並列推論マシンと基本ソフトウェア総論

内田 俊一

(財団法人) 新世代コンピュータ技術開発機構、研究所
東京都港区三田一丁目 4-28 三田国際ビル 21F
uchida@icot.or.jp

概要

知識情報処理に適した新しいコンピュータ技術の開発を目指して1982年度から開始された第五世代コンピュータプロジェクトも、10年が経過し、その最終段階に到達している。このプロジェクトは、論理プログラミングを核として、知識処理の分野と高度並列処理の分野を結びつけ、新しいコンピュータ技術を創造しようと試みた。

現在、10年間の研究成果を総合的に評価するために、主要な研究成果を統合した、第五世代コンピュータプロトタイプシステムが試作され、稼働を開始したところである。その中核は、並列推論マシン (PIM) とその基本ソフトウェアである。

これらは、知識処理や記号処理の応用問題を並列処理により解くために必要な汎用の並列プログラミング言語と環境を備えている。これによって、非定型、不均質な並列処理を、自由に、効率良く記述できるようになり、知識処理や記号処理の応用問題を並列処理により解くことが可能となった、

これは、論理型言語のもつ言語機能を実現するために必要なメモリ管理の自動化機構やプロセス間の同期の自動化機構が、効率良く実装されたこと、 いくつもの並列応用実験ソフトウェアの開発をとおして、アルゴリズムやプログラムのパラダイムや効率的な負荷分散の手法が開発されたことによる。

総合的にみると、並列推論マシンおよび、基本ソフトウェアの研究成果はプロジェクトの当初の目標を、十分に満たすものとなっている。この論文は、並列推論マシン (PIM) と基本ソフトウェアについてのコンパクトな解説を行うことを意図したものである。

1 はじめに

知識情報処理に適した新しいコンピュータ技術の開発を目指して1982年度から開始された第五世代コンピュータプロジェクトも、10年が経過し、その最終段階に到達している。 このプロジェクトは、将来の知識処理の理論的バックボーンは『論理』であるとの前提に立ち、第五世代コンピュータシステムの中核となるプログラミング言語として、論理プログラミングを採用した。また、実用的な知識処理を行なうためには、強力な記号処理能力が不可欠との認識から、大規模並列処理によって、これを実現することを目指した。このように、論理プログラミングを核として、知識処理の分野と高度並列処理の分野を結びつけ、新しいコンピュータ技術を創造しようと試みたのである。

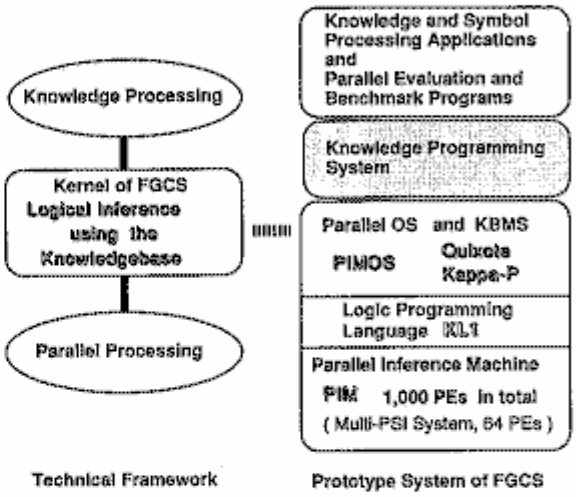


図 1: 第五世代コンピュータプロジェクトの枠組

現在、10年間の研究成果を総合的に評価するために、主要な研究成果を統合した、第五世代コンピュータプロトタイプシステムが試作され、稼働を開始したところである。その中核は、並列推論マシン (PIM) とその基本ソフトウェアである。並列論理プログラミング言語、KL1を効果的にサポートするPIMは、アーキテクチャの異なる5種のハードウェアモジュールを含んでいる。全体の要素プロセッサ数は、約1,000台であり、知識処理用としては世界最大の規模である。 PIMは、KL1の並列

言語処理系ソフトウェアをその中に含んでいる。また、その基本ソフトウェアは、PIMを制御する並列OS、PIMOSと知識ベース管理システム、KBMSから成る。KBMSは、その下層として、並列データベース管理システム、Kappa-Pを持ち、その上層に演繹オブジェクト指向データベースを基盤とする知識表現言語、Quixoteとその管理システムを備えている。これらは、すでに、PIM上で稼働しており、世界で最も完成度の高い並列基本ソフトウェアとなっている。

