Computers chem. Engng, Vol. 14, No. 11, pp. 1259–1270, 1990
Printed in Great Britain

0098-1354/90 $3.00 + 0.00
Pergamon Press plc

ARTIFICIAL INTELLIGENCE IN PROCESS ENGINEERING—CURRENT STATE AND FUTURE TRENDS

G. STEPHANOPOULOS
Laboratory for Intelligent Systems in Process Engineering, Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.

(*Received* 19 *July* 1990; *received for publication* 26 *July* 1990)

Abstract—Recent advances in artificial intelligence have changed the fundamental assumptions upon which the progress of computer-aided process engineering (modeling and methodologies) during the last 30 yr has been founded. Thus, in certain instances, numerical computations today constitute inferior alternatives to qualitative and/or semi-quantitative models and procedures which can capture and utilize more broadly-based sources of knowledge. In this paper it will be shown how process development and design, as well as planning, scheduling, monitoring, analysis and control of process operations can benefit from improved knowledge-representation schemes and advanced reasoning control strategies. It will also be argued that the central challenge coming from research advances in artificial intelligence is "modeling the knowledge", i.e. modeling: (a) physical phenomena and the systems in which they occur; (b) information handling and processing systems; and (c) problem-solving strategies in design, operations and control. Thus, different strategies require different forms of declarative knowledge, and the success or failure of various design, planning, diagnostic and control systems depends on the extent of actively utilizable knowledge. Furthermore, this paper will outline the theoretical scope of important contributions from AI and what their impact has been and will be on the formulation and solution of process engineering problems.

Zusammenfassung—Neuere Entwicklungen bei künstlicher Intelligenz haben die grundlegenden Annahmen verändert, auf denen der Fortschritt der computergestützten Fertigungsplanung (Formgebung und Methodenlehren) während der letzten 30 Jahre begründet war. Daher stellen in bestimmten Fällen numerische Berechnungen schlechtere Alternativen für qualitativ und/oder halbquantitative Modelle und Verfahren dar, die breiter fundierte Wissensquellen einnehmen und benutzen können. In diesem Artikel wird gezeigt, wie Prozeßentwicklung and -gestaltung sowie Planung, Steuerung, Oberwachung, Analyse und Kontrolle der Prozeßtätigkeiten von verbesserten Wissensdarstellungsschemen und modernen vernünftigen Kontrollstrategien profitieren können. Es wird auch argumentiert, daß die zentrale Herausforderung, die aus Fortschritten bei der Entwicklung künstlicher Intelligenz kommt, "Modellierung des Wissens" ist, d.h. Modellierung: (a) physikalischer Phänomene und der System, in denen sie sich abspielen; (b) von Informationsbearbeitungs- und verarbeitungssystemen; und (c) problemlösender Strategien bei Planung, Betrieb und Kontrolle. Daher erfordern unterschiedliche Argumentationsstrategien unterschiedliche Formen des erklärenden Wissens, und der Erfolg oder der Mißerfolg verschiedener Entwurfsplanungen, Diagnosen und Kontrollsysteme hängt vom Umfang des tatsächlich nutzbaren Wissens ab. Weiterhin wird in diesem Artikel der theoretische Umfang wichtiger Beiträge durch künstliche Intelligenz dargelegt und deren Auswirkung auf die Formulierung und Lösung von Problemen der Fertigungsplanung in der Vergangenheit und Zukunft.

Résumé—Les nouveaux développements en matière d'intelligence artificielle ont modifié les suppositions de base sur lesquelles le progrès de la planification de l'usinage assistée par ordinateur (modelage et méthodologies) était fondé pendant les 30 dernières années. Les calculs numériques représentent donc dans certains cas des alternatives moins bonnes pour les modèles qualitatifs et/ou semi-quantitatifs ainsi que pour les procédés basés et utilisant des sources scientifiques à fondement plus large. Cet article montre comment le développement et la conception des processus ainsi que la planification, la commande, la surveillance, l'analyse et le contrôle des activités de processus peuvent profiter des schémas de représentation scientifique améliorés et des stratégies de contrôle modernes résonnables. Il est aussi argumenté que le défi central qui résulte des progrès accomplis dans le développement d'une intelligence artificielle est un "modelage des connaissances", c'est-à-dire un modelage: (a) des phénomènes physiques et des systèmes dans lesquels ils se déroulent; (b) des systèmes de traitement et de transformation des informations; et (c) des stratégies qui résolvent les problèmes lors de la planification, de l'exploitation et du contrôle. C'est pourquoi les diverses stratégies d'argumentation exigent diverses formes de savoir explicatif et le succès ou l'échec de diverses planifications d'ébauches, de diagnostics et de systèmes de contrôle dépent de l'étendue du savoir réellement utile. Cet article représente également l'étendue théorique d'importants apports par l'intelligence artificielle ainsi que leurs répercussions sur la formulation et la solution apportées aux problèmes de la planification de la production dans le passé et à l'avenir.

1. THE SCOPE OF COMPUTER-AIDED PROCESS ENGINEERING

The role of computers in process engineering research and development has grown continuously over the last 30 yr to a point of pervasive and self-propelling reliance on the machines. In the area of chemical process development and design, computers are used for: (a) process simulation and analysis; (b) equipment sizing and costing; (c) optimization; (d) integrated design of energy management systems, reactor networks, separation sequences; (e) layout of piping networks; and (f) project planning. Chemical process control is relying more and more on computers for the implementation of low-level feedback control, on-line parameter estimation and controller adaptation, as well as for the execution of higher-level tasks such as optimization, planning and scheduling of plant-wide operations.

Although the variety of computing applications is extensive and growing, the underlying paradigm has been unique and a simple one, namely, "*numerically solve a set of equations*." This is due to the fact that computers have been perceived to be "computational machines" only. Thus, even when the desirable is a qualitative description of a physical system's behavior, it will be established quantitatively under a limiting set of parametric values and initial conditions. Furthermore, the exclusive reliance on numerical computations forces one to "*use only the knowledge which can be represented by a quantitative scheme*," thus limiting the range and utility of the ensuing numerical results. Quite often, the exclusion of available qualitative and approximate-quantitative (e.g. ordinal, or order-of-magnitude relations) knowledge is very detrimental because the user either overlooks fundamentally sound scientific knowledge, or distorts such knowledge to fit it into a quantitative representational scheme. Needless to say, one should resist the down-grading of reliable quantitative knowledge to inferior qualitative or semi-quantitative forms. Instead, one should strive to *articulate, represent and utilize all forms of available knowledge*. This is the new dictum imposed by the needs in chemical engineering and made possible by present state-of-the-art advances in computer science technology.

