Two Dimensional Airfoil Optimisation Using CFD in a Grid Computing Environment

Wenbin Song, Andy Keane, Hakki Eres, Graeme Pound, and Simon Cox

School of Engineering Sciences
University of Southampton
Highfield, Southampton, SO17 1BJ, UK
{w.song, ajk, eres, gep, sjc}@soton.ac.uk
http://www.geodise.org

Abstract. In this paper, a two-dimensional airfoil shape optimisation problem is investigated using CFD within a grid computing environment (GCE) implemented in Matlab. The feature-based parametric CAD tool ProEngineer is used for geometry modelling. The industrial level mesh generation tool Gambit and flow solver Fluent are employed as remote services using the Globus Toolkit as the low level API. The objective of the optimisation problem is to minimize the drag-to-lift coefficient ratio for the given operating condition. A Matlab interface to the design exploration system (OPTIONS) is used to obtain solutions for the problem. The adoption of grid technologies not only simplifies the integration of proprietary software, but also makes it possible to harness distributed computational power in a consistent and flexible manner.

1 Introduction

Computational fluid dynamics (CFD) has been constantly developed over the past few decades and now both commercial and in-house codes can provide more and more robust and accurate results. Combined with the use of wind tunnel test data, CFD can be used in the design process to drive geometry change instead of being used mainly as a design validation tool. This aspect can be further exploited by bringing optimisation tools into the design process. Automation of the design process can significantly shorten the design cycle and lead to better designs compared to previous manual design modification approaches. Such manual approaches are still adopted by most engineers due to various reasons: lack of robustness and flexibility in automating the design process, the high computational cost associated with large numbers of iterations of high fidelity simulation codes, difficulties of collobaration in a heterogeneous computational environments, etc. In fact, the revolution brought by the World Wide Web with respect to information sharing has not yet delivered fundamental changes to engineering design practice for a number of reasons, including security problems in collaborative environments. The emerging Grid computing technologies [1] aim to fill this gap in providing a general architecture for building a geographically distributed, collaborative computational environment.

Design search and optimisation is a process that can be used to improve the design of complex engineering products, systems, and processes. There are a large number of numerical optimisation methods available for this purpose, along with various strategies that provide even more complex scenarios for design search such as hybrid methods incorporating machine learning methods. In order to evaluate the strength and/or weakness of different designs, it is often necessary to use complicated computational tools such as CFD. One of the important aspects of these tools is the high computational cost related to the solution of large numbers of simultaneous algebraic equations, such as in computational fluid dynamics or structural analysis. The combination of the repetitive nature of search processes and the constant increase in demand for computational power of high-fidelity computational models has prompted much effort in the study of various aspects of the problem solving environment. For example, approximation techniques are often used to provide surrogate models for the high fidelity codes and to decouple the strong interactions between codes from different disciplines. However, an efficient approximation framework often requires capabilities to start the analysis codes on-demand for points anywhere in the parameter space and mining new data into existing datasets. It is clear that the strong coupling between optimisation methods and domain codes partly limits the ability to prototype different complicated search strategies, and thus impedes the wider use of optimisation technologies in the engineering design offices on a daily basis.

The aim of this paper is to provide an exemplar of CFD-based shape optimisation using emerging grid technologies that address some of these issues. The paper is organized as follows. The next section gives a brief introduction on grid computing technology with a focus on the architecture and various techniques used. The third section defines a two-dimensional airfoil shape optimisation problem. The optimisation method used and some results are given in section four, with concluding remarks and future work described in section five.

2 Grid Computing

The design of increasingly complex engineering systems relies on knowledge drawn from various disciplines, and a multidisciplinary approach to tackle the interrelations between different domains. The availability of a generic infrastructure addressing the integration of software packages with a rich user interface is vital. Emerging grid computing techniques seem to be able to provide a general approach for integration and collaboration while retaining the division and autonomy of disciplinary domain experts. To address the organizational challenges that prevent a wider application of a multidisciplinary optimisation approach in a generic manner, a flexible environment supporting a powerful scripting language, rich visualization tools, and common mathematical computation capabilities is desirable. In this work, Matlab is chosen as the central stage for the problem solving environment, as it provides a broad spectrum of functions and algorithms for a wide range of applications including visualization and Graphical User Interface (GUI) building features. However, our approach allows that the toolkits developed can be easily integrated with other environments such as Python. [2]

