ENGINEERING CONTROL SYSTEMS AND COMPUTING IN THE 1990s*

John L. Casti

May 1985
WP-85-31

*Written version of a presentation given at the Oslo Symposium on "Applied Mathematics, Systems Analysis and Computer Science, September 17-21, 1984.

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

FOREWORD

In September 1984 a workshop was held in Geilo, Norway, jointly sponsored by IIASA and the Nordic Council of Ministers, to explore the projected developments in applied mathematics, systems analysis and computer science in the 1990s. This Working Paper is a written version of one of the invited presentations.

Boris Segerstahl
Leader
Science and Technology
Program

ABSTRACT

The relationship between computing hardware/software and engineering control systems is projected into the next decade, and conjectures are made as to the areas of control and system theory that will most benefit from various types of computing advances.

ENGINEERING CONTROL SYSTEMS AND COMPUTING IN THE '90s.

J. Casti

1. Control Science and Computing

Theoretical developments in the control and system sciences have always gone hand-in-hand with advances in computing technology. In the pre-computer era, theoretical formulations emphasized aspects of control processes (e.g., stability) that could be relatively easily ascertained by analytic means. With the development of analog computers in the 1930s and 1940s, emphasis was placed upon theoretical formulations that took advantage of the analog computer's capacity to solve boundary-value problems; the Euler-Lagrange formulations for variational problems and the Wiener-Kolmogorov filter being good illustrations of this point. Finally, with the widespread availability of the von Neumann architecture digital computers from the late 1950s, the theoretical emphasis shifted to initial-value formulations of control problems, as evidenced by the development of dynamic programming aoproaches to variational problems and the widespread use of the Kalman filter.

We are now in the midst of another major discontinuity in computing hardware and software technology, with new architectures, operating systems, programming languages and theoretical constructs emerging daily. It is clear that these developments will profoundly affect both the theory and applications of control and systems science for years to come. In this paper we shall consider some of the major trends in computing and speculate on their implications for theoretical control science, as well as try to indicate some of the engineering areas in which new applications of control and system techniques are being made possible through these enhanced computing capabilities.

2. Control Systems: Problems and Concepts

We consider a control process represented in internal, or state-variable form,

$$\dot{x} = \phi(x,u) \quad , \qquad x(0) = x_0 \quad , \qquad (\Sigma)$$
$$y(t) = h(x) \quad ,$$

where the control function $u \in \Omega$, a set of admissible inputs, while the observed output $y \in \Gamma$, a set of output functions. The quantity $x(t) \in X$ is the state of the system Σ. For our purposes, we shall assume $u(t) \in R^m$, $y(t) \in R^p$, $x(t) \in R^n$. The functions ϕ and h are assumed to belong to some class of functions possessing suitable analytic properties.

In the foregoing set-up, it is clear that if the initial state x_0 and an input function $u(t)$ are given, a corresponding output function $y(t)$ is generated. The map

$$f : \Omega \to \Gamma \qquad (\xi)$$

represents the so-called <u>input/output</u> (or <u>external</u>) description of the process (here we suppress the dependence on the initial state). Most of the interesting questions of system modeling and control revolve about the interplay between the description (ξ) and (Σ), under various hypotheses concerning the sets Ω, Γ, X and the maps ϕ, h and f.

- <u>Main Problem of System Modeling</u> - the key question of system theory upon which all else depends is the so-called <u>problem of realization</u>:

> Given the sets Ω and Γ, together with the external description f, determine "good" internal descriptions $\Sigma = (X, \phi, h)$ whose external behavior agrees with f.

The interpretation of this problem is clear. The external descriptions (Ω, Γ, f) represents the experimental evidence, the data. The objective is to find a model Σ that "explains" the data and, at the same time, is the "best" of all such models that agree with the observed data.

In order to make precise what is meant by a "good" model Σ, we must introduce the concepts of reachability and observability. Intuitively, a good model is one that is minimal, in some sense. In passing from an external description to the model Σ, the only mathematical construction involved is the state space X. Thus, it is natural to ask that X be "as small as possible". If X is a vector space, then we require dim X to be minimal; however, for nonlinear f, X is not generally a vector space, so the concept of dimension loses its meaning. Nonetheless, by use of the properties of reachability and observability we may still impose a minimality requirement on X that first of all, agrees with one's system-theoretic sense of compactness and secondly, reduces to the minimal dimensionality requirement when X is a vector space. Consequently, we call a model Σ "good" if it is both <u>completely reachable</u> and <u>completely observable</u>. Let us examine what these properties involve.

- <u>Reachability</u> - given an internal model Σ, the essence of the reachability question is to determine all those states $x \varepsilon X$ reachable from the initial state x_o in some time T (possibly infinite) using admissible input functions $u \varepsilon \Omega$. If all states $x \varepsilon X$ are reachable, then we call Σ <u>completely reachable</u>. It is clear that the reachability of a given state x depends upon several factors: the initial state x_o, the allowed time T, the admissible inputs Ω and, finally, the dynamics ϕ. We shall examine the computational aspects of this question below.

