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

Towards the General-Purpose Parallel Prosessing System

Kazuo Taki

Institute for New Generation Computer Technology
4-28, Mita 1-chome, Minato-ku, Tokyo 108, JAPAN
taki@icot.or.jp

1 Introduction

Processing power of the recent microprocessors grows very rapidly; It almost gets over the power of mainframe computers. Trends of the continuous improvement of the semi-conductor technology suggest that the processing power of one-chip processor devices will reach 2000 MIPS until the end of 1990s, and that the parallel computer system with 1000 processors will be installed in a cabinet which will realize 2 TIPS (tera instructions per second) peek speed.

Such a gigantic power hardware is no longer hard to imagine because recent large-scale parallel computers for scientific processing, just appeared in the market, suggests a trend of large parallel computers.

However, the software technology on the scientific parallel computers focuses on very limited application domains. Hardware design is also shifted to the applications somewhat. The parallel processing paradigm on those systems is the *data parallelism*. Problem modeling, language specification, compiling technique, a part of OS design, etc. are all based on the *data parallelism*.

The characteristics of the *data-parallel computation* are regular computation on uniform data or synchronous computation in other word. The coverage of this paradigm is limited to non-wide area of application domains, such as dense matrices computation, image processing, and other problems with regular algorithms on uniform data.

To make full use of the gigantic power parallel machines in the future, the other parallel processing paradigms, that cover much wider range of application domains, are longed to be developed.

2 New Domain of Parallel Application

Knowledge processing is the target application domain of FGCS project. Characteristics of knowledge processing problems are different much from that of scientific computations based on the data-parallel paradigm.

Dynamic and non-uniform computation often appear in the knowledge processing. For example, when a heuristic search problem is mapped on a parallel computer, workload of each computation node changes drastically depending on expansion and pruning of the search tree. Also, when a knowledge processing program is constructed from many heterogeneous objects, each object arises non-uniform computation. Computation loads of these problems are hardly estimated before execution.

These large computation problems with dynamism and non-uniformity are called *the dynamic and non-uniform problems* in this paper. When a system supports the new computation paradigm suitable for *the dynamic and non-uniform problems*, its coverage of the application domain must expand not only to the knowledge processing but also to some classes of large numerical and symbolic computation that have less data-parallelism.

3 Research Themes

The dynamic and non-uniform problems arise *new* requirements mainly on the software technology. They need more complex program structure and more sophisticated load balancing scheme than that of the data-parallel paradigm.

These items, listed below, have not been studied enough for *the dynamic and non-uniform problems* with large computation.

1. Modeling scheme to realize large concurrency
2. Concurrent algorithms
3. Programming techniques
4. Load balancing schemes
5. Language design
6. Language implementation
7. OS implementation
8. Debug and performance monitoring supports

The latter five items should be included in the topics of design and implementation of the system layer. The former three items should be included in the application layer or more general framework of software development.

4 Approach

Such an approach has been taken in the FGCS project that the system layer (including the topics 5 to 8 in section 3) was carefully tailored to suit *the dynamic and non-uniform problems* and topics of the upper layer (1 to 4) were studied on the system.

Key Features in the System Layer : The system layer satisfies these items to realize efficient programming and execution of the target problems.

1. Strong descriptive power for complex concurrent programs
2. Easy to remove bugs
3. Ease of dynamic load balancing
4. Flexibility for changing the load allocation and scheduling schemes to cope with difficulty on estimating actual computation loads before execution

Mainly, the language feature realizes these characteristics and the language implementation supports efficiency. The key language features are listed below.

- **Small-grain concurrent processes** : A lot of communicating processes with complex structure can be easily described, realizing large concurrency.
- **Implicit synchronization/communication** : They are performed between concurrent processes even in remote processors, which helps to write less buggy programs.
- **Separation of concurrency description and mapping** : Programmers firstly describe concurrency of the program without concerning with mapping (load allocation). Mapping can be specified with a clearly separated syntax after the concurrency description is finished. Runtime support for the implicit remote synchronization enables it.
- **Handling a scheduling** without destroying the clear semantics of the single-assignment language
- **Handling a group of small-grain processes** as a task

The language implementation realizes an efficient execution of these features, including a efficient OS kernel implementation of memory management, process scheduling, communication, virtual global name space, etc. [Taki 1992]. The other OS functions, which are written in the language, realize an research and development environment of parallel software including a programming system, task management functions, etc.

Research for the Upper Layer : Research topics of 1 to 4 in section 3 have been studied. After toy problems have been tested enough, R & D on practical large applications become important.

Strong cooperation of experts on application domain and on parallel processing is indispensable for those R & D. Several R & D teams have been made for each application development. Firstly, the research topics have been studied focusing on each application, then commonly applicable paradigms and schemes are extracted and supported by the system as libraries, OS functions or programming samples.

5 Current Status

System Implementation : A concurrent logic programming language KL1, which has those features listed in section 4, has been efficiently implemented on the parallel inference machine PIM. A parallel operating system PIMOS, which is written in KL1, supports an R & D environment for parallel software.

Very low-cost implementation of those features [Taki 1992] encourages the research of load balancing schemes. The language features helps the research of various concurrent algorithms and programming techniques.

Application Development : Practical large applications have been implemented [Nitta 1992], such as:

- LSI-CAD system : Logic simulation / Placement
- Genome analysis system : Protein sequence analysis / folding simulation / structure analysis
- Legal reasoning system
- *Go* game playing system
- Other eight application programs with different knowledge processing paradigms

Most of them arise dynamic or non-uniform computation. Some measurements show very good speedup and absolute speed by parallel processing.

Common Paradigms and Schemes : Efforts on extracting common paradigms and schemes from each application development have been continuing. Categorizing dynamic process structures and load distribution schemes have been carried on. Performance analysis methodologies have also been studied [Nitta 1992].

A multi-level dynamic load distribution scheme for search problems is already supported as a library program. A modeling, programming and mapping scheme based on *a lot of small concurrent objects* have been commonly used among several application programs.

6 Conclusion

New paradigms of parallel processing, that can cover *the dynamic and non-uniform problems*, are expected to expand application domains of parallel processing much larger than ever.

The dynamic and non-uniform problems must be a large application domain of parallel processing, coming next to the applications based on the *data-parallelism*. Parallel processing systems, that support efficient programming and execution of the dynamic and non-uniform problems, will get close to the general-purpose parallel processing system.

The KL1 language system, developed in the FGCS project, realize many useful features for efficient programming and execution of that problem domain. Many application developments have been proving effectiveness of the language features and their implementation.

R & D of problem modeling schemes, concurrent algorithms, programming techniques and load balancing schemes for that problem domain have started in the project, and still have to be continued. The accumulation of those software technology must make the true general-purpose parallel processing system.

References

[Nitta 1992] K. Nitta, K. Taki and N. Ichiyoshi. Experimental Parallel Inference Software. In *Proc. of the Int. Conf. on FGCS*, 1992.

[Taki 1992] K. Taki. Parallel Inference Machine PIM. In *Proc. of the Int. Conf. on FGCS*, 1992.