Progress and Challenges in the Application of Artificial Intelligence to Computational Fluid Dynamics

Alison E. Andrews*
NASA Ames Research Center, Moffett Field, California

This decade has seen the appearance of several attempts to apply artificial intelligence (AI) to problems in computational fluid dynamics (CFD). The purpose of this article is to propose an approach to analyzing such AI/CFD systems, to apply that analysis to four first-generation systems, and to use the results to assess the progress that has been made and highlight the remaining challenges. These first AI/CFD systems demonstrate that present AI technology can be successfully applied to well-formulated problems that are solved by means of classification or selection of pre-enumerated solutions (as opposed to construction, where solutions must be synthesized). Attempts to incorporate topics that are still in the realm of AI research or to apply AI technology to poorly understood or poorly formalized CFD tasks have some benefits but generally result in long system development times and a large investment of effort with no guaranteed payoff. A clear idea of the objective of an AI/CFD project and an awareness of the factors that affect its success are therefore of primary importance.

Introduction

ALL phases of computational fluid dynamics (CFD) research and practice can be regarded as being composed of two types of activities: those for which a computer is preferable or necessary, such as high-speed numerical processing, and those for which a human is necessary, such as making judgments based on knowledge, reasoning, perception, and common sense. Efforts to enhance CFD methodology have focused on both types of activities. Improvements to grid generators, solution methods, and algorithms have streamlined the computational part of obtaining a solution. The introduction of high-speed color graphics has aided humans in the decision-making associated with CFD problem setup and interpretation of solutions. Note that in all of these efforts, however, the traditional separation of labor between humans and computers has remained: humans make the decisions and computers do the number crunching. This would be an eminently practical and satisfactory arrangement if the kinds of decisions humans are called upon to make were always interesting, within their own range of knowledge, and could be made on a time scale compatible with human response time. This is not the case in CFD. Much of the reasoning involved in geometry definition, discretization, parameter adjustment, data format, code execution, graphical display, and intermediate solution assessment is routine, sensitive to error, requires some experience or expertise, and has become a rate-limiting step in the process of obtaining a solution. If it were possible, CFD researchers and practioners would happily delegate many of these sorts of tasks to a computer.

Several years ago, CFD researchers began to look to the field of artificial intelligence (AI) for approaches to automating some of the tasks still being performed by humans. Among the expected benefits of automating such tasks were relief from tedium; the codification, preservation, and distribution of expertise; consistency in application; and reduced solution turnaround time in code development and use. One of the earliest areas to be investigated was computer symbolic mathematics. Symbolic manipulation codes (of which Macsyma is perhaps the most well-known) have been used in CFD to perform stability and accuracy analyses and to generate Fortran code. Roache and Steinberg[1] report on the use of Macsyma in three-dimensional, boundary-fitted coordinate transformations. Reference 1 also gives a brief history of symbolic manipulation in CFD, outlines potential uses, and provides a useful reference to other work in the area. Progress is being made in addressing the speed and memory limitations encountered in early work,[2] but the use of such symbolic manipulation tools in CFD has not yet become widespread. This is due mainly to the difficulty of effective use of these tools; one must learn a new language (repeatedly, if use is infrequent), and obtaining an answer in its simplest, most usable form can be challenging.

The emphasis of this paper is on another area of great interest, that of knowledge-based systems (expert systems), an AI problem-solving approach that has met with some success in the solution of real-world problems in a variety of domains. Before much work had been done to combine expert systems and CFD, many predictions were made regarding the potential of such a union.[3-5] It was stated that AI would help in the acquisition of new fluid dynamic knowledge and render the use of CFD in aerodynamic design more effective.[3] Depending on the complexity of the problem and the availability of appropriate tools, it was estimated that knowledge-based system development could take from several months to more than a decade. None of the predictions included realistic information as to the difficulty of any particular AI/CFD application.

Since then, researchers have implemented knowledge-based systems in the areas of aerodynamic design, consultation in the use of large CFD computer codes, solution-adaptive grid refinement, and flow field zoning (in progress). It is time to step back and analyze these existing systems in terms of the nature of their targeted tasks, what kinds of knowledge they possess, how much effort was required to codify that knowledge, how quickly they were constructed, their performance, and what the return on investment (ROI) has been. Analysis of this sort helps to identify which aspects of CFD are most readily automated using AI. Some aspects of human

Received Nov. 18, 1986; revision received May 26, 1987.

*Research Scientist. Member AIAA.

decision-making in CFD are well formulated and admit straightforward application of existing AI techniques, whereas some require extensive conceptual groundwork to be laid and may be intrinsically more difficult to automate. In some cases, researchers will have to wait for (or contribute to) advances in the state-of-the-art in AI before they can hope to use it effectively in their target application. This information is necessary to engineers, scientists, and managers interested in applications of AI to CFD in both the near and long term. This paper proposes an approach to analyzing AI/CFD systems, applies that analysis to several existing systems, and uses the results to highlight the progress and remaining challenges in the application of artificial intelligence to computational fluid dynamics.

