NASA Technical Memorandum 85994

NASA-TM-85994 19840023019

Computational Aerodynamics and Artificial Intelligence

Unmeel B. Mehta and Paul Kutler

June 1984

NASA
National Aeronautics and
Space Administration

NASA Technical Memorandum 85994

Computational Aerodynamics and Artificial Intelligence

Unmeel B. Mehta
Paul Kutler, Ames Research Center, Moffett Field, California

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

TABLE OF CONTENTS

Computational Aerodynamics and Artificial Intelligence*

Unmeel B. Mehta and Paul Kutler
NASA Ames Research Center, Moffett Field, California

SUMMARY

A major role of computational aerodynamics in the future is that of acquiring new knowledge in a number of complex aeronautical problems. It will also play an increasingly effective role in the aerodynamic design and development process. The science of artificial intelligence can advantageously be applied to both these roles of computational aerodynamics. This report considers some aspects of artificial intelligence and *speculates* on how knowledge-based systems can accelerate the process of acquiring new knowledge in aerodynamics, how computational fluid dynamics may use expert systems, and how expert systems may speed the design and development process. In addition, the anatomy of an idealized expert system called AERODYNAMICIST is discussed. The report concludes with a discussion of resource requirements for using artificial intelligence in computational fluid dynamics and aerodynamics. Three main conclusions of this report are presented. First, there are two related aspects of computational aerodynamics: reasoning and calculating. Second, a substantial portion of reasoning can be achieved with artificial intelligence. It offers the opportunity of using computers as reasoning machines to set the stage for efficient calculating. Third, expert systems are likely to be new assets of institutions involved in aeronautics for various tasks of computational aerodynamics.

INTRODUCTION

The knowledge base of aerodynamics has experienced tremendous growth during this century (fig. 1). Before World War I, the foundations of aerodynamics were established. During the years between the beginning of World War I and the end of World War II, there was a surge in the establishment of low-speed aerodynamic knowledge. Following World War II, there was a great increase in high-speed aerodynamic knowledge. At present, we are at the threshold of acquiring new aerodynamic knowledge primarily through computational aerodynamics. This new knowledge is sought in a number of complex aeronautical problems, such as the understanding of helicopter flows, drag-reduction devices, high-lift devices, turbulence-control mechanisms, vortex flows and

* This is the corrected version, with six additional figures, of AIAA Paper 84-1531, Snowmass, Colo., 1984.

flows with separation, and highly maneuverable aircraft.

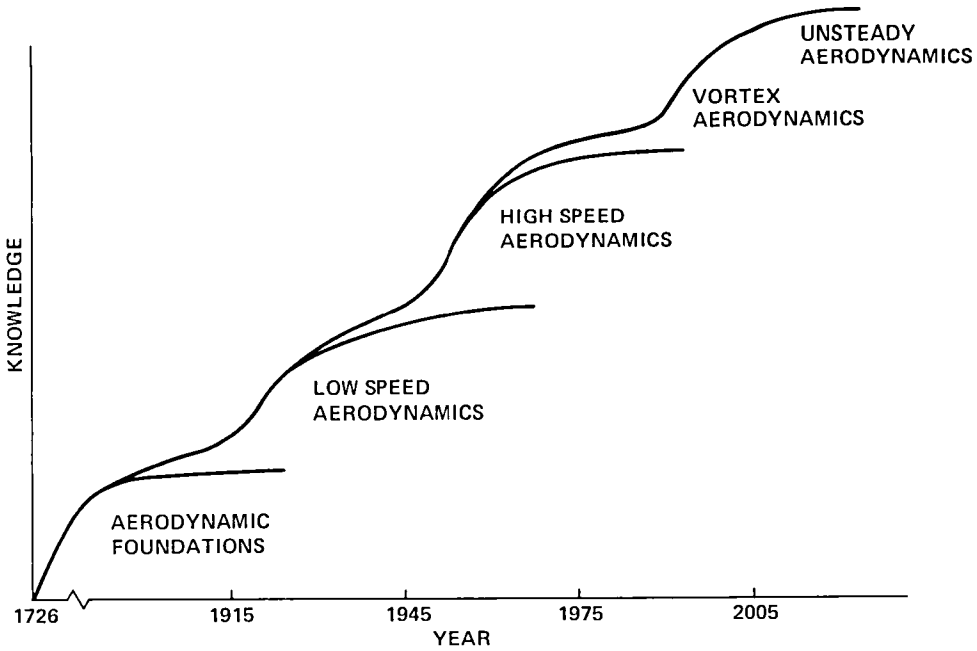


Figure 1. - Growth of aerodynamic knowledge (partly based on reference 1).

Another area in which computational aerodynamics will play a major role is aerodynamic design (fig. 2). Currently, computations based on the linearized inviscid and linearized inviscid plus boundary-layer approximations are routinely used in the design process, and those based on nonlinear inviscid and nonlinear inviscid plus boundary layer are in limited use (refs. 2 and 3). One can safely predict that before the end of this century, numerical solutions of the Reynolds-averaged Navier-Stokes equations will likely be routinely applied to solve aeronautical flow problems.

There are two possible applications of artificial intelligence (AI) to computational aerodynamics. First, in the process of acquiring new knowledge, it can make some of the complex research problems manageable by efficiently using available knowledge. Second, in the process of applying available knowledge to tractable aeronautical design and development problems, it can reduce the time required for their numerical solutions. This report speculates on these roles of AI in computational aerodynamics.

The extent to which computational aerodynamics can provide new knowledge depends on computational efficiency, manageability of computational complexity, and improvement of turbulence models (fig. 3). These factors are pacing the advances in computational aerodynamics (see also references 4-7). Computational efficiency includes both computer speed and memory and the speed of numerical algorithms. Manageability of computational complexity includes grid generation, body definition, data-base and programming logic management, data analysis, and computational method formulation. On the other hand, how effective computational aerodynamics is likely to be in applying knowledge to the aerodynamic design process depends on the speed

at which a design process can be completed with the use of computational aerodynamics, and on the degree of confidence the designer has in computational aerodynamics relative to his confidence in experimentation and analytical methods (fig. 4). These are the factors pacing the acceptance of computational aerodynamics by management. Artificial intelligence is likely to play a useful role in all of the above factors (figs. 3 and 4).

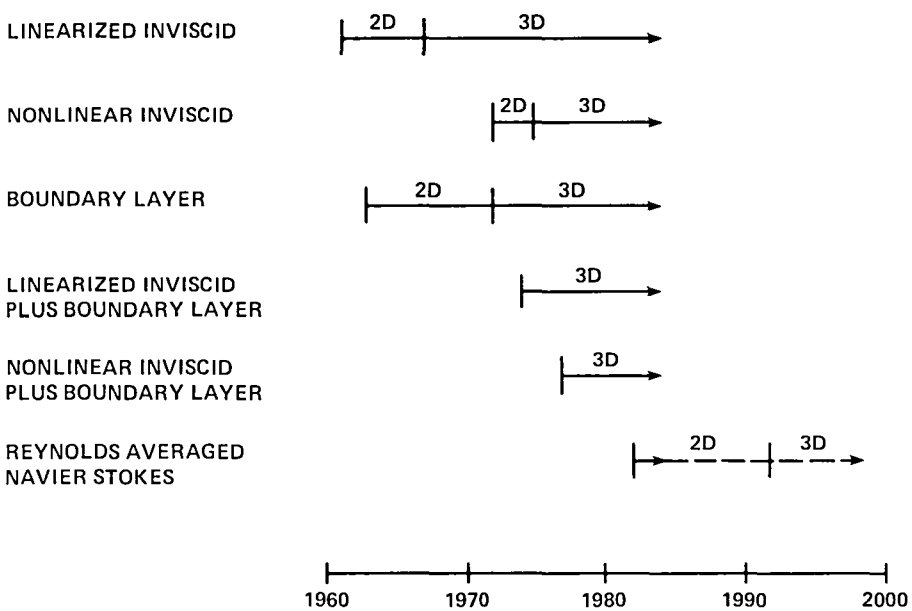


Figure 2. - Computational aerodynamics in design and development of flight vehicles (past dates are based on reference 7).

The potential applications of knowledge-based systems, an applied side of AI, are enormous in many human endeavors dealing with problems of a scientific, technical, and commercial nature. Realizing this, Japan in 1981 embarked on a plan to develop the fifth generation of computers; these computers will not be information (data) processing systems, but knowledge-information processing systems (ref. 8). The U. S. industrial response was the creation in 1983 of the Microelectronics and Computer Technology Corp. Its largest and most expensive program is nicknamed alpha-omega for advanced computer-architectures (ref. 9). Also in 1983, the Defense Advanced Research Projects Agency (DARPA) announced a new program called Strategic Computing and Survivability, to promote development of a new generation of superintelligent computers for military use (ref. 10). Great Britain, in response to the Japanese plan, has formulated the Alvey Programme for Advanced Information Technology with emphasis on the development of knowledge-based systems (ref. 11). The European Common Market countries have also begun the European Strategic Programme on Research in Information Technology (ESPRIT) with one major area of research being information and knowledge engineering (ref. 12). These activities have come about because of the belief that "knowledge is the new wealth of nations" (ref. 8). Thus, it is possible to state that computational aerodynamics and aerodynamic knowledge in knowledge-based systems can be the new asset of institutions involved in aeronautics.

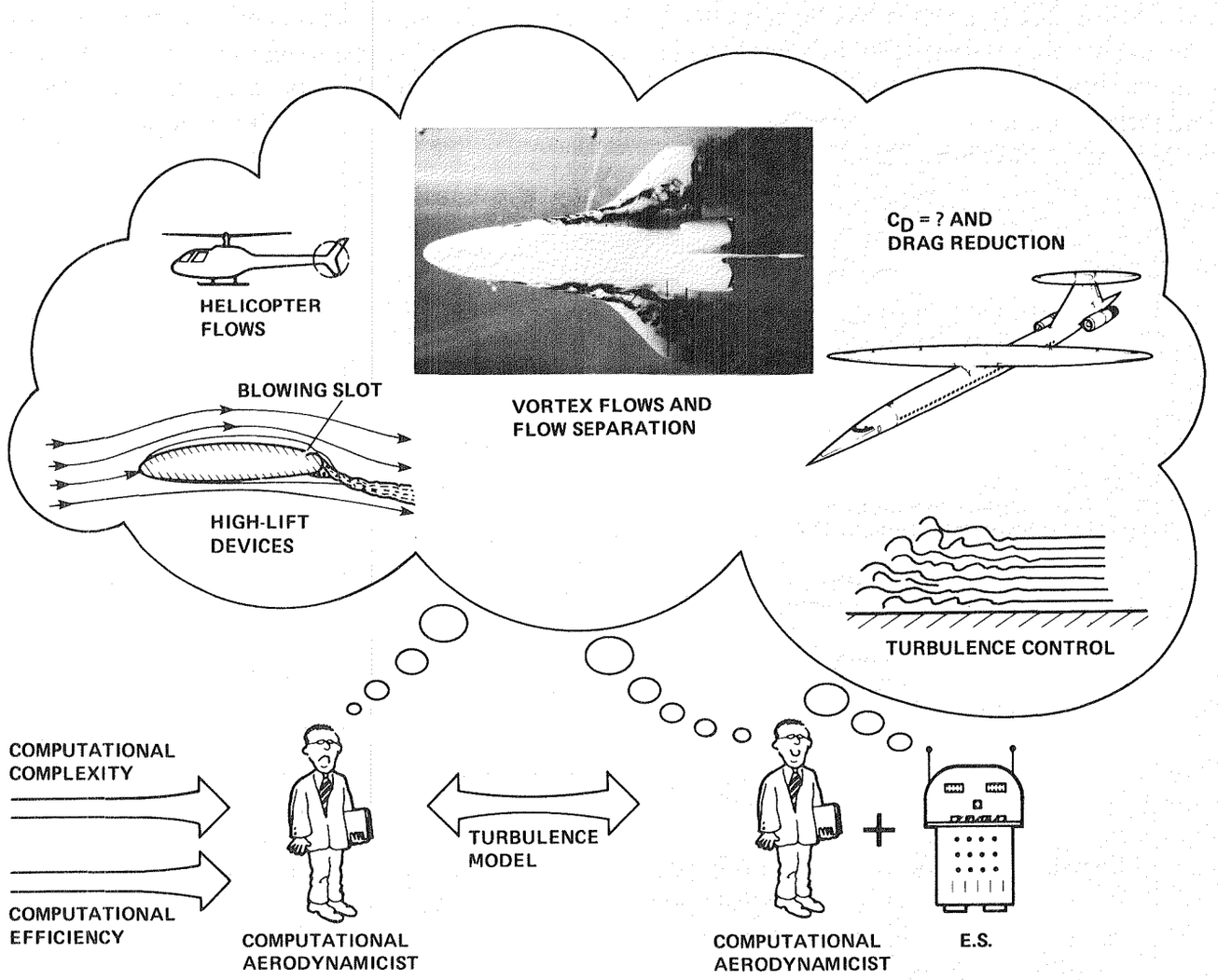


Figure 3. - Artificial intelligence to assist computational aerodynamics in the acquisition of new knowledge.

This report has four purposes: (1) to survey some aspects of AI that may be relevant to computational fluid dynamics (CFD) and in particular to computational aerodynamics; (2) to describe how expert systems, a class of knowledge-based programs, are likely to accelerate advances in aerodynamics; (3) to discuss possible expert systems in CFD; and (4) to discuss how expert systems are likely to speed the design and development process. A brief description of the origins of AI and of its various disciplines is first presented. What are expert systems? What can they do? Where can they be utilized? These questions will be addressed. Then, a few examples of existing expert systems are presented. Next, basic concepts and tools for constructing expert systems are discussed. After speculating on how AI is likely to influence computational aerodynamics, the anatomy of an idealized expert system called AERODYNAMICIST is described. Further speculations are presented on the likely use of expert systems in CFD and in the aerodynamic design and development process. A discussion of resource requirements for effective applications of AI in computational aerodynamics concludes this report.