- Observability - in contrast to reachability, which deals with what is possible using inputs from Ω, the problem of observability centers about what can be known about the system Σ from observation of its output $y \in \Gamma$. More particularly, we ask if knowledge of $y(t)$ over some time horizon $0 \leq t \leq T \leq \infty$ is sufficient to determine uniquely the initial state x_o. Clearly, the answer to this question is bound up in the interplay between the system dynamics ϕ, the output function h and the inputs $u \in \Omega$, as well as the time horizon T. A state $x \in X$ that is identifiable is called observable, and if all $x_o \in X$ are observable, then we say that Σ is completely observable.

  Remark: From the foregoing considerations, it is easy to see why complete reachability and complete observability are natural requirements to impose upon any internal model Σ purporting to be a "good" representation for experimental data (Ω, Γ, f): a state that is unreachable cannot arise from the application of any input $u \in \Omega$ and, consequently, is irrelevant to the characterization of the data. Similarly, if two distinct states x_o, x_o' give rise to the same observed output, they are indistinguishable as far as the external behavior is concerned and can be treated as the same state.

In addition to the above-stated problems of realization, reachability and observability, two other broad problem classes comprise the full spectra of general topics of concern to system and control theorists. These are problems of stability and optimality.

- Stability - problems of stability come in two conceptually different forms, depending upon whether one is interested in the stability properties of a single system or the stability of a family of systems. The first class of problems come under the general heading of classical Lyapunov stability, while the second class forms the basis for what is termed structural stability.

The fundamental question addressed in Lyapunov theory is the following: if the origin is an equilibrium for the system Σ (i.e., $\phi(0,0) = 0$) and $x_o \neq 0$, will the state $x(t) \to 0$ as $t \to \infty$? This is termed the problem of asymptotic stability of Σ (in the sense of Lyapunov). A related question is: given $\varepsilon > 0$, if $||x_o|| \leq \varepsilon$, does there exist a $\delta(\varepsilon) > 0$ such that $||x(t)|| \leq \delta$ for all $t > 0$? This is the problem of stability (in the sense of Lyapunov).

For general nonlinear ϕ, it is usually very difficult to answer these stability questions globally (i.e., for arbitrary x_o); however, if we restrict x_o to a sufficiently small neighborhood of the origin, then the linear approximation of Σ can be used to address the Lyapunov problems. If ϕ is differentiable and we let

$$F = \frac{\partial \phi}{\partial x}(0,0) \quad ,$$

then the stability and asymptotic stability of the origin is determined by the location of the characteristic values of F relative to the imaginary axis.

The preceding considerations have all been based upon the so-called uncontrolled, or free, motion of the system Σ, i.e., with $u \equiv 0$. One of the central questions in control theory is to what degree the stability characteristics of the system can be altered by suitably chosen feedback control $u = u(x)$. In other words, if we use the control law $u(x)$, the new system dynamics

$$\dot{x} = \phi(x,u(x)) \doteq \Psi(x) \quad ,$$

and we are interested in what manner the stability properties of $\Psi(x)$ can differ from those of $\phi(x,0)$. In the case of autonomous linear systems ($\phi(x,u) = Fx + Gu$), if we use linear feedback $u(x) = -Kx$, we have $\Psi(x) = (F-GK)x$ and the degree to which the characteristic roots of F can be "shifted" by the feedback law u is termed the Pole-Shifting problem. Its solution is intimately tied-up with the reachability properties of the pair (F,G). The references [1, 2] give a good account of the status of this key question.

When we move from the study of the stability properties of a single system Σ to a family of such systems, the basic questions shift from the behavior of a single trajectory to the collective behavior of a family of trajectories under a perturbation from one member of the system family to another. The simplest such situation is when the dynamics $\phi(x,u)$ contains a parameter a, i.e., $\phi = \phi_a(x,u)$. For simplicity, let us consider only the free motion of the system ($u \equiv 0$). Each fixed value of the parameter a generates a trajectory $x_a(t)$ and we are interested in the qualitative behavior of the family of trajectories $\{x_a(t)\}$ as we vary a. In particular, we are concerned with whether there exist parameter values a* for which the topological nature of the trajectory $x_{a*}(t)$ is different from that of $x_a(t)$ for all a in a neighborhood of a*. Such a value a* is called a <u>bifurcation point</u> of the family. The simplest example of this type is the damped harmonic oscillator

$$\Sigma_a : \ddot{x} + a\dot{x} + x = 0 \quad , \quad a \text{ real.}$$

Here, the phase plane trajectories are as in Figure 1.