Analysis of Knowledge-Based Systems

Excitement over the prospect of combining AI and CFD to produce more powerful research and design tools has been attended by a certain amount of confusion. Which issues merit attention at the outset of an AI/CFD project? What can be learned from analysis and evaluation of existing AI/CFD systems? There is a tendency to focus too early on implementation issues such as representation, language, and machine. It is also common to rely on superficial descriptions of problem types, such as diagnosis or design, to guide selection of a solution approach or comparison of existing systems. To alleviate this confusion, it is first necessary to understand an intended or existing system at the knowledge level, a concept that has been proposed by Allen Newell:[6] "The knowledge level permits predicting and understanding behavior without having an operational model of the processing that is actually being done by the agent." A knowledge-level analysis for knowledge-based systems is analogous to the more familiar "specification" for conventional computer programs. Once the system has been analyzed at the knowledge level, it is then useful to look at where the knowledge is located, how much codification of the knowledge is required, and how the knowledge is used to solve the problem. These steps in the overall analysis approach are described more fully herein.

William Clancey has proposed a model for the knowledge-level analysis of a certain class of knowledge-based systems—those that solve problems by means of heuristic classification.[7,8] He makes a distinction between solution by classification/selection and solution by construction (analysis vs synthesis is another way to state this familiar idea). One section is devoted to a discussion of constructive problem solving,[8] but it does not suggest a model for a knowledge-level analysis of this class of systems. In the cases of several of the AI/CFD systems discussed herein, aspects of each class are presented. To attempt a knowledge-level analysis of these AI/CFD systems, elements of Clancey's model for classification will be appropriated, and new elements to handle construction operations will be added.

Clancey applies a knowledge-level analysis to a dozen well-known expert systems (including MYCIN, SACON, GRUNDY, and Drilling Advisor) that vary in problem type and representation. By distilling from each system what he terms its inference structure, he is able to detect a pattern: "These programs proceed through easily identifiable phases of data abstraction, heuristic mapping onto a hierarchy of pre-enumerated solutions, and refinement within this hierarchy. In short, these programs do what is commonly called classification."[7] An example of data abstraction is the translation of the statement "$M_\infty = 3.1$" to the statement "the freestream flow is supersonic." An heuristic mapping is a "direct, non-hierarchical association" between concepts in different classes,[7] "based on assumptions of typicality ... sometimes just a poorly understood correlation."[8] Refinement can be thought of as the inverse of the abstraction operation. Based on that common inference structure, some of the terms he proposes for analysis of programs that perform classification are data abstraction (quantitative-to-qualitative, definitional, or generalization), heuristic match, and data refinement (refinement, categorization, discrimination). The inference structure of a generic classification problem solver is shown in Fig. 1.

Additional terminology is required for the analysis of systems that synthesize solutions (as opposed to selecting them through heuristic classification). Such systems are generally difficult to characterize because of a lack of consensus as to the nature of the process; human constructive problem solvers often have different philosophies on the subject. However, elements that commonly appear in the process are decomposition (of a large problem into smaller subproblems), analysis (often numerical), construction (by composition of components or primitives, by generation of whole solutions, or by modification/perturbation of existing solutions), and evaluation (leading to another pass through the system or to a finally acceptable solution satisfying goals and constraints). The inference structure of one approach to constructive problem solving, heuristic design-by-redesign,[9] is described in Fig. 2. This approach usually does not contain the decomposition element described.

The first step in the analysis method proposed in the present study consists of a knowledge-level analysis that makes explicit what knowledge is needed by the system to solve its target problem. The knowledge-level analysis is performed by determining the system's inference structure with the aid of Clancey's classification model terms and relations and the additional terminology proposed for synthesis systems. The next step in the analysis takes into account the fact that existing numerical computer codes often play a part in the overall system; it is a determination of what knowledge resides where (including how much of the knowledge resides in the numerics). Then comes an evaluation of the amount of effort required to codify the knowledge. Little effort may be required if the task is well-formulated and some of it has already been encoded (perhaps in the form of a numerical code, as was mentioned, or in the form of a code user's manual). Much ef-

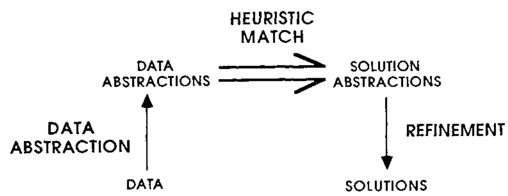

Fig. 1 Inference structure of a typical classification problem solver (from Ref. 7).