To better understand the limitations of the paradigm based on numerical computations alone, let us examine a series of representative problems from chemical engineering:

Process design—Figure 1 shows an advanced computer-aided design environment for chemical process design. It is composed of a database management system (DBMS) which retrieves and directs information among the various facilities such as process simulators, optimizers, process units' sizing and costing routines, estimators of physical or chemical properties, etc. under the guidance of the human designer. But, such a computer-aided design system "does not know" how the design is done and cannot encode and answer questions like: Where does the design start from? What is to be done next? What simplifications and assumptions should be made for the design to proceed? The design strategy and methodologies for design decision-making reside in the expert human designer's mind and they never become articulated into automatic, computer-implemented procedures. As a result, a CAD system as the one of Fig. 1 cannot even encode and replay the "history of a design." Models of the design process which allow the development of "human-aided" design systems will be discussed later in this paper.

Product and process development—Scientists and engineers involved in the design of new products (materials, solvents, pharmaceuticals, specialty chemicals) or the conceptualization of new processing schemes stemming from basic chemical or biochemical reaction schemes do not use computers in their creative tasks because their essential needs are not numerical computations. Thus, fundamental qualitative scientific knowledge, or accumulated experimental facts are never formally articulated and represented.

Modeling system's behavior—Suppose that we want to establish the qualitative behavior of an assumed model for a catalytic reaction and thus investigate the effects of the postulated mechanistic steps. Numerical simulation depends heavily on the assumed values of the inherent parameters and provides only local information. Mathematical analysis, on the other hand, can establish for small size problems explicit global properties of the assumed

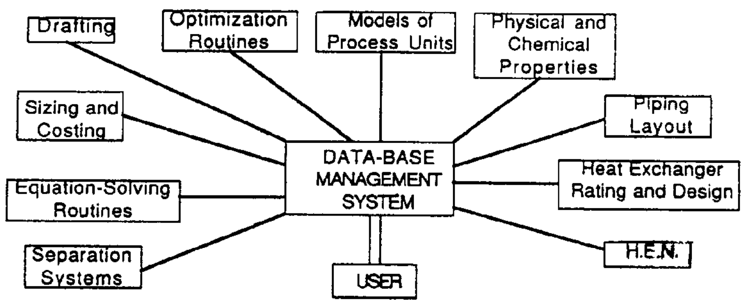

Fig. 1. Typical computer-aided design environment.

model such as number of steady state solutions and their character. Consequently, one should attempt to capture all fundamental results from mathematical analysis and formalize them through proper representations into a software system, which can produce the global qualitative behavior of the assumed catalytic mechanism over various regions of parametric values.

Feedback control—Quantitative models and numerical computations are and will continue to be central for the implementation of feedback control. Unfortunately, numerical computations alone are weak or not robust in answering questions such as: How well is a control system running? Are the disturbances normal? Why is derivative action not needed in a loop? What loops need dead time compensation; should it be increased or reduced? Have the stability margins of certain loops changed, and if so, how should the controllers be automatically retuned?

Monitoring and diagnosing process operations—Computer technology has caused an explosion in the process information that can be conveyed to human operators. Present-day control rooms display thousands of analog or digital process data, and hundreds of alarms. During the course of steady state operations, simple observation of scores of displays is sufficient to confirm the process' status. But, when the process is in transient or crises occur, the dynamic evolution of displayed data can confound even the best operators. Quantitative computations are inadequate to provide a robust "mental model" as to what is going on and carry out routinely tasks like: distingish normal from abnormal operating conditions, assess current process trends and anticipate future operational states, identify causes of process trends (e.g. external disturbances, process faults, operator-induced mishandling, operational degradation due to parametric changes). The key cognitive skill is the formation of a mental model of the chemical process that fits the current facts and enables the operator to correctly assess the process' behavior and predict the effects of possible control actions. The automatic development of such a mental model requires more knowledge than that provided by numerical computations alone.

Planning and scheduling of process operations—The planning of process operations involves specifying an ordered sequence of operations, or a partially-ordered set of operations which, when carried out, will perturb the state of the chemical plant from some initial state and cause it to eventually attain some prespecified final, or goal state. Conceivably, one could formulate this problem as a mixed-integer, nonlinear optimization problem and solve it numerically, if it were not for the following difficulties: (1) for realistic size industrial problems it can be shown that the problem is intractable; (2) nontemporal constraints introduce restrictions on the temporal ordering of process operations; and (3) the objective function cannot be fully articulated *a priori*. Consequently, additional forms of knowledge and symbolic generation and manipulation of primitive operations are essential for the synthesis of operating procedures either *a priori* (i.e. at the process design stage) such as start-up, shut-down, change-over or on-line (i.e. during operation) such as response to faults, coordinated plant-wide optimization.

It is clear from the previous discussion that present and future needs in chemical engineering cannot be met by the traditional computing paradigms. But, before we explore how artificial intelligence is impacting the reformulation of computing paradigms in various areas of chemical engineering, it is important to elucidate what the essential premises are on which current application of artificial intelligence in engineering problems is based.

2. THE ESSENTIAL FRAMEWORK OF ARTIFICIAL INTELLIGENCE

It is generally accepted that artificial intelligence is part of computer science and, in the words of Elaine Rich (1983), "... is the study of how to make computers do things at which, at the moment, people are better." Thus, while computers outperform humans in: (a) carrying out large-scale numerical computations: (b) storing and efficiently retrieving massive records of detailed data; and (c) efficiently executing repetitive operations, they are currently quite inferior in: (i) responding to situations very flexibly; (ii) making sense out of ambiguous or contradictory messages; (iii) recognizing the relative importance of different elements within a situation; and (iv) finding similarities despite differences, and drawing distinctions despite similarities among various situations; all considered to be manifestations of human intelligence (Hofstadter, 1980) and subjects of study in the realm of artificial intelligence.

Making a mind vs modeling the brain (Dreyfus and Dreyfus, 1988)—This definition could be construed as implying that AI is trying to "make computers think exactly like humans," i.e. creating a model of the brain. For engineering work such interpretation is wrong and obviously sterile. Instead, "making a mind" is a more accurate description of what AI applications in engineering are trying to do, i.e. they tackle the same problems humans do, with solutions that possess the robustness and flexibility characteristic of human approaches. Such a shift in emphasis has produced excellent examples of computer systems which exploit symbolic processing, novel models to represent all forms of knowledge, and a series of successful problem-solving paradigms, all results of research work in AI.