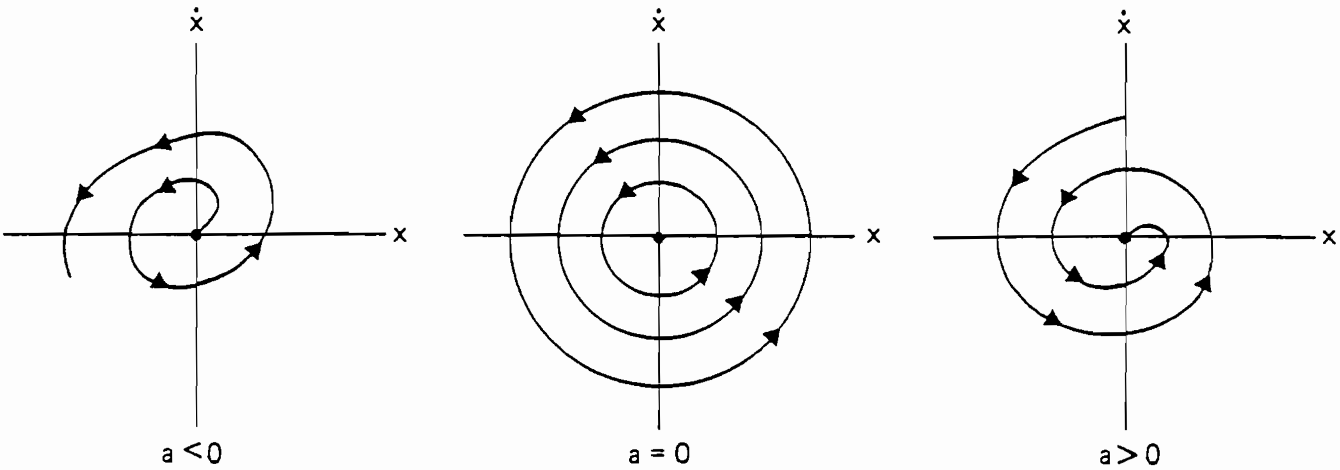

Figure 1. Phase Plane Trajectories for Damped Harmonic Oscillator.

For this system, the trajectories shift from positive to negative spirals as a passes through the bifurcation point $a^* = 0$. Thus, the family Σ_a is not structurally stable for any perturbation of a that includes $a = 0$, although it is a structurally stable family for $a > 0$ or $a < 0$.

The problems of structural stability of uncontrolled systems have been extensively studied in recent years and a good account of developments is given in the book [3]. The analogous questions for controlled systems ($u \neq 0$) have been little examined as of the date of writing (early 1985).

- Optimality - for historical reasons, the most well-studied problems in system and control theory involve superimposing a scalar criterion J upon the dynamics Σ, and seeking a control law u that minimizes J. The most common form for J is an integral, in which case we seek a control u that minimizes

$$J = \int_0^T g(x,u)\,dt \quad ,$$

subject to the dynamics

$$\dot{x} = \phi(x,u) \quad , \qquad x(0) = x_o \quad .$$

As is well-known [4, 5], there are two quite distinct approaches to the solutions of the above optimal control problem:

I) Maximum Principle approach - we form the Hamiltonian of the control problem

$$H(x,u,\lambda) = g(x,u) + \lambda(t)\phi(x,u)$$

and seek a control u to miminize H. This procedure leads to the solution of the two-point boundary-value problem

$$\dot{x} = \frac{\partial H}{\partial \lambda} = \phi(x,u) \quad , \qquad x(0) = x_o$$

$$-\dot{\lambda} = \frac{\partial H}{\partial x} = (\frac{\partial g}{\partial x}) + \lambda (\frac{\partial \phi}{\partial x}) \quad , \qquad \lambda(T) = 0 \quad .$$

In principle, the solution of this problem determines x and λ as functions of u and these functions are then used to reduce H to a function of u alone. The function that then minimizes H is the optimal control for the problem. Details of this procedure can be found, for example, in [5, 6].

II. Dynamic Programming approach - we introduce the optimal value function

$$I(x_o,T) = \min_u J \quad .$$

Employing Bellman's Principle of Optimality [4, 7], it can be shown that I satisfies the partial differential equation

$$\frac{\partial J}{\partial T} = \min_v [g(x_o,v) + \phi(x_o,v) \frac{\partial I}{\partial x_o}] \quad , \quad T > 0 \quad ,$$

$$I(x_o,0) = 0$$

Solution of this initial-value problem produces both the function $I(x_o,T)$, and the optimal control function

$$u^*(x_o,T) = \arg \min_v [g(x_o,v) + \phi(x_o,v)) \frac{\partial I}{\partial x_o}] \quad .$$

Note that u* is a feedback policy, giving the optimal control as a function of the current state x_o and the time to go T, while the control law determined via the Maximum Principle is an open-loop control, giving the optimal input only as a function of the current time t. From a computational point of view, there are pluses and minuses associated with each approach, as we shall discuss in more detail below.