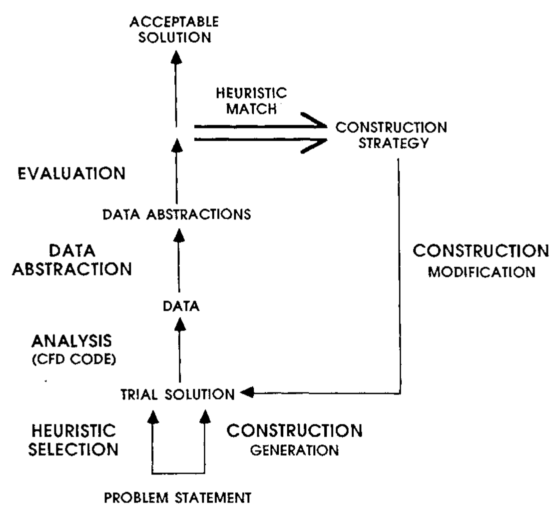

Fig. 2 Inference structure of a typical design-by-redesign construction problem solver.

fort may be required if the task involves laying conceptual foundations and little or none of the knowledge has been encoded in any form. Where the task lies on this spectrum has a great impact on the eventual system's ROI. Qualitative determination of the knowledge engineering effort required by a system involves both estimating the task's state of formulation and comparing the amount of knowledge encoding already done to that which remains to be done. The final step in an analysis of AI/CFD systems is a look at how the knowledge is used. Whether the knowledge is applied many times in an iterative portion of an often-run program (which has the effect of amplifying any value) or whether it is applied once in an infrequently-used system makes a difference to the ultimate value of the system. Each step of the described analysis contributes to an overall estimate of the probable ROI of a system. This approach to the analysis of AI/CFD knowledge-based systems is illustrated in the next section.

Case Studies

There have been several attempts to apply AI concepts to CFD problems in this decade, and they have met with varying degrees of success. Four of these systems are examined using the proposed analysis.

Expert Cooling Fan Design System (EXFAN)

One of the first implemented AI/CFD systems that attracted notice was Siu Shing Tong's Expert Cooling Fan Design System called EXFAN, which is a demonstration version of the more general Expert Design System (EDS) architecture he proposes.[10] EXFAN performs aerodynamic design of turbomachinery components by starting with an initial design and then iterating through analysis and redesign until the design goals are met within the specified constraints. The purpose of EXFAN was to demonstrate that such an AI/CFD design system could broaden the applicability of the CFD design code on which it is based, thereby reducing labor costs and achieving better design results. EXFAN is a rule-based expert system written in Lisp and coupled with a Fortran CFD analysis code. It was developed on a DEC VAX 11/780 in only a few person-weeks. Tong estimates that a practical design tool based on EXFAN could be built in several person-months. EXFAN performs better than an inexperienced designer and has shown promise of being able to outperform an expert (for the relatively simple cases tried) simply through an ability to analyze many more trial geometries in a given amount of time (this is possible when the human is the rate-limiting step in the process).

This information is interesting, but its usefulness to other system designers is limited. It is necessary to examine EXFAN at the knowledge level. EXFAN contains three main types of knowledge: 1) knowledge about the relationship between flow conditions and blade geometry parameters; 2) knowledge about how to use the CFD code (input format, execution, error-handling, and output interpretation); and 3) expert design knowledge (how to adjust geometric parameters to achieve goals and satisfy constraints, and when constraints must be relaxed in order to obtain a solution). An attempt to sketch the inference structure of the system is shown in Fig. 3. Reflected in this sketch are the iterative nature of the solution process and the elements of the process. Identification of the process elements aids in the identification of the necessary system knowledge. In this and all subsequent inference structure sketches, the system being described is the entire AI/CFD system. Most of the elements shown belong to the expert system portion. Any element that is not part of the expert system portion is indicated by including its source in parentheses below that element (e.g., the analysis element of the EXFAN system is provided by a CFD code, and the construction of the initial trial design is performed by a human user).

It should be noted that the first type of knowledge listed, that of how blade geometry affects flow conditions, resides implicitly in the CFD analysis code. It is extremely valuable knowledge if used properly. The second type of knowledge resides in the expert system, and its purpose is to make the valuable fluid dynamics knowledge in the analysis code more usable. The design knowledge also resides in the expert system, where it can be applied quickly and systematically to design problems.

The next issue to address is the amount of effort required to capture this knowledge. In the case of EXFAN, the CFD analysis code already existed and was in general use, so both the analysis knowledge and the code-use knowledge were in place (although the latter required some coding). The expert design knowledge had to be retrieved from an expert but was well-formulated and fairly straightforward to impart. In this sort of design approach, the knowledge is used repeatedly as the trial design is iteratively modified to obtain a solution, leading to a high amplification factor.

