PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT.

The Role of Logic Programming in the 21st Century

Ross Overbeek

Mathematics and Computer Science Division
Argonne National Laboratory, Argonne, Illinois 60439
overbeek@mcs

1 The Changing Role

Logic programming currently plays a relatively minor role in scientific problem-solving. Whether this role will increase in the twenty-first century hinges on one question: When will there be a large and growing number of applications for which the best available software is based on logic programming? It is not enough that there exist a few small, peripheral applications successfully based on logic programming; the virtues of logic programming must be substantial enough to translate into superior applications programs.

1.1 Applications

Applications based on logic programming are starting to emerge in three distinct contexts:

1. Applications based on the expressive power of logic programming in the dialects that support backtracking and the use of constraints. These applications frequently center on small, but highly structured databases and benefit dramatically from the ability to develop prototypes rapidly.

2. Parallel applications in which the expressive power of the software environment is the key issue. These applications arise from issues of real-time control. As soon as the performance adequately supports the necessary response, the simplicity and elegance of the solution become most important. In this context, dialects of committed-choice logic programming have made valuable contributions.

3. Parallel applications in which performance is the dominant issue. In these applications, successful solutions have been developed in which the upper levels of the algorithm are all implemented in a committed-choice dialect, and the lower layers in C or Fortran.

What is striking about these three contexts is that they have not been successfully addressed within a unified framework. Indeed, we are still far from achieving such a framework.

1.2 Unification of Viewpoints

Will a successful unification based on logic programming and parallelism emerge as a dominant technology? Two distinct answers to this question arise:

No, the use of logic programming will expand within the distinct application areas that are emerging. Logic programming will play an expanding role in the area of information processing (based on complex databases), which will see explosive growth in the Unix/C, workstation, mass software, and networking markets. On the other hand, logic programming will play quite a different role in the context of parallel computation. While an integration of the two roles is theoretically achievable, in practice it will not occur.

Yes, a single technology will be adopted that is capable of reducing complexity. Developing such a technology is an extremely difficult task, and it is doubtful that integration could have proceeded substantially faster. Now, however, the fundamental insights required to achieve an integration are beginning to occur. The computational framework of the twenty-first century—a framework dominated by advanced automation, parallel applications, and distributed processing—must be based on a technology that allows simple software solutions.

I do not consider these viewpoints to be essentially contradictory; there is an element of truth in each. It seems clear to me that the development and adoption of an integrated solution must be guided by attempts to solve demanding applications requirements. In the short run, this will mean attempts to build systems upon existing, proven technology. The successful development of a unified computational framework based on logic programming will almost certainly not occur unless there is a short-term effort that develops successful applications for the current computing market. However, the complex automation applications that will characterize the next century simply cannot be adequately addressed from within a computational framework that fails to solve the needs of both distributed computation and knowledge information processing.

The continued development of logic programming will require a serious effort to produce new solutions to sig-

nificant applications—solutions that are better than any existing solutions. The logic programming community has viewed its central goal as the development of an elegant computational paradigm, along with a demonstration that such a paradigm could be realized. Relatively few individuals have taken seriously the task of demonstrating the superiority of the new technology in the context of applications development. It is now time to change this situation. The logic programming community must form relationships with the very best researchers in important application areas, and learn what is required to produce superior software in these areas.

2 My Own Experiences

Let me speak briefly about my own experiences in working on computational problems associated with the analysis of biological genomes. Certainly, the advances in molecular biology are leading to a wonderful opportunity for mankind. In particular, computer scientists can make a significant contribution to our understanding of the fundamental processes that sustain life. Molecular biology has also provided a framework for investigating the utility of technologies like logic programming and parallel processing.

I believe that the first successful integrated databases to support investigations of genomic data will be based on logic programming. The reason is that logic programming offers the ability to do rapid prototyping, to integrate database access with computation, and to handle complex data. Other approaches simply lack the capabilities required to develop successful genomic databases. Current work in Europe, Japan, and America on databases to maintain sequence data, mapping data, and metabolic data all convince me that the best systems will emerge from those groups who base their efforts on logic programming.

Now let me move to a second area—parallel processing. Only a very limited set of applications really requires the use of parallel processing; however, some of these applications are of major importance. As an example, let me cite a project in which I was involved. Our group at Argonne participated in a successful collaboration with Gary Olsen, a biologist at the University of Illinois, and with Hideo Matsuda of Kobe University. We were able to create a tool for inferring phylogenetic trees using a maximum likelihood algorithm, and to produce trees that were 30–40 times more complex than any reported in the literature. This work was done using the Intel Touchstone DELTA System, a massively parallel system containing 540 i860 nodes.

For a number of reasons, our original code was developed in C. We created a successful tool that exhibited the required performance on both uniprocessors and larger parallel systems. We find ourselves limited, however, because of load-balancing problems, which are difficult to address properly in the context of the tools we chose.

We are now rewriting the code using bilingual programming, with the upper levels coded in PCN (a language deriving much of its basic structure from committed-choice logic programming languages) and its lower levels in C. This approach will provide a framework for addressing the parallel processing issues in the most suitable manner, while allowing us to optimize the critical lower-level floating-point computations. This experience seems typical to me, and I believe that future systems will evolve to support this programming paradigm.

3 Summary

Balance between the short-term view and the longer-range issues relating to an adequate integration is necessary to achieve success for logic programming. The need to create an environment to support distributed applications will grow dramatically during the 1990s. Exactly when a solution will emerge is still in doubt; however, it does seem likely that such an environment will become a fundamental technology in the early twenty-first century. Whether logic programming plays a central role will depend critically on efforts in Japan, Europe, and America during this decade. If these efforts are not successful, less elegant solutions will become adopted and entrenched.

This issue represents both a grand challenge and a grand opportunity. Which approach will dominate in the next century has not yet been determined; only the significance of developing the appropriate technology is completely clear.

Acknowledgments

This work was supported in part by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.