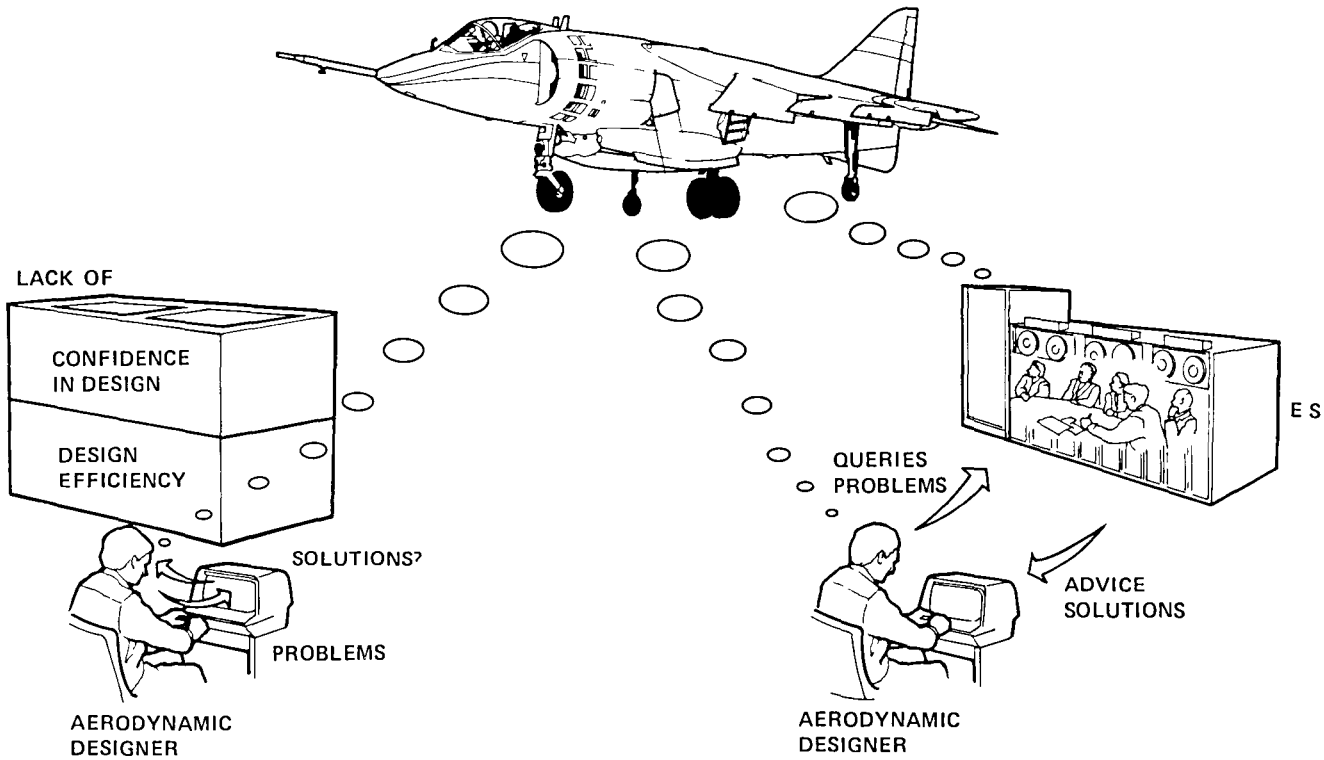


Figure 4. - Artificial intelligence to assist computational aerodynamics in aerodynamic design and development process.

As a caveat, the reader should keep in mind that the perspective of the authors is that of active *researchers* in computational aerodynamics and in CFD, but not in the field of AI.

The authors would like to thank William B. Gevarter for his comments and suggestions, which have helped to improve the presented discussion of the science of artificial intelligence.

ARTIFICIAL INTELLIGENCE

In 1844, Augusta Ada Byron, Countess of Lovelace, possibly the first computer programmer, recognized that the dichotomy between the programmer and the computer was false (ref. 13). Today, a computational aerodynamicist manipulates algebraic symbols leading to the development of a computer program, whereas a computer is confined to arithmetic calculations or data processing (number crunching). This division of labor between the computational aerodynamicist and computer is neither necessary nor appropriate. Computers are capable of manipulating any sort of symbol. In 1937, Alan Turing outlined a machine, now known as the Turing machine, that contained the notion of a universal machine capable of doing anything that can be done by any machine, including symbolic manipulation, and of what can or cannot be computed.

The origins of AI can be traced to mathematical logic, computation, and cybernetics (refs. 14 and 15). The developments of mathematical logic and new ideas concerning computations during the 1930's and 1940's led to the development of the modern computer, a descendent of the "analytical engine," a machine conceived by Charles Babbage around 1843. Complexity of doing computations with a computer and interest in simulating intelligent behavior in the computer led to the emergence of the science of AI. In 1950, Turing proposed a test, now known as Turing's Test which would indicate whether or not a machine was intelligent. In this test, if an interrogator, separated from the person (or machine) under interrogation, could not tell for certain whether he or she was communicating with a person or with a machine, then the machine could be said to be intelligent. The concepts of cybernetics and self-organizing systems also affected the emergence of AI by influencing the fields of logic and computation. The phrase "artificial intelligence" became popular at the Dartmouth Summer Research Project on Artificial Intelligence in 1956 (ref. 15). At that meeting, the discipline of AI is considered to have been born. John McCarthy is credited with coining the above phrase; Turing is called the father of AI.

The word "intelligence" means the ability to learn, understand, or deal with new or trying situations (ref. 16). Another meaning of this word is the ability to apply knowledge to manipulate one's environment or to think abstractly as measured by objective criteria. It derives from the Latin "legere" meaning to gather, and hence to select. The Latin "intellegere" means to select among and, therefore, to know, to understand, and to perceive (ref. 8). If an artifact or a device can gather, select among, know, understand, and perceive, then we have an artificial intelligence or an intelligent machine. Artificial intelligence is the part of computer science concerned with concepts and methods of symbolic inference by a computer and with the symbolic representation of knowledge to be used in making inferences (ref. 8). Thus, computer programs are designed to create computer systems exhibiting the characteristics we associate with intelligence in human behavior.

There are two principal disciplines of research in AI (fig. 5): cognitive science and intelligent machines. The first is the theoretical side of AI, and it deals with modeling human processing of information. The second is the applied side of AI, and it is concerned with constructing computer programs that perform expertly in cognitive tasks of problem solving. This second branch breaks down into subdisciplines based on applications: language understanding, vision systems, automatic programming, theorem proving, and intelligent robots. At present, those applications which do not fall into one of these subdisciplines are put into the subdiscipline called problem solving.

If a comparison were made between current intelligent machines and intelligent humans, it would be observed that both the computer system and the human brain would have certain limitations. The computer system does not have all five sophisticated human senses (sight, hearing, smell, taste, and touch), although some computer systems do have sight and hearing capabilities. The computer system generally lacks common-sense reasoning. Finally, it does not usually have an instantaneous flash of recognition or intuition. On the other hand, the conscious state of a human brain has a short attention span. It is not capable of entertaining more than a few, say three or four, hypotheses simultaneously, and it is limited in speed. At present, the difference between future intelligent machines and intelligent humans is unknown because of a lack of sufficient information (ref. 17).

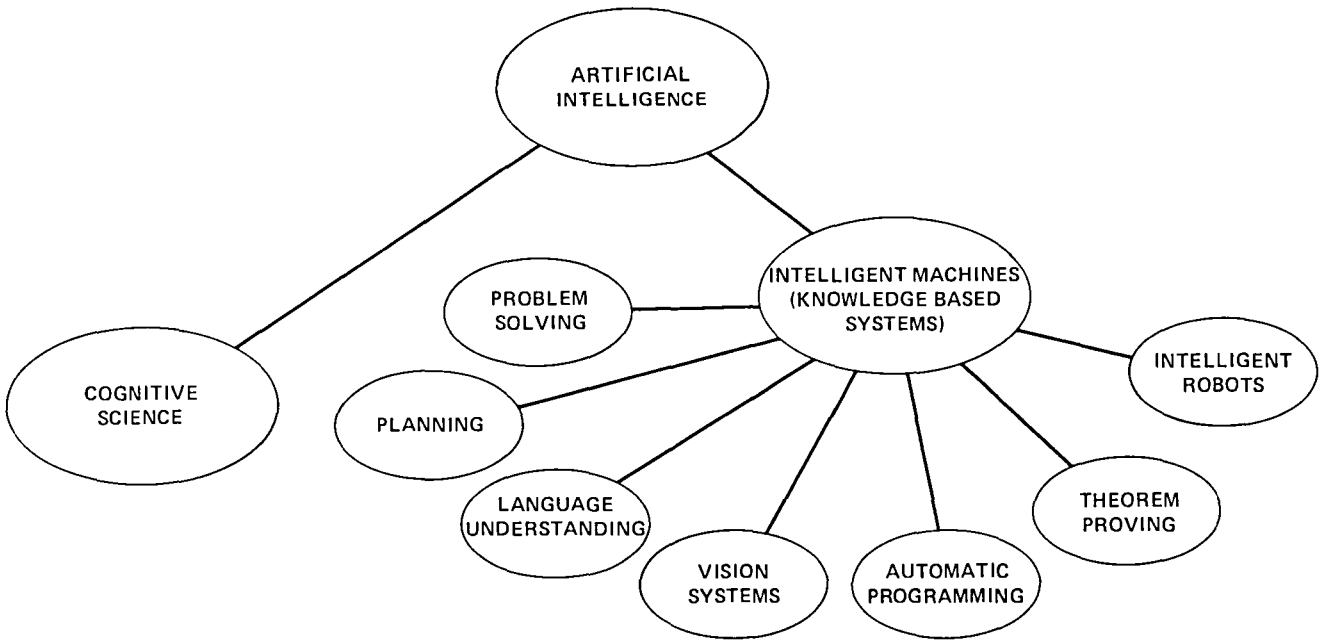


Figure 5. - Branches of artificial intelligence.

Knowledge-based systems are computer programs that can reason, make judgments, and draw conclusions in a specific domain by drawing on encoded factual and heuristic knowledge, problem-solving methods, and inference procedures (fig. 6). Some programs can even understand written and spoken words. Knowledge is usually inexact, incomplete and ill-defined. *Just as amateurs become experts incrementally, amateur knowledge-based systems can be caused to evolve into expert knowledge-based systems.* Therefore, those knowledge-based systems are known as expert systems.

EXPERT SYSTEMS

An expert system is a knowledge-based computer program in some domain that performs a specialized, usually difficult, professional task at the level of (or sometimes beyond the level of) a human expert. The minimum accepted range of behaviors associated with the nature of expertise in expert systems consists of knowledge, public and private, about a narrow specialized domain, and skill at solving some of the domain problems. MACSYMA (ref. 18) and DENDRAL (ref. 19) are examples of such systems. These systems will be described below. Some expert systems have begun to explain results, for example, PUFF (ref. 20), and to learn, for example, EURISKO (ref. 21). In addition to these capabilities, systems capable of exhibiting the other behavioral characteristics associated with the word "expertise" were nonexistent as of 1981 (ref. 22). These are abilities of reorganizing knowledge, breaking rules by understanding not only the letter of the rule but the spirit as well, knowing when a problem is outside the sphere of expertise,

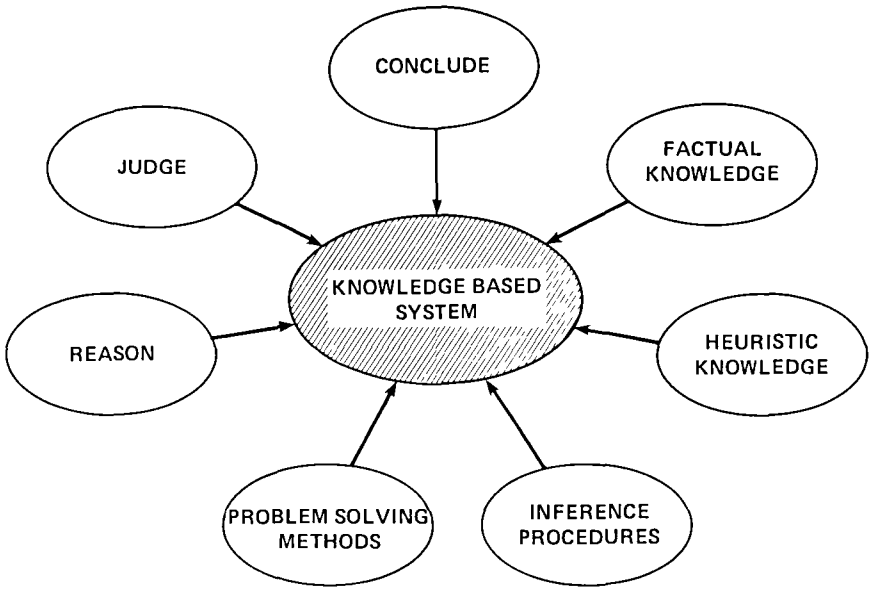


Figure 6. - Features of knowledge-based systems.

and of gradually decreasing expertise as the boundaries of expertise are approached. Further, expert performance means, for example, the level of performance of expert aerodynamicists doing research, design, or development in aerodynamics, or of very experienced people doing scientific, technical, or managerial tasks. The power of an expert system lies in the knowledge it possesses rather than in a collection of domain-independent solution methods (ref. 23).

In most technical and scientific fields, experts select among difficult choices by symbolic manipulations, inferential procedures, and by drawing upon working rules of thumb or heuristics based on experiential knowledge. Expert systems are particularly suited for the mode of thinking that is primarily reasoning rather than calculating.

Expert systems work especially well in dealing with two generic kinds of problems. First are the pure combinatorial problems in which pursuit of an exact or optimal solution would lead to a combinatorial explosion of computation. Second are the data-interpretation problems requiring analysis of a large amount of data. When successfully applied to these problems, expert systems can have substantial economic value. They are particularly powerful when expert performance is based largely on compiled experience. Expert systems can make, and have made, a difference of millions of dollars on the bottom line. They can also alter the perception of management concerning realities. These benefits are achieved when expert systems are designed with one or more of the following objectives (fig. 7): (1) sustaining growth through duplication of expertise, (2) capturing and distributing expertise for new ventures, (3) combining the expertise of many experts, (4) solving intrinsically complex problems that resist solution by other means, (5) managing knowledge, (6) providing a competitive advantage, (7) preserving perishable expertise, and (8) improving skills of low-skill workers. Some of these objectives are discussed with case studies in reference 8. The net effect of meeting any one of these objectives is to increase the number of available experts (fig. 8).