EXFAN was intended to be a demonstration system, and as such has probably not been overly profitable. It is, however, an excellent example of an AI/CFD system with large ROI potential. This potential stems from several factors: 1) much valuable knowledge was already encoded in the form of a CFD analysis program; 2) the additional design knowledge required was well-formulated and not difficult to encode; 3) the application of AI to capture the code-use and design knowledge added value to the existing CFD code; 4) AI was applied to a recognized bottleneck in the design process (turnaround in analyzing and resubmitting cases); and 5) the added value is amplified by the iterative nature of this design method. Tong attributes the large ROI primarily to the fact that the system is hybrid, as opposed to pure AI. It may be that he is referring to the advantages of working with existing codified knowledge and the potential for leveraging that knowledge with a modest AI effort. These advantages are not necessarily peculiar to hybrid systems, however, nor do all hybrid systems enjoy them. Tong's more recent work[11] involves two other AI/CFD systems: one that uses heuristic search to find an acceptable turbine cascade design in a situation where no human design knowledge existed, and one that uses a similar approach for distributed data as opposed to discrete design parameters. He has also been involved in the development of a design system incorporating some limited learning capabilities.[12]

PAN AIR Knowledge Sytem (PAKS)

PAKS, or PAN AIR Knowledge System, developed by Conner, Purdon, and Wamsley, is an AI/CFD system that aids users of PAN AIR, a well-known panel-method code widely used in airplane design.[13,14] The PAKS system is actually the first implemented part of a more comprehensive PAN AIR ap-

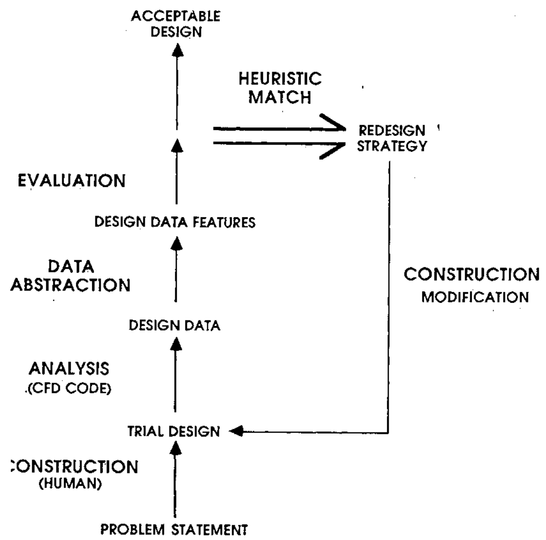

Fig. 3 Inference structure of EXFAN.

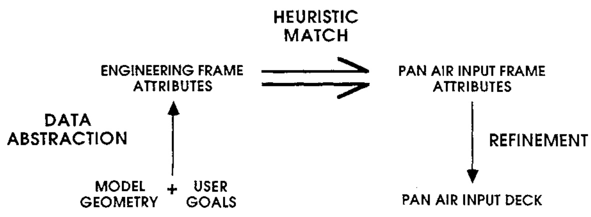

Fig. 4 Inference structure of PAKS.

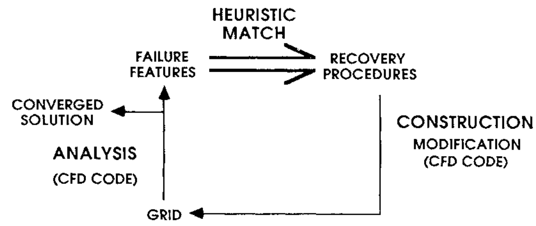

Fig. 5 Inference structure of ADAPT.

plication system, which is envisioned as consisting of PAN AIR plus three cooperating expert systems: one for geometry input preparation, one for the remaining nongeometrical input, and one for output analysis, interpretation, and selected graphical display. Nongeometrical input preparation was selected as the target for this first prototype system. PAKS takes the user-defined geometry and the user's goals and constructs a PAN AIR input deck. The motivation for this work stems from the scarcity of experienced users, whose experience is critical to the success and timeliness of design projects using PAN AIR. PAKS is an attempt to increase the impact of PAN AIR on the design community by making it easier to use. PAKS was originally developed on a XEROX Lisp machine using a MYCIN-derivative, rule-based, expert system building tool. It has since been reimplemented in S.1, a similar tool developed at Teknowledge, and the delivery system will be on a DEC VAX 11/780. Approximately half a person-year was invested in system development. PAKS performs input deck preparation at the level of an experienced user of PAN AIR.