The overall architecture adopted in this paper is shown in Figure 1, which is a simplified version of the general Grid-Enabled Optimisation and Design Search for Engineering (GEODISE) architecture [3] (http://www.geodise.org/). In the current implementation, Globus Toolkit 2.2 [4] is used to provide various low-level functionalities such as authentication, resource management, job submission, etc. These functionalities can then be exposed to end users via the interface to commodity technologies, such as Java, Python, Perl, etc. In the

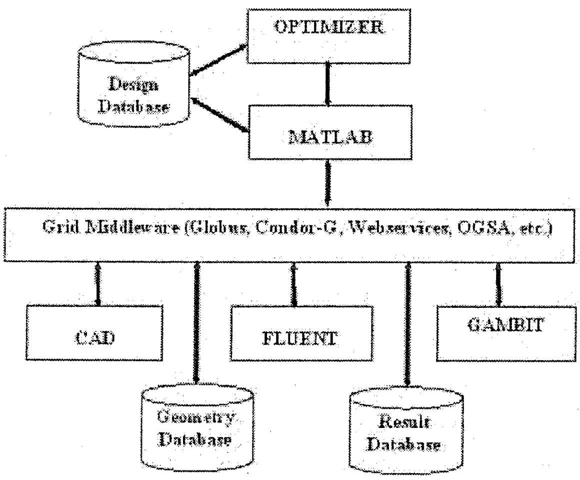

Fig. 1. Overall structure of the grid-enabled design optimisation

Geodise project, The Java Cog Kit [5] is used to expose Globus functionalities in the Matlab environment. Matlab is widely used in academia and industry to prototype algorithms, and to analyse and visualize data. Matlab also enables programmers to access Java classes, and therefore provides the functionality for code re-use and further developments. Detailed discussion on how various technologies such as Condor, Globus, etc., are used in Geodise can be found in [6]. Via the use of the Matlab functions provided in the Geodise Toolkit, the user is also able to submit his/her own code to computing resources, or to run software packages installed on the server. Access to database functionalities such as file archiving and retrieval and user notifications are also provided in the form of Matlab functions. Table 1 lists the functions implemented in the current version of the Geodise toolkit.

Table 1. Implemented commands in the Geodise toolkits

gd_archive	Stores a file in repository with associated metadata
gd_proxyinfo	Returns information about the user's proxy certificate
gd_proxyquery	Queries whether a valid proxy certificate exists
gd_createproxy	Creates a proxy certificate using user's credentials
gd_destroyproxy	Destroys the local copy of user's proxy certificate
gd_getfile	Retrieves a file from a remote host using GridFTP
gd_putfile	Transfers a file to a remote host using GridFTP
gd_jobsubmit	Submits a GRAM job to a Globus server
gd_jobstatus	Returns the status of the GRAM job specified a job handle
gd_jobpoll	Queries the status of a Globus GRAM job until complete
gd_jobkill	Terminates the GRAM job specified by a job handle
gd_listjobs	Returns job handles for all GRAM jobs
gd_query	Query metadata about a file based on certain criteria
gd_retrieve	Retrieves a file from the repository to the local machine

3 Two dimensional airfoil design using orthogonal basis functions

A two-dimensional airfoil design optimisation problem appropriate for concept wing design is studied here, using the Geodise toolkit. Instead of using airfoil coordinates directly, as is the case in many airfoil design applications, six basis airfoil functions are used to form a set of orthogonal functions to define the airfoil shape [7]. The goal of the optimisation is to minimize the drag/lift ratio by changing the weights of the six basis functions and the thickness-to-chord ratio. The basis airfoil functions were derived from a family of nine NASA SC(2) supercritical airfoils. The first three of these six basis functions are shown in Figure 2. A similar approach of using basis functions for defining airfoil shape was adopted by Ahn J. *et al.* [8], however, the basis functions used were not orthogonal and not derived from airfoil shapes. The use of orthogonal basis functions leads to a unique mapping from parameter space to the airfoil geometries, which is a desirable feature when optimisation methods are adopted to search for good designs. The airfoil geometry is defined first by upto six weight coefficients, followed by the adjustment to the specified thickness-to-chord ratio. A detailed discussion on geometry definition, meshing and solution is given in the next sections.