Figure 7. - Objectives for which expert systems are advantageous.

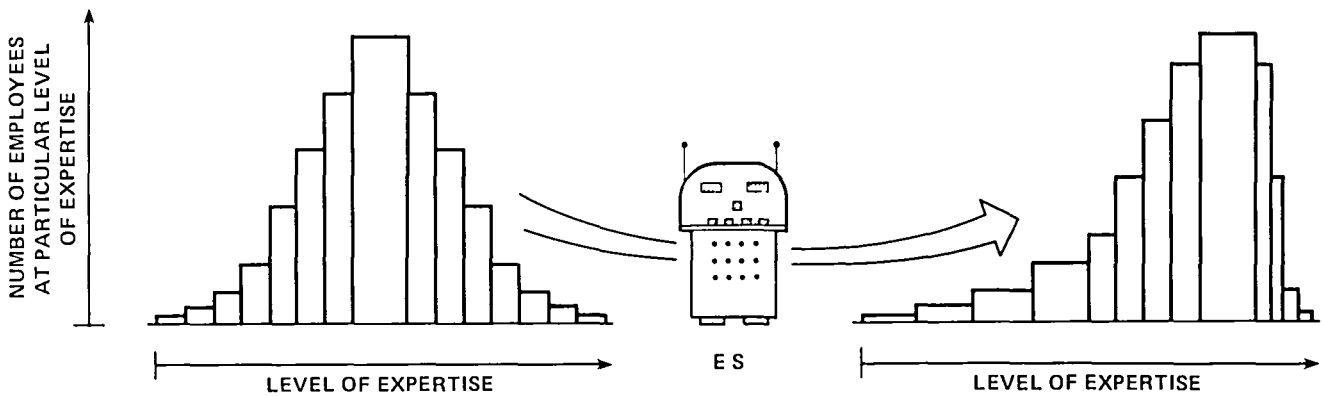


Figure 8. - Expert system increases the availability of expertise.

During the past 20 years, a number of experimental and operational expert systems have been developed for diverse domains. Such systems were developed in areas of bioengineering, the chemical industry, computer science, education, engineering, genetics, law, management science, manufacturing, mathematics, medicine, the military, and resource exploration. Furthermore, there are expert systems for developing expert systems. The functions of some of the existing systems are briefly explained in references 8 and 24-28. Applications that demonstrate the usefulness of expert systems are described below for routinely solving difficult real-world problems.

An expert system in mathematics, called MACSYMA (ref. 18), is a general-purpose set of programs that can solve a variety of mathematical problems far beyond the abilities of most mathematicians. It has algebraic-manipulation capabilities, as well as the numerical subroutine library, IMSL, and an extensive plotting package. It can perform at least 600 distinct mathematical operations (ref. 29). These include the four basic arithmetic operations, differentiation, definite and indefinite integrations, solution of equations including ordinary differential equations, solution of systems of linear or nonlinear algebraic equations, simplification, factorization, matrix operations, vector algebra, order analysis, and finite or infinite sums of Taylor series expansions. MACSYMA

uses both algorithmic and nonalgorithmic procedures, mathematical knowledge, and heuristics. It does not use search and reasoning under uncertainty; it does not explain its reasoning. It is used extensively by mathematicians, physicists and engineers.

In the domain of the chemical industry the DENDRAL expert system surpasses all humans in determining the relatively small set of possible molecular structures of known atomic constituents generated by given mass spectrographic, nuclear magnetic resonance, and other chemical experimental data (ref. 19). This system has three functional parts; namely, planning, generating, and testing. The planning part reduces the effort of the problem solver by using encoded heuristic knowledge of expert chemists. This limits the search for matching the generated structures with the data. The generating part generates systematically all possible partial organic structures that meet the constraints set by the planner and those apparent in the data. Then it expands these substructures in all plausible ways, and it simulates the behavior of the resulting structures in a mass spectrometer. The testing part compares the simulated spectra with the experimental spectrum. This whole process leads to accurate, high-quality results in a fraction of the time that an exhaustive search would take. DENDRAL is widely used by chemists. A significant point about this system is that the theory of mass spectrometry was incomplete, and the rules about it were inexact. This led to the development of the META-DENDRAL system, which automatically inferred these rules from examples successfully analyzed by humans.

CADUCEUS, a knowledge-intensive system in the domain of internal medicine, attempts to make a diagnosis at a level of expertise that permits it to solve most of the clinical pathological problems which baffle human experts (refs. 24 and 30). It covers more than 80% of all internal medicine; it incorporates more knowledge of internal medicine than any human can. Its knowledge base includes about 500 diseases and more than 3,500 manifestations of disease. It consists of an extremely large semantic network of relationships between diseases and symptoms with approximately 100,000 associations. This system is undergoing formal clinical trials, and it is likely to be used eventually as a diagnostic aid to physicians' assistants, in rural health clinics, in military medicine, and in space travel.

R1/XCON is an expert system that configures requests of customers for VAX computer systems at Digital Equipment Corporation from a customer's order of various standard and optional components (ref. 24). It breaks up the problem into subtasks and it does each of them in order as follows: (1) checks orders for completeness and gross problems, (2) (theoretically) puts components into CPU cabinets, (3) places unibus boxes filled with components in unibus cabinets, (4) places panels in unibus expansion cabinets, (5) performs a system floor layout, and (6) lays out the system cabling. After completing these subtasks, R1/XCON provides a configuration overview diagram, cabinet assignment detail diagrams, unbundled component ordered list, unconfigured component list, forgotten component list, unused capacity list, etc. Before this system was developed, the resident DEC experts thought that this job could not be done by a computer system. R1/XCON has been reported as saving DEC millions of dollars each year.

Knowledge, reasoning, expertise, and complexity are the key words associated with these and similar systems. Knowledge is the grist of the expert mill.

Hayes-Roth et al. (ref. 24) consider *ideal* expert systems to have seven fundamental characteristic features. First, they are considered to act expertly if they produce high-quality results in a minimal amount of time. Although high performance is sometimes difficult to quantify, quality

and adroitness of performance are necessary for systems to be experts. Second, expert systems employ symbolic reasoning. This allows explicit symbolic representation of knowledge based on logical foundations and on first-order predicate calculus. Third, expert systems have general problem-solving ability in the domain of their expertise, along with knowledge of basic principles. This determines the degree of intelligence of an expert system. Fourth, expert systems are utilized in complex and difficult domains. If a particular domain is not complex and difficult, then that domain does not qualify for an expert system. Fifth, expert systems take a problem stated in some arbitrary initial form and reformulate it in the form appropriate for processing. Sixth, expert systems have the ability to reason about their own processes. Seventh, an expert system is characterized by the generic task that the system is constructed to perform. Most of these tasks are control, design, diagnosis, debugging, instruction, interpretation, monitoring, planning, prediction, and repair. The above seven features are current fundamental goals of building expert systems. However, no existing system has all of these features.

The expert-system domain must be well chosen. Not all domains are amenable to building expert systems. In domains that are well bounded, expert systems work the best. There are generally seven stages of development of an expert system – conception, feasibility demonstration, prototype construction, extended use in a prototype environment, acceptance of the performance of the prototype system, commercial system construction, and commercial system release. Very few existing systems have gone through all of these stages. The unfavorable characteristics of the domain for building an expert system may terminate the expert system project in its initial stages of development. Another factor which may limit the development of the expert system is the lack of knowledge-engineering tools.

A knowledge engineer builds an expert system in the domain of interest by tapping the knowledge and expertise of an expert in this domain. The expert must be acknowledged to perform the problem-solving task well. He must be to able to articulate his expertise, which comprises: (1) factual and heuristic knowledge; (2) inferential, judgmental, and experiential procedures; and (3) problem-solving methods. Further, he must be available for constructing an expert system. If there are a number of experts that meet these requirements, they may have differing and conflicting views. In such cases, one of them is appointed as knowledge czar, and the expert system is built based on his views.

BUILDING EXPERT SYSTEMS

Two basic concepts are sufficient for building expert systems: symbols and search (ref. 31). Expert systems basically manipulate symbols and perform problem-solving tasks generally using search. Symbols are sufficient for any intelligent action (ref. 31). They are used for encoding knowledge of experts into systems that would act intelligently. Depending on the nature or complexity of the problem domain, search is required for efficient use of the expert knowledge. The concept of symbols underlies symbolic representation of knowledge and the manipulation of that knowledge (symbolic reasoning). The process of building expert systems with these basic concepts can require years of effort. However, the expert system technology has evolved to the

point where a few building tools and languages, which are derived from basic AI programming languages, exist for constructing expert systems. In this section we describe the above basic concepts, the basic languages, and some of these tools. In addition to these, we indicate how expert systems differ from conventional programs.

Symbols are strings of characters, for example, *transonic, 20.0,* and *jogging.* The symbol structures usually employed are a type of data structure, called list structures, containing symbols, for example, *(Multiply 7 Z)* and *(difference 8 (divide 7 8)).*

There are different methods for representing knowledge (ref. 14). In expert systems for computational aerodynamics and fluid dynamics, knowledge is most likely to be represented by the propositional doctrine (ref. 32) and by procedures. The propositional doctrine utilizes a declarative knowledge base plus an inference mechanism. In declarative representations, the static aspects of knowledge, such as facts about objects, events, and their relations, are stressed. Knowledge is represented declaratively by methods based on mathematical logic, such as first-order predicate calculus which allows manipulation of knowledge by logical operations and rules of inference. Examples of logical and arithmetic inferences are illustrated in figure 9. Production systems (ref. 14) are equivalent to expressions of first-order predicate calculus. The basic concept of these systems is that the knowledge base consists of production rules in the form of condition-action pairs, that is, IF-THEN expressions. In this, the action part of the rule pair is executed only if the condition side is invoked for a piece of knowledge or in the reverse manner when substantiating hypotheses. On the other hand, procedural representations of knowledge stress how to use knowledge in well-specified situations (ref. 14). Knowledge is contained in procedures that know how to do specific things, how to find relevant facts and make inferences.

LOGICAL INFERENCE	ARITHMETIC INFERENCE
GIVEN { RULE 1 IF R = P + Q THEN R = Q + P RULE 2 IF R = P + Q THEN P = R − Q	
IF z = x + y z = 36 x = 10	IF x = 10 y = 26 z = x + y
THEN z = y + x (RULE 1) y = z − x (RULE 2) y = 26	THEN z = 36

Figure 9. - Logical inference versus arithmetic inference.

Expert systems are problem-solving systems. These usually have three main components, a data base, a set of operators, and a control strategy. The data base is the knowledge base associated with the current task-domain and the desired goal. Operators, such as rules of inference, help achieve the goal by manipulating the data base. A control strategy is used for deciding what operator to apply and where to apply it. A method of search for an appropriate operator sequence for achieving the goal is an example of a control strategy. A problem-solving method is

formulated as a search procedure in many expert systems. Problem-solving methods are required for intelligent action based on symbolically represented knowledge.

Search is generally unavoidable, but unlike the knowledge base it is not critical in expert systems. There are different search techniques (refs. 14, 24, 33, and 34): (1) the data-directed or forward chaining strategy, which starts from known conditions and proceeds toward the goal; (2) the goal-directed or backward chaining method which begins from the goal to be achieved and progresses toward facts; (3) the heuristic search technique, which generates and tests solutions; (4) the difference reduction or mean-ends analysis, which tries to reduce the difference between the current and the goal state; (5) the blackboard model, which mixes forward and backward chaining strategies, and in which multiple sources of knowledge check the status of the emerging solution on the blackboard in order to decide what these sources can contribute; and (6) the bidirectional search, which involves searching from both ends of a search space. The data-directed, goal-directed, and bidirectional search strategies could involve blind search strategies, which are not practical for large problems. The heuristic search technique reduces the amount of computation by using heuristic knowledge about the problem. Feigenbaum considers the blackboard model to be the most powerful (ref. 33).

For manipulating symbols, the language development and selection are driven by convenience, efficiency, and availability of special features, although any programming language may be used. There are three major families of languages: functional languages, for example, LISP; logic programming languages, for example, PROLOG; and object-oriented languages, for example, SMALLTALK. These languages produce descriptive programs emphasizing what is to be done. On the other hand, languages such as FORTRAN, PASCAL, ADA, BASIC, and PL/1 produce procedural programs emphasizing how the computation is to be done.

In an attempt to create intelligent programs, list-processing languages were developed during the 1950's. LISP (*LIS*t *P*rocessing) evolved out of experience with list-processing. It was created by McCarthy (ref. 35) in 1958. It is the second oldest language after FORTRAN. LISP and its variants are extensively used as the AI programming languages in the United States. The functional nature of LISP is illustrated in figure 10 by comparing a LISP program with a FORTRAN program for computing a factorial. The LISP program is very much like the mathematical task of computing the factorial. Functions in a LISP program define formal mathematical expressions. The program FACTORIAL is represented in the same data structure as all the other data, namely, list structure. This program is recursive, unlike a FORTRAN program, which cannot call itself.

Mathematically, the formal language of symbol structures is generally predicate calculus. Therefore, there is considerable work being done in logic programming, for example, PROLOG (*PRO*gramming in *LOG*ic). It is primarily used in Europe and Japan. In PROLOG, relations or predicates are defined by sentences of predicate calculus.

Since it takes a long time to build an expert system, special tools have recently begun to be developed for knowledge-engineering tasks. These tools are themselves expert systems. Most of them have evolved from existing expert systems. In this case, these tools are either skeletons of existing systems or of parts of these systems. Some of these tools are for knowledge representation, some are for building inference systems, and some are for both. Tools for building inference systems are useful only in domains that can utilize control strategies similar to those from which

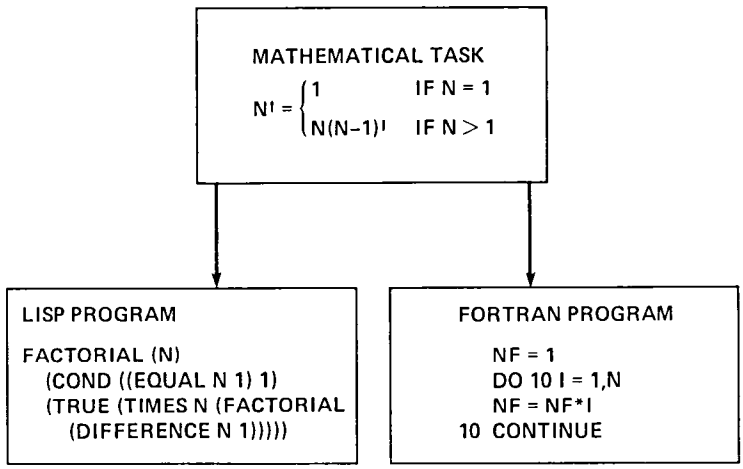


Figure 10. - Functional programming versus von Neumann programming.

these expert tools were derived. It is a difficult yet essential task to choose an appropriate tool for building a particular system. Problem characteristics determine solution features. These, along with the required capability of the expert system, determine system features. Solution features and system features guide the choice of tool features for building the expert system. These considerations and guidelines for choosing an appropriate tool for building an expert system are discussed in reference 24. Furthermore, that reference describes and evaluates a number of available tools.