PAKS contains several main types of knowledge: 1) documented knowledge about code operation, 2) experiential knowledge about code operation, 3) knowledge about consulting in the use of PAN AIR (what are common difficulties), and 4) knowledge of panel methods and general aerodynamics. A sketch of the inference structure based on the system description[14] is shown in Fig. 4. The input model geometry and user goals are abstracted to a set of "engineering frame attributes," higher-level concepts (such as speed regime and panel surface type) needed for problem definition and boundary condition selection. These engineering attributes are then matched to PAN AIR input frame attributes, which are the various data group subdivisions (global, network, flow properties) found in a PAN AIR input deck. Specific input commands are generated by refinement of the input deck attributes. Conner et al. correctly state that PAKS solves problems by means of structured selection, as can be seen by comparison of Fig. 4 with the characteristic inference structure for heuristic classification in Fig. 1. Although PAN AIR is not included in the figure, its presence as part of the overall capability (PAN AIR + PAKS) is taken into account in this analysis.

The documented knowledge about PAN AIR can be found in the PAN AIR User's Manual. This manual was the major source of the expert system's knowledge, accounting for approximately two-thirds of the rules in the knowledge base. Manuals are excellent sources of information in an already-encoded form that often needs little modification. The experiential knowledge about how to operate PAN AIR was not already encoded, but since that program has existed for many years, that knowledge is available and is probably the type that can be captured with rules. Although they are a relatively small percentage of the overall rule base, these experiential rules can make a substantial difference in system performance. Knowledge about consulting for an established CFD code is also in the category of not-previously-encoded-but straightforward-to-encoded knowledge. Knowledge about panel methods and aerodynamics is found both in PAN AIR (in the numerics) and in PAKS (in the relations pertinent to the problem setup phase, such as determining the proper boundary conditions for a discrete set of surface and wake types). In PAKS, the knowledge is used once per run. Since it forms part of a design program, system runs can be frequent, and amplification of the system's value results.

Conner et al. candidly admit[13] that the model geometry design and construction portion of the full application system would have the greatest impact on the present design process since that portion is presently the most challenging and time-consuming. However, they were wise to choose the nongeometrical input portion of the task as their first prototype. An attempt to automate the geometry task would probably involve some difficult knowledge representation issues and might wander from application into research. PAKS has a good ROI potential for some of the same reasons as does EXFAN: 1) PAN AIR already contains very valuable knowledge; 2) much of PAKS' knowledge was already encoded in the form of the user's manual, and the rest was not too difficult to encode; 3) through not technically a hybrid system, PAKS adds value to an associated existing CFD code (PAN AIR); and 4) the added value is multiplied by the number of runs needed to produce a design and the number of designs undertaken with PAN AIR.

Grid Adaptation

Dannenhoffer and Baron[15] report on an expert system embedded in a two-dimensional grid adaptation scheme. Their adaptation approach consists of successive grid refinement or coarsening by means of cell-by-cell division or collapse, depending upon local flowfield features. Features are determined by a simple thresholding algorithm applied to the flow data obtained from a Euler flow solver. Rather than attempt to find a threshold value that can always give good results, Dannenhoffer chose to use a value that works well most of the time and to incorporate a failure recovery capability in the form of an expert system to handle the rest of the cases. (For the purpose of ease of reference during this discussion, this grid adaptation system will be referred to as ADAPT. This is merely a convenience for the present author—the adaptation scheme is never referred to by name in Dannenhoffer et al.[15]) The primary purpose of building this expert system was to make the adaptation scheme more robust in a flexible, easy-to-extend way. The expert system is rule-based and is written in Fortran to avoid the need for inter-language interfacing. There is no indication of the length of development time, but it is assumed to be small when compared to that required for development of the CFD portion of the system. The performance of ADAPT is quite good, as evidenced by the five adapted grids Dannenhoffer presents as results, two of which required failure recovery.

ADAPT contains three types of knowledge: 1) how to solve the two-dimensional Euler equations to obtain fluid dynamic data; 2) when and how to refine or coarsen the computational grid; and 3) how to recover from a flow solver failure caused by grid effects. The conventional portion contains the first two types of knowledge described, thereby making up the bulk of the system, and is denoted by "(CFD code)" in the inference structure sketch of Fig. 5. The expert system portion contains only the failure recovery knowledge, which it uses to match failure features to appropriate recovery procedures. There is no attempt to abstract or refine information.

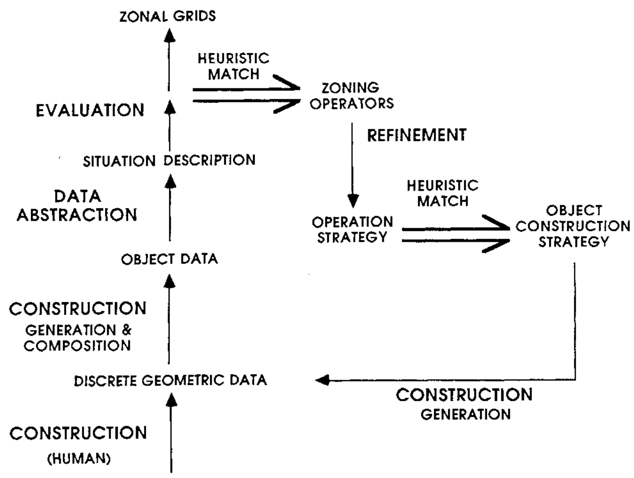

Fig. 6 Inference structure of EZGrid.