3. Control and Computing

Before embarking upon a consideration of the impact upon control and system science of current and projected trends in computer science, let us briefly examine the principal types of computing requirements necessary to address the questions posed in the preceding section. As with most areas of science, the computational requirements are both numeric and non-numeric.

- Numerical Computing - almost all questions involving reachability, observability, realization and stability ultimately reduce to numerical problems of linear algebra: determination of the linear dependence or independence of a collection of vectors, calculation of the rank of a certain matrix, ascertaining the locations of the characteristic roots of a matrix and so forth.

  For example, for the linear system

  $$\dot{x} = Fx + Gu \quad , \quad x(0) = 0 \quad , \quad x \in R^n \quad , \quad u \in R^m \quad ,$$

  the reachable states from the origin using bounded, measurable inputs can be shown to be characterized by the range of the n x nm matrix

  $$\mathcal{C} = [G|FG|F^2G|\ldots|F^{n-1}G] \quad .$$

  With some extension and qualification, basically the same result can be obtained locally for nonlinear processes [8]. In addition, through duality exactly the same sort of results apply for problems of observability. On the other hand, if we are interested in asymptotic stability of the origin for the above system, then we must consider the location in the complex plane of the characteristic roots of the matrix F. If Re $\lambda_i(F) < 0$, i=1,2,...n, then the origin is stable; otherwise it is not.

  Computational problems of optimal control involve a somewhat different set of numerical requirements. As we have seen above, the Maximum Principle approach involves the solution of a two-point boundary-value problem for the system state x and co-state λ, followed by determination of the minimum of the Hamiltonian $H(x,u,\lambda)$. Numerically, this implies various finite-difference schemes for integrating the equations for x and λ, together with appropriate methods, such as gradient schemes, for unconstrained (or possibly constrained) optimization. The dynamic programming approach also involves a combination of a numerical integration procedure for the optimal value function $I(x_o,T)$, coupled with an optimization procedure at each step for the optimal policy function $v(x_o,T)$.

In passing, it is worthwhile to note that memory requirements for the two procedures differ considerably, growing linearly in n for the Maximum Principle approach, geometrically in n for dynamic programming. This fact is the Achilles heel for the dynamic programming approach, which in almost every other respect is preferable to the Maximum Principle method. It is the alleviation of this "memory gap" that some of the recent and projected developments in computer hardware and software may turn out to have their greatest impact in the control area.

- <u>Non-Numerical Computing</u> - a considerable number of important system and control problems involve computing in symbols, rather than numbers. For instance, the local reachability properties of the nonlinear system

$$x = \phi(x,u) \quad , \qquad x(0) = x_o \quad ,$$

$$y = h(x) \quad ,$$

involve determination of the Lie algebra of vector fields $\{f^i(x_o)\}$, where $f^i(x_o) \doteq \phi(x_o,u^i)$, u^i = constant input, $i=1,2,\ldots,M$. This calculation requires that we compute the Lie bracket of two vector fields

$$[f^i,f^j](x_o) = \frac{\partial f^j}{\partial x}(x_o)f^i(x_o) - \frac{\partial f^i}{\partial x}(x_o)f^j(x_o) \quad ,$$

an operation involving symbolic computation of the Jacobian $\frac{\partial f^i}{\partial x}$. Furthermore, to determine the relevant Lie algebra we must test the linear independence of a set of such brackets, together with the $\{f^i(x_o)\}$. Similar remarks apply to studying the observability properties of the system.

In a similar vein, for linear system problems ($\phi(x,u) = Fx+Gu$, $h(x) = Hx$), most of the important system properties can be determined in terms the rational transfer matrix

$$W(\lambda) = H(\lambda I-F)^{-1}G \quad .$$

Letting $\chi_F(\lambda)$ denote the characteristic polynomial of F, we can write

$$W(\lambda) = P(\lambda)/\chi_F(\lambda) \quad ,$$

where $P(\lambda)$ is a polynomial matrix. Thus, study of W is often reduced to the study of the entries of P, i.e., we need symbolic computational routines designed specifically to operate upon polynomials.

In connection with problems in structural stability, one of the most effective means for studying the structural properties of a vector field is to reduce the field to its so-called "normal form". Basically this involves a symbolic nonlinear coordinate change from the original basis into a coordinate frame that makes the topological properties of the vector field transparent. Details of this procedure can be found in [9]; the point we make here is the need for good symbolic computing languages to execute these coordinate changes (and their inverses). We note, as an aside, that the same ideas (and programs) can also be used in catastrophe theory applications.

