. The KSR1 compiler has two levels of optimisation; -O1 for scalar optimisation and -O2 combines this with loop unrolling. Sequential performance may also be enhanced by running the code through KAP with only scalar optimisation switched on. Times for the optimisation, together with similar data for the i860 and HP Snake are shown in Table 4.

Subsequent parallelism introduced into this code is highly nested, and was used at several levels. Parallelism at the innermost level involves two parallel subroutine calls. These calls are located inside the shooting method integration loop, which is sequential. The next level of parallelism is over an independent loop. At the outermost level there is a loop with an ordered critical region. This gives a parallel pipelining effect. Because of a loop carried

Processor	Clock rate (MHz)	Optimisation level	run time (seconds)
KSR1	20	none	333
KSR1	20	-O	206
KSR1	20	-O2	196
i860	40	-uniproc -Og	183
HP Snake	66	none	273
HP Snake	66	+O3	96

Table 4:
Single Processor Performance for laminar flow control code (CNC, Manchester University)

cells	time
1	3438 s
8	458 s
16	252 s

Table 5:
Parallel speedup for the laminar flow control code on KSR-I (CNC, Manchester University)

dependency, the subsequent iteration cannot proceed until a value has been written to by the previous iteration (the generated approximate solution). Moreover there is I/O within this loop which needs to be ordered so as to maintain the form of the sequential output.

The above parallelisation steps led to very respectable speedups levels, as shown in Table 5. This gives data for one of the larger aerofoils investigated, involving 40 frequencies across 100 stations.

4.4 Transonic Impinging Jets

The CNC has also worked with the Thermodynamics and Fluid Mechanics Division of Mechanical Engineering at UMIST on the implementation of a compressible pressure correction method for impinging Jets. This code was designed for predicting the type of flow beneath a vertical take-off and landing aircraft whilst hovering above the ground. The aim of this collaboration was to first port the code to the KSR1 and optimise it for parallel execution, with as few changes to the original codes as possible. The Jet code was originally written and optimised for a Cray XMP. Efforts thus focused on examining how this code, written for a vector architecture machine, could be restructured so as to suit the KSR1's architecture and on investigating which forms of parallelism are most useful to the KSR1 system, in particular its hierarchical memory model. The work was carried out as part of the AMUS Work-package 6.2 funded by ESPRIT as part of project 2716-AMUS.

5. Parallel CFD at the School of Computer Studies, University of Leeds (Peter Jimack)

5.1 Background

The Scientific Computation group in the School of Computer Studies has a history of producing high quality software for the solution of differential equations. The main aim of current CFD research within the group is to continue this work by producing good quality, reliable and efficient general purpose software capable of solving a wide range of flow problems. The techniques used to develop suitable algorithms for such software include the use of both finite element and finite volume schemes on unstructured meshes, combined with automatic adaptivity of these meshes through local refinement, unrefinement and movement based upon built-in error estimates. The use of parallel algorithms within this CFD software is also a major area of current research within the School. The main objective of this work is to ensure that the high quality, efficiency and generality of the software that is produced is maintained when implemented in parallel. This means

addressing the two key issues of parallel software: portability and scalability.

The School of Computer Studies is a member of a number of multidisciplinary research groups within the University including the Centre for the Development of Computational Fluid Dynamics and the recently established Keyworth Institute of Manufacturing and Information Systems. The links established through these groups allow extra breadth to be added to the School's own expertise, especially in the area of engineering applications and, in the case of the Keyworth Institute, in the exploitation and transfer of technology into industry. In addition, the cost of computer facilities, such as a 34 node Meiko parallel computer system, can be shared with other departments within these groups.

Below is a brief introduction to two of the projects that are currently being undertaken within the School, designed to give a sample of the School's research in parallel CFD and related issues. Other similar work being pursued at present includes projects developing parallel scientific visualisation and programming environments, projects developing parallel linear algebra and PDE software, and projects porting sequential CFD codes onto parallel machines.

