CHAPTER 7. SUMMARY AND CONCLUSIONS

7.1 Summary

There are many real-world process control problems for which heuristic, or rule-based, control is preferable. An example of such a problem is a Tandem particle accelerator, which is mathematically intractable, and typically is under the control of an experienced human operator. The cost and difficulty of training operators has spurred efforts to capture and automate this expert knowledge.

Conventional expert systems, alas, are ill-suited to process control. Most expert systems are large and slow, requiring fast processors, large memories, and frequent delays for garbage collection. Few have a knowledge of time; most are limited to reasoning about the present instant. In general, the only way to solve a larger problem is to buy a larger computer. Those systems which do allow a problem to be subdivided across multiple computers, retain a central "blackboard" machine as a bottleneck.

This research has developed a novel expert system for process control, which makes four original contributions to the field:

a) The expert system achieves extremely high inferencing performance on "low end" processors, by compiling backward and forward inference networks to procedural code. The inference engine and knowledge base require less than 6 Kbytes of program storage, and no dynamically-allocated data storage. Speeds in excess of 300 Logical Inferences Per Second have been achieved on a 68HC16 microcontroller, and over 3000 LIPS on a 33 MHz 486SX computer.

b) The system uses a unique model to perform cooperative reasoning across a distributed network of inferencing agents: the "consultant/advisory" model. A new portable representation of knowledge (rules) has been devised, as well as a portable object-based representation of data (facts).

c) The system can manipulate time-valued data, and automatically handles the expiration and renewal of input data. A conceptual extension incorporates, for the first time, the formal relations of temporal logic, allowing statements to be made about temporal relationships in the knowledge base.

d) The system has successfully operated a process control problem never before solved with inference-driven techniques: controlling the terminal voltage and corona current of an FN Tandem Accelerator. In doing so, it achieves comparable or better speed, better control of slew rate, less overshoot, and better long-term regulation than a human operator.

Three incidental, but no less important, contributions have been made in the course of this research:

e) The expert system allows an unprecedented freedom to mix procedural and inferential programming. Inference routines may freely call procedures and may directly interact with sensors and actuators. Procedural code may access facts and invoke selected rules. One benefit is extremely rapid prototyping of control strategies.

f) A new local-area network, the Asynchronous Token Ring (Appendix B), was developed during this research. This is an open-ended, deterministic network that requires no special hardware beyond that offered by most microcontrollers, and which can be directly used with fiber optic transmission. It is unique in that it can sequence and acknowledge broadcast messages.

g) New hardware has been developed for "technology insertion" into the 1960's-vintage control electronics of the Tandem Accelerator (Appendix A). The new interface techniques include a transparent monitoring of panel meters, and a retrofit to high-voltage power supplies to allow computer control.

7.2 Conclusions

Heuristic control problems are now within the scope of inexpensive, embedded microcontrollers. This new expert system, TEXMEX, provides ample inferencing speed with limited resources. The problem of nondeterministic response time, commonly associated with expert systems, has been overcome. Furthermore, a central inference engine is not necessary; process control problems of greater complexity can easily be distributed over a network of small processors.

Commercial interest has been expressed in TEXMEX, and development will likely continue with the intent of converting this research software into a marketable product.

7.3 Future Work

Unfortunately, the McMaster University Tandem Accelerator was closed down permanently, just after the control experiments described herein. There was no opportunity to automate more than the terminal charging subsystem. It would be highly instructive to bring more of the accelerator under expert control -- both to test the performance of the system with a more complex problem, and to study the more subtle interactions that can occur between the accelerator subsystems.

The implementation of formal temporal logic is still immature. Research is needed to explore different means to achieve the temporal relations described in Chapter 3. These relations must also be tested for utility in a variety of control problems; perhaps a less formal logic will suffice.

Chapter 3 also suggested several other areas for further development of the expert system. For example, automatic rule relocation is likely to be a large and fruitful research subject.[1] "Just in time" compilation, offering the speed of compiled code with the portability of tokens, is another likely focus.

Only a minimal Inferencing Token Language has been implemented, and the design of this language has been driven by the immediate needs of the accelerator control problem. This is inadequate for the design of what should be a general-purpose language. This class of problem has recently become of significant interest,[2] and touches upon both language design and computer architecture (instruction representation).

This work was originally intended to be portable to a variety of embedded microprocessors. At present, only the 68HC16 and 8086 (IBM PC) microprocessors have been used. The system should be

1. One complex aspect of this problem is balancing the often-conflicting demands of processor time and network bandwidth. For example, moving a rule from a heavily-loaded processor to a lightly-loaded processor may be desirable from the standpoint of CPU utilization, but may cause an undesirable increase in network traffic.

2. Primarily, thanks to the advent of the Java language and its tokenized representation.

tested on more platforms, particularly the ubiquitous but resource-limited 8-bit microcontrollers such as the 8051 and 68HC11.

The Asynchronous Token Ring network is still in a primitive state. While only incidental to the main goals of this research, this network revealed itself to be quite an interesting -- and a potentially profitable -- problem. Further development, as outlined in Appendix B, could produce a valuable networking product for embedded processors.