この上には、制約論理プログラミング言語の処理系や並列定理証明プログラム、自然言語処理システムなど、知識プログラミングソフトウェアと総称される、より高度な推論や知識処理のためのツールとなるソフトウェア群がある。また、並列推論マシンと基本ソフトウェアの総合的な評価と、並列記号処理や知識処理の応用を開拓することを目的として、法的推論システム、遺伝子情報処理システム、VLSI-CADシステム、各種エキスパートシステムなどの、実験的応用システムが試作され、PIM上で稼働している。これらは、知識処理の応用問題は多くの並列性を含んでおり、並列化は、処理速度向上にきわめて有効であることを示している。また、同時に、より高い台数効果を得るためには、並列アルゴリズムや負荷分散など汎用的な並列ソフトウェア技術の研究が重要であることを示している。

総合的にみると、並列推論マシンおよび、基本ソフトウェアの研究成果はプロジェクトの当初の目標を、十分に満たすものとなっている。この論文は、並列推論マシン(PIM)と基本ソフトウェアについてのコンパクトな解説を行うことを意図している。

本プロジェクトは、日本の国家プロジェクトとしては、初めて、コンピュータ科学への貢献と国際的な共同研究の推進を、その目的として掲げ、これを実践するために研究成果の公開を広く行なった。この結果、世界の多くの論理プログラミングや並列処理の研究者の協力を得ることができた。並列推論マシンとその基本ソフトウェアの周辺には、多くのソフトウェアや理論研究の成果が、豊かに積み上げられている。これらの成果は、世界中の論理プログラミングや並列処理の研究者達に帰属するものである。

2 研究開発目標と研究計画

2.1 研究開発の範囲と試作目標

このプロジェクトは、知識情報処理を指向した新しいコンピュータの基礎技術の開発を目指した。この新技術の理論的な基盤を『論理』におき、これに基づいて知識処理に適したソフトウェア技術やハードウェア技術の研究開発テーマを設定した。

2.1.1 並列推論システム

研究開発の中核は、論理型言語を効率良く実行するハードウェア技術とソフトウェア技術の開発であり、これらは並列推論システムと呼ぶ新しいコンピュータシステムの試作により、具体的に提示される。並列推論システムは、並列推論マシン(PIM)のハードウェア、KL1言語処理系ファームウェアおよびソフトウェア、並列OS、PIMOSより構成される。研究のゴールを明確にするために、並列推論システムや多くのソフトウェアを統合したプロトタイプシステムを試作目標として掲げた。

プロトタイプシステムの規模としては、1,000台規模の要素プロセッサを含むものとし、性能目標としては、100MLIPSから、1GLIPSとした。プロトタイプシステムの基本ソフトウェアとしては、記号処理、知識処理の応用を、自由にプログラムでき、かつ、並列処理によって高速に実行できるものを目指した。

以上のようなハードウェアやソフトウェアを、バラバラではなく、一体化したものとして作り上げて行くことを目指したことも、本プロジェクトの大きな特徴であった。

2.1.2 知識ベース管理システムと、知識プログラミングソフトウェア

並列推論システムの上には、本格的な知識処理の実現を目指して、知識処理の中核として必要となる種々のソフトウェア技術の構築を目指し、次のようなソフトウェアの研究開発を実施した。

- 知識表現言語と知識ベース管理システム(KBMS)
- 高度な問題解決のための高次推論ソフトウェア
- 自然言語処理ソフトウェア

これらの研究においては、『論理』に基盤を置いた新しい知識処理のソフトウェア技術の確立を意図していた。人間の社会システムの中で生み出される種々の知識を蓄えるためには、知識表現言語によって記述し、これを蓄積して、知識ベース化されなければならない。

このように表現され、蓄えられた知識ベースを柔軟に扱うためには、機能の高い推論機構が必要となる。自然言語処理は、人間との対話機能の実現のために必要な技術であるが、同時に、このような高度な推論機構や知識表現を生み出すのに適した研究テーマでもある。

このような高レベルの機能を持つソフトウェアは、述語論理を扱う定理証明プログラムのように、記号処理の高速処理能力に裏打ちされて、初めて実用に供し得るものとなる。このような高速処理の実現は、並列処理によってのみ可能であり、この実現に必要な「並列知識処理」のためのソフトウェア技術も研究開発目標に含めることとした。