5.2 Load Balancing for Unstructured Problems with Adaptivity

The use of unstructured grids for the finite element or finite volume solution of CFD problems is essential when considering geometrically complex domains or the use of mesh adaptivity, the latter necessary for the reliable solution of most practical problems. One of the drawbacks of such an approach stems from the fact that most algorithms for solving these problems in parallel require the grid to be decomposed into a number of subgrids, each of which is allocated to a separate processor. On static structured meshes, it is a relatively simple task to perform this partitioning in an efficient manner, where each processor is allocated a roughly equal subgrid (for the purposes of load balancing) and the boundary between each subgrid is as short as possible (this boundary length, when measured in terms of shared element or volume edges, determines the amount of inter-processor communication that will be required). For unstructured grids it is a far more difficult task to partition the mesh in an optimal or nearly optimal manner, especially when this has to be done dynamically due to mesh adaptivity.

The Scientific Computation group has been studying this issue by investigating parallel mesh generation and partitioning for steady problems and, in collaboration with Shell UK, load balancing of adaptive grids for time-dependent problems. A number of new graph-based algorithms have been developed for automatically controlling the distribution of mesh data within parallel finite element and finite volume solvers in such a way as to be both fast and efficient ([8], [9]). Analysis of these methods combined with computational experiments on CFD problems demonstrate their superiority over other popular partitioning strategies for important classes of problems. The following simple example (taken from [9]) integrates the Euler equations in two space dimensions with an inflow of air at Mach 2.5 hitting a 10 degree wedge and forming a shock front. In its original form this is a steady problem, however, in the time-dependent form used here the shock starts off along the wedge and rises to its steady state position as time proceeds. The unstructured mesh becomes heavily refined around the front as it forms but remains coarse away from the wedge and the shock. Figure 4 shows the solution grids after the first and final remeshes.

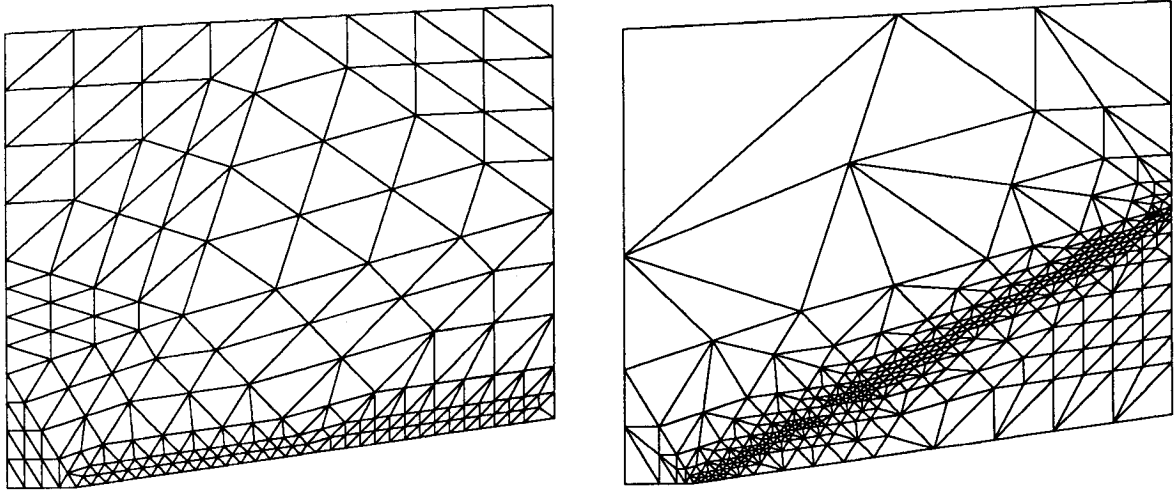


Fig. 4:
Wedge shock grids after the first and final remeshes (Leeds University)

P	CB		DRSB		RSB	
	T _s	T _b	T _s	T _b	T _s	T _b
16	2,415	20	2,074	228	2,079	486

Table 6: Inviscid wedge shock calculation with unstructured adaptive grid - solution time in transputer CPU seconds (T_s) and associated load-balancing time on SPARC Workstations (Leeds University)

The problem has been solved three times on 16 (P = 16) processors: once using the dynamic partitioning algorithm developed at Leeds (denotes as DRGB) and then using two other popular partitioning algorithms, recursive coordinate bisection (RCB) and recursive spectral bisection (RSB), both described in [10], for example. Table 6 shows the parallel solution times (T_s) in transputer CPU seconds and also the time spent performing the load balancing *sequentially* on a SPARC workstation (T_b) in CPU seconds. It can be seen that the new method has the best solution time (and therefore has provided the best partitions) yet the cost of obtaining these good partitions does not offset their benefits (as with the RSB algorithm for example). A parallel implementation of the partitioning algorithm will reduce these overheads still further and allow scalability of the algorithm by removing this sequential bottleneck.

