Chapter 1

Introduction

1.1 Motivation

Computational Fluid Dynamics (CFD), enable us to deal with a wide range of engineering problems related with heat and mass transfer (HMT) by means of the numerical solution of the governing equations, i.e. the conservation of mass, momentum and energy.

With the help of digital computers it has been possible to solve numerically these equations and simulate engineering problems including different phenomena (from pure diffusion to natural or forced convection), flow structures (laminar and turbulent), states of flows (single or two phase flows) and geometries (from simple to complex cavities, channels, bluff bodies, etc.).

The purpose of these simultations is that they may be a laboratory of accurate numerical experiments for predicting complex phenomenologies and a supply of information to less detailed solutions for engineering problems. However, the experimental research (not done in this work) cannot be underestimated in front of the numerical experiments. These experimental cases are chosen carefully to show either the limitations or the range of application of the models implemented. Once the code has been "checked", the simulations that it provide are a tool for the design and optimization of engineering solutions.

The target of this work is to contribute to the development of numerical techniques that will allow the detailed numerical simulation of complex heat and mass transfer phenomena inside thermal equipments, such as melting for heat storage [1], heat looses due to the natural convection in solar colectors [2] and ventilated facades [3], turbulence inside the chamber or through the valves of a reciprocating compressor for refrigeration [4], thermal stratification in tanks for heat storage [5], etc.. All of them, should be simulated in detail for an improvement of their performances.

For a given equipment, the different levels of complexity of modellization could be discussed from the accuracy and feasibility points of view of the results. If a relatively simple model is used with a global conservation of mass, momentum and energy balances over the parts of the equipment, and additionally, with the help of experimental coefficients (i.e. friction factor, heat transfer coefficient in convection, void fraction, pressure-loss coefficient, etc.) we can expect to obtain, with low computational resources (i.e. memory storage and cpu time), global results. Indeed, the accuracy of results depends on the accuracy of the experimental coefficients and their suitability for such model.

Conversely, a complex and detailed model with less use of experimental coefficients would give very accurate and detailed results, but it demands large computational resources. For instance, the modellization of an element such as the reciprocating compressor is based on the integration of the governing equations in the whole compressor domain (compression chamber, valves, manifolds, mufflers, connecting tubes, parallel paths, shell, motor, etc.) featured by complex heat transfer and fluid flow phenomena: 3D turbulent, pulsatory and compressible flow, fast transient processes, complex geometries, moving surfaces, etc.

Hence, for an accurated solution of these phenomena, in order to be used in the design and optimization processes, a very fine spatial and temporal discretization of the governing equations is required, resulting into a large number of systems of equations with hundreds of thousands and even millions of unknowns. Moreover, the treatment of the high convective terms of the governing equations, the pressure-velocity coupling and the boundary conditions lead to non linear and ill-conditioned systems of equations which are more difficult to solve.

Therefore, the feasibility to solve in detail problems with sets of large systems of equations is constrained by the current technology of the computing science applied to CFD problems: large computational resources and powerful solvers.

1.2 Overview of parallel computing in CFD problems

In order to deal with a problem such as the presented above, the parallel computation on multiprocessor architectures and the numerical algorithms which exploit the capabilities of these machines offers the possibility to solve large CFD problems. To do so, current research efforts are being stressed in two areas: computational resources based on distributed memory architectures and numerical algorithms (i.e. the discretization and solution of the governing equations). It is worth noting that, the developement of high performance software follows the current software enginering philosofy: the layering or block-building technique.

1.2.1 Computer technology

The computer technology is continuously changing in order to afford the present and future of a wide number of disciplines. Few of them are work-office, entertainment (multimedia), data base and High Performance Computing (HPC). Here we are interested on the application of the emerging technologies to HPC, and more precisely, to the simulation of CFD problems.

Single-processor architecture machines, in spite of their increasing improvements in computational power, are incapable to afford the numerical simulations of our interest. For instance, to carry out a numerical simulation within a reasonable time it would require an speed of the processor over the physical limits of the current technology. Moreover, the size of the problem is too large and hence, its storage in RAM is another limiting factor. The development and progressive introduction of the operating system Linux in PCs gives the equivalent reliability and performance features to the work-stations with UNIX.

Conventional parallel computers with either shared or distributed memories and with high speeds of communication among processors, represent an important advance respect

to the above machines. These machines use UNIX based operating systems. However, this computational power is tightly linked with an expensive cost.

Nowadays, it is being both a decreasing cost of personal computers (PCs) with fast processors and a rapid development of the communication technologies that improve data transfer rates in networks. As a result of this scenario, the PC clusters are an alternative to the supercomputation of distributed applications.

For similar computational power per processor, the PC clusters have a cost per processor lower than supercomputers. However, the main disadvantage of PC clusters relays on their limited capacity of communication, roughly 10 times slower than in the case of the supercomputers. Since the parallel performance is directly affected by the communication, it is very difficult to design the numerical algorithms, i.e. the parallel solvers for CFD problems.

