CHAPTER 1. INTRODUCTION

1.1 Introduction

Computers are daily finding wider and wider use in the control of physical processes -- from automobile engines to automobile factories. Often the computer merely embodies in inexpensive silicon a "classic" control methodology, such as linear feedback or "ladder logic." But as more complex machines and processes are subjected to computer control, these conventional "algorithmic" techniques become intractable -- or may be totally unable to manage the control problem[1]. Thus there is increasing interest in adopting the "heuristic" techniques of artificial intelligence for process control. Of particular interest are the inference-driven techniques -- such as expert systems and fuzzy logic[2] -- because of their more predictable behavior.

Unfortunately, artificial intelligence was not invented in a process control environment. The state of the art in inference engines is a poor fit to the state of the art in process control.

Commercial expert systems (e.g. Nexpert, Kappa PC) typically run on large PCs or workstations.[3] Moreover, they typically run on a *single* PC or workstation, assuming that all "reasoning" shall be performed by one central unit. Also implicit in their design is the assumption of timelessness -- that all data shall be presented in an instant, and a conclusion drawn from (and for) that snapshot.

Process control, on the other hand, is increasingly the province of small embedded microprocessors. Large processes (such as an automobile assembly plant) no longer are automated with a single central mainframe computer; instead, a number of small computers are distributed throughout the plant and connected via a local-area network (Jones, 1988). And only the simplest of process control

---

1. (Zhang and Grant, 1990) cite the inverted pendulum as an example of a problem with a simple heuristic solution, but a forbiddingly complex analytical solution.
2. Throughout this paper, fuzzy logic systems shall be considered a subset of expert systems.
3. Some simple fuzzy logic engines *have* been developed for microcontrollers, e.g. (Motorola, 1992; Sibigtroth, 1991).

systems can function without some knowledge of time and change, even if only the first derivative of a feedback signal.

A process control expert system should reside on the distributed hardware used for such applications. For fastest response, decisions should be made as near to the point of control as is feasible, that is, in the "local" controllers. Centralizing all decisions limits the inferencing power to that which can be achieved by a single computer. Thus it is preferable to distribute the complex reasoning tasks over the network of computers. These systems commonly employ small microprocessors and local-area networks, so the expert system can expect only limited resources, and should use network communications to share information. Finally, the process control expert should be "aware" of time, and allow temporal relationships (such as the change of a data value) to be stated in its logical language.

1.2 High Performance Inferencing

Expert systems were originally developed on mainframe computers in languages such as LISP, a legacy which lingers. Most modern systems still require the resources of a large computer. Many are still written in LISP or similar languages, with their requirement for large "heap" memory and frequent garbage collection. This results in nondeterministic performance, which is unacceptable in a real-time control system.

Some systems (e.g. CLIPS) attempt to circumvent the problem by compiling the expert system to an efficient intermediate language, such as C. Fuzzy logic systems for microcontrollers (Sibigtroth, 1991) may be written entirely in assembly language. In either case, interactive and incremental development -- a major attraction of LISP -- is lost.

1.3 Distributed Inferencing

Contemporary expert systems typically assume either a single processor, or a small number of processors with a shared, central, "blackboard" memory where information can be posted and read. This "logical star" topology, illustrated by Figure 1-1a, is reminiscent of the multicomputer architectures of the 1960s (Hwang and Briggs 1984).

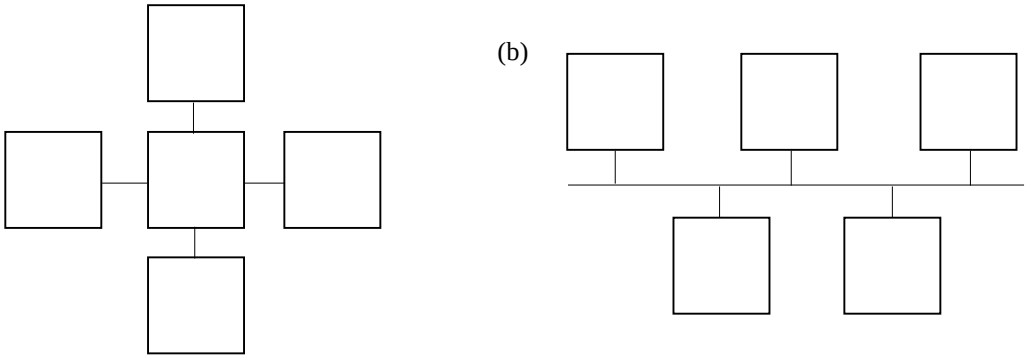

Figure 1-1. Multiprocessor Topologies. (a) star, (b) bus.

The trend in multicomputer systems is toward distributed processing. Even embedded control systems are increasingly using networks of small processors, rather than a cluster of large computers. This "logical bus" topology of a network, illustrated in Figure 1-1b, is a more suitable model for a control-oriented expert system for two reasons. First, it is a closer match to the physical networks commonly used in process control systems. More significantly, a bus topology avoids the processing limitations imposed by a central "bottleneck."

1.4 Temporal Logic

Unlike many subjects to which expert systems have been applied, process control is not a static problem. *Time* is frequently a consideration when making a control decision. Examples include:

- ensuring that events occur in the proper sequence
- monitoring inputs for a change in status
- keeping input data "current"
- computing rate of change of control variables
- recognizing and analyzing trends
- maintaining a history of input or output values

In short, information must be treated as having a temporal aspect. Data may have a time value, and a history log may be needed for some variables. Furthermore, the logical language of the knowledge base must be able to clearly express temporal relationships.