4. Computer Science in the '90s - What's in it for systems and control?

Just as all Gaul is divided into three parts, computer science is divided into two — hardware and software, with several subdivisions of each. To gain some indication as to how current and projected hardware and software developments will impact systems and control over the coming decade, we focus upon the following trees in the computer sciences forest:

- computer architecture
- algorithms and data structures
- operating systems
- programming languages
- complexity theory
- human interfaces

Let us examine each of these topics in turn.

A. Parallel Architectures - most likely, the only hardware development that will significantly influence the control sciences over the coming decade is the ever-increasing trend toward non-serial types of information processing. Ranging from totally asynchronous machines through tightly coupled systems of a few high-performance processors like the Cray X-MP to lock-step vector processors, the basic research challenge in parallel processing involves finding algorithms, programming languages and parallel architectures that, when used as a system, yield a large amount of work processed in parallel at the cost of a minimum number of additional instructions. The following formula represents the speedup with p processors over that achieved with a single processor [10]:

$$S(p,\alpha,\sigma) = \frac{1}{(1-\alpha) + \alpha/p + \sigma(p)},$$

where

α = the fraction of the work in the application that can be done in parallel,

σ = excess work required in the instructions due to parallelism, i.e., instructions dealing with synchronization and communication between the processors,

p = number of processors

Assuming $\sigma = 0$, Figure 2 shows the speedup as a function of parallelism.

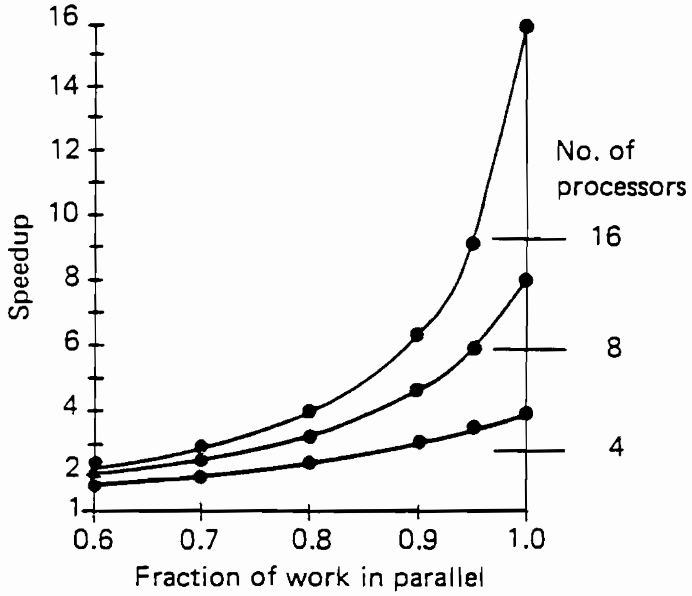

Figure 2. Speedup as a Function of Parallelism and Number of Processors [10].

As noted in [10], vector processors may be the least promising hardware development since to achieve maximum performance requires vectorizing at least 90% of the operations, but experience has shown that only about 50% of a typical problem can be vectorized. Nevertheless, in many problems of control involving purely linear-algebraic computations, the magic 90% figure will be attainable and vector processor would then play a significant role.

As noted in the preceding section, a very large number of the numerical computations for systems and control processes involve determination of the number of linearly independent vectors from a given set. It is hard to imagine a computing application that lends itself to parallelism in a more direct fashion than this. Consequently, in our above speedup formula, it seems reasonable to assume that $\alpha = 1$ for such control calculations. The mystery factor is σ, the computational overhead associated with parallelism. Since σ depends not only upon p, but also upon the particular algorithm and the specific architecture in use, no fixed value can be attached to σ without this information. But for a synchonous parallel machine or for a vector processor, $\sigma = 0$ indicating that at least for control problems, such a machine may be a better bet than the grander (and costlier) totally asynchronous, multiple-instruction stream "supercomputer".

In closing this topic, we note that some specific algorithms making use of such synchronous vector processors for dynamic programming calculations were reported almost 15 years ago in [11], considerably in advance of widespread availability of the hardware to realize them. It is our conjecture that these algorithms and their relevant refinements will play an increasingly significant role in control computations in the coming decade.

B. Algorithms and Data Structures - in addition to new algorithms dictated by parallel architectures, new serial algorithms and data structures will greatly influence control and system computations in the future. Of particular interest is the overall question of optimal algorithms, either in regard to computing time or storage requirements. As an illustration, in the solution of the optimal control problem

$$\min \int_0^T [(x,Qx) + (u,Ru)]dt \quad , \qquad Q \geq 0 \ , \qquad R > 0 \quad ,$$

$$\frac{dx}{dt} = Fx + Gu \quad , \qquad F = nxn \quad , \qquad G = nxm \quad ,$$

the standard approaches require numerical integration of the matrix Riccati equation

$$-\frac{dP}{dt} = Q + PF + F'P - PGR^{-1}G'P \quad ,$$

$$P(T) = 0 \quad ,$$

a calcuation involving $O(n^2)$ computations per time step, where n is the dimension of the state vector x. In [1], it was shown that if rank Q = p, then an alternate algorithm could be provided that involved only O(n(p+m)) computations per step, a substantial improvement if $p + m << n$, a situation that is often the case in application. This is just a single example of the general question of optimality of algorithms, a topic treated in more detail in [12].