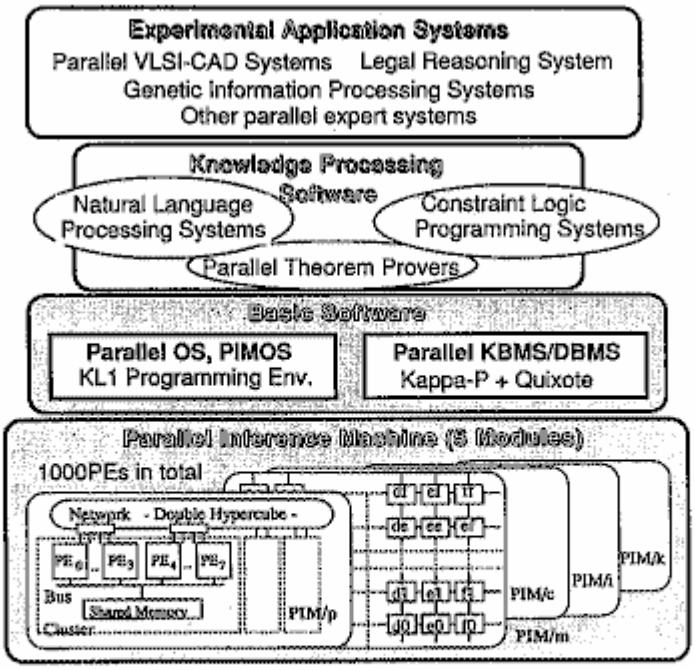


図 2: プロトタイプシステムの構成

2.1.3 評価および実験的応用ソフトウェア

多くの要素技術の基礎研究を実施するためには、その研究テーマに適した典型的な問題を取り上げて、ソフトウェアの試作実験を行うことが必須である。

このため、VLSI や電子機器の設計や故障診断などの工学的な問題を始めとして、会社の経営や行政、医療分野における問題など多くの社会システムにおいて生み出され利用される知識や規則を扱う、知識処理システムを対象として取り上げることとした。これによって、人間社会で生み出されるさまざまな知識を扱い得る、汎用的な知識処理のソフトウェア技術の開発を行った。

このような知識処理の分野は、科学技術計算の分野などと異なり、コンピュータ科学における未開拓分野の一つである。最近実用化されはじめたエキスパートシステムなどの研究において、この分野の研究が徐々に進められている。そこで作られている知識ベースの規模は、ルール数でせいぜい数 100 である。これは、知識を整理し、プログラムする技術が未熟であることが大きな要因であろう。大規模な並列推論システムの有効性を評価するためにも、大規模な問題を扱う応用プログラムの開発が必要であり、このために、広く学際的な研究領域から応用問題を探すこととした。

以上のようにこのプロジェクトの研究開発の対象分野は、極めて広く、かつ未開拓であることから、個々の研究テーマ間相互の結び付きを確保するための工夫が必要であった。

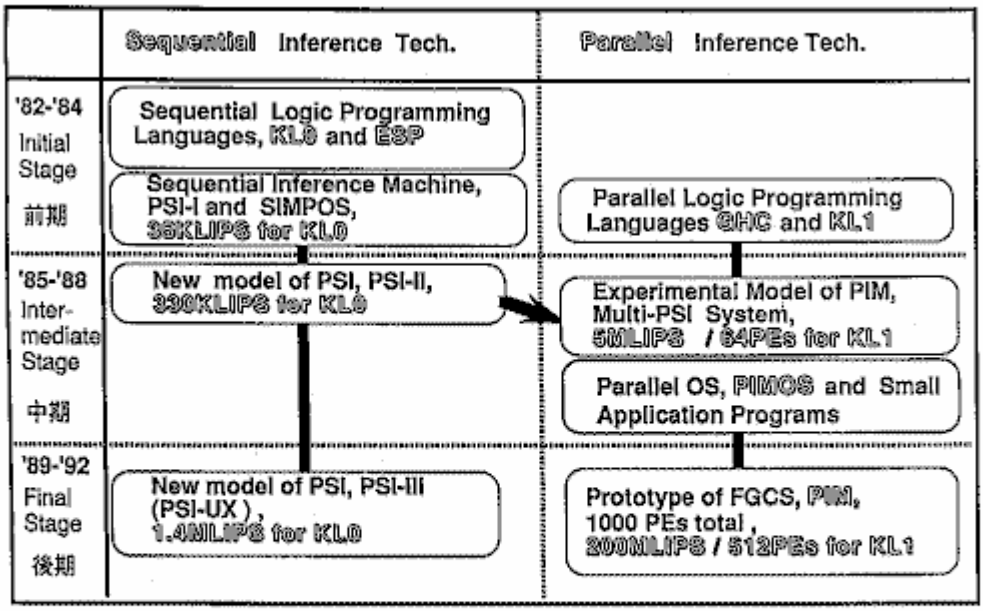
これは、論理型言語をもった並列推論システムの試作という明確な目標と、『論理』という共通的な理論的基盤を、持つことによって達成された。そしてこれが、このプロジェクトの、極めて重要な、そして、また大きな特徴となった。すべての研究テーマの研究成果は、中核となる並列推論マシン上に統合することを意図したことで、異なる分野の研究者の間の密接な交流が生まれ、これまで未開拓だった、学際的な研究分野をも作り出すことができた。