The basic concepts of AI and the characteristics of conventional programs result in a number of differences between expert-system programs and current computational aerodynamic programs. These are listed in table 1 for different features of these programs. Because of some of the features, namely programming style, program architecture, types of data, and programming tools that are used in developing expert-system programs, these programs grow incrementally, and they are usually easy to modify. On the other hand, because of the same features, the conventional programs for computational aerodynamics grow by revision, and they are difficult to modify. Consequently, expert-system programs are more useful and can be quickly developed, as illustrated in figure 11.

Another way of comparing these programs is to consider the hierarchy of solution methods, as shown in figure 12. Algorithmic programs for narrow domains, for example conventional computational aerodynamic programs, are usually powerful but not general or domain-independent. On the other hand, programs that are domain-independent are not that dependable. In between these two we have expert system programs. Those expert programs that do not contain algorithms are more general but less powerful than comparable programs using only algorithms. However, expert systems that contain both algorithms and non-algorithmic procedures are both powerful and more general than comparable conventional (algorithmic) programs. An example of such a system is MACSYMA. Note that almost all expert system programs for computational aerodynamics will be of this type.

FEATURE	EXPERT-SYSTEM PROGRAMMING	COMPUTATIONAL AERODYNAMIC PROGRAMMING
EMPHASIS	WHAT IS TO BE DONE	HOW IT IS TO BE DONE
SOLUTION METHOD	INFERENCE RULES AND HEURISTIC SEARCH	ALGORITHMIC (PROCEDURAL)
PROGRAM ARCHITECTURE	CONTROL STRUCTURE SEPARATE FROM DOMAIN KNOWLEDGE	DATA AND CONTROL INTEGRATED
DATA BASE	PRIMARILY SYMBOLIC	PRIMARILY NUMERIC
TYPES OF DATA	IDEAS, KNOWLEDGE, NUMBERS, AND PROGRAMS	NUMBERS
DATA RELATIONSHIPS	COMPLEX	SIMPLE
PROGRAMMING STYLES	FUNCTIONAL, OR LOGICAL, OR OBJECT-ORIENTED, OR COMBINATIONS OF THESE	VON NEUMANN STYLE
PRIMARY LANGUAGES	LISP, PROLOG, SMALLTALK, AND VARIANTS OF THESE	FORTRAN
PROGRAM GROWTH	INCREMENTAL GROWTH	GROWTH BY REVISION
PROGRAMMING TOOLS	AVAILABLE	BECOMING TOOL CONSCIOUS
MODIFICATIONS	USUALLY EASY TO MODIFY	DIFFICULT TO MODIFY

Table 1. Comparison of expert-system programming and computational aerodynamic programming for various programming features

EFFECTS OF ARTIFICIAL INTELLIGENCE ON COMPUTATIONAL AERODYNAMICS

Computational efficiency, computational complexity, and modeling of turbulence determine the extent to which computational aerodynamics is likely to be effective in acquiring new knowledge in aerodynamics. Computational aerodynamics is now being applied to increasingly time-consuming and complex problems, and to problems in which the nature of turbulence is poorly understood. The science of artificial intelligence will indirectly help improve computational efficiency. It is also the contention here that expert systems will assist in managing computational complexity. Further, the reasoning power of expert systems will aid in development of turbulence models. Therefore, AI will accelerate the process of acquisition of new aerodynamic knowledge (fig. 13). In this section, we discuss how AI is likely to assist computational aerodynamics.

Computer hardware has advanced tremendously in the last 29 years, from the vacuum tubes of early computers to very large-scale integrated (VLSI) circuits, producing a net improvement in computation cost effectiveness of 10^3 (refs. 2 and 4). In the same period, both the architecture of computers and the programming languages used to control them have remained substantially the same. This has limited the full exploitation of VLSI circuit technology. Although there are vast differences in hardware and performance of the first four generations of computers, they are

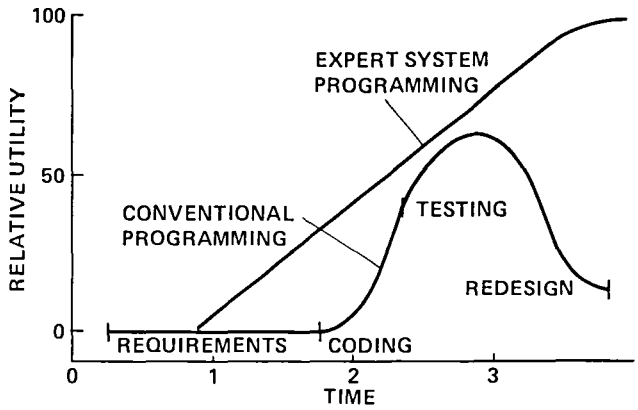


Figure 11. - Advantage of expert-system programming over conventional programming (partly based on reference 36).

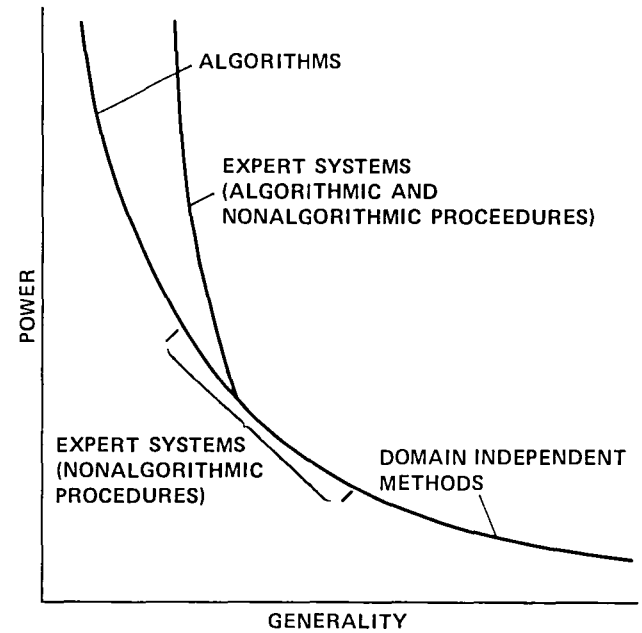


Figure 12. - Hierarchy of solution methods.

all based on a single basic design – the von Neumann processor. Even array processors that have single-instruction, multiple-data (SIMD) path architectures use sequential instruction-stream, von Neumann programming methods. Since programming languages, such as FORTRAN, are modeled on the von Neumann processor, programs are complex, concerned with the smallest data entities, and rarely reusable in constructing new programs. Consequently, the writing of software is becoming more expensive every year (refs. 37-40). Software is the limiting factor in putting computer power to its maximum use. It is generally believed that AI is the cornerstone of next-generation computer technology (ref. 41). This will definitely be advantageous to computational

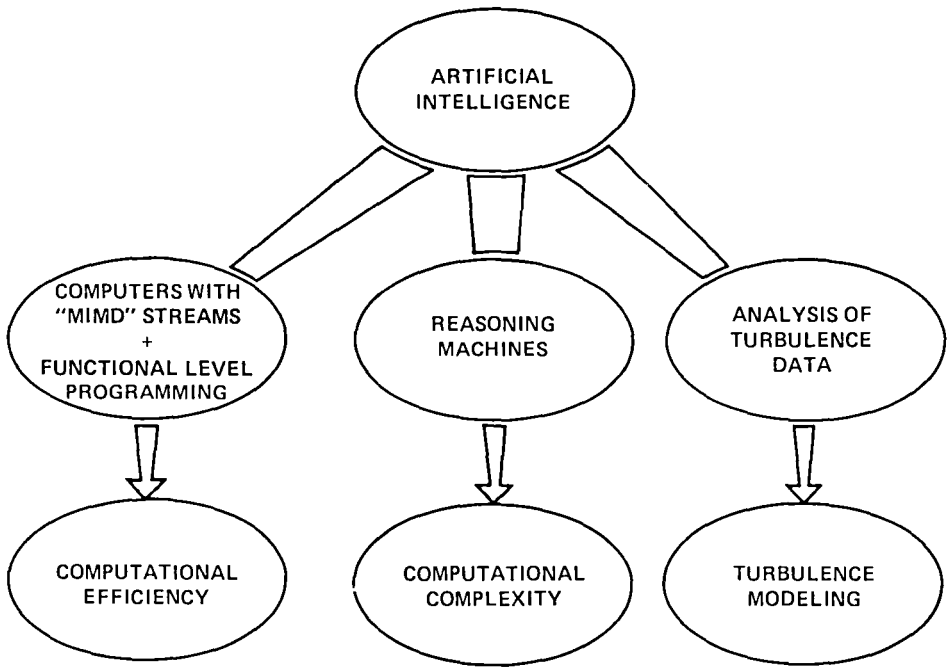


Figure 13. - Effects of artificial intelligence on computational aerodynamics.

aerodynamics. Note that it may be the worldwide interest in knowledge-based systems rather than in CFD that is going to be the driving force behind the development of the fifth-generation computer.

In the view of many researchers in the field of computer architecture, the fifth-generation computers will use multiple-instruction, multiple-data (MIMD) streams and many rather than just a few processors (refs. 41-43). This will make a break with the von Neumann architectures and programming languages. Two of the likely language types to emerge as predominant languages on the fifth-generation computers are the functional-level programming (refs. 44 and 45) and a combination of LISP and PROLOG. In the functional-level programming, programs are put together with so-called program-forming operations to build new programs. These in turn can be used to construct even larger ones. This approach allows parallel or concurrent operations to be expressed easily. LISP is an example of functional programming style, but it retains some features of von Neumann programming. The functional-level of programming along with expert systems for automatic programming (refs. 29, and 46-50) will greatly reduce the software cost and complexity of FORTRAN-like computer codes.

There are two related aspects of computational aerodynamics (fig. 14): *reasoning* and *calculating*. Reasoning is based on factual and heuristic knowledge of both computational procedures and aerodynamics; and it is carried out by symbolic manipulation and inferential procedures. Calculating is based on reasoning. It is impossible to calculate without doing some reasoning. Although limitations in computational speed are considered to be a pacing item of the advances in computational aerodynamics, the mere existence of a piece of hardware that can execute instructions rapidly does not, by itself, guarantee that a particular problem can be solved

quickly. Again, it is reasoning that is going to extract high performance out of a computing machine. The science of artificial intelligence offers the opportunity of using computers as reasoning machines to set the stage for efficient computations.

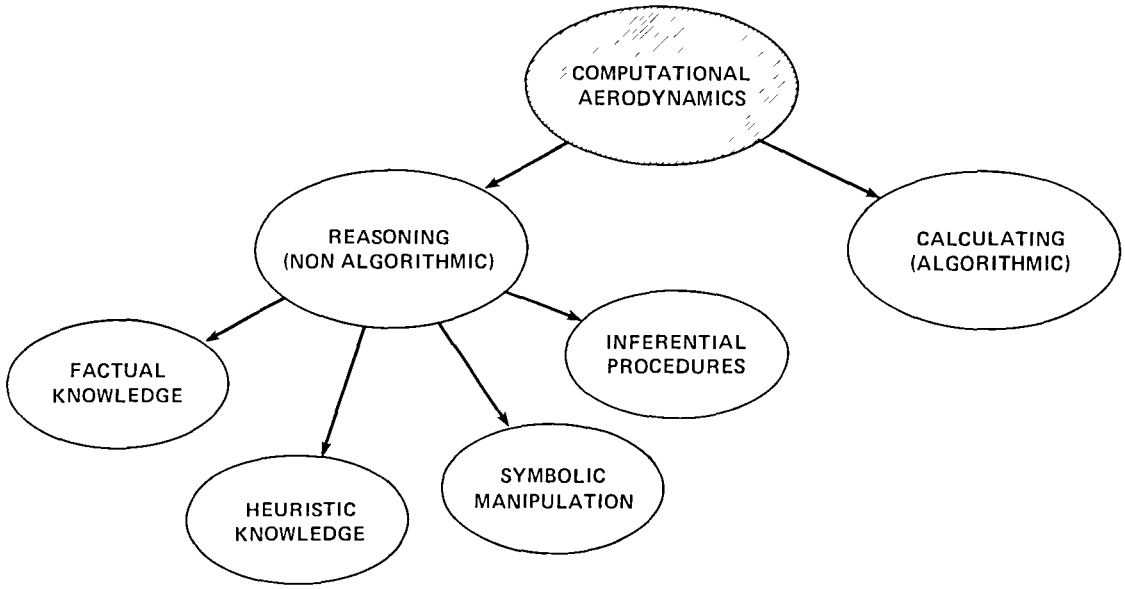


Figure 14. - Aspects of computational aerodynamics.

It behooves the computational fluid dynamicist to exploit the expertise of systems like MACSYMA for reasoning before computation. There are at least two advantages of algebraic programs over purely numerical ones. First, a result in algebraic form is more valuable than one in numerical form. Second, simplifying an expression algebraically before evaluating it numerically reduces the computation time. Furthermore, there are at least four advantages of automatic algebraic computations. First, months or years of work with pencil and paper can be done in a few minutes. For example, in 1847 Charles Delauney began algebraic computation of the position of the Moon as a function of time (ref. 13). It took him 20 years to complete this calculation. In the process he also committed three errors. In 1970, this computation was repeated on a computer in 20 hrs. Second, a human being is more likely to make a mistake than is a computer system. Third, a general mathematical expression stating a scientific theory can be easily simplified under certain assumptions by algebraic manipulations which may be very difficult for the human being. Fourth, a human being is freed from usually tedious algebraic manipulations to pursue more stimulating and interesting mental scientific activities.