AI and computer programming—Rich's definition of AI has another important corollary: the research results should lead to an executable computer program. It is this requirement that places AI squarely in the area of computer science and distinguishes it from operations research, information science, systems theory, mathematical logic and other fields from which it has been, and still is, drawing in ideas and methodologies. Such requisite computer program, based on a computer language which "... is a novel formal medium for expressing ideas about methodology ... (and) ... control(ling) the intellectual complexity ..." (Abelson and Sussman, 1985) should, ideally, possess provable properties such as tractability, correctness and completeness. Unfortunately, this is a very hard proposition and for many applications, impossible to establish.

Modeling knowledge—Looking more closely at the various research advances one quiclky realizes that the practical thrust of AI in engineering applications is to enforce systematic and organized *modeling of knowledge*, i.e.: (a) modeling of physical systems (e.g. at the Boolean, qualitative, semi-quantitative or quantitative level); (b) modeling of information processing systems; and (c) modeling of problem-solving paradigms, such as diagnostic, planning, design. Without expressive representations of the requisite *declarative* (i.e. "what is ...?") and *procedural knowledge* (i.e. "how to ..."), no computer programs can be written. It is this preoccupation with all forms of knowledge and their representation that distinguishes current efforts from earlier ones in chemical engineering, and enables people to deliver what in the past was an "idea." But, developing the proper models to represent knowledge and generating programs with, ideally, provable properties, are highly-challenging propositions with significant intellectual content. This is the message that current research efforts of AI applications in chemical engineering convey. Current research has outgrown simple-minded rule-based systems.

Problem-solving paradigms—Complex codes for numerical computations have enjoyed a significant advantage: the existence of a concise, advanced mathematical background that provides the proof of the computer program's properties; e.g. stability, rate of convergence, residual errors. Most of the computer programs developed in the field of artificial intelligence are based on problem-solving paradigms, which are not as fortunate as their numerical counterparts. But, significant advances in mathematical logic, approximate algebras, qualitative and semi-quantitative calculus are providing the theoretical background for developing algorithms with provable properties.

Software design—Procedural programming based on a bottoms-up design of software systems has been the traditional mode in computer-aided process engineering applications. This is quite unsatisfactory for large-scale, diverse information processing required by process engineering. Thus, top-down design approaches based on object-oriented programming (Stephanopoulos *et al.*, 1987; Fikes and Kehler, 1985; Stefik and Bobrow, 1985) are evolving as the dominant paradigm, requiring a complete inversion of traditional thinking.

3. ARTIFICIAL INTELLIGENCE IN PROCESS DEVELOPMENT AND DESIGN

The engineering design of processing systems is a dialectic process (Stefik *et al.*, 1982) between goals (i.e. what is desired) and possibilities (i.e. what is actually realizable). But, the diversity of the top-level (initial) goals makes design quite an informal activity in the province of expert designers. Indeed, no general theory exists for the systematic and rigorous development of the procedure that leads to the design of the desired "artifacts." Various attempts to formalize the overall design procedure as mathematical programming problems have yielded limited success with rather narrowly focused problem definitions and quite rigid solution methodologies. Thus, these advances can be viewed as a *set of support tools*, rather than a *theoretical framework for design.*

In the absence of a general theory on how design is done, research in the field of AI has been addressing the following two distinct, but complementary, areas of inpuiry:

Axiomatic theory of design, with the objective to establish a theoretically firm ground for the definition of design and thus bring it into the realm of "science" rather than "art," where it presently stands. Advances in this direction have been rather recent, but significant research effort is currently underway.

Engineering science of knowledge-based design (Tong, 1987), aiming at the development of a rational framework for organizing, evaluating and formulating knowledge-based models of how the design is done. The central issue to resolve here is the identification and structuring of various forms of knowledge, which is pertinent to the design tasks. Two major areas of advancement have resulted from the use of AI-related research: (i) systematic modeling of the process of design; and (ii) new effective programming styles, which depart from the conventional computer-aided design paradigms and allow the development of large highly-complex computer programs.

3.1. The human-aided computer-based design paradigm

Despite the continuous enrichment of traditional CAD environments, the character of the overall design procedure remains the same: "*the human does the design and the computer provides the support tools, without understanding the design process, its rationale, or the design decisions.*" But, the complete structure of tasks during the design of an engineering artifact can be very large, detailed and complex for any human to document mentally and carry with him/her. To the extent that we can untangle and make explicit the design procedure, thus emulating the designer's own methodology, the process can be mechanized.

But, in this case, we are moving towards a "human-aided, machine-based design" paradigm (Stephanopoulos and Kriticos, 1987), where the computer through human guidance can carry out significant portions of a design by "understanding" the design process itself, its rationale and the reasoning behind a number of design decisions. This is the paradigm whose development and computer implementation has been significantly advanced by research in AI, and which we will discuss in the following sections. The benefits from the availability of such mechanized models for design are many and diverse: (1) improvements in cost and reliability; (2) explicit documentation of the design process itself: why certain goals were set during the design and how they were achieved; how design decisions were made; what assumptions and simplifications were involved; what models were used at the various stages of design; what alternative designs were examined, and why certain ones were selected over others; (3) explicit documentation of the designed artifact itself; i.e. what are its components and their characteristics, how are they interconnected, what are its functional and performance characteristics, what are the critical design variables and the intrinsic trade-offs; (4) easy verification and modification of the resulting design. Having an explicit documentation of the intermediate design tasks, generated alternatives, rationale behind various design decisions, assumptions, conjectures and simplifications, one can replay the design scenario and easily verify the validity of the derived design, or modify its design premises for further improvements; and (5) the mechanized model of a design methodology offers an excellent depository for the organization of new empirical knowledge and/or the systematic incorporation of new theoretical results and analytic tools. Such inclusion of new knowledge will progressively increase the automation of the design procedure itself.

3.1.1. Modeling the process of design. Figure 2 shows a generic model, suggested by Tong (1986), which captures the essential features of the human-aided design paradigm. It is composed of three distinct facilities, an *Advisor*, a *Planner* and a *Designer*, which can interact in both top-down and bottom-up directions. Specifically, in a top-down architecture the Advisor develops and maintains a network of design goals (a partial ordering of the design steps). It also assesses the qualitative impact on selected goals by identifying bottlenecks in a proposed design plan. It embodies a theory of how goals are created, prioritized, decomposed, how they interact and how they are satisfied. The Planner receives a goal from the Advisor and simulates the design steps using a detailed planning theory. As a result, it completes the design plan, initially sketched by the Advisor, by identifying all the specific engineering design tasks. The Designer maintains the representation of the engineering artifact being designed and other domain-specific knowledge. Thus, given a design step (from the Planner), the Designer must simulate the step and update the artifact's representation accordingly. In a bottom-up architecture the Designer detects conflicts, and/or using the domain-specific data, generates extensions to the design plan which are communicated to the Planner.