The computer technology advances are also concerned with faster communications. Therefore, they have also to be considered for the improvement of our parallel implementations. These advances are summarized below from the hardware and software points of view.

- With respect to the hardware for networks, the advances are directed to increase the transfer rate of data per second, i.e. the bandwidth and to reduce the start-up time of the communication, i.e. the latency. Examples of network connections are Fastethernet(100Mb/sec) and Myrinet(1Gb/sec) [6]. The connections are joined in the switch elements that accept transference of data from all the processors simultaneously and in both directions, i.e. in full duplex communication. Furthermore, these elements enable the clustering of several tens of processors, and hence, the increase of the potential scalability or computational power in distributed applications.

- With respect to the software for networks, the advances have been directed to the development of efficient, reliable and portable message passing libraries. For example, the Message Passing Interface [7] (MPI) is becoming an standard de facto.

1.2.2 Numerical algorithms

Current research efforts on numerical algorithms are being stressed in the development of accurate and fast solutions of CFD and HT problems by means of new discretization procedures of the governing equations and new solvers for both single and distributed memory multiprocessor architectures.

The accuracy of results is achieved by either a grid refinement on the spatial and temporal coordinates for a given discretization scheme or by an improved discretization scheme for a given grid. However, the grid refinement leads to larger systems of equations while the discretization with higher order schemes, following conservative and stability criteria, leads to more complex computational molecules, and hence, to more dense matrices. Moreover, the discretization procedure has to consider the coupling problem among the mass, momentum and energy systems of equations. Segregated and coupled discretization procedures and solvers are continuously being revised [8, 9].

Since the most computation-intensive part of a simulation is the solution of the algebraic systems of equations, the research is also focused on fast and robust solvers. These solvers

are featured by an increase of efficiency and scalability to deal with difficult and large systems of equations. Furthermore, the research and development of new solvers is closely related with the sequential and parallel computers, and hence, leading to sequential and parallel solvers respectively.

Within the framework of solvers for CFD problems, current sequential solvers are based in iterative methods (e.g. the incomplete factorizations SIP [10], MSIP [11] and SIS [12, 13]) which exploit, better than direct methods (e.g. the complete factorization [14]), the sparsity of the large matrices that arise in the discretization of the governing equations.

Apart from the improvements of the iterative solvers on the reduction of both the number of floating point operations and memory storage requirements, the main improvements are obtained when combining any of them with a Multi Grid acceleration technique [15, 16] (MG) for the solution of large linear systems of equations. It is well known the degradation of the efficiency of these iterative solvers, so called smoothers in the context of MG, for these systems of equations. The major improvements are done in the development of algebraic restriction and prolongation transfer operators which lead to the Algebraic Multi Grid (AMG) [17, 18, 19] as well as in the implementation of different solvers at different grid levels (e.g. the use of a iterative solvers at finer levels plus a direct solver at the coarsest level). An emerging approach to the acceleration techniques with similar results to AMG is the Multi Resolution Analysis (MRA) [20, 21, 22] with wavelets [23]. It offers a generalization of the transfer operators, and hence, a designing tool of them.

Nevertheless, the elliptic nature of the governing equations and the coupling among the fluid-flow variables constrain their fast solution with sequential solvers to not too large size problems (e.g. up to a half million of unknowns).

The strategy based on parallel solvers is the key to scale the problem size to several millions of unknowns at low computational cost time. Although the solvers mentioned above are powerful in sequential (i.e. execution in single processor architectures), the inherent sequential parts contained within the algorithm (e.g. incomplete LU factorizations) and the large number of communications disable their efficient implementation in distributed memory multiprocessor architectures. For instance, in a parallel implementation of the algebraic multigrid, several communication steps are required at each level leading to a low ratio of the time of computation with the time of communication for the coarser levels.

Furthermore, the efficient parallelization of CFD problems with large systems of equations not only depends on this ratio but also on the distribution of data (or load) among the processors (i.e. the load balancing). The addition of the domain decomposition [24] with the Krylov space based iterative methods [25] is the most widely adopted technique. However, the efficiency of this strategy is strongly dependent on the partitioning directions of the domain and on the parallel efficiency of the preconditioner for the Krylov solver. The most common preconditioners are based in incomplete factorizations [26], polynomial preconditioners [27] and approximate inverses [28, 29]. However, due to the inherent sequential parts of the incomplete factorization, the efficiency decreases with the number of processors. The polynomial preconditioner parallelize well but it needs an accurate approximation of the singular values, task as difficult as to solve the system of equations. And, the approximate inverse preconditioner requires, in advance, the knowledge of the pattern of the inverse of the matrix problem.

1.2.3 Software engineering: layering

Software engineers often recommend the abstraction and encapsulation by layers in developing software components[30]. They recommend the layering of new components on top of existing components, using only information about the functionality and interfaces provided by the existing components. This layering approach is in contrast to a direct implementation of new components, utilizing unencapsulated access to the representation data structures and code present in the existing components. By reusing the existing components, the layering approach intuitively result in reduced development costs, and in increased quality for the new components.