The domains of computation and aerodynamics are complex and difficult. Obviously, the domain of computational aerodynamics is then complex and difficult. Expert systems in these domains will be a great benefit to the computational aerodynamics community as discussed in some of the following sections. These systems will become intelligent assistants to a computational aerodynamicist. They will accelerate the advances in computational aerodynamics, and they will speed design and development processes. These achievements are possible simply because these assistants are the reservoirs of knowledge, and because they are capable of making manageable

the complexity associated with difficult computational problems. In addition, the expertise in continuously updated software of an expert system remains an asset to the problem-solving process forever. Unlike its human counterpart it never cries, dies, moves on, or contains obsolete knowledge.

An expert system can be of tremendous utility in analysis of experimental data. For example, a laser Doppler system produces a huge amount of data. These data can be statistically processed; they can be used for developing theories of turbulence. The expert system can seek orderly patterns in irregular data and thereby hit on predictable laws of turbulence. This is not far-fetched; already, a system called BACON simulates some of the important processes of scientific discovery. It has independently rediscovered laws of planetary motion, electrical resistance, and Black's law of temperature equilibrium, as well as concepts of atomic weight and specific heat from physical data (refs. 51 and 52). Another example is EURISKO, which develops its own theories and ideas once it is given the principles of a discipline (refs. 21 and 53). A third example is LOGIC THEORIST, which follows rigid rules and logic to find alternative proofs of theorems in "Principia Mathematica" (Russell and Whitehead). And the fourth example is the development of elaborate mathematical theories from a few simple axioms of the set theory (ref. 54).

ANATOMY OF AN EXPERT SYSTEM: AERODYNAMICIST

Figure 15 shows the *idealized* anatomy of an expert system called AERODYNAMICIST, which processes knowledge and information (data). An expert system with such an anatomy could be used in the domain of computational aerodynamics. It could also be used in various subdomains of computational aerodynamics and CFD. None of the existing expert systems has the anatomy of this idealized system, although some parts of it are contained in every existing system. The idealized system is based on requirements of the above domains. It is designed considering what is technologically feasible and is likely to be feasible. Further, some parts of it are concocted from generic structures of expert systems described in references 8, 24, 27, 36, and 55.

AERODYNAMICIST can be used in three modes: the expert mode, the learning mode, and the teach mode. It contains the following subsystems: (1) a language processor or an input/output system, (2) a system manager, (3) a knowledge base comprising factual and heuristic knowledge, (4) an inference system or a method of reasoning and of solving problems, (5) a knowledge and inference acquisition system, (6) an expertise controller, (7) a status recorder, (8) an explanation system, and (9) an information processor. Each component of this system is described briefly below.

An aerodynamicist or a student communicates with the expert system using the language processor in domain-oriented language, which is generally some limited variant of English, or by means of a graphics editor. In addition to these modes of communication, a knowledge engineer may use a structure editor for knowledge and inference acquisition. The aerodynamicist employs the system as an expert. The knowledge engineer communicates with the system to build its knowledge base and its inference system. The student learns how to use the system. The language

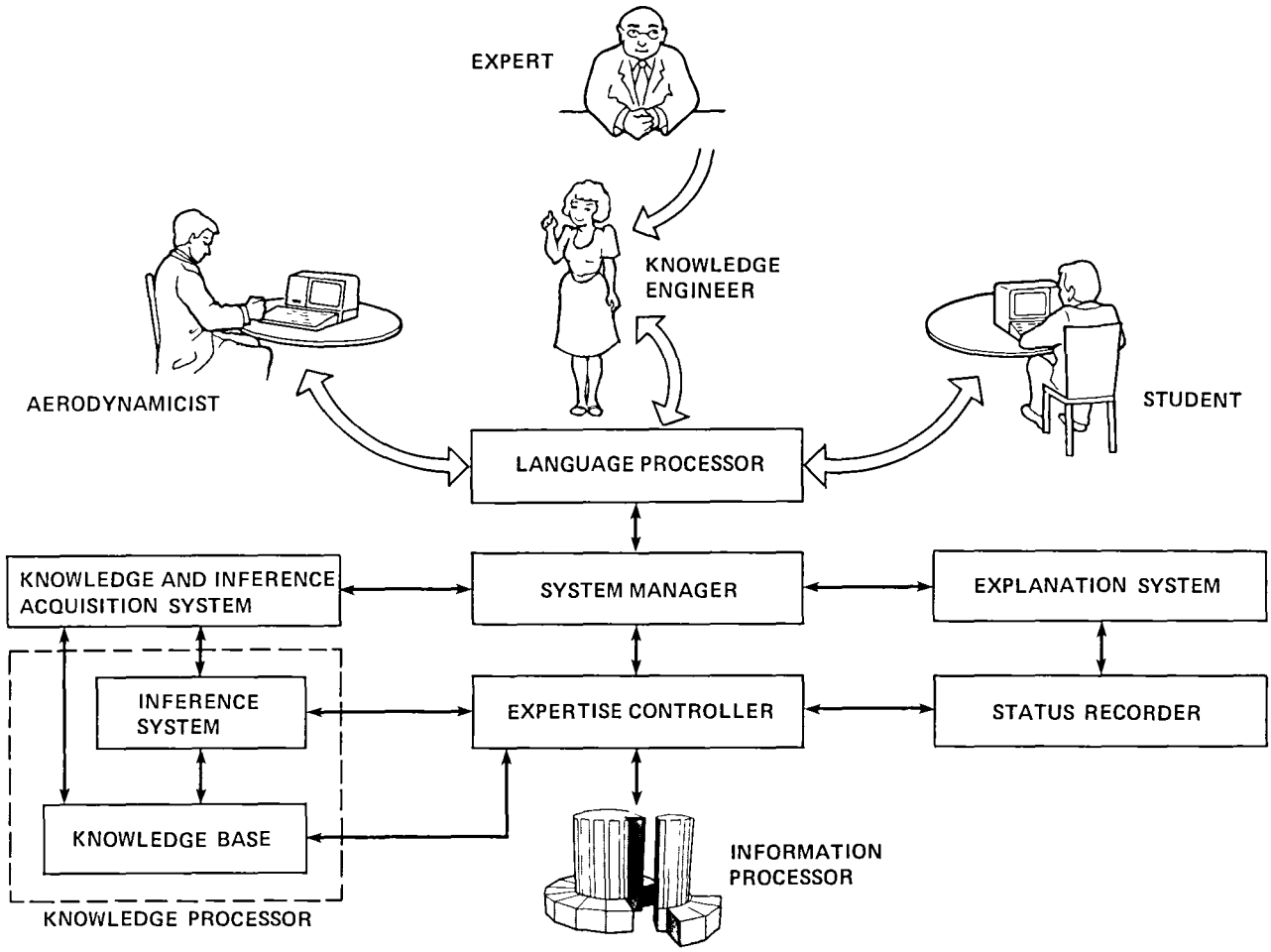


Figure 15. - Anatomy of AERODYNAMICIST, an idealized expert system for computational aerodynamics.

processor acts as an intermediary between any one of the above users and the expert system, and also transforms communication of the user into a language that can be understood by the system manager. This processor also transmits to the user the response of the expert system to the user directives.

The system manager directly or indirectly controls the other subsystems of AERODYNAMICIST. It sends directives to and receives responses from the knowledge and inference acquisition system, expertise controller, and the explanation system. It receives commands from and sends its response back to the language processor. It monitors the errors and interrupts of systems that are under its control.

The most important part of the anatomy of any expert system is the knowledge-base system, because the performance of the expert system depends on the knowledge it has. In the domain of utility, the expert system contains the following different types of knowledge that would make it behave knowledgeably: facts about objects, actions and events; the time-history of a sequence

of events; cause-and-effect relations; performance knowledge, that is, knowledge about how to do things; meta-knowledge, that is, knowledge about what is known; the theory of domain; and heuristic knowledge, that is, experiential and judgmental knowledge and rules of good guessing. Some domains may not contain all of these types of knowledge. A discussion of some of these is given in reference 14.

The next most important part of any expert system is the inference system. It contains inference procedures or problem-solving methods, which reason by manipulating and acting upon the combination of knowledge contained in the knowledge-base system and the problem submitted by the user. The inference system may contain different kinds of reasoning, some of which are discussed in reference 14. Formal reasoning based on mathematical logic deduces new data structures from given data structures by syntactic manipulation of given data. Procedural reasoning involves simulating answering questions and solving problems as, for example, the information (data) processing done in computational aerodynamics. Meta-level reasoning deals with meta-knowledge, that is, knowledge about what is known. Nonmonotonic reasoning withdraws a previously deduced conclusion in response to learning some new fact; see, for example, reference 56. Various combinations of these three kinds of reasoning are contained in existing expert systems. Reasoning by analogy, reasoning through generalization, and abstract reasoning are not yet feasible for implementation in expert systems. Common-sense reasoning is another kind of reasoning that is difficult at present to implement. Some of these are likely to be feasible in the future.

Knowledge and inference acquisition are equivalent to machine learning. When AERODYNAMICIST is being used in the learning mode or when it needs to capture its own experience, the knowledge and inference acquisition system is used. The knowledge engineer encodes the knowledge of the expert and inference rules utilizing this acquisition system as an aid. There are three other ways of acquiring knowledge (ref. 24): (1) The expert can interact directly with this acquisition system using an "intelligent editing program"; (2) induction programs may generate knowledge based on experience, obtained from the expert and from textbook cases; and (3) knowledge may be acquired directly from textbooks using text-understanding programs. The last two ways are not yet demonstrated. The knowledge and inference acquisition system decides how knowledge and inference shall be organized, controlled, propagated, and updated. It handles the difficult task of knowledge-base management. It determines how the inference system will use knowledge. An example of an acquisition system is TEIRESIAS (ref. 57). It assists the expert in acquiring, correcting, and using new inference rules without calling on a knowledge engineer.

The expertise controller actually controls the solution process for solving the problem submitted by the aerodynamicist or the student. It performs three functions: scheduling, interpreting, and enforcing consistency. It solves the problem by using the knowledge system, the inference system, and the information processor. It formulates a list of things to do, controls that agenda, and determines what action should be executed next. It interprets the inference procedures and applies them to the knowledge base and problem data. It ensures that plausible solutions or conclusions are arrived at, and that inconsistent ones are rejected.

The status recorder records all the activities of the expertise controller. Particularly, the status recorder has the following information: (1) status of three types of actions taken by the expertise controller, (2) record of the strategy embarked upon in order to solve the problem,

(3) agenda of potential actions awaiting execution, (4) record of the intermediate hypotheses, decisions and successful solutions, and their dependencies, and (5) list of failures during the problem solving process. In addition to this record keeping activity, it provides information to the explanation system.

The explanation system explains the actions of the expertise controller either to the aerodynamicist or the student user by tapping the information generated in the status recoder. The explanation system answers questions about how the solution of the problem was determined, why the expert system failed to provide the answer, why some decision, conclusion or hypothesis was reached while some other was rejected. The explanation system assists the user in evaluating AERODYNAMICIST, deciding whether to use the built-in expertise, and in debugging, modifying, and expanding the built-in expertise. This is particularly important for building confidence of designers of aerodynamic vehicles in computational aerodynamics.

The knowledge base and the inference system primarily compute by reasoning. On the other hand, the information processor primarily computes by calculating. This is its only function in AERODYNAMICIST. The information processor is essential, particularly when a flow field is to be determined.

EXPERT SYSTEMS IN COMPUTATIONAL FLUID DYNAMICS

The design and application of software for CFD basically involves four related phases: software method formulation, programming, validation, and utilization. Each phase requires decisions by one or some of the following: the theoretical expert, software designer, or applications engineer. Further, CFD involves the synthesis of many facets (fig. 16), each of which requires expertise. It is conceivable that an expert system could be designed that would act as a flow-field synthesizer; that is, act on all of the 10 facets depicted in figure 16 for a computation. Because of the knowledge-accumulation concept used in such a system, wrong paths and mistakes often made by researchers can be avoided.

Present expert-systems techniques show promise of being used in at least five aspects of the CFD simulation process that would involve some of the above facets: definition of the flow-problem, grid generation (one of the computational complexities associated with advances in CFD), construction and analysis of numerical schemes, flow-solver selection and use, and data reduction, analysis and display (fig. 17).

Before a computational fluid dynamicist begins constructing a computer program to solve a flow problem, he has to completely define the problem based on the physical aspects of it; see, for example, reference 58 for viscous flows. This process could be performed by an expert system with extensive fluid dynamic knowledge encompassing flow regimes, local flow characteristics, acceptable assumptions and their shortcomings, governing equations, physical boundary and initial conditions, and solution methods (see figure 17). The expert system in essence could develop a blueprint for tackling the flow problem. Such a system could be used for education; it could also be used by a person not familiar with fluid dynamics.

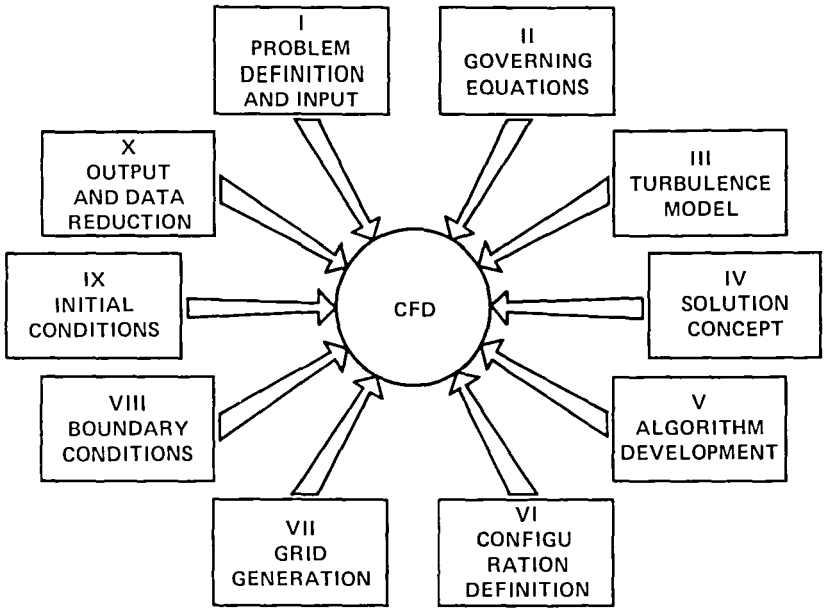
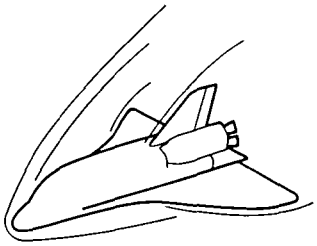


Figure 16. - Various facets of computational fluid dynamics.