3.1 Problem and Geometry Definition

Shape parameterization methods in the context of multidisciplinary design optimisation have been investigated by a number of researchers. A comprehensive overview can be found in [9]. A combined shape, topology and configuration optimisation process for structures was recently reported in [10], in which a parametric model was constructed using the CAD tool ProEngineer. In most cases, the transfer of geometry between codes is implemented using standard neutral formats such as STEP, IGES, PATRAN, etc. Here, a parametric airfoil

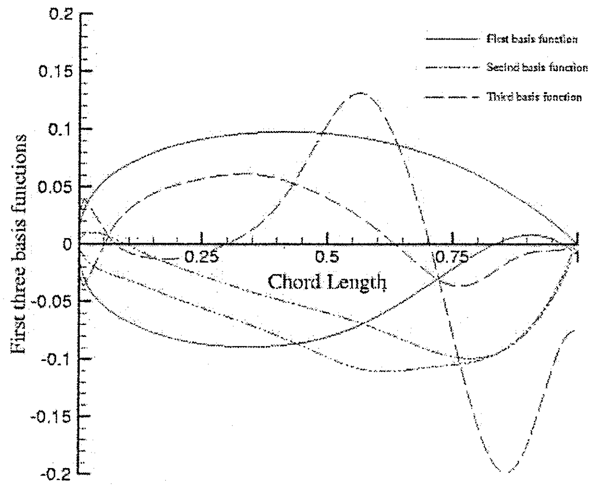

Fig. 2. First three airfoil basis functions used for airfoil geometry definition

geometry has been defined using ProEngineer. In this case the basis functions have been normalized with respect to the first member of the set and the thickness to chord ratio. Moreover, it has been shown from [7] that by adopting this approach good representation of the original airfoil can be recovered by simply varying the thickness to chord ratio and second weight, leaving the first function weight at unity and the remaining four at zero. This leads to a two-dimensional specification which is very simple to implement and interpret. These parameters, their initial values and ranges are listed in Table 2.

3.2 Mesh generation and CFD solution

To carry out optimization, mesh generation must be carried out in a batch mode, based on the geometry imported from ProEngineer. However, in general, a lot of work is needed to clean up the geometry in order to create robust, efficient meshes, to remove undesired features such as very short edges and surfaces (via surface merge or removal of short edges), and this process is often interactive in nature involving much graphic picking. Unique tagging can be used to replace the graphic picking and to deal with topology changes. However, this is not always possible, especially for complex models generated using a top-down design approach. To begin with here, an interactive session was used to generate a script file that is later used to run the meshing program in batch mode. The limitation of this approach is that the geometry topology must be maintained the same, and also that when a different version of meshing tool is to be used, another

Table 2. Design variable for the airfoil optimisation problem

Name	Lower bound	Inital value	Upper bound	Meaning
$w(2)$	-1.0	-0.98	1.0	Weight for the second basis function
tc_ratio	0.06	0.08	0.18	Thickness-to-chord ratio

interactive session may be needed to generate the script file. The generated mesh file is then imported into Fluent [11]. Node spacing is specified on the edges to control the mesh density. A triangular mesh is used for the flow field. Boundary layer meshes are attached to the upper and lower edges of the airfoil. A Reynolds-averaged two-dimensional compressible Navier-Stokes solver is used to obtain the flow field solutions.

Meshing jobs are submitted onto the computing server using the `gd_jobsubmit` function after a valid proxy has been established using the user's credentials. A unique job handle is returned and later used to check the status of the meshing job. Fluent CFD jobs are then submitted using the same mechanism. Results are stored into data files on the scratch directory on the server. After the jobs finish, `gd_getfile` is used to retrieve the results files and values of lift and drag coefficients are obtained by analyzing the resulting files.

4 Optimisation using Genetic Algorithm

An implementation of a population-based genetic algorithm (GA) [12] is adopted in the search process due to its robustness in finding the location of globally optimum designs. However, the design search is not conducted on the original landscape of the objective function, instead, a surrogate model is used in lieu of the true objective function. The surrogate model was constructed from results evaluated at twenty sample points in design space using a random Latin hypercube sampling technique. The landscape of the lift/drag ratio is shown in Figure 3 based on the results at these twenty points. Here, a kriging surrogate model [13] is constructed and searched using the Genetic algorithm. The best design found on the krigging surrogate model is then validated using a full CFD solution. The resultant airfoil shape and values of design variables and objective function are given in Figure 4. It can be seen that using a surrogate model can significantly speed up the search process and enables the exploration of multiple search methods, which would be impossible on the high fidelity codes. Furthermore, discrepancies between the landscapes of the true objective function and the approximated one could be addressed by evaluating the approximation error or introducing some adaptive update scheme based on the criteria of expected improvements. It is expected that such consideration will improve the effectiveness of approximation methods, and this will be studied in the future work.