5.3 Scalability and Portability

For parallel programming and architectures to be fully accepted within the field of Computational Fluid Dynamics, or any other computationally intensive subject area, the twin issues of scalability and portability will have to be more comprehensively addressed and analysed than is presently the case. In particular, any large piece of software, when written, will be expected to have a lifetime of many years and so must be guaranteed to be able to port and run efficiently onto more advanced hardware than initially used. Such hardware will still only be at the design or concept stages when the software is being written. Moreover, since future improvements in computer hardware are likely to be achieved mainly by increasing the number of individual processors available to contribute towards a task (rather than improving the computational

performance of each processor), it is vital that parallel programs and architectures be able to scale efficiently - to many hundreds of thousands or even millions of processors if possible.

This project has successfully introduced the XPRAM programming model ([11]) and is now involved in a number of theoretical and practical applications of it. This programming model is designed to be layered on top of a distributed memory machine, the Bulk Synchronous Parallel (BSP) computer proposed by Valiant [12], yet have an interface and programming paradigm akin to a shared memory computer. The reason for this design, which was completed in conjunction with Inmos Ltd., is that scalability can only be achieved through a distributed memory architecture whilst the programming interface provided by a shared memory machine is far more straightforward and therefore more portable.

Unlike for sequential computing, however, there is far more to the issue of portability than just having a consistent programming interface. The XPRAM model has been designed so that the performance characteristics of any algorithm will also be preserved across any BSP machine - a fact that can be demonstrated analytically. This preservation of the performance characteristics of a code is a far more important and complex portability issue than that of simply getting a code to run on different parallel machines.

Present work in this area is continuing through a substantial SERC funded research grant and further formal collaboration with industry is also planned. A simulator has been written to implement the XPRAM model and the work on scalable algorithms for large problems in areas such as linear algebra for example will have direct and immediate applications in CFD codes when it is completed.

References

1. Carter, J.G., Blake, R.J. and Allan, R.J., «Computation of incompressible turbulent flow on novel architecture systems», CFD Algorithms and Applications for Parallel Processors, ASME Fluids Engineering Conference, 1993.
2. Gu, X., Bradley, D., Gaskell, P.H., Emerson, D.R., Carter, J.G. Blake, R.J. and Allan, R.J., «Parallel computation of turbulent recirculating combustion processes and chemically reacting industrial flows», Parallel CFD '93.
4. Golby, D. and Leschziner, M.A., «Implementation of a cell vertex code for turbulent transonic flows on a Meiko Computing Surface», 4th Annual Conference of the Meiko User Society, Southampton, 15-16 April 1993.
5. Golby, D. and Leschziner, M.A., «A domain-decomposed cell vertex scheme for turbulent transonic flows implemented on a MIMD architecture», Proc. Parallel CFD '93, Paris, France May 10-12, pp. 43-48, 1993.
6. Ford, R., «Experiences parallelising an Aeronautics code on the KSR1», Proc. of Supercomputing Europe, February 1993
7. Ford, R., «CNC Application Analysis Report, Laminar To Turbulent Transition», December 1991.
8. Hodgson, D.C. and Jimack, P.K. «Efficient mesh partitioning for parallel PDE solvers on distributed memory machines», in Proc. 6th SIAM Conf. on Parallel Processing for Scientific Computing, (eds. R.F. Sincovec et al.) SIAM 1993.
9. Walshaw, C. and Berzins, M., «Enhanced dynamic load-balancing of adaptive unstructured meshes» in Proc 6th SIAM Conf. on Parallel Processing for Scientific Computing, (eds. R.F. Sincovec et al), SIAM 1993.
10. Simon, H.D., «Partitioning of unstructured problems for parallel processing», Computing Systems in Engineering, 2, pp. 135-148, 1991.
11. Nash, J., Dew, P.M., Dyer, M.E. and Davy, J.R., «Parallel program design on the CPRAM model», in Abstract Machine Models for Highly Parallel Computers, (eds. J.R. Davey & P.M. Dew), OUP 1993.
12. Valiant, L.G., «A bridging model for parallel computation», CACM 33, No. 8, pp. 103-111, 1990. ■