Although the prototype of Fig. 2 is quite generic, various models of the design process result from the following considerations: goal-driven and data-driven strategies are present in every model of the design process, but their contributions may differ widely: (i) different theories are employed to stipulate different networks of goals, i.e. create, prioritize, decompose, satisfy design goals; (ii) different planning methodologies are used to order the design steps, leading to design strategies with different computational complexity; (iii) different control strategies are invoked to coordinate the interaction between goal-setting and planning the design steps; and (iv) different representational schemes are used to describe the design states of the engineering artifact under design.

3.1.2. Modeling components. But, what are the essential components of a model of the design methodology? To answer this question, Mostow (1985) has summarized the various efforts in modeling the design process and has suggested that a comprehensive, computer-based model of design should address the methodological and/or representational aspects of the following elements: (a) *the state of the design*, i.e. descriptions of the artifact under design at various levels of detail; (b) *the goal structure of the design process*. If the design is to be mechanized, it must be a purposeful activity with clear goals, which guide what is to be done at each point of the design process. These goals are not "descriptions of the artifact under design, but prescriptions as to how these descriptions should be

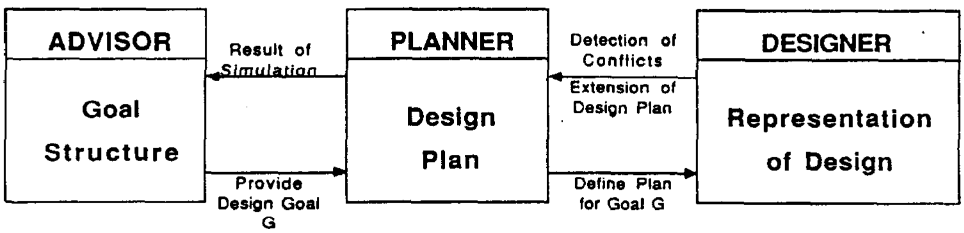

Fig. 2. A generic model for the design process.

manipulated;" (c) *design decisions*. Once a goal has been selected, there may be several paths for achieving it. Design decisions represent choices among them and should be clear and explicit; (d) *rationale for design decisions*. They justify the goal selection and the choice of the best plan to accomplish it. The need for explicit rationalization forces the designer to evaluate his/her reasoning and allows the design procedure to evolve systematically as new knowledge becomes available; (e) *control of the design process*. Provides the navigation through the design alternatives by guiding the designer in how to choose the goal to work on at each point and selecting the best plan with which to achieve it. This is the most critical component of the design methodology; and (f) *learning in design*. Emulates the designer's own ability to learn both general knowledge about the domain and specific knowledge about the problem at hand, from the accumulation of factual and strategic knowledge.

3.1.3. Issues to be resolved. Current (future) research efforts are (will be) focusing on resolving the following issues, which are presently inhibiting the full deployment of human-aided machine-based design environments: (i) develop concise representations of the state of design which allow the description of the process at various levels of detail; (ii) formalize the generation, aggregation, disaggregation, elimination or propagation of design constraints as the design evolves; (iii) compose efficient and provable algorithms for the logical control of design activities, which can tame the complexity of the overall problem; and (iv) mechanize automatic learning from past design experience.

3.2. Cooperative process design

The design of a chemical process goes through various stages of evolution, requiring the input of diverse expertise to deal with: (i) process technology characteristics; (ii) safety considerations; (iii) sizing, costing and economic optimization; (iv) operability specifications; and (v) desired controlled operation. Presently, all of these activities are carried out in a largely sequential and segmented fashion. The availability of software systems emulating a concise model of how process design is done can integrate these activities and lead to a *Cooperative Process Design* environment, functioning as follows: (i) different designers with different expertise are operating from different engineering workstations; (ii) object-oriented intelligent database systems provide a global database reflecting the current state of process design and private databases retaining different versions of the evolving design; (iii) the various expert designers use the global database to assert their preferences, introduce their design choices and critique decisions made by others. Assumptions, decisions and methodologies are explicitly available in the global database; and (iv) various versions of the process design are developed as noncommensurable decisions are stipulated by various experts for subsequent study and analysis. Such systems of cooperative engineering design are already available in experimental forms for the design of manufacturing systems.

4. ARTIFICIAL INTELLIGENCE IN PROCESS OPERATIONS AND CONTROL

The design of process control systems and the on-line control of chemical plants require the systematic coordination of a multitude of tasks. In turn, each of these tasks has many facets requiring data, numerical algorithms, decision-making procedures and human intervention to provide experiential knowledge. AI can play an important role in automating many of these tasks through the use of computers.

4.1. The autonomous process control system

A chemical plant is part of a corporation-wide network of processing systems. Operations Planning (Ladson and Baker, 1986) establishes multi-plant production logistics (production plans, budget, raw materials, desired products, sales, inventories). Within the scope of these constraints, Plant-Operations Management defines the multi-period production plans, resolves crises, plans scheduled maintenance and shutdown and evaluates system's performance. This hierarchy of control tasks continues with the middle level of Control Strategies and Coordination and the lower level of Direct Control and Adaptation. Although this hierarchy is an old idea (Lefkowitz, 1966; Mesarovic *et al.*, 1970; Findeisen *et al.*, 1980), past work did not address the fundamental issues which prevented its full implementation. AI is providing the enabling theory and technology, thus leading to the emergence of the so-called Autonomous Process Control System (Stephanopoulos, 1989).