5 Concluding remarks and future work

A two-dimensional airfoil design optimisation using CFD has been tackled using emerging grid technologies. Although the problem does not require as much computing power as a full-wing CFD analysis, it has demonstrated all the elements in a typical CFD-based shape optimisation problem. The Geodise computation toolkit is used in the study, along with a Genetic Algorithm and surrogate modeling methods. Grid technologies are still fast evolving, and the Open Grid Service Architecture (OGSA) [14] seems to be the future direction for grid services developments. Other issues may involve the use of a job scheduler for job farming and monitoring. Additionally, a rich GUI is desired for novice designers as well as the scripting language for expert users and further developments. More complicated geometries and approximation and optimisation frameworks will also be studied in due course.

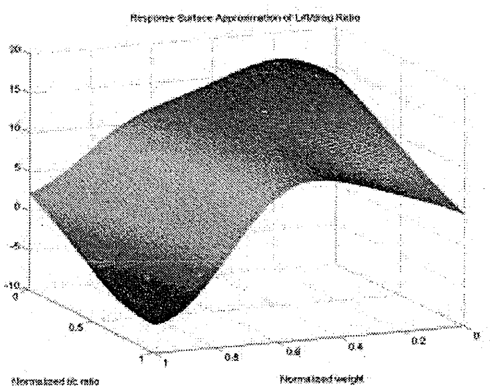

Fig. 3. Landscape of lift/drag coefficient ratio for airfoils with two design variables

Acknowledgements This work is supported by the UK e-Science Pilot project (UK EPSRC GR/R67705/01). The authors gratefully acknowledge many helpful discussions with the Geodise team. The authors also wish to thank Dr. Neil Bressloff and Mr. Andras Sobester for helpful discussions.

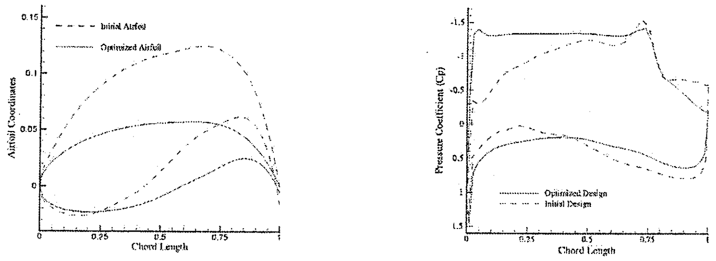

Fig. 4. Initial and Optimized airfoil (left) and corresponding pressure distribution (right). For the initial and optimized design, the weight for the second basis function is -0.9815/-0.5985, the thickness-to-chord ratio is 0.116/0.0716, and lift/drag ratio is 7.8363/14.9815

References

[1] Foster I., Kesselman C., Tuecke S.: The Anatomy of the Grid: Enabing Scalable Virtual Organizations. International Journal of Supercomputer Applications **15** (3) (2001) 200-222

[2] Python(2003) http://www.python.org

[3] Cox,S.J., Fairman,M.J., Xue,G., Wason,J.L., Keane,A.J.: The GRID: Computational and data resource sharing in engineering optimisation and design search.(2001) IEEE Computer Society Press

[4] The Globus Project (2003) http://www.globus.org

[5] Laszewski,G., Foster I., Gawor,J., Lane P.: A Java Commodity Grid Toolkit. Concurrency: Practice and Experience **13** (8-9) (2001) 643-662

[6] Pound,G., Eres,M.H., Wason,J.L., Jiao,Z., Cox,S.J., Keane,A.J.: A Grid-enabled Problem solving environment (PSE) for design Optimisation within MATLAB.(2003) International Parallel and Distributed Processing Symposium, April 22-26, 2003, Nice, France

[7] Robinson,G.M., Keane,A.J.: Concise Orthogonal Representation of Supercritical Airfoils. Journal of Aircraft **38** (3) (2001) 580-583

[8] Ahn,J., Kim,H.-J., Lee,D.-H., Rho,O.-H.: Response Surface Method for Airfoil Design in Transonic Flow. Journal of Aircraft **38** (2) (2001) 231-238

[9] Samareh,J.A.: Survey of shape parameterization techniques for high-fidelity multidisciplinary shape optimisation. AIAA Journal **39** (5) (2001) 877-884

[10] Langer,H., Puhlhofer,T., Baier,H.: An approach for shape and topology optimization integrating CAD parameterization and evolutionary algorithms.(2002)

[11] Fluent (2003) http://www.fluent.com

[12] Keane, A.J.: OPTIONS Design exploration system, (2003) http://www.soton.ac.uk/~ajk/options.ps

[13] Jones D.R., Schonlau,M., Welch W.J.: Efficient global optimisation of expensive black-box functions. Journal of global optimisation **13** (1998) 455-492

[14] Foster,I., Kesselman C., Nick,J., Tuecke,S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002