One of the most important steps required to accurately solve a three-dimensional CFD problem using finite-difference procedures involves the proper location of the nodal points in the flow region to be resolved. There are basically two decision stages and a feedback stage involved in the discretization process. The two decision stages involve (1) the grid topology, and (2) the grid-generation scheme; the feedback stage involves an analysis and modification of the grid based on the grid metrics, the flow-solver algorithm to be used, and the flow-solution generated. Grid generation is an intrinsically complex and difficult reasoning process. It is a combinatorial problem because the degrees of freedom in distributing grid-points are directly proportional to the number of grid-points. Therefore, AI techniques can greatly reduce the complexity associated with the creation of grid networks.

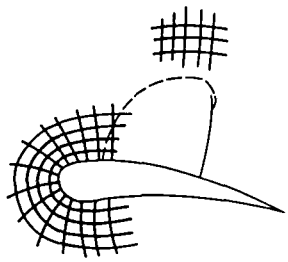
The *conceptualized* expert grid-generation system SPIDER shown in figure 18 is based on the three major elements of an expert system mentioned earlier (see figure 15). It depicts a few essential ingredients of some of these elements. The input phase consists of three groups of input information required by SPIDER. The first part of this input information consists of various flow parameters such as the Mach number, angles of attack and yaw, and Reynolds number. These parameters would determine, for example, whether planes of symmetry can be employed, the position of the outer computational domain, and the possible regions of nodal point clustering. The second group of input information is the geometric data; that is, body coordinates that form the boundary of the computational domain. The third group of input data consists of such qualitative program control information as the desired level of accuracy required and the permissible level of expense to be incurred.

Contained in the knowledge base of SPIDER would be the facts and heuristics. In this case the facts would consist of the grid-generation schemes and the grid-analysis procedures.



FLOW-PROBLEM DEFINITION

- PHYSICAL ASPECTS OF FLOW
  - SPEED REGIME
  - VISCOUS OR INVISCID
  - STEADY OR UNSTEADY
  - THIN, SLENDER OR COMPLEX-SHEAR LAYERS
  - SHOCK WAVES – FLOW DISCONTINUITIES
- SURFACE AND FIELD VARIABLES
- PHYSICAL BOUNDARY AND INITIAL CONDITIONS
- SUGGESTIONS AS TO SOLUTION METHODOLOGY



GRID GENERATION

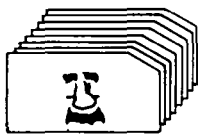
- CONCEPTUAL OR TOPOLOGICAL DISCRETIZATION
- DEFINITION AND APPLICATION OF GRID-GENERATION PROCEDURE
- GRID-QUALITY ANALYSIS
- GRID-POINT LOCATION ADJUSTMENTS

$$\frac{\partial U}{\partial t} = \frac{\partial^2 U}{\partial x^2}$$

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \frac{U_{i+1}^{n+1} - 2U_i^{n+1} + U_{i-1}^{n+1}}{\Delta x^2}$$

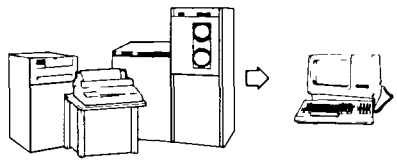
CONSTRUCTION AND ANALYSIS OF NUMERICAL METHODS

- DEVELOPMENT OF SCHEMES WITH SPECIFIC PROPERTIES
- STABILITY ANALYSIS
- ACCURACY ANALYSIS



FLOW SOLVERS

- NAVIER-STOKES SOLVER
- PNS SOLVER
- BOUNDARY LAYER SOLVER
- INVISCID SOLVER



DATA REDUCTION (ANALYSIS AND DISPLAY)

- ENGINEERING QUANTITIES
- FLOW VISUALIZATION
- ERROR ANALYSIS

Figure 17. - Expert systems in computational fluid dynamics.

The heuristics knowledge would consist of good practice and good judgment of constructing acceptable grid systems. For example, there currently exists no theory that can determine the zoning or grid patchwork for either two- or three-dimensional problems. Therefore, heuristics would be employed.

Once the flow field has been zoned, each zone can then be discretized using the procedures denoted in figure 18 which include both algebraic and differential approaches. With the flow region discretized, various levels of grid-analysis procedures can be used to judge the quality of the created grid. These vary in complexity from procedures that simply look at the grid parameters (such as the transformation Jacobian, geometric derivatives, and ratio of the metrics) to procedures that combine these functions with the flow-solver algorithm and flow solution to yield a modified grid. Thus, this system must interact with the flow solver to receive information on the flow solution and transmit information on the new nodal point locations.

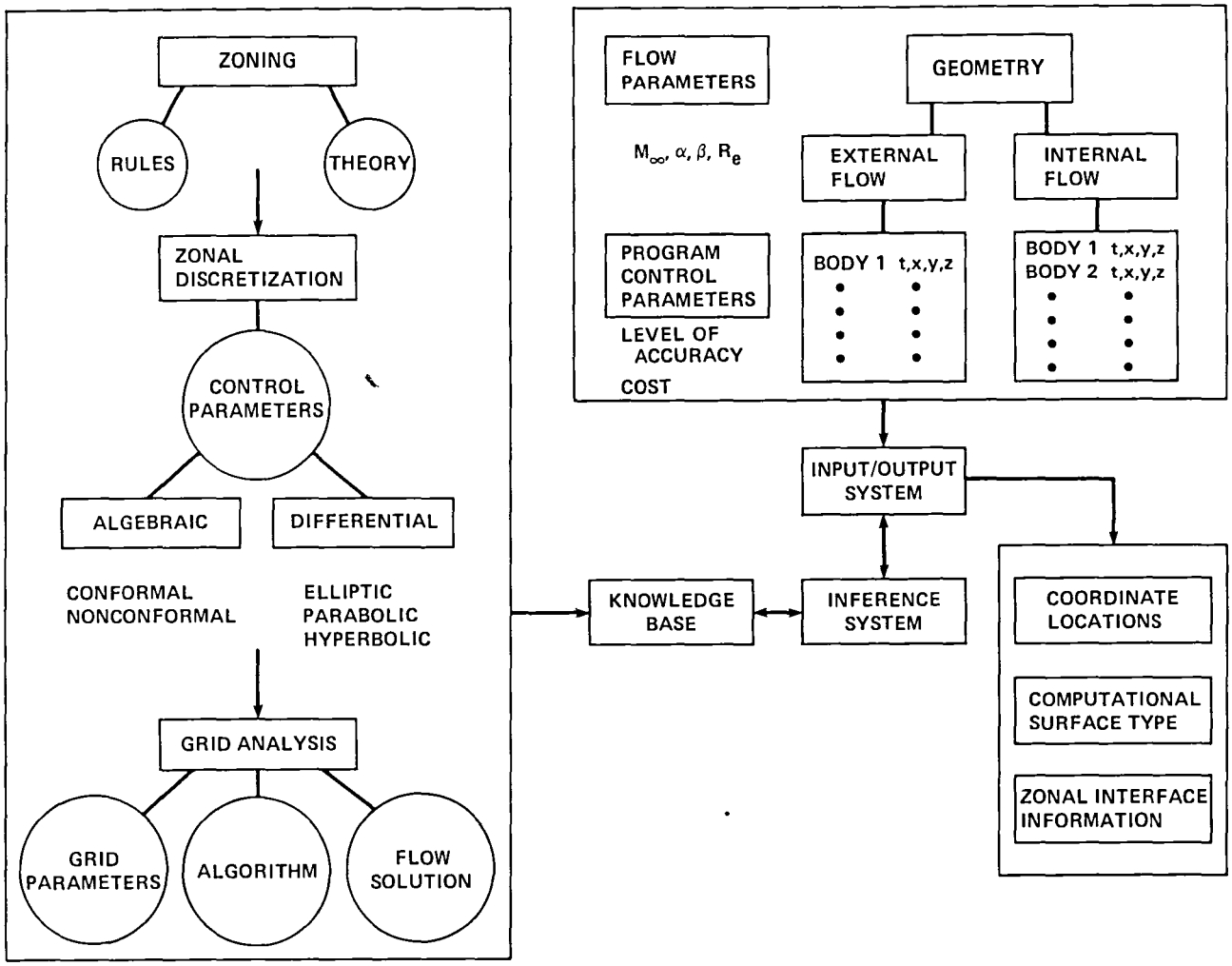


Figure 18. - SPIDER, an expert system for grid generation.

The expert grid-generation system simply produces as output three pieces of information to be used by the flow solver: (1) the coordinate location of the nodal points, (2) the definition of each surface of the computational cube (e.g., plane of symmetry, body, and shock wave), and (3) the zonal interface control parameters. The latter piece of information tells the flow solver which zonal boundaries are adjacent to each other. This is required by the boundary-condition routines in the flow solver.

An expert system can be constructed for building and analyzing numerical methods. Such a system could develop a numerical method that has specified properties and capabilities. It could perform a stability analysis, and determine phase and amplitude errors of any numerical method. It could be made to compute test problems and present sample results. It could summarize all the characteristics of the numerical method. It could check for and construct compatible numerical schemes that might be used in different parts of the flow field. Sometimes it is difficult to perform a stability analysis; for example, it is generally difficult to perform a stability analysis with

nonperiodic boundary conditions. For such situations, the expert system could be advantageous.

Just as MACSYMA solves ordinary differential equations, an expert system could be developed to solve the partial differential equations governing fluid flows. This system would contain knowledge of existing flow solvers. It would automatically select the solution method and generate the corresponding code for a specific problem. In addition, the system would assist in determining a correct solution by checking whether the problem is well-posed. This system would be widely used by aerodynamicists in aerodynamic research, design, and development. It would not be necessary that the user have any knowledge of the solution procedures used by this system.

Three-dimensional computational fluid dynamic solutions generate vast amounts of data. The processing of these data for analysis and display is somewhat analogous to signal interpretation in military and space applications. It is highly conceivable that an expert system could be constructed to make decisions, and to process and display useful data in an appropriate form based on known desired displays. This system would decide what part of the flow field to analyze, what analysis to perform, what to display, and how to display it. It could also summarize the significant physical, computational, and design aspects of the computed results.

The above five examples are a just a few of the potential applications of expert systems in CFD. These are felt to be the ones most promising and immediately possible.

EXPERT SYSTEMS FOR AERODYNAMIC DESIGN AND DEVELOPMENT

Flight vehicle designers need sophisticated computational aerodynamic tools to simulate the flow field about complicated three-dimensional flight configurations. They would prefer to have expert simulation tools at their disposal in a relatively short time. They would also like to use these tools without the need to know exactly how they work. Furthermore, they would like these tools to be easy to use, versatile, reliable (thus improving productivity), robust, and accurate (so that the design and development of more efficient configurations would be feasible). The development of sophisticated computational aerodynamic tools requires vast expertise and enormous resources in terms of both manpower and computer capacity (i.e., speed and storage). The creation of these simulation tools requires knowledge; for example, in the disciplines of aerodynamics, structural dynamics, propulsion, design methods, fluid dynamics, numerical analysis, and computer science. Such tools demand an inordinate amount of development time. To provide simulation programs that are easy to use, reliable, and versatile, and to alleviate a long development process, expert systems can be employed (fig. 19).

There are at least nine areas in which expert systems can be developed. These areas are potential flow, Euler equations, thin-shear layer (boundary-layer) equations, Reynolds-averaged Navier-Stokes equations, parabolized Navier-Stokes equations, computational design, aerodynamic knowledge-base, structural knowledge-base, and aerodynamic design knowledge-base.

Abundant knowledge is available for determining the potential flow around aerodynamic vehicles. The numerical procedures and techniques are well understood. For transonic flows, the aircraft industry is routinely using potential-flow codes, such as the series of FLO codes

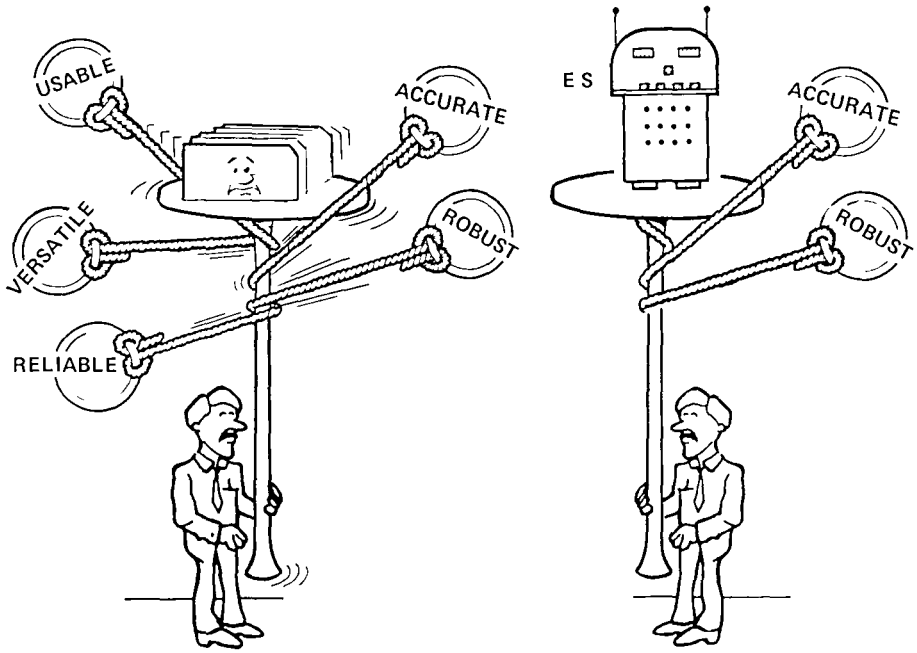


Figure 19. - Expert systems provide the flight vehicle designers with usable, versatile, and reliable design tools.