4.1.1. Plant-operations management. The essential structure of functions is shown in Fig. 3. The *Plant-Operations Planner* is entrusted with the development of a feasible plan of plant operations over a period of time. To achieve its objective: (a) it defines the scope of the planning problem; and (b) it calls on other functions such as *Production Optimizer*, *Start-up/Shut-down Planner* or *Crisis Manager* to solve the planning problem. To define the scope of the planning problem, the *Plant-Operations Planner* receives information from the following sources: (i) from corporate production plans it learns the production goals; (ii) from the *Plant/Operations Assessor* it receives information about the status of processing units in the plant, the available processing capabilities, and whether a crisis has developed and needs attention, or is developing and should be deflected; and (iii) from the *Maintenance Scheduler* acquires information as to what units are marked for preventive or corrective maintenance. Defining the planning problem requires significant intelligence on the part of the *Plant-Operations Planner*, which must be able

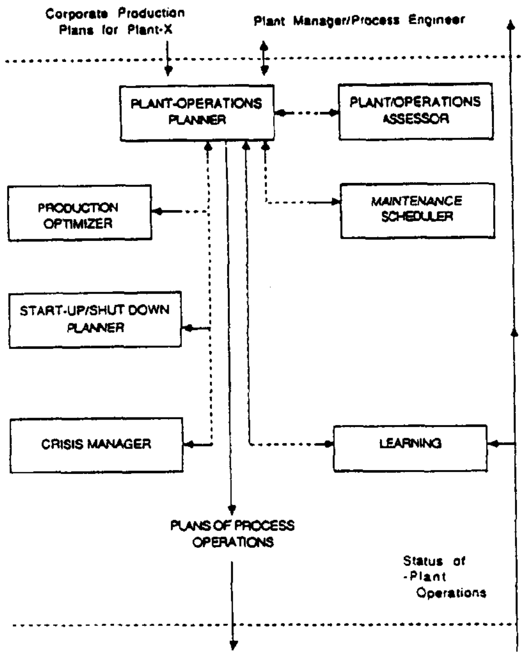

Fig. 3. The structure and functions of the Plant-Operations Planning segment.

to do the following: (a) generate symbolic descriptions of states (initial, goal), objectives, constraints and subjective preferences; (b) have decision-making abilities to identify and resolve conflicts among desired goals, constraints, preferences and select the appropriate function to carry out the planning; (c) adapt and expand its knowledge, based on the performance of plans already executed; and (d) identify infeasibilities in the corporate production plans, resolve them and return the information to corporate planning. The *Production Optimizer* is called to produce a multi-period optimal plan of operations. The methodology here is fundamentally algorithmic (mathematical programming), but the essential difficulty lies in the formulation of the optimization problem. The *Start-up/Shut-down Planner*, under normal conditions, encapsulates standard operating procedures. It is essentially symbolic in character and incorporates significant amounts of knowledge and decision-making. *Crisis Manager* is entrusted with the "creative" planning of operating procedures, which will deflect incipient problems in production. The recognition of crisis rests with the *Plant-Operations Planner*. Since the *Crisis Manager* does not contain but a limited number of crisis scenarios, it must possess extensive decision-making abilities to synthesize novel plans. The *Learning* function attempts to capture new knowledge by comparing the performance of plant operations with the intended goals, and construct *ad hoc* rules or generalizations of fragmented observations on how planning at this level can be improved.

4.1.2. Control strategies and coordination. The net outcome of the upper-level tasks is the formulation of multi-period operational plans. The functions of the middle-level tasks (Fig. 4) will: (a) convert these plans to control strategies; or (b) adapt the current control strategies using information from the *Direct Control and Adaptation* level tasks. The *Control-Strategies Synthesizer* and the *Control-Strategies Executive* are the two central functions at this level. The first schedules sequences of control actions, while the second coordinates the use of various resources needed for the implementation of the control actions. The *Control-Strategies Synthesizer* first defines the scope of the control-strategies synthesis problem, by receiving the following information: (i) plans of process operations established at the upper-level by the *Plant-Operations Planner*; (ii) evaluation of current process trends from the *Process-Trends Analyzer*; (iii) equipment, processing or performance faults from the *Process-Trends Interpreter*; and (iv) commands, rules or preferences from the process operator. This information is used to establish: (1) the current operating state and trends; (2) desired control objectives; (3) processing constraints; and (4) human preferences on the desired sequence of control actions. Second, it determines a schedule of control actions, which satisfies the established scope, by calling on the functions *Optimizing Control* or *Response-to-Faults Scheduler*. It requires significant intelligence and must possess skills in symbolic manipulation of data, models and plans, and decision-making in resolving conflicts.

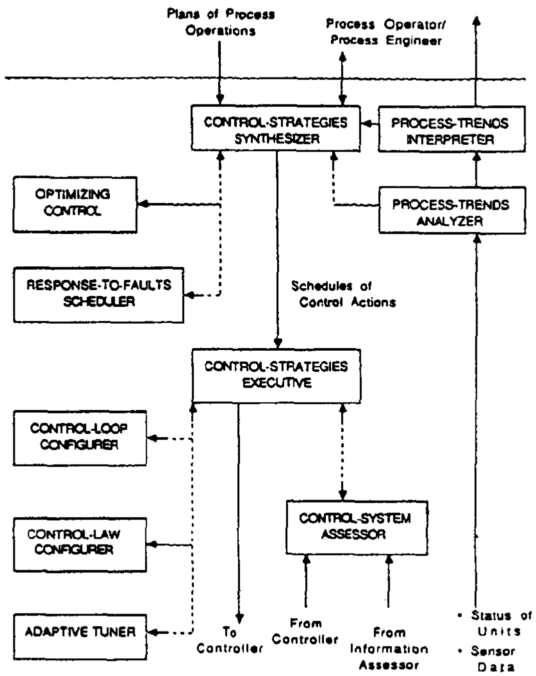

Fig. 4. The structure and functions of the Control-Strategies Synthesis and Coordinating segment.

The *Optimizing Control* function carries out optimization of a functional objective subject to a set of differential-algebraic constraints. It is essentially a numerical procedure, but is must be complemented with symbolic manipulations and decision-making skills to allow: (a) adaptation of the models as the process moves through an expanded region of operation; (b) adaptation of existing or specification of new operating constraints; (c) specification or adaptation of the objective function(s); and (d) specification of preferences, trade-offs, given that optimizing control is essentially multi-objective. The *Response-to-Faults Scheduler* is basically a scheduling facility, operating at a higher frequency than the *Crisis Manager* at the upper level. It is invoked when the *Control-Strategies Synthesizer* has decided that the information from the *Process-Trends Interpreter* indicates the presence of faults to contend with. It carries out the following tasks: (1) establishes the goal-state that the desired control strategy should make possible to reach; i.e. retain or return operation to the feasible region; (2) determines the list of available resources; i.e. pumps, valves, control loops, stand-by units, etc. which can be used for the synthesis of the control strategy; (3) defines the set of constraints (hard and soft) which must be obeyed; and (4) schedules a sequence of operating steps; e.g. changes in controller set-points, turning pumps on or off, closing or opening valves, switching units on or off. An essential feature of the *Response-to-Faults Scheduler* is its ability to predict future operating states and trends, through qualitative and semi-quantitative simulation. This is needed: (i) to evaluate the consequences of existing abnormalities (faults); and (ii) to evaluate and compare alternative schedules, given that the available constraints are not sufficient to establish a unique schedule.