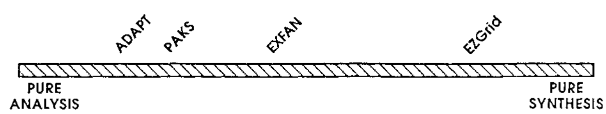

Fig. 7 Proportion of analytic/synthetic elements.

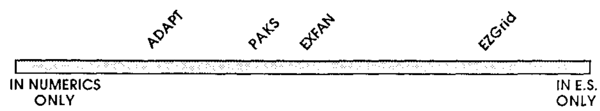

Fig. 8 Location of knowledge.

Compared to the effort involved in building the CFD code portion, the effort required to encode the failure recovery knowledge is relatively small. The knowledge base has fewer than 20 rules so far, enabling it to react to failures resulting from insufficient resolution at a body surface, refinement regions that are too small, and several other common problems. This knowledge may have existed previously in some form, but in general, Dannenhoffer must learn it rule by rule as he applies the code to new cases. The encoding of the knowledge is straightforward; its control structure remains intact in its Fortran representation. The overall system is iterative. A grid may be modified by cell refinement or deletion as many times as is required to achieve a prescribed accuracy in the flow solution. It is only when a solution does not converge that the expert system is invoked. Therefore, although it is associated with an iterative process, the expert system portion may not ever be called upon for a particular case, or it may be invoked only a few times.

ADAPT has the potential for a good return on investment. The main factors contributing to its good ROI are: 1) the flow solver/grid adapter already contained most of the knowledge involved in the program; 2) a relatively small effort was invested for the expert system portion since the failure recovery knowledge was fairly easy to acquire and encode; 3) value is added to an existing CFD code by extending its range of applicability; and 4) any aid to grid generation has the advantage that discretization is a necessary preliminary to most CFD analyses.

Expert Zonal Grid Generation (EZGrid)

EZGrid is an Expert Zonal Grid Generation system that partitions a two-dimensional flow field into four-side, well-shaped zones that are then individually discretized.[16] A zonal approach to computing a flow field is well documented.[17,18] EZGRID's method of solution is to design a flow-field zoning by step-wise application of zone construction operators until the entire flow field has been zoned. The purpose of EZGrid is to capture and codify the knowledge of CFD researchers who design and evaluate flow field zonings, leading to the possibility of streamlining grid generation for complex configurations and achieving more consistent (and perhaps better) discretizations. EZGrid is a rule-based expert system written in Lisp, MRS (a Lisp-based logic programming language developed at Stanford[19]), and C (for the computationally intensive portion). Development was begun on a DEC VAX 11/780 and is being continued on a Silicon Graphics Iris Workstation. To date, just over one person-year has been invested in the prototype, with probably one additional year required for a useful system. Present performance on easy problems is that of a novice; more difficult problems are not yet possible.

EZGrid has several types of knowledge: 1) knowledge of how to describe geometric objects in a way that can be reasoned about on one level and used in computation on another; 2) zoning knowledge (for a given situation, what are the possible/desirable next steps in the design of the zoning); and 3) knowledge of how to actually construct the zonal boundary curves. All of this knowledge resides in the knowledge base of the expert system. The final system will also contain knowledge about the relationship between zoning/discretization and the flow-field solution, grid generation knowledge (residing in an existing grid generation code), and knowledge about how to run the grid generation code. A sketch of the inference structure of EZGrid is found in Fig. 6. Note that the loop is not an iterative one but rather reflects the step-wise nature of the solution process.

Of the knowledge contained by EZGrid in its present form, little was either already existing or particularly easy to access. Some work has been done on geometric description and reasoning in the fields of computer vision and robotics, but nothing is in a form suitable for use in zoning. This type of knowledge has a perceptual basis, making it difficult to capture and encode for a computer. Zoning knowledge exists, to some extent, but only in the minds of a few experts, and it is not well-formulated. It is also common to have experts disagree on what constitutes the best zoning. Throughout the construction of EZGrid, much effort has been devoted to the formulation of a zoning methodology. More work is required in that endeavor as well as in the devising of analytical tests for zoning evaluation. Finally, knowledge about how to construct the zonal boundaries in two dimensions could probably be found in a user of CAD/CAM systems, but such resources were not available during EZGrid's development. EZGrid's knowledge is used repeatedly in each run, as it is applied at each step of the design of the zoning. EZGrid's role in the more general and frequent process of flow field discretization may lead to further amplification of its value.