プロジェクトの開始時点では、以上述べた、研究開発目標のうちで、2.1.1 の試作目標のみが、比較的明確に設定されていた。残りの目標は、プロジェクトの進捗に従って、徐々に明確化され、追加されていったものである。

2.2 プロジェクトの研究計画

本プロジェクトの計画立案は、1979 年から 3 年間に渡って行われた。そして、その詳細計画と予算は、1981 年度末に決定された。この当時は、論理型言語が、まだ世に出たばかりであり、論理型言語自身の能力が、未解明な段階であった。このため OS のような大規模かつ複雑なソフトウェアが記述できないのではないか、とか、実行のオーバヘッドが大きく使い物にならないのではないか、といった危惧も持たれていた。

表 1: 推論システムの開発経緯



また、高級言語と関連づけられた並列アーキテクチャの研究としては、関数型言語を実行するデータフローマシンの先駆的な研究があったが、知識処理、記号処理用のマシンとしての能力は、まだ、明らかではなかった。エキスパートシステムなどの、知的なソフトウェアの研究も小規模な実験的プログラムを試作している段階であった。

このように、並列推論システムを構築するに必要な要素技術のほとんどが、まだ、未成熟な段階にあった。このような状況の中で、このプロジェクトの実施計画を作成するにあたり、10年の期間を、前期3年、中期4年、後期3年の、3つの期に分け、それぞれの期の研究内容をおおまかに設定した。

- 前期
  ハードウェア、ソフトウェア、基礎理論の多岐にわたる要素技術の研究と、研究開発ツールの開発を行う。

- 中期
  要素技術の評価と取捨選択。見通しの立った要素技術、優れた技術を選び、さらに発展させる。中期末に、主要技術を統合して、中規模の実験システムを試作する。

- 後期
  中規模実験システムの技術を評価し、優れた技術を選択。それらをさらに発展させて、大規模プロトタイプシステムを試作する。

プロジェクトの開始時点では、前期についてのみ詳細計画を作成し、中期や後期の詳細計画は、ひとつ前の期の終わりに作成することとした。これは前期開始時点では、中期や後期に何ができるのかは、予測できなかったためであった。予算や研究開発体制も、研究開発の進捗に評価しつつ決定することとした。十分な進捗がなければ、中途で打切ることもあり得るプロジェクトであった。

3 前期の推論システムの研究開発経緯

3.1 逐次型推論マシン (PSI) の開発

1,000台規模の要素プロセッサを結合した並列推論マシンと、それを制御する基礎ソフトウェア、特に、並列OS, PIMOSを作り上げるためには、その構成要素となる個々の技術を、ひとつひとつ確立し、積み上げていく必要があった。その過程では、多くの技術的な選択枝があり、その評価と絞り込みのプロセスを的確に行うことが必須であった。評価のためには、できるだけ実際的なシステムを試作することが評価の信頼性をあげるために重要である。

前期においては、逐次論理型言語のパーソナルワークステーションを試作し、論理型言語の記述能力と実行性能の評価を行なうこととした。同時に、これを共通的に用いるソフトウェア研究開発ツールとすることとした。(当時は、関数型言語LISPのワークステーションが、開発されており、このようなワークステーションの論理型言語版を目指した。)当時の最も高速な論理型言語の処理系は、DEC20システム上で動作するDEC10 Prologであり、これと同程度の性能を目指した。