The *Process-Trends Analyzer* and *Process-Trends Interpreter* provide information on the current status of processing equipment and operations. The former establishes the temporal trends of operating variables through declarative descriptions (Cheung and Stephanopoulos, 1988). The latter interprets these trends and determines whether the operation is normal or abnormal. In the second case it identifies the source(s) of abnormalities, e.g. equipment faults, processing malfunctions (e.g. sintering, poisoning of catalysts, caking, fouling of heat exchangers) or performance deterioration (e.g. decreased yields, shifts in products composition). Both are highly symbolic in character and they involve qualitative and semi-quantitative modeling and logical decision-making.

The *Control-Strategies Executive* receives schedules of control actions and coordinates the resources for their implementation. To accomplish its tasks, it uses information from the *Control-System Assessor*, which provides the following data: (i) current control loop configuration, process models and constraints used in the control law, executable control laws and controller adaptation mechanisms; (ii) evaluation of control system performance; (iii) loss of control due to control valve failure, controller saturation or faults in communication lines; and (iv) loss of information due to faulty sensor(s) or communication lines. The above failures are rather abrupt and are detected directly at the lower level (see *Direct Fault-Detector*) in contrast to equipment or processing faults which are detected indirectly at the middle level (see *Process-Trends Interpreter*).

The information from the *Control-System Assessor* defines the scope within which the *Control-Strategies Executive* determines the control-loop configuration, control laws and the adaptation of controller tuners. Each of these tasks is carried out by special functions. The *Control-Loops Configurer* determines the structure of control loops. It uses algorithmic, control-theoretical methods and a set of rules to account for *ad hoc* past experience (Tzouanas *et al.*, 1988), or to propagate semi-quantitative modeling constraints. Since the *Control-Loops Configurer* may change the present configuration, it must also provide an operating schedule for switching from the present to the next configuration. Such schedule provides the interface between numerical controller designs and logic-based decision-making.

The *Control-Loops Configurer* seeks information from the *Control-Law Selector*, which encompasses theoretical and heuristic knowledge to suggest the preferred algorithm for the implementation of direct control.

The *Adaptive Tuner* determines whether certain conditions are met in order to adjust, tune, certain parameters in the adaptation laws. It uses predetermines criteria based on excessive output, state and parameter errors, all of which is information supplied by the *Control-System Assessor*.

Machine learning is, in principle, possible at two points: (a) comparing the resulting process trends with the schedules of control action produced by the *Control-Strategies Synthesizer*, one can evaluate the merits of decisions made by the *Optimizing Control* or the *Response-to-Faults Scheduler*; and (b) the merits of decisions orchestrated by the *Control-Strategies Executive* can be judged against the performance evaluation offered by the *Control-System Assessor*. Very little work has been done along these lines, because the declarative descriptions of process trends and performance evaluation has been of low expressive value.

4.1.3. Direct control and adaptation. The central component here is the *Controller* (Fig. 5), which encompasses the numerical algorithms of "Control Law" and "Estimation/Controller Adaptation." It can accept any of the available conventional methodologies, in a direct or indirect adaptation strategy. The *Controller* determines the law governing the actions of the actuators after it receives the following information from the *Control-Strategies Executive*: (i) whether to change the control-loops configuration

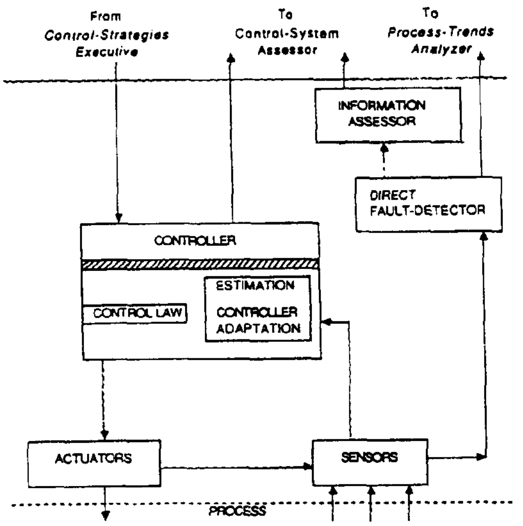

Fig. 5. The structure and functions of the Direct Control and Adaption segment

or not; (ii) whether to change the control law or not; (iii) a schedule of actions to transfer operation from the old to the new control-loop configuration and/or control law, if a change is warranted; (iv) in case of a control law change, the new process models and constraints, on which the new control law is based; (v) whether to adapt the parameters of the estimator and/or adaptive mechanism; and (vi) a schedule of actions to transfer operation from the old to the new estimator/adaptor. In turn, it sends to the *Control-System Assessor* information regarding its current control-loop configuration, control laws, identification and controller adaptation mechanisms.

Several papers have recently suggested the use of rule-based expert systems for the adaptive tuning of the control law (usually PID), and/or the protection of the control law through a series of logical conditions. Such solutions are basically *ad hoc*. Analytic rules can be derived to "protect" the stability of adaptive mechanisms which, nevertheless, can and should be developed algorithmically. The use of rules as "safety jackets" around PID or other control laws is a poor way to account for the inherent constraints, which should be part of algorithmic control laws.

The *Direct Fault-Detector* uses raw data from sensors and actuators to detect abrupt faults in sensors, actuators or communication lines. The fault-detection methodologies, used at this level, are based on simple qualitative rules which are presently being hardwired into local processors attached to individual sensors and actuators. The *Information Assessor*: (a) confirms/rejects the detected faults; (b) enables or disables the filtering of information to the *Control-System Assessor*; and (c) evaluates the performance of the controller with particular emphasis on the identification of excessive output, state or parameter errors. It passes this information to the *Control-System Assessor.*

4.2. *Planning and scheduling process operations*

Within the scope of the Autonomous Process Control System, planning and scheduling of process operations are manifestly critical activities. In the problem of planning and scheduling process operations, this automation can take two forms: (a) *a priori* synthesis of process operations for subsequent implementation; and (b) automatic, on-line planning and scheduling in real time, while responding to feedback of information about changing conditions and process trends. The first of these options is appropriate for planning operations such as start-up, routine shutdown or equipment and process changeover and can be extended to include a series of predetermined safety fall-back operations. The second option accounts for a wide variety of operating conditions and involves on-line decision-making. Typical examples are automatic response to process faults and planned evolution to a new operating state for the purpose of optimizing operational performance.