(ref. 59). The strengths and weaknesses of small-disturbance and full-potential methods are well known. The characteristics and capabilities of various numerical methods are also well known. For incompressible flows, the panel method and the Douglas-Neumann code exist. In short, the numerical determination of potential flow is no longer a basic research topic. It is just a matter of properly utilizing the available knowledge. All this knowledge can be codified into an expert system that would automatically generate the grid system using SPIDER and solve for the potential flow around any vehicle. The expert system could make all the relevant decisions based on the requirements specified by the user, and then it would direct a supercomputer to perform the calculations. Upon receiving the computed data, it would analyze the data and report the significant results. A similar case, based on availability of enormous knowledge, can be made for developing an expert system for automatically and expertly computing thin-shear layers.

During the last 4 years, there has been significant progress in computing solutions of the Euler equations. Although there is still a lot to learn about solution procedures for these equations, it makes more sense to begin building an expert system now than developing a portfolio of codes for each application that arises. The expert Euler equation system could be continuously updated, and thus its capability would incrementally grow. Conventional programming techniques generally produce codes that need to be redesigned when their requirements are modified (fig. 11). An expert system that is incrementally developed, on the other hand, would result in a code that would remain operational with changing requirements within the same domain. A similar case can be made for developing an expert system that would solve Reynolds-averaged Navier-Stokes equations and the parabolized Navier-Stokes equations.

Aerodynamic design methods are classified as either inverse solution methods or numerical

optimization methods. Inverse methods determine the aerodynamic body surface that produces the desired pressure distribution. These methods, at present, are best suited for initial design because it is difficult even for an experienced designer to specify the pressure distribution that would achieve the design objective and because structural constraints are difficult to impose (ref. 60). But an expert system can search through a knowledge base containing various pressure distributions associated with the shapes of the bodies and their design characteristics, and it can construct the likely pressure distribution that would satisfy the design conditions. This pressure distribution can then be used for the inverse problem.

On the other hand, numerical optimization methods couple flow solvers with numerical minimization schemes to produce designs that achieve user-specified design objectives. In these methods, the designer specifies an initial aerodynamic body, design objectives, design constraints, and design conditions. The effect on the design objectives and constraints is determined by computing the flow field. A search direction is established that minimizes one of the design objectives while satisfying the constraints. In this fashion the design variables are perturbed independently, and the design objective and constraints are again evaluated. When the objective can no longer be reduced in the current direction, a new search direction is chosen, and the process is repeated. The number of design points that can be considered by numerical optimization is theoretically unlimited; however, computer capacity limits the number of these points (ref. 61). In such instances, an expert system with heuristic search procedures utilizing a knowledge base of previous designs and experience related to the optimum way to perturb the variables could benefit the optimization process (fig. 20).

Knowledge of fluid dynamics, structural mechanics, and mechanical design pertinent to the design and development process of an aerodynamic configuration in knowledge-based systems could be a great asset to designers and developers. A lot of heuristic and experiential knowledge is used in design and development of aircraft operating in the transonic regime (refs. 62-64). Imagine what an asset it would be if the aerodynamic design expertise of Kelley Johnson, designer of planes such as U-2 and SR-71 for Lockheed Aircraft Company, were to be encoded.

RESOURCE REQUIREMENTS

Constructing an expert system requires a substantial investment of time and manpower. Davis (ref. 22) conducted a survey to determine the time required to build an expert system. The results of his survey are shown in figure 21. The systems mentioned in this figure vary drastically with respect to the size of the problem they are trying to solve and the amount of knowledge they need to perform expertly. For example, the size of the problem solved by MACSYMA is probably two orders of magnitude larger than that of PUFF. Furthermore, the level of performance that was achieved by the various systems was not the same. Some of these systems routinely solved practical problems, while others never made it beyond the research stage. Nevertheless, there is a decline in the time required to build a system. The curve suggests that it would take about 5 man-years to build an expert system that can be routinely used. Davis has qualified this estimate by requiring that the problems must be clearly defined, and that experienced researchers and

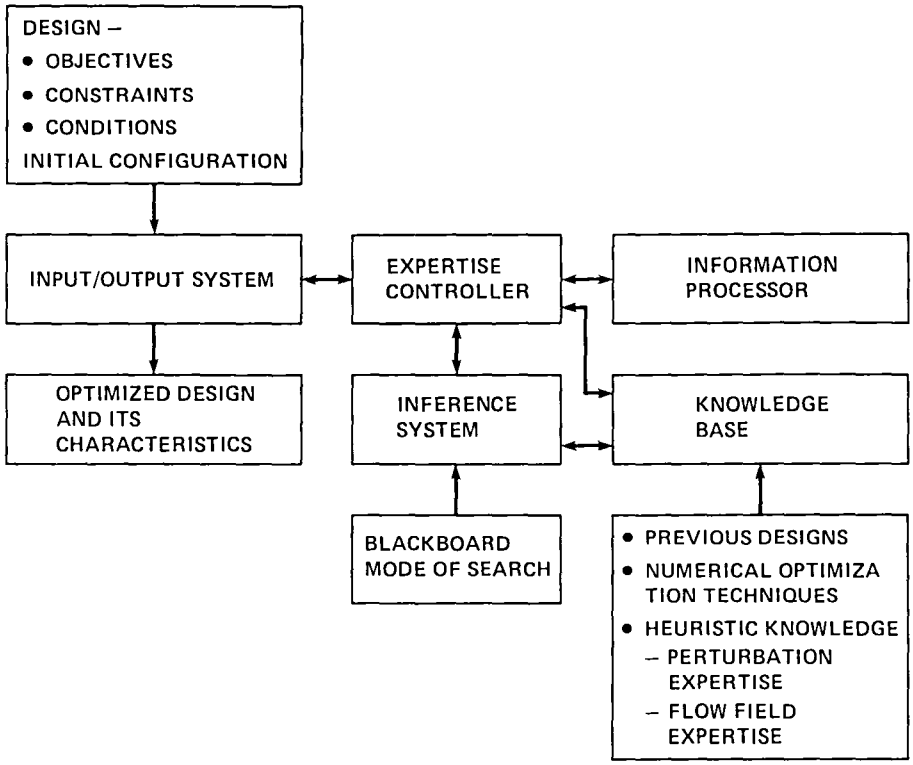


Figure 20. - OPTIMIZER, an expert system for aerodynamic design optimization.

mature technologies be used for building the expert system.

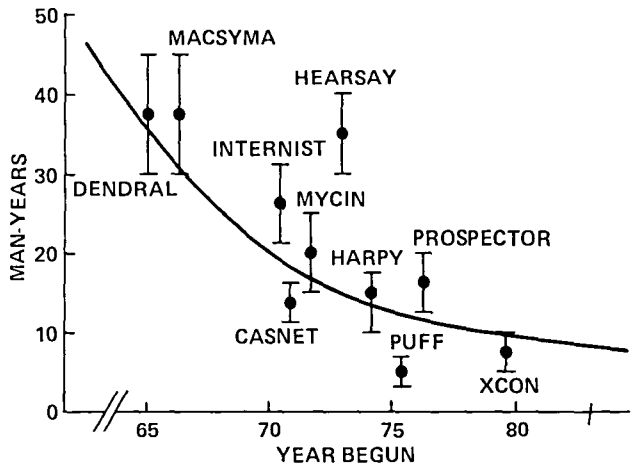


Figure 21. - Development times in man-years of some of the existing expert systems (from reference 22); the vertical bars indicate the error in estimating man-years.

Hayes-Roth has given an estimate for building a knowledge-based system that ranges from 7

months for simple systems in a friendly environment with existing tools to 15 years for complex systems in demanding environments where fundamental research and development are required (ref. 65). This time estimate covers the period from conception of the knowledge system solution to application. This includes building a prototype system for feasibility demonstration that a knowledge system can be of use in the given context, designing and developing the actual system, and integrating the hardware, software, user interface, and knowledge base. Simple systems generally contain 50 pieces of knowledge - a piece of knowledge is the smallest piece of information; for example, a single IF-THEN rule. A current complex system generally requires 10^5 pieces of knowledge. It is projected that at a noneducational institute the cost will range from \$175,000 for a simple system with 14 man-months of work to \$15,000,000 for a complex system with 50 man-years of work. These estimates include the costs for design, development, knowledge engineers, computing, overhead and expert-system building tools.

The discipline of knowledge engineering is the most crucial in development of expert systems. It is the art of designing and building knowledge-based systems. It involves knowledge representation, knowledge acquisition, knowledge refinement, knowledge system architecture, and knowledge system performance. Knowledge engineering is the pacing item for advances in expert system technology because, at present, knowledge engineering has four significant shortcomings (fig. 22). First, there is a scarcity of knowledge engineers. Currently about six computer science doctoral-level graduates per year are involved in construction of knowledge-based systems (ref. 36). There are about 70 persons in the world who have a high level of experience in knowledge-based system design and development (ref. 36). Second, there is a lack of knowledge-engineering tools. Existing tools for knowledge-representation and problem-solving structures are not always amenable to new applications. Often laboratory tools must be modified or new tools must be constructed. Third, knowledge engineering is still in a research stage. Although there are ways machines can learn, knowledge is often handcrafted because human-like knowledge-acquisition modes are not yet thoroughly understood. Fourth, there is a knowledge-representation mismatch between the way a human expert normally states knowledge and the manner in which it must be represented in the expert system. Therefore, the expertise of a knowledge engineer in programming the expertise of a human expert is extremely crucial in building expert systems (ref. 65).

To date, expert systems have been created and exercised on third-generation computers only. Programming languages used in AI must be available on existing fourth-generation computers for using AI in computational aerodynamics. Since computational aerodynamics generally requires supercomputers, it is highly desirable that the reasoning aspect of computational aerodynamics be accomplished on the same or similar computer. In addition, tools for building expert systems should be available on this computer.

CONCLUDING REMARKS

There are two ways the science of artificial intelligence is likely to benefit computational aerodynamics. Directly, it will greatly influence the reasoning aspect of computational aerodynamics

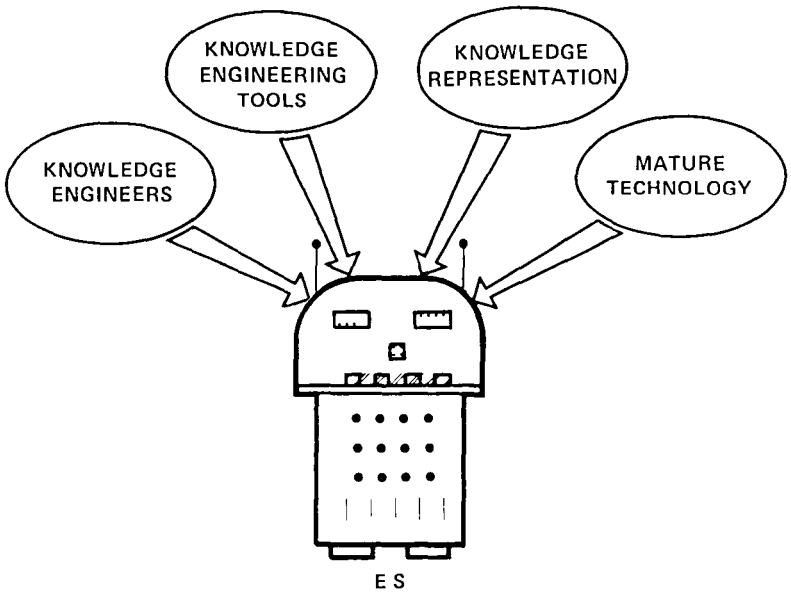


Figure 22. - Pacing items for knowledge engineering.

based on available knowledge and expertise. It offers the opportunity of using computers as reasoning machines to set the stage for efficient calculating. This will be accomplished through the use of knowledge-based systems. Artificial intelligence will make knowledge easily available and usable. It will make computational complexity manageable. It will provide designers of aerodynamic vehicles with reliable, easy to use, and versatile expert computational aerodynamic tools, as well as related design tools. Indirectly, a major worldwide interest in knowledge-based systems is likely to lead to the development of the next generation of computers and the higher-level programming languages based on functional programming style that will replace von Neumann computers and von Neumann programming languages such as FORTRAN. This will result in faster computers and in greater simplicity in programming, both of which will be advantageous to computational aerodynamics.

Expert systems are likely to be a new asset of institutions involved in aeronautics, particularly for various aspects and tasks of computational aerodynamics. Expert systems will introduce a new way of preserving knowledge, and, consequently, they will change the way knowledge is passed from one generation to the next. They will liberate people from concentrating on narrow domains of knowledge and allow them to entertain broad-range problems. On the other hand, the widespread use of only one or a few expert systems may lead to uniformity and homogeneity. A tendency toward overdependence on expert systems may stifle individual creativity and innovation.

There are three major stages of the computational aerodynamic development cycle (see also reference 2): (1) research, (2) technology transfer, and (3) design and development. The first stage, research, is the stage in which a computational procedure is conceptualized, algorithms are developed, and pioneering applications are performed. After feasibility of the computational tool has been demonstrated, technology transfer or dissemination from the research mode to the design

and development mode occurs. In this stage, the engineering user requirements are considered and different technology components are assembled for engineering applications. When this is completed, the design and development stage begins, in which the developed technology from the research stage is utilized in the design process, as well as in creative and innovative developments. These three stages are likely to be greatly accelerated by the use of artificial intelligence concepts (fig. 23).

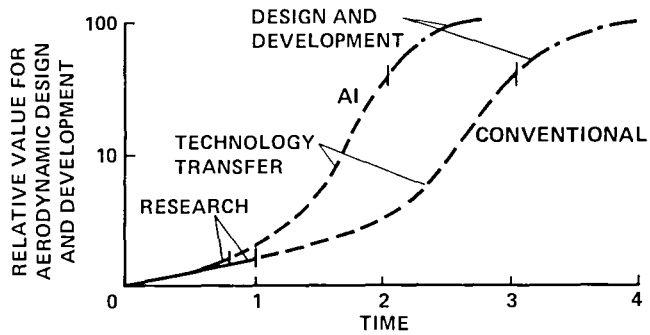


Figure 23. - Artificial intelligence accelerates research, technology transfer, and design and development stages of computational aerodynamics development.