In a somewhat different direction, we can ask about the "typical" behavior of a given algorithm on a class of control problems. It is well-known in linear programming, for example, that the classical simplex method can break down under certain pathological circumstances. Some recent work, surveyed in [13], has been devoted to addressing the question of how pathological is pathological? Empirical evidence indicates that the simplex method will produce the correct solution "almost always", and that the bad cases are indeed rare. Given the close relationship between optimization problems in operations research and analogous problems in control theory, it is reasonable to conjecture that similar studies will be devoted to an analysis of standard computational algorithms for "bread-and-butter" system problems like Kalman filtering, linear-quadratic control and Luenberger observers.

Besides algorithmic analysis, availability of cheap mass storage capacity will focus attention on new ways of storing data for efficient accessibility. One way control theory could benefit from such developments is in the alleviation of the dynamic programming "curse of dimensionality" for certain classes of control processes. If we have the system dynamics written in the control-canonical form

$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= x_3 \\
&\vdots \\
\dot{x}_n &= f(x_1,\ldots,x_n,u) \quad ,
\end{aligned}
$$

then while computing the optimal value function I(x,t) at time t, instead of storing I(x,t+Δ) at all states, we need only to store it on a restricted hypersurface in R^n (see Figure 3).

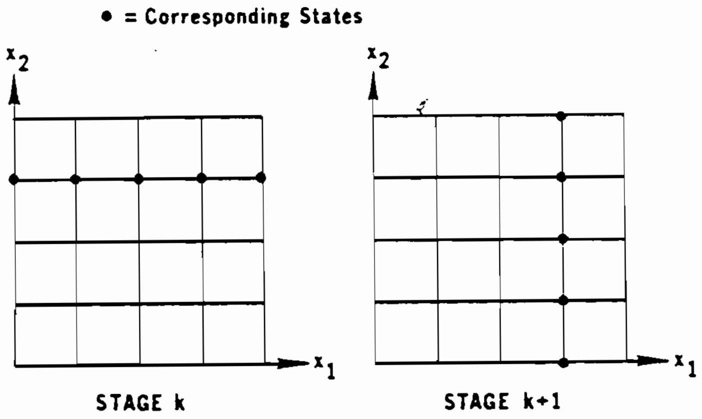

Figure 3. States at time T for which I(x,t+Δ) must be stored to compute I(x,t).

This data structure scheme for control processes is called the "shift-vector" method and is described in greater detail in [14]. It is reasonable to suppose that similar contributions of problem structure and new data storage ideas will combine to greatly reduce computing burdens for large classes of control problems in the near future.

C. Operating Systems - as cheap, powerful computers become ever more widespread, a greater emphasis will be placed on the real-time control of several geographically distinct processes. For example, automotive firms are currently exploring the possibility of computer control via satellites for large fleets of automobiles. Such "distributed" control processes involve both hierarchies of controllers, as well as the synchronization of many different processors utilizing different data sets. Applications of this sort generate the need for new types of computer operating systems specifically tailored to allow multiple resources to be efficiently coordinated in the execution of control programs.

Of special interest for control applications are operating systems for networks of interconnected computers. The basic question is how to design a system that efficiently checks for errors in the network and corrects them. One of the most novel approaches to this question has been the "order from disorder" principle invoked by M. Rabin and his co-workers and described in [15]. Their idea is to employ probabalistic algorithms that insure the operating system will properly coordinate the nodes in the network almost all the time. So, rather than demand 100% perfect operation, they trade a small possibility of error for a vastly increased overall efficiency of operation in the network. The probabalistic approach has proven to be particularly effective in the so-called "Byzantine Generals" problem, which models the situation in which several processors are faulty and give conflicting information to the supervisory program. Which information should be trusted and how should decisions be made? Rabin's algorithm solves the problem by trading a perfect notion of what all processors know for a small measure of uncertainty and a great deal of simplification. It is tempting to speculate that operating systems constructed along probabalistic lines will result of necessity as networks emerge that are orders of magnitude larger than those currently in use.

Another very important development in operating systems that will dramatically affect control calculations is in the area of memory management, including optimal swapping policies for virtual memory, file access methods and off-line storage optimization. We have already seen that such memory management schemes may spell the difference between success and failure in dynamic programming calculations, and this holds especially true in areas of real-time process control, as well.