1.5 Control Requirements of the Tandem Accelerator

An example of a process control problem exhibiting all these requirements is the McMaster University Tandem particle accelerator. Operation of particle accelerators is largely heuristic (Poehlman, Garland, and Stark, 1991), with many temporal aspects. This demands a high-performance inference engine, running on a set of embedded control processors.

The High Voltage Engineering Corporation model FN Tandem Accelerator, as installed at McMaster University, is shown schematically in Figure 1-2. The purpose of this device is to accelerate

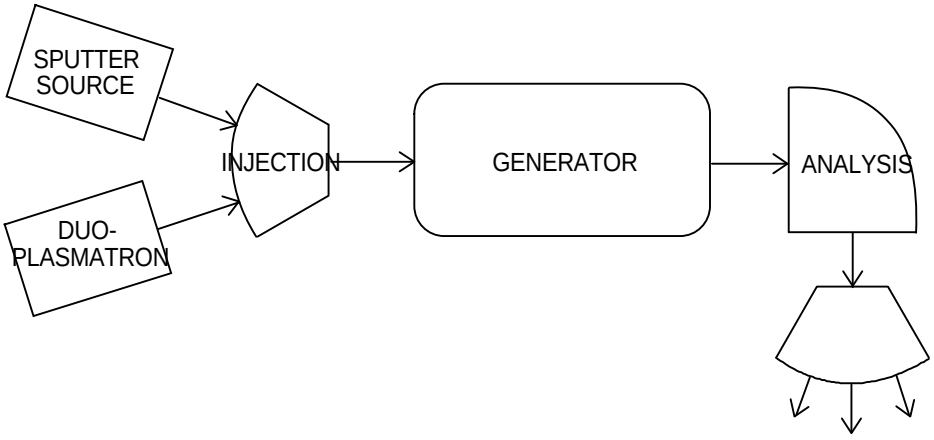

Figure 1-2. The Model FN Tandem Accelerator

various atomic nuclei to energies of several tens of MeV. Either of two ion sources (Duoplasmatron or Sputter Source) produces a beam of negative ions of the desired element. This beam is injected into the Generator tank. At the center of the tank is the high voltage terminal, which is charged positively to a potential of several megavolts by two Van de Graaff belts[4]. The negative ions accelerate towards and enter the positive terminal. Inside the terminal, the ions are stripped of some electrons as they pass through a thin foil or gas. The now-positive ions accelerate away from the terminal and out of the tank. The analyzing magnet bends the path of the ions having the desired energy 90 degrees; ions of lower or

4. In the McMaster University accelerator, chains are used rather than belts. The operating principle is the same.

higher energy are bent more or less and are blocked by a slit. This "tandem" arrangement doubles the effective acceleration, and allows both the ion sources and the beam targets to be at a low potential.

Heuristic control

In theory, the physics of this device are susceptible to an exact analytical solution. In practice, dimensions are never exact, alignment is never perfect, and currents and voltages are never precisely known. Many stray influences can perturb the particle beam. Thus the accelerator poses an intractable control problem. While some subsystems of the Tandem Accelerator have been automated (see Chapter 2), overall control of the accelerator is almost completely heuristic, and is usually performed by a human operator.

Temporal

Time is a factor in most aspects of accelerator control. For instance, it is not permissible to allow the generator terminal to charge too quickly; this requires knowledge of short-term rate of change. The stripper foil in the generator can wear out and require replacement; this is deduced from long-term trends in the measured beam current.

Embedded Multicomputer

Even a small accelerator such as the FN has hundreds of inputs and outputs.[5] It is expensive and impractical to wire all of these to a single computer, and the input/output processing alone would heavily load that computer. It is preferable to divide this burden among several processors. Ideally, these processors and their I/O wiring would be located near the accelerator subsystems, concentrating the data and reducing the number of cables to the control room. These considerations, and the hostile environment[6], suggest the use of embedded microcontrollers communicating via a local-area network.

1.6 Research Objectives

The task of the distributed expert system shall be to control a major subsystem of the FN

5. Just from Injection through Analysis, there are over 60 measurable quantities.

6. Among other problems: lack of air conditioning, unregulated power, electrical noise created by machinery, and huge electrical transients caused by megavolt sparks in the generator tank.

accelerator. As a test problem in process control, the accelerator is advantageous because most response times are measured in seconds, rather than milliseconds. Also, few "small" accelerators have ever been automated; still fewer with inference-driven control software (Lind, Poehlman, and Stark 1993).

This research project will make four contributions to the field:

a. High performance knowledge processing on simple CPUs. An expert system shell and inference engine, suitable for use on small microcontrollers, shall be developed. This software must use only the few kilobytes of memory available on these controllers. Heap storage, which is extravagant in memory use and which introduces nondeterministic delays, should be avoided. Speeds of hundreds of Inferences per Second, using a typical single-chip microcomputer, are desired.

b. Distributed reasoning. A method shall be devised to allow a set of microprocessors, each with an independent inference engine, to cooperate on the solution of a problem. This method shall be suitable for microprocessors communicating via a local-area network.

c. Integration of time into knowledge-based systems. The expert system shall be able to draw conclusions from time-constrained data. It shall also be able to represent, in its knowledge base, logical statements about temporal conditions.

d. Real-time control of a dynamic physical system. To demonstrate the efficacy of the distributed temporal inference engine, it shall operate a real-time process that has never before been automated: the FN Tandem Accelerator.