At this point, the measurable return on investment for EZGrid is low. Its primary task, flow-field zoning, does not build on any existing computer program and, in fact, is a new area in itself. Building an AI/CFD system for flow-field zoning is an attempt to create value, not to enhance existing value, which tends to increase investment with no guarantee of commensurately increasing return. There are several factors, however, that suggest potential for a better ROI: 1) if the "real payoff" in automatic zonal grid generation is in three dimensions, then a two-dimensional system is at least a good starting point; 2) the codification of zoning methodology is valuable in promoting a better understanding of the process (this contribution to ROI is difficult to quantify; and 3) discretization is ubiquitous since it is an essential step in most CFD analyses, so any value created in this area will be multiplied many times over.

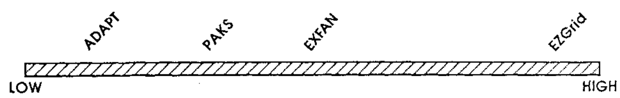

Fig. 9 Knowledge engineering effort.

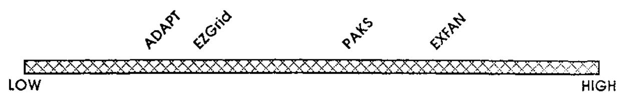

Fig. 10 Knowledge amplification factor.

Conclusions

The existence of implemented AI/CFD systems is an indication of progress in the application of AI to CFD. In order to gauge the amount of process as well as what challenges lie ahead, it is necessary to analyze existing systems in a meaningful way. An approach to analyzing CFD knowledge-based systems has been proposed, based, in part, on the concept of knowledge-level analysis. Examination of the knowledge level of a system by looking at its inference structure reveals the type of knowledge required by the system. It is then possible to look at where the knowledge resides (in a typical AI/CFD system, neither the AI nor the CFD portion contains all of the knowledge), to estimate how much knowledge encoding is required (and how difficult it may be), and to notice how the knowledge is used in the system (e.g., within an iterative loop).

The analyses performed in the previous section furnish useful insights into the factors that promote a successful AI/CFD effort. The findings of the case studies are summarized graphically in Figs. 7–10. In these figures, the combination of PAKS and PAN AIR is denoted simply by PAKS. Figure 7 shows the relative proportion of analysis/classification and synthesis/construction problem-solving methods in each AI/CFD system studied. The farther right one travels on this spectrum, the less guidance one can obtain, since there is generally less experience in automating this type of problem solving. Lack of precedent tends to lengthen system development time.

Figure 8 gives a qualitative indication of where the systems' knowledge resides. None of the AI/CFD systems (by definition) fall at the extremes of this spectrum; however, there is a notable spread among them. The larger the expert system portion of the AI/CFD system, the more effort required to build the overall system (assuming the CFD portion already exists), and the smaller the resulting leverage.

The relative amount of effort to formulate and encode the systems' knowledge is shown in Fig. 9. EZGrid is found at the high end of the scale because of the unformulated state of flowfield zoning knowledge and the difficulty inherent in representing it. Any "how-to" seminar on building expert systems will emphasize the importance of: 1) the existence of the knowledge (an expert must be able to describe how the problem is solved), and 2) the basis of the knowledge (knowledge with a cognitive basis is easier to handle than that based on perception). Flouting this wisdom can lead to a project with a definite "research" flavor.

Figure 10 shows a comparison of the knowledge amplification factors of the four systems. This factor is intended to reflect the way the system uses its knowledge (e.g., within an iterative process) and the frequency of use of the entire system. Large amplification can offset a large expenditure of effort, whereas small amplification has little effect on a system's payoff. A very high return on investment would correspond to positions in the left half of the scale in Figs. 7–9 and those in the right half of the scale in Fig. 10. EXFAN's modest effort combined with a large amplification factor will probably result in a higher ROI than that given by the smaller effort and smaller amplfication for ADAPT.

Several general observations can be distilled:

1) At present, automation of analytic or classification problem solving is better understood than automation of constructive problem solving.

2) It is presently feasible to perform limited constructive operations such as modification or perturbation of discrete parameters.

3) A relatively modest AI effort based on one or more existing numerical codes can leverage the knowledge that already resides in those codes.

4) Some aspects of CFD require extensive conceptual groundwork to be laid before AI can be introduced. For quicker system development, the knowledge must exist in a form that is straightforward to encode.

5) Valuable knowledge can be amplified by how it is used within the overall system and how frequently the system itself is used.

6) Addressing a recognized bottleneck tends to boost the ROI of a system.

7) Though not a direct contributor to measurable ROI, the formalization of expert knowledge and experience required in the development of a knowledge-based system can greatly benefit CFD experts by advancing the understanding of the discipline.

Assessment of progress leads to awareness of the remaining challenges in the application of this relatively new capability (both present and future AI technology) to CFD:

Challenge 1: Identify additional aspects of CFD to which present AI technology can be applied (all possible bottlenecks have not been addressed), and refine or build on existing first-generation systems.