Finally, we can hope that operating systems will appear enabling users to operate on idealized versions of resources without concern for physical detail. For instance, processes instead of processors, files instead of disks, data streams instead of program I/O. Such operating systems would greatly streamline many of the control and system calculations described earlier, especially those involving the generation of an internal model from behavioral data (the realization problem).

D. Programming Languages and Control Software - from a systems engineering and design point of view, there is a great need for languages that efficiently express algorithms and data in a form compatible with the customary abstract or mathematical formulations of the problem. Procedure-oriented programming languages like FORTRAN, functional languages like LISP and object-manipulation languages such as Smalltalk, all have significant drawbacks for control applications. A language that deals directly with the basic concepts of control and system design such as frequency response, gain, feedback sensitivity and so on would surely greatly streamline most investigations.

In much the same direction, there is an increasing need for special control software packages expressed in one of the standard programming languages. While various packages of this sort exist at research centers around the world, it is not unreasonable to expect that future developments in programming methodology will result in the codification of basic concepts such as data types and control structures, and that this codification will act to uniformize the many packages currently in existence with a major advance in accessibility to the international controls community.

E. The Human Interface - the efficient transfer of information between humans and the machine is one of the neglected backwaters of computer science research. Nevertheless, just as in commercial applications, advances in this area will dramatically increase the utility and effectiveness of computers in control and system studies. My personal bet is that the most significant impact will be in the area of advanced graphics, enabling a system controller or designer to display optimal value surfaces, alternate stochastic realizations of trajectories, intersections of controllability subspaces and the like with an efficiency and detail heretofore impossible to provide. In addition to such static "snapshots" of the system, future graphics capability will admit the possibility of generating "full-color" moving pictures of the system's design behavior in the face of a variety of alternative environments.

In addition to vastly enhanced graphics capability, it seems a safe bet to expect that new interactive methods for computer-aided design and advanced forms of input and output such as optical readers, voice input, touch-sensitive pads and light pens will also contribute to an increased flexibility for future designers and system controllers to interact with the machine.

F. Complexity Theory - as a final point of contact between future computer science and control, we note the theoretical work currently underway in complexity theory [16]. While most of the work carried out thus far has been focused on algorithms for classical mathematical operations such as linear equation solving, numerical integration and so forth, there is every reason to believe the same questions can (and will) be profitably studied in connection with control processes. For example, work devoted toward identifying the time and space requirements of a given problem, and the relationship between the problem's size (perhaps as measured by the number of states, inputs and outputs) and the best or worst case performance of algorithms to solve the problem would be of great value.

Very closely related to the time/space question is the classification of control problems into complexity classes. It is of great theoretical and practical interest to determine those problems that are solvable deterministically in polynomially bounded time (P-problems) and those that solvable non-deterministically in polynomial time (NP-problems). Most of these questions seem relatively straightforward for linear problems, but such a classification for nonlinear systems seems very far away, at present.

Finally, we could list the following general theoretical questions for the role of the computer in control and system calculations:

i) What problems can machines solve?

ii) What are optimal algorithms for given classes of control problems?

iii) What is the intrinsic best-and-worst-case performance of given classes of machines for given classes of problems?

iv) What control problems are equivalent to each other in computational difficulty?

5. New Applications for the 90s

The analytic and computational problems of engineering system and control theory were generated principally by the applications of the 1950s and 60s, most significantly in the areas of navigation, aerospace and chemical process control. It would be remiss in a paper of this sort, devoted to a look into the 90s, not to engaged in a bit of not-so-speculative speculation about new applications that will provide the impetus for the theoretical and applied problems of the future. In this spirit, we briefly examine three engineering areas in which control system thinking is only now coming forward as a major component of systems design: speech synthesis, automobile systems control and satellite nagivation for boats and cars.

- Speech Synthesis - enhanced human interaction with computers has generated an immense need to be able to communicate directly through natural languages in a spoken, rather than written, mode. One of the most powerful means for synthesizing such spoken output is the so-called linear predictive coding method in which speech is modeled as a stationary autoregressive discrete-time random process

$$y_t + A_{N,1}\, y_{t-1} + \ldots + A_{N,N}\, y_{t-N} = e_{N,t} \quad ,$$

where $\{y_t\}$ is the observed speech signal and $\{e_{N,t}\}$ is a zero-mean white noise process representing the air issuing from the lungs which is then modulated by the vocal system to produce the speech waveform. The problem here is to choose the order N, the coefficients $\{A_{N,i}\}$ and the noise variance R_N so as to best fit the observed speech signal $\{y_t = t \geq 0\}$.