The planning of process operations involves specifying an ordered sequence of operations, or a partially ordered set of operations which, when carried out (applied), will perturb the state of the chemical plant from some (given) initial state, and cause it to eventually attain some pre-specified final or "goal," state. Typically, this transformation cannot be achieved in a single step, and the plant must be taken through a series of intermediate states. The operation of the plant at these intermediate states must be consistent with physical constraints such as conservation of mass, energy and momentum, and equilibrium and rate phenomena. An operations plan which satisfies this condition is termed "physically feasible." In addition to the physical constraints, intermediate states are required to satisfy certain "operational" constraints imposed on the plan (practical feasibility), such as upper bounds on reactor temperatures, prohibition of explosive mixtures, etc. Finally, for underconstrained planning problems, more than one feasible plan may exist. In such cases, more stringent performance criteria are required to select a small set of "efficient" plans from among the many alternatives.

Research in the area of planning falls into two categories. One school of work focuses on those theoretical aspects of the problem which do not depend on the particular area of application, the so-called "domain-independent" theory of planning. Other research has concentrated on exploiting characteristics specific to a given problem domain in order to construct special-purpose planning programs. Clearly, if domain-independent planning theory could handle all of the complexities of practical planning problems, this approach would be preferable to one which was restricted to a particular domain of application. However, there is theoretical evidence to suggest that building a provably correct, complete, domain-independent planner that is versatile enough to solve real-world planning problems, is impossible. Lakshmanan and Stephanopoulos (1998a, b) developed a complete methodology for the synthesis of operating procedures, based on functional operators within the scope of nonlinear planning strategies, initiated in the field of AI. The complete methodology consists of two phases: (a) the *Problem Formulation Phase*; and (b) the *Plan-Synthesis Phase*. During the Problem Formulation, the user describes: (i) the initial state; and (ii) the desired goal state. Special algorithms have been developed to ensure automatic completeness and consistency in the specification of the initial and goal states. Also, the computer generates automatically concurrent goals. The Problem Formulation ends with the (iii) specification/identification of operational constraints on the desired plan.

Having defined the operations planning problem in a form which is understood by the computer, the next phase of the planning methodology consists of synthesizing plans to solve the stated problem. This generation of plans is carried out in three stages: (1) *Identification of Primitive Actions*. First, the primitive operations required to carry out the transformation from initial to final states must be identified. In this methodology, the means-ends analysis paradigm is employed to identify these primitive "operators." (2) *Construction of a Partial Plan*. The second stage involves the construction of a "partial plan." This consists of deriving a partial ordering on the primitive operations. This partial ordering stems from the operational constraints placed on the operating plan, together with the physical laws which govern plant operation, and is analogous to the constraint-posting philosophy of nonlinear planning. (3) *Synthesis of Complete, Feasible Plans*. At present, the nonlinear planning methodology only manipulates operational constraints which are temporal in nature. Some operational constraints stated by the user may not be amenable to direct transformation into temporal constraints between goals. In such cases, a systematic, linear, generate-and-test strategy based on the partial plan generated in the previous step, is used to develop a feasible plan (or a set of feasible plans) which solve the problem.

4.3. *Representation and analysis of process trends*

Informational overload is a typical symptom of modern-day computer-aided process operations and control. The Autonomous Process Control System requires that systematic and formal representations are available for modeling and analyzing the process trends, as these are generated from the vast amounts of sensor data. Unfortunately, process trends are either modeled on a very fine scale based on the sequence of discrete sensor data, or through curve fitting and interpolation in every conceivable way. These representations are very poor, cannot reflect the generic properties of the trends, are often inconsistent and create problems as one tries to coordinate feedback control, adaptive control, diagnosis and optimization within the scope of an Autonomous Process Control System.

Cheung and Stephanopoulos (1990) have developed a formal representation of process trends based on the *triangular modeling of temporal episodes* (Fig. 6). It can be proven that this representation is: (a) complete; (b) correct; (c) robust to scaling and modeling errors; and (d) quite compact. It allows the generation of quantitative, qualitative and semi-quantitative relationships among process variables and provides consistent models of the process behavior at various levels of detail. Such representation schemes need to be deployed in real-time environments to support the construction of "mental models" of what is happening in a chemical process during operation.

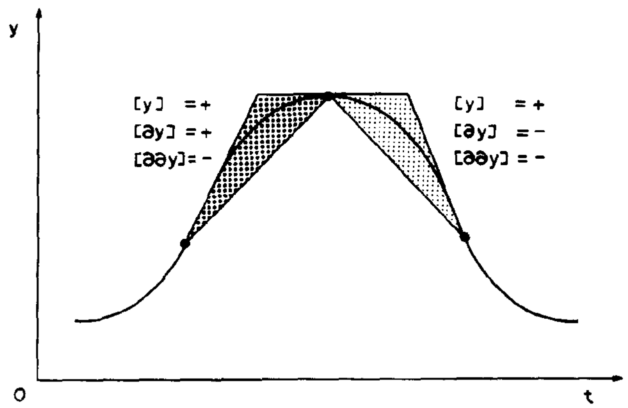

Fig. 6. Example of two consecutive triangular episodes.

4.4. Neural networks and recognition of operating patterns

In the past few years, neural networks have evolved to powerful computational paradigms. They are programmable, dynamic systems which process information from sensors through activation and inhibition rather than through the transmission of symbolic information. Knowledge is represented by the strength of the interconnections among the nodes of a neural network. A neural network model can learn the relevant features of its input environment in spite of noisy or contradictory inputs, producing useful patterns among the features of its environment. It is adaptive in character and possesses short-term and long-term memory. Neural networks have been proposed to be used for adaptive process control and process fault diagnosis (Hoskins and Himmelblau, 1988; Jones and Hoskins, 1987). But, certain clarifications are in order before one embarks on an instructive adoption of neural networks in process operations and control: (1) neural networks should be used to uncover unknown patterns and not to model existing ones; (2) neural networks should not be viewed as alternative parameter or state estimation techniques; (3) neural networks require extensive training based on input–output data. If a problem area uses input–output data based on existing models, neural networks are inferior solutions; (4) neural networks should be used on-line in process operations to identify hidden patterns and relationships among process variables; and (5) neural networks are excellent in analyzing vast amounts of records of data one accumulates during the operation of chemical plants.

5. MODELING LANGUAGES FOR PROCESS ENGINEERING

Modeling has long been the cornerstone of the engineering approach to problem-solving. Models are essential components of problem-solving methodologies used to tackle process engineering.