Challenge 2: Increase the understanding and formalization of various aspects of CFD methodology. An attempt to apply AI to the automation of these aspects, while having low payoff in the short run, often spurs such development by exposing gaps or weaknesses in the accepted knowledge of a particular domain.

Challenge 3: Identify aspects of CFD that should be automated (such as geometry definition, design, code generation, methodology selection) but require AI techniques that are still research topics (such as geometric reasoning, representation of perceptual knowledge, constructive problem solving, learning, automatic programming, and qualitative physics). Attempts to apply those AI techniques before they are out of the research lab, though requiring a substantial investment of effort, can result in relatively early advanced capability. An additional benefit is that of valuable feedback to the AI researchers, which can serve to advance the state of the art more quickly.

It is not necessary to avoid constructive problem solving or always make an existing CFD code the centerpiece of any AI/CFD system, especially if a research objective is being pursued. It is simply important to have a clear idea of the objective of any AI/CFD undertaking and to be aware of the factors affecting such things as system development time and near-term payoff so that expectations are realistic.

References

[1]Roache, P.J. and Steinberg, S., "Symbolic Manipulation and Computational Fluid Dynamics," AIAA Paper 83-1952, July 1983. See also *AIAA Journal,* Vol. 22, Oct. 1984, pp. 1390–1394.

[2]Steinberg, S. and Roache, P.J., "A Toolkit of Symbol Manipulation Programs for Variational Grid Generation," AIAA Paper 86-0241, Jan. 1986.

[3]Mehta, U.B. and Kutler, P., "Computational Aerodynamics and Artificial Intelligence," NASA TM-85994, June 1984.

[4]Kutler, P., Mehta, U.B., and Andrews, A.E., "Potential Application of Artificial Intelligence Concepts to Numerical Aerodynamic Simulation," NASA TM-85976, June 1984.

[5]McCroskey, W.J., Kutler, P., and Bridgeman, J.O., "Status and Prospects of Computational Fluid Dynamics for Unsteady Transonic Viscous Flows," NASA TM-86018, USAAVSCOM TM-84-A-8, Oct. 1984.

[6]Newell, A., "The Knowledge Level," *Artificial Intelligence,* Vol. 18, No. 1, Jan. 1982, pp. 87–127.

[7]Clancey, W.J., "Classification Problem Solving," *Proceedings of the National Conference on Artificial Intelligence,* American Association for Artificial Intelligence, Aug. 1984, pp. 49–55.

[8]Clancey, W.J., "Heuristic Classification," *Knowledge Based Problem Solving,* Prentice-Hall, NJ, 1986, pp. 1–67.

[9]Dixon, J.R. and Simmons, M.K., "Computers That Design: Expert Systems for Mechanical Engineers," *CIME (Computers in Mechanical Engineering),* Nov. 1983, pp. 10–18.

[10]Tong, S.S., "Design of Aerodynamic Bodies Using Artificial Intelligence/Expert System Technique," AIAA Paper 85-0112, Jan. 1985.

[11]Tong, S.S., "Coupling Artificial Intelligence and Numerical Computation for Engineering Design," AIAA Paper 86-0242, Jan. 1986.

[12]Nicklaus, D.J., Tong, S.S., and Russo, C.J., "Engineous: A Knowledge Directed Computer Aided Design Shell," *Proceedings of the 3rd Conference on Artificial Intelligence Applications,* Computer Society of the IEEE, 1987, pp. 308–314.

[13]Conner, R.S., Purdon, D.J., and Wamsley, F.K., "Pan Air Consultation Using Expert System Techniques," AIAA Paper 85-4094, Oct. 1985.

[14]Conner, R.S., and Purdon, D.J., "PAN AIR Knowledge System," AIAA Paper 86-0239, Jan. 1986.

[15]Dannenhoffer, J.F. III and Baron, J.R., "Robust Grid Adaptation for Complex Transonic Flows," AIAA Paper 86-0495, Jan. 1986.

[16]Andrews, A.E., "A Knowledge-Based Approach to Flow Field Zoning for Computational Fluid Dynamics," Ph.D. Thesis proposal, Dept. of Mechanical Engineering, Stanford University, CA, 1985.

[17]Rai, M.M., "A Conservative Treatment of Zonal Boundaries for Euler Equation Calculations," AIAA Paper 84-0164, Jan. 1984.

[18]Hessenius, K.A. and Rai, M.M., "Applications of a Conservative Zonal Scheme to Transient and Geometrically Complex Problems," AIAA Paper 84-1532, June 1984, see also *Computers & Fluids,* Vol. 14, No. 1, 1986, pp. 43–58.

[19]Genesereth, M.R., Greiner, R., Grinberg, M.R., and Smith, D.E., "The MRS Dictionary," Stanford Heuristic Programming Project Rept. HPP-80-24, Dec. 1980, revised Jan. 1984.