Predicting the future is a difficult and hazardous business. Nonetheless, we have ventured into the unknown world of computational aerodynamics and artificial intelligence (CAAI) by speculating on the role that artificial intelligence may play in computational aerodynamics; the ultimate goal being to bring intelligent machines to bear on the present and future complex computational tasks of the fluid dynamicist and aerodynamic designer. Further, we have assumed that challenges facing the science of artificial intelligence (ref. 32), particularly in the field of knowledge engineering (refs. 65 and 66), will be successfully resolved by those engaged in artificial intelligence research.

REFERENCES

1. Rogers, E. W. E.: Aerodynamics – Retrospect and Prospect, *Aeronatical J.*, Feb. 1982, pp. 1-22.
2. The Influence of Computational Fluid Dynamics on Experimental Aerospace Facilities, A Fifteen Year Projection, Prepared by the Committee on Computational Aerodynamics Simulation Technology Developments, Aeronautical and Space Engineering Board, Commission on Engineering and Technical Systems, National Research Council, National Academy Press, Washington, D. C., 1983.
3. Hall, M. G.: Computational Fluid Dynamics – A Revolutionary Force in Aerodynamics, AIAA Paper 81-1014, Palo Alto, Calif., 1981.

4. Peterson, V. L.: Impact of Computers on Aerodynamics Research and Development, *Proceedings of the IEEE*, vol. 72, no. 1, 1984, pp. 68-79.

5. Kutler, P.: A Perspective of Theoretical and Applied Computational Fluid Dynamics, AIAA Paper 83-0037, Reno, Nev., 1983.

6. Mehta, U. B.; and Lomax, H.: Reynolds Averaged Navier-Stokes Computations of Transonic Flows – The State-of-the-Art, *Transonic Aerodynamics*, D. Nixon, ed., AIAA Progress in Astronautics an Aeronautics, vol. 81, 1982, pp. 297-375.

7. Chapman, D. R.: Trends and Pacing Items in Computational Aerodynamics, *Lecture Notes in Physics*, vol. 141, W. C. Reynolds and R. W. MacCormack, eds., 1981, pp. 1-11.

8. Feigenbaum, E. A.; and McCorduck, P.: *The Fifth Generation*, Addison-Wesley Publishing Company, 1983.

9. Fischetti, M. A.: MCC – An Industry Response to the Japanese Challenge, *IEEE Spectrum*, Nov. 1983, pp. 55-56.

10. Cooper, R. S.; and Kahn, R. E.: SCS – Toward Supersmart Computer for the Military, *IEEE Spectrum*, Nov. 1983, pp. 53-55.

11. Oakley, B. W.: *IEEE Spectrum*, Nov. 1983, pp. 69-71.

12. Nasko, H: *IEEE Spectrum*, Nov. 1983, pp. 71-72.

13. Pavelle, R.; Rothstein, M.; and Fitch, J.: Computer Algebra, *Scientific American*, Vol. 245, No. 6, Dec. 1981, pp. 136-152.

14. Barr, A.; and Feigenbaum, E. A.: *The Handbook of Artificial Intelligence*, vol. 1, William Kaufmann, Inc., 1981.

15. McCorduck, P.: *Machines Who Think*, W. H. Freeman & Co., New York, 1979.

16. *Webster's New Collegiate Dictionary*, (s. v. "intelligence"), G. & C. Merriam Co., Springfield, Mass., 1976.

17. Minsky, M.: Why People Think Computers Can't, *The AI Magazine*, Fall 1982, pp. 3-15.

18. MACSYMA Reference Manual, version 10, Mathlab Group, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Mass., 1983.

19. Lindsay, R. K.; Buchanan, E. A.; Feigenbaum, E. A.; and Lederberg, J.: *Applications of Artificial Intelligence for Organic Chemistry - The DENDRAL Project*, McGraw-Hill, New York, 1980.

20. Kunz, J. C.; Fallat, R. J.; McClung, D. H.; Osborn, J. J.; Votteri, R. A.; Nii, H. P.; Aikins, J. S.; Fagan, L. M.; and Feigenbaum, E. A.: A Physiological Rule-Based System for Interpreting Pulmonary Function Test Results, Report HPP-78–19, Heuristic Programming Project, Computer Science Department, Standford University, Stanford, Calif., 1978.

21. Lenat, D. B.: The Nature of Heuristics, *Artificial Intelligence*, vol. 19, 1982, pp. 189-249.

22. Davis, R.: Expert Systems – Where Are We? And Where Do We Go from Here? *The AI Magazine*, Spring 1982, pp. 3-22.

23. Feigenbaum, E. A.: The Art of Artificial Intelligence – Themes and Case Studies of Knowledge Engineering, *Proceedings of the International Joint Conference on Artificial Intelligence* , vol. 5, 1977, pp. 1014-1029.

24. *Building Expert Systems*, F. Hayes-Roth; D. A. Waterman; and D. B. Lenat; eds., Teknowledge Series in Knowledge Engineering, vol. 1, Addison-Wesley Publishing Co., Inc., 1983.

25. Gevarter, W. B.: Expert Systems – Limited but Powerful, *IEEE Spectrum*, Aug. 1983, pp. 39-45.

26. Gevarter, W. B.: An Overview of Artificial Intelligence and Robotics, Vol. I - Artificial Intelligence, Part A - The Core Ingredients, NASA TM-85836, 1983.

27. Gevarter, W. B.: An Overview of Expert Systems, NBSIR 82-2505, National Bureau of Standards, 1982.

28. Duda, R. O.; and Shortliffe, E. H.: Expert Systems Research, *Science*, vol. 220, no. 4594, 1983, pp. 261-268.

29. Barr, A.; and Feigenbaum, E. A.: *The Handbook of Artificial Intelligence*, vol. 2, William Kaufmann, Inc., 1982.

30. Miller, R. A.; Pople, H. E.; and Myers, J. D.: INTERNIST-I, an Experimental Computer-based diagnostic Consultant for General Internal Medicine, *New Engand Journal of Medicine*, vol. 19, 1982, pp. 468-476.

31. Newell, A.; and Simon, H. A.: Computer Science as Empirical Enquiry – Symbols and Search, The 1975 ACM Turing Lecture, *Communications of the Association for Computing Machinery*, vol. 19, no. 3, 1976, pp. 113-126.

32. Nilsson, N. J.: Artificial Intelligence Prepares for 2001, *The AI Magazine*, Winter 1983, pp. 7-14.

33. Feigenbaum, E. A.: Artificial Intelligence, *IEEE Spectrum*, Nov. 1983, pp. 77-78.

34. Gevarter, W. B.: An Overview of Artificial Intelligence and Robotics, Vol. I - Artificial Intelligence, Part C - Basic AI Topics, NASA TM-85839, 1983.

35. McCarthy, J.: Recursive Functions of Symbolic Expressions and their Computation by Machine, *Communications of the Association for Computing Machinery*, vol. 3, 1960, pp. 184-195.

36. Partridge, D. R.: Expert Systems, seminar notes, sponsored by Tecnology Transfer Society, Palo, Alto, Calif., Jan. 1984.

37. Martin, E. W.: Strategy for a DoD Software Initiative, *Computer*, Mar. 1983. pp. 52-59.

38. Fleckenstein, W. O.: Challenges in Software Development, *Computer*, Mar. 1983. pp. 60-64.

39. Mizuno, Y.: Software Quality Improvement, *Computer*, Mar. 1983, pp. 66-72.

40. Mueller, G. E.: The Future of Data Processing in Aerospace, *Aeronautical Journal*, Apr. 1979, pp. 149-158.

41. Wallich, P.: Designing the Next Generation, *IEEE Spectrum*, Nov. 1983, pp. 73-77.

42. Lerner, E. J.: Data-Flow Architecture, *IEEE Spectrum*, Apr. 1984, pp. 57-62.

43. Davis, A. L: Computer Architecture, *IEEE Spectrum*, Nov. 1983, pp. 94-99.

44. Backus, J.: Functional-Level Computing, *IEEE Spectrum*, Aug. 1982, pp. 22-27.

45. Backus, J.: Can Programming be Liberated from the von Neumann Style? A Functional Style and its Algebra of Programs, 1977 ACM Turing Award Lecture, *Communications of the Association for Computing Machinery*, vol. 21, no. 8, 1978, pp. 613-641.

46. Barstow, D.: A perspective on Automatic Programming, *The AI Magazine*, Spring 1984, pp. 5-27.

47. Balzer, R.; Cheatham, Jr., T. E.; and Green, C.: Software Technology in the 1990's – Using a New Paradigm, *Computer*, Nov. 1983, pp. 39-45.

48. Lerner, E. J.: Automating Programming, *IEEE Spectrum*, Aug. 1982, pp. 28-33.

49. Green, C.; and Westfold, S.: Knowledge-Based Programming Self Applied, *Machine Intelligence 10*, Ellis Horwood Limited, Halsted Press (John Wiley & Sons), 1982.

50. Green, C.; Phillips, J.; Westfold, S.; Pressburger, T.; Kedzierski, B.; Angebranndt, S.; Mont-Reynaud, B.: and Tappel, S.: Research on Knowledge-Based Programming and Algorithm Design – 1981, KES.U.81.2, Kestrel Institute, Palo Alto, Calif., Sept. 1982.

51. Boraiko, A. A.: The Chip – Electonic Mini-Marvel That Is Changing Your Life, *National Geographic*, Oct. 1982, pp. 427-450.

52. Bradshaw, G. F.; Langely, P. W.; and Simon, H. A.: Studying Scientific Discovery by Computer Simulation, *Science*, vol. 222, no. 4627, DEc. 1983, pp. 971-975.

53. Artificial Intelligence – The Second Computer Age Begins, *Business Week*, Mar. 8, 1982, pp. 66-75.

54. Lenat, D. B.: Automated Theory Formulation in Mathematics, *International Joint Conference Artificial Intelligence*, vol. 5, 1977, pp. 833-842.

55. Sharma, A.; and Minkowycz, W. J.: KNOWTRAN – An Artificial Intelligence System for Solving Heat Transfer Problems, *Journal of Heat and Mass Transfer*, vol. 25, no. 9, 1982, pp. 1279-1289.

56. McCarthy, J.: Circumscription – A Form of Non-Monotonic Reasoning, *Artificial Intelligence*, vol. 13, 1980, pp. 27-40.

57. Davis, R.: Interactive Transfer of Expertise – Acquisition of New Inference Rules, *International Joint Conference Artificial Intelligence*, vol. 5, 1977, pp. 321-328.

58. Mehta, U. B.: Physical Aspects of Computing the Flow of a Viscous Fluid, NASA TM-85893, 1984.

59. *Transonic Aerodynamics*, D. Nixon, ed., AIAA Progress in Astronautics an Aeronautics, vol. 81, 1982.

60. Lores, M. E.; and Hinson, B. L.: Transonic Computational Design, *Transonic Aerodynamics*, D. Nixon, ed., AIAA Progress in Astronautics an Aeronautics, Vol. 81, 1982. pp. 377-402.

61. Hicks, R. M.; and Vanderplaats, G. N.: Application of Numerical Optimization to the Design of Supercritical Airfoils without Drag-Creep, SAE Paper 770440, 1977.

62. Lynch, F. T.: Commercial Transports – Aerodynamic Design for Cruise Performance Efficiency, *Transonic Aerodynamics*, D. Nixon, ed., AIAA Progress in Astronautics an Aeronautics, vol. 81, 1982, pp. 81-147.

63. Bradley R. G.: Practical Aerodynamic Problems – Military Aircraft, *Transonic Aerodynamics*, D. Nixon, ed., AIAA Progress in Astronautics an Aeronautics, vol. 81, 1982, pp. 149-187.

64. *Subsonic/Transonic Configuration Aerodynamics*, AGARD-CP-285, 1980.

65. Hayes-Roth, F.: Codifying Human Knowledge for Machine Reading, *IEEE Spectrum*, Nov. 1983, pp. 79-81.

66. Feigenbaum, E. A.: Knowledge Engineering for the 1980's, Computer Science Dept., Stanford University, Stanford, Calif. 1982.

1 Report No	2 Government Accession No	3 Recipient's Catalog No
NASA TM-85994		

4 Title and Subtitle	5 Report Date
COMPUTATIONAL AERODYNAMICS AND ARTIFICIAL INTELLIGENCE	June 1984
	6 Performing Organization Code

7 Author(s)	8 Performing Organization Report No
Unmeel B. Mehta and Paul Kutler	A-9807
	10 Work Unit No
	T-6465

9 Performing Organization Name and Address	11 Contract or Grant No
NASA Ames Research Center Moffett Field, CA 94035	
	13 Type of Report and Period Covered
	Technical Memorandum

12 Sponsoring Agency Name and Address	14 Sponsoring Agency Code
National Aeronautics and Space Administration Washington, DC 20546	505-31-01

15 Supplementary Notes

Point of Contact: Unmeel B. Mehta, Ames Research Center, MS 202A-1, Moffett Field, CA 94035. (415) 965-5548 or FTS 448-5548.

16 Abstract

A major role of computational aerodynamics in the future is that of acquiring new knowledge in a number of complex aeronautical problems. It will also play an increasingly effective role in the aerodynamic design and development process. The science of artificial intelligence can advantageously be applied to both these roles of computational aerodynamics. This report considers some aspects of artificial intelligence and speculates on how knowledge-based systems can accelerate the process of acquiring new knowledge in aerodynamics, how computational fluid dynamics may use expert systems, and how expert systems may speed the design and development process. In addition, the anatomy of an idealized expert system called AERODYNAMICIST is discussed. The report concludes with a discussion of resource requirements for using artificial intelligence in computational fluid dynamics and aerodynamics. Three main conclusions of this report are presented. First, there are two related aspects of computational aerodynamics: reasoning and calculating. Second, a substantial portion of reasoning can be achieved with artificial intelligence. It offers the opportunity of using computers as reasoning machines to set the stage for efficient calculating. Third, expert systems are likely to be new assets of institutions involved in aeronautics for various tasks of computational aerodynamics.

17 Key Words (Suggested by Author(s))	18 Distribution Statement
Computational aerodynamics Computational fluid dynamics Artificial intelligence Expert systems; Aerodynamic design Knowledge-based systems	Unlimited Subject Category - 02

19 Security Classif (of this report)	20 Security Classif (of this page)	21 No of Pages	22 Price*
Unclassified	Unclassified	39	A03

*For sale by the National Technical Information Service, Springfield, Virginia 22161

End of Document