Following this idea a CFD code based on a structure of four layers may be proposed: the user layer, the solver layer, the algebra layer and the communication layer. These layers are linked and integrated to build an structure which looks like an iceberg (see Fig. 1.1).

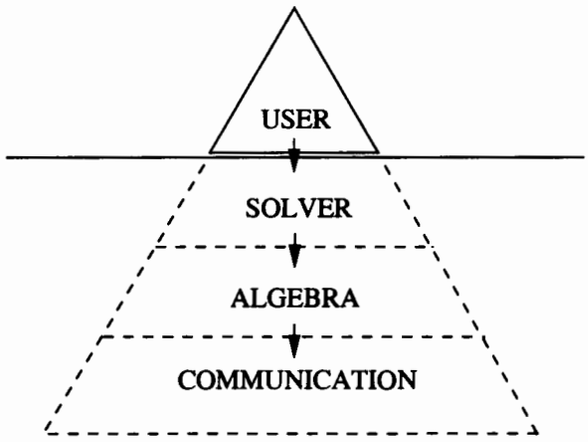

Figure 1.1: Iceberg built by layers: user layer, solver layer, algebra layer and communication layer.

The concept of the iceberg comes from the idea that only the top of the iceberg or the last layer is visible while the rest of the iceberg or layers are hidden below the water. This top end is called the user layer, the only layer visible to the end user of the CFD code. The rest of layers are hidden from the user, i.e. the user does not access to the remaining layers.

In order to build the CFD code in layers it is very important to specify clearly which are the subroutines of each layer and which layer is supported by another, i.e, to define the dependencies between layers in only one direction. This dependency goes from the top to the base as shown with the arrows in figure 1.1. The detailled description of these layers is given in appendix.

1.3 Scope of the work

This work is aimed to contribute to the development of numerical algorithms for CFD problems using both sequential and parallel computers based on clusters of commodity

computers. The purpose of this work is fourfold; it is to provide with:

- The derivation of the systems of equations of a CFD model problem.
- A review of the sequential and parallel solvers for these systems.
- A parallel implementation of the domain decomposition and Krylov solvers.
- A performance study of such implementation for a benchmark problem.

These items are described in the forthcoming chapters as follows. In chapter 2, the fundamentals of fluid dynamics and heat transfer in 3D phenomena are reviewed through the governing equations (i.e. the conservation of mass, momentum and energy) and the boundary conditions. The discretization of the governing equations for both cartesian and cylindrical coordinates is based on the Finite Volume Method (FVM) on staggered grids. Numerical techniques such the discretization schemes, implementation of boundary conditions and solution procedures for incompressible and unsteady flow problems are reviewed.

The chapter 3 is concerned with the analysis and solution of these systems of equations. This analysis is focused on the matrix properties of systems of equations such the sparsity of matrices, the stability criteria associated with the matrix coefficients and the convergence behavior for different boundary conditions. On the other hand, this chapter is a review of the implementation details of several suitable solvers for these systems of equations. This review covers the basic iterative methods based on incomplete factorizations, the algebraic multigrid, the multiresolution analysis with wavelets and the Krylov space based solvers with incomplete LU factorizations as preconditioners. These implementation descriptions are completed with a set of performance tests by means of the comparison of the time of computation and the memory requirements for a sequential execution.

In chapter 4, the concepts of parallel computing in distributed memory machines and performance parameters are defined and measured for two well differentiated architectures: the Cray T3E and a PC cluster (see Appendix). In this work the MPI library has been adopted for the implementation of the exchange of data among processors. This MPI implementation is submitted to a set of performance tests where the communication and computational efficiencies (i.e. the speed-up and the scalability) of both the algebraic operations and the Krylov solvers are measured. Since these tests have been executed in both architectures, the comparison of the communication and computation performances has reported valuable information about the range of application of these solvers (i.e. the scalability and the speed-up) and the cost-effectiveness of each machine.

This is the base of the parallel implementation of the systems of equations and solvers under an algebraic implementation of the domain decomposition technique. The idea is to decompose the algebraic operations involved in all procedures by means of the distribution of vector and matrix data among the processors. The operations over data and the data follow the so called Single Program and Multiple Data (SPMD) paradigm. This paradigm suits well for distributed memory machines such as the PC clusters.

In chapter 5, several of the above techniques are applied to the solution of a CFD problem: the domain decomposition technique, the incomplete factorization solver for the solution of momentum equations and BiCGSTAB preconditioned for the solution of the pressure correction equation. The well known 3D lid driven cavity problem is chosen to

firstly benchmark the accuracy of results and secondly to analyze the speed-up and the scalability of the parallel implementation for several number of processors, partitioning configurations and problem sizes. Bottlenecks within the algorithm and solution alternatives to the current implementation are also outlined.

The concluding remarks of the solution of large systems of equations in PC clusters and further trends on parallel computing of CFD problems are discussed in chapter 6.