まず、逐次型の機械語、KL0を設計し、これを効率的に実行する逐次型推論マシン (PSI-I) を開発した。ハード

ウェアとしては、タグアーキテクチャとマイクロプログラム制御を採用した。　次に、KL0の上にシステム記述言語ESPを開発した。ESPは、Prologにオブジェクト指向のモジュール化機構を付加した言語であり、PSI用のOS、SIMPOSの開発や、その後の、知識処理の実験ソフトウェアの試作に広く用いられた。

この言語のソフトウェア生産性は極めて高く、我々は論理型言語の優秀性を確信した。その実行性能も、35KLIPSを達成し、当初の目標をクリアーした。しかし、同時に、コンパイラの最適化機能やアーキテクチャーの工夫により、さらなる性能の向上が可能であることも見出した。このようにして、PSI-IとそのOS, SIMPOSの開発は、成功裏に終了した。PSI-Iは、約100台が製造され、本プロジェクトの最初の共通ツールとして用いられた。共通ツール化は、また、ICOTや委託先の研究グループの間で、ソフトウェアの自由な流通と分業を可能とし、プロジェクト全体としてのソフトウェアの生産性を向上することに、大きく貢献した。

前期には、PSI-IとSIMPOSの開発と並行して、並列論理型言語の研究も行なった。この言語の設計に当たっては、記述能力の高さと共に、マシン言語としての実装上の簡潔さに重点をおいた。並列型の論理型言語については、PARLOGやCPなどの先駆的な研究があり、これらの研究者達との交流をもとに、PIMの機械語に適した簡潔な並列論理型言語、GHCを設計した。

GHCは、さらに機能を簡単化したFGHCが作られたのち、資源管理や実行制御機能を実装するためのメタコールなどの機能が付加され、実用的な並列論理型言語、KL1に発展していった。

3.2　PSI開発の意義

PSI-IとそのOS、SIMPOSの開発によって、論理型言語とその実行機構についての定量的な評価が可能となった。この評価結果は、中期以降の研究開発計画の策定に大きな影響を及ぼした。

3.2.1　記述能力およびソフトウェアの生産性

論理型言語の記述能力については、OSのような細かな制御を能率良く記述できるか、また、規模の大きなソフトウェアが作成可能かという点が第一の問題であった。

これについては、SIMPOSの開発が、解答を与えてくれた。SIMPOSの第一版のサイズは、マルチウィンドウベースのインタフェースやプログラミング環境を含み約10万行であった。この規模のソフトウェアを論理型言語に不慣れな初心者を多く含む開発チームが約1年半で作り上げた。

この印象は極めて強く、この時点で、並列推論マシンのOSも、論理型言語で記述すべきであるとの決心をした。

SIMPOSの記述には、論理型言語にオブジェクト指向のモジュール化機能を追加したESPという言語を用いた。論理型言語とオブジェクト指向言語の融合は、ソフトウェアの生産性と保守性を高める上で極めて効果的であることを実感した。

3.2.2　論理型言語の実行性能

タグアーキテクチャを用いたPSI-Iの、ハードウェアは、DEC10 Prologと同等の、約35KLIPSの性能を達成した。この速度は、当初の知識処理の応用ソフトウェア開発には十分であった。　また、主記憶は、80MBの大容量のものを付加した。主記憶が大きいことで、ソフトウェアのプロトタイピングが容易となり、生産性が向上した。この速度と主記憶容量は、大規模な論理型言語の応用ソフトウェア開発を、多いに促進した。

PSI-Iのハードウェアの規模は、カスタムVLSIを用いなかったこともあり、11枚のプリント基板を要した。また、コンパイラによるオブジェクトコードの最適化を導入すれば、さらに性能を向上させ得ることがわかった。これは、後の、PSI-IIの開発において、D.H.D. Warrenの提案による抽象マシン命令セット(WAM)を導入することで実現された。

PSI-IのCPUの実装に必要なハードウェア量が明確化したことから、VLSI技術を用い小型化すれば、並列推論マシンの要素プロセッサも実現可能であることが推測できた。

このように、PSI-IとSIMPOSの開発は、論理型言語の記述能力の高さと実用性を証明した。また、その高速実行のためのCPUの実現方法についての見通しも得ることができ、中期計画の内容が明確化したのである。