The standard solution to the problem involves forming the covariance estimate

$$R_k = E[y_t y_{t+k}]$$

and to solve the system

$$[A_{N,N} \cdots A_{N,1} \; I] \begin{bmatrix} R_0 R_1 & \cdots & R_N \\ R_1 \; R_0 & \cdots & \\ \vdots & & \\ R_N R_{N-1} & \cdots & R_0 \end{bmatrix} = [0 \; \ldots \; 0I]$$

for the unknown weights $\{A_{N,i}\}$. Since the coefficient matrix of this system is Toeplitz, the Krein-Levinson algorithm produces a solution in $O(N^2)$ operations. However, recent work by Kailath and his associates [17] has shown that if a parallel computer of the synchronous type is available with, say, N processors, then a modification of an old algorithm due to Schur can result in a solution of the above system in O(N) operations. This figure should be constrasted with the O(N log N) operations needed with a parallel implementation of the Krein-Levinson procedure. Thus, the Schur algorithm, together with widespread availability of VLSI technology and special purpose parallel computation, opens up the very real possibility of high-fidelity speech synthesis in the coming decade.

- Automobile Control Systems - in order to meet increasingly stringent fuel consumption and exhaust emissions standards, almost all of the world's automobile manufacturers have turned to widespread use of on-line computers to monitor and regulate most of the performance of their engines. This computer takeover of the automobile motor is being carried out in the face of very restrictive hardware constraints involving low costs, high reliability, and response over many time-scales ranging from 1 millisecond to 100 seconds.

The typical control variables are air-fuel ratios, spark timing and level of exhaust gas recirculation. The question for control theorists and software designers is to find "fast" algorithms using readily measurable information that carry out these control actions in real-time. Current algorithms and procedures do a marginally satisfactory job, but major advances in this area remain and the payoff for even partial success will be measured in the millions of dollars.

In passing, let us note that many of the same remarks as for motors apply to other automotive systems such as transmission, suspension and braking. The anti-blocking braking system (ABS) and the just now emerging computer-controlled suspension systems are only the tip of the rapidly surfacing iceberg of such computer regulation of the overall automobile system.

- Satellite Navigation - for a number of years now it has been possible for boat and shipowners to find their position at sea through satellite communication. For private use, units priced around $2000 can provide accuracy to within 10-15 meters, while more expensive commercial units can reduce the margin of error to a few meters. This is all quite satisfactory for boats but far too inaccurate for automobiles. Systems with an accuracy of less than 1 meter are currently undergoing experimental test in automobiles in conjuction with an on-board computer mapping system that enables the car to be put on "automatic pilot" even in urban environments.

  It is clear that any widespread use of such satellite navigation systems for cars will create an enormous control and coordination problem, one that can only be dealt with by major advances in the kind of network operating systems that we spoke of in the last section. This type of application is an area in which hardware and software failures may very well be fatal; consequently, whatever new algorithms and procedures that emerge from these systems, the emphasis must be upon both speed and reliability, again strongly suggesting a high degree of parallelism.

REFERENCES

1. Casti, J., Dynamical Systems and their Applications: Linear Theory, Academic Press, New York, 1977.

2. Kalman, R., P. Falb and M. Arbib, Topics in Mathematical System Theory, McGraw-Hill, New York, 1969.

3. Arnol'd, V.I., Geometrical Methods in the Theory of Ordinary Differential Equations, Springer, New York, 1983.

4. Bellman, R., Dynamic Programming, Princeton University Press, Princeton, 1957.

5. Pontryagin, L. et al., The Mathematical Theory of Optimal Processes, Interscience, New York, 1961.

6. Gabasov, R. and F. Kirillova, The Qualitative Theory of Optimal Processes, Dekker, New York, 1976.

7. Larson, R. and J. Casti, Principles of Dynamic Programming - Part I, Dekker, New York, 1978.

8. Casti, J., Nonlinear System Theory, Academic Press, New York, 1985.

9. Guckenheimer, J. and P. Holmes, Nonlinear Oscillations, Dynamical Systems and Bifurcations of Vector Fields, New York, 1983.

10. Buzbee, B.L. and D.M. Sharp, "Perspectives on Supercomputing", Science, 227 (1985), 591-597.

11. Casti, J., M. Richardson and R. Larson, "Dynamic Programming and Parallel Computers", J. Optim. Theory & Applic., 12 (1973), 423-438.

12. Taub, J. and H. Wozniakowski, A General Theory of Optimal Algorithms, Academic Press, New York, 1980.

13. Garey, M. and D. Johnson, Computers and Intractability, Freeman, San Fransisco, 1979.

14. Larson, R. and J. Casti, Principles of Dynamic Programming - Part II, Dekker, New York, 1982.

15. "Order Out of Chaos in Computers", Science, March 2, 1984, 917-919.

16. Taub, J., A. Wasilkowski and H. Wozniakowski, Information, Uncertainty, Complexity, Addision-Wesley, Reading, Mass., 1983.

17. Kailath, T. "Signal Processing in the VLSI Era", in Modern Signal Processing and VLSI, S. Kung, H. Whitehouse and T. Kailath, eds., Prentice-Hall, Englewood Cliffs, NJ, 1984.