Since early efforts, computer-aided modeling has been generally organized around unilateral computations, which perform predetermined operations on fixed inputs to yield values of desired outputs. But, a series of inherent weaknesses have pointed out the disadvantages of the traditional approach: (a) the time and cost associated with computer-model development are high; (b) the resulting models are difficult to document and maintain adequately; (c) the re-use of computer-aided models is minimal; they tend to be task-specific and are often intrinsically linked to solution procedures; (d) the models cannot be synthesized automatically by the computer in the course of automatic execution of an engineering task; and (e) for interactive modeling the modeler is required to be highly skilled in programming. As a result of these weaknesses, the duplication of modeling efforts has been enormous. Accumulated modeling knowledge is almost impossible to use, since the underlying modeling context (purpose, assumptions, simplifications) has never been documented and rationalized. So, why must every new modeling effort start from scratch?

All of the above weaknesses are due to the following two essential premises on which past and current computer-aided modeling has been based: (1) The contextual "declarative" knowledge, i.e. the "what is ..." knowledge, around a processing unit is partly articulated. For example, process models represented by sets of equations do not include *explicitly* information such as the following: (i) underlying assumptions; (ii) simplifications, made by the modeler to limit model's validity over a given range of conditions or to underscore the relative importance of various physico-chemical phenomena; (iii) scope of engineering task, i.e. what the model was intended for. Thus, different models are needed to represent the process at the overall input–output level, the process-segment level, the process-unit level or the process-sub-unit level; and (iv) missing relationships, including inequalities, order-of-magnitude and qualitative relationships (Mavrovouniotis and Stephanopoulos, 1988). These are not normally part of process models because conventional analytic techniques cannot handle them. (2) The declarative knowledge, which is articulated in a model, is often intrinsically integrated with the "procedural" knowledge, i.e. the "how to ..." methodologies (e.g. algorithms for the solution of equations of optimization problems), which in turn depend on the specific characteristics of the modeling relationship. Stephanopoulos *et al.* (1990) have developed MODEL.LA, a formal language for the modeling of processing systems to be used in various process engineering tasks. It was designed to achieve the following major objectives: (a) represent a processing system at any level of detail by using multiple coexisting levels of abstraction which can communicate betwen each other and which can explicitly keep track of interrelated units, and their associated modeling relations. The ability to represent multiple views is a critical feature of the system that distinguishes it from other modeling systems; (b) automatically generate the set of basic relationships (balances, reaction and transport rates, equilibrium equations, etc.) that describe the system. This requirement implies the development of explicitly structured mathematical models, involving variables, terms and relationship objects; (c) capture and utilize qualitative, semi-quantitative relationships (ordinal, order-of-magnitude) or Boolean relationships. Such requirement will allow the modeling system to be used beyond the scope of traditional simulators, e.g. for diagnostic systems, autonomous process control systems, automated process design, automated planning and schedule of process operations; and (d) offer explicit documentation of all the hypotheses, assumptions and simplifications that give rise to a particular model. The model should have all the characteristics of the knowledge it embodies—it should be transmittable from one person to another and open to modification, improvement and combination with another knowledge. Thus it should allow a direct mapping between the ideas of process model and the knowledge-base of a project.

MODEL.LA is composed of the following items: (i) a set of six elementary objects used to describe any structural or behavioral characteristic of processing systems; Generic-Unit, Port, Stream, Context, Constraint, Variable; (ii) a set of 10 semantic relationships describing all possible interactions among the six objects; and (iii) a precise set of rules determining the language's syntax which is an extended BNF (Backus–Naur Form). MODEL-LA is becoming the modeling language of DESIGN-KIT (Stephanopoulos *et al*., 1987), an object-oriented system for process engineering activities.

REFERENCES

Abelson H. and G. J. Sussman, *Structure and Interpretation of Computer Programs*. MIT Press, Cambridge, Mass. (1985).

Cheung J. T.-Y. and G. Stephanopoulos, *Computers chem. Engng* **14**, 495 and 511 (1990).

Dreyfus H. L. and S. E. Dreyfus, *DAEDALUS*, Winter (1988).

Fikes R. and T. Kehler, *Commun. ACM* **28**, 904 (1985).

Findeisen W. *et al*., *Control and Coordination in Hierarchical Systems*, Wiley, New York (1980).

Hofstadter D. R., *Gödel, Escher, Bach: An Eternal Golden Braid*. Vintage Books, New York (1980).

Hoskins J. C. and D. M. Himmelblau, *Computers chem. Engng* **12**, 881 (1988).

Jones B. and J. C. Hoskins, *BYTE* **12**, 155 (1987).

Lakshmanan R. and G. Stephanopoulos, *Computers chem. Engng* **12**, 985 (1988a).

Lakshmanan R. and G. Stephanopoulos, *Computers chem. Engng* **12**, 1003 (1988b).

Lasdon L. and T. Baker, *Chemical Process Control—CPC III* (M. Morari and T. J. McAvoy, Eds). CACHE-Elsevier, New York (1986).

Lefkowitz I., *ASME, J. Basic Engng* June, 392 (1966).

Mavrovouniotis M. and G. Stephanopoulos, *Computers chem. Engng* **12**, 867 (1988).

Mesarovic M., D. Maco and Y. U. Takahara, *Theory of Hierarchical, Multilevel Systems*. Academic Press, New York (1970).

Mostow J., *AI Mag.* **6**, 44 (1985).

Rich E., *Artificial Intelligence*. McGraw-Hill, New York (1983).

Stephanopoulos G., *The Shell Process Control Workshop—II* (D. M. Prett and C. Garcia, Eds). Butterworths, London (1989).

Stephanopoulos G. and Th. Kriticos, *NSF-AAAI Workshop on AI in Process Engng*, Columbia University (1987).

Stephanopoulos G. J., J. Johnston, T. Kriticos, R. Lakshmanan, M. Mavrovouniotis and C. Siletti, *Computers chem. Engng* **11**, 655 (1987).

Stephanopoulos G., G. Henning and H. Leone, *Computers chem. Engng* **14**, 813 and 847 (1990).

Stefik M. and D. G. Bobrow, *AI Mag.* **6**, 40 (1985).

Stefik M., D. G. Bobrow, A. Bell, H. Brown, L. Conway and C. Tong, *Proc. Conf. Adv. Res. in VLSI*, Massachusetts Institute of Technology, p. 43 (1982).

Tong C., AI/VLSI Project. Working Paper 34, Rutgers University (1986).

Tong C., *AI Engng* **2**, 133 (1987).

Tzouanas V. K., C. Georgakis, W. L. Luyben and L. H. Ungar, *Computers chem. Engng* **12**, 1065 (1988).