4　中期における推論システムの研究開発

4.1　並列推論システムの開発の開始

4.1.1　並列言語処理系と並列OSの開発

中期における第一の目標は、前期に設計した並列論理型言語KL1 (当時はまだ、FGHCと呼んでいた。)の並列ハードウェア上への実装と、並列推論マシン用のOS、PIMOSの開発であった。KL1は、AND並列型の並列論理型言語であり、言語処理系は、記憶セルの自動管理機構とデータフロー型のプロセス間の同期機構という、大規模な並列ハードウェア上では、実装されたことのない機構の実現を前提としていた。

これら2つの機構は、大規模かつ汎用的な並列処理を記述するプログラミング言語には、不可欠のものと考えられ、本プロジェクトにおける重要な技術上のブレークスルーと考えられた。そのため、これらの機構を効率良く実装できるか否かが第一の問題でとなった。これには、さらに実行の高速化のためにどのようなハードウェアサポートを行なうべきかの問題も含まれていた。

KL1が、このような機構を備えることから、PIMOSは、主記憶やプロセッサなどの資源管理とユーザプログラムの実行管理を行うだけでなく、さらにユーザプログラムの並列プロセスへの分割や、それらの要素プロセッサへの効率的な割り当てを支援する役割を果たすことが求められた。

将来においては、このようなプログラム負荷の分割と割り当てを自動化することが望ましい。しかし、このためには多くの並列応用ソフトウェアの負荷の分割と割り当ての最適パターンを蓄積する必要があり、当面は、プログラマが指定する方法をとることとした。

このようなKL1の実装、PIMOSの持つべき機能の明確化は、実際にプログラムを書きながら、試行錯誤的に進めるしか、方法はないと考えられた。このためには、このようなソフトウェア実験を行なう使い易く安定した並列ハードウェアが必要との結論に至った。

4.1.2 PSI-IIとマルチPSIの開発

KL1やPIMOSの研究開発と並行して、論理型言語を用いた知識処理のソフトウェアや理論の研究も、積極的に推進された。この結果、前期に開発したPSI-Iの性能を上回るソフトウェア開発ツールが不可欠となった。そこで、前期末より、検討を開始し、中期に入ってから、PSI-Iの小型化版PSI-IIの開発を開始した。

VLSIの進歩によるハードウェアの世代交代は早く、この種のワークステーションの寿命は3年位であった。PSI-IIの開発は、2つの目的をもって始められた。一石二鳥を狙ったわけである。ひとつは、小型かつ高性能のワークステーションを開発し、知識処理ソフトウェアの試作実験を促進することであった。

もう一つの役割は、そのCPUを要素プロセッサとして、並列ソフトウェア開発用の並列ハードウェアを生み出すことであった。この並列ハードウェアは、並列推論マシンの実験機、マルチPSIと呼ばれた。並列ハードウェアの要素プロセッサとしての信頼性を確保する上で、PSI-IIは、共通ツールとして、多数製造されることから、有利であると期待したわけである。

PSI-IIは、カスタムVLSIチップを用いて実装したことにより、ハードウェアのサイズは、PSI-Iに比べて、約6分の1となった。性能は、D.H.D. Warrenが新たに提案したWAMを採用し、コンパイラによるオブジェクトコードの最適化をはかると共に、アーキテクチャを改良したことにより、PSI-Iの約10倍の、330KLIPSを達成した。また、主記憶の最大容量も、320MBに拡張し、大規模の知識処理の実験ソフトウェアの試作を大きく加速した。

中期においては、PSI-IとPSI-IIを用いて、種々のエキスパートシステムや自然言語処理システム、制約論理プログラミングシステム、データベース管理システムなどの知識処理の基礎となるソフトウェア研究が、大きく前進した。中期の4年のうち、最初の2年間は、主に、PSI-Iが用いられ、後半の2年は、PSI-IIが用いられた。

PSI-IIの開発と並行してマルチPSIの開発も進められた。マルチPSIは、8×8のメッシュ構造を持ち、最大構成で、64台の要素プロセッサを接続することができた。また、おのおのの要素プロセッサには、80MB主記憶を付加した。マルチPSIの最大構成では、約5GBのメモリを有する大きなシステムとなった。

4.1.3 KL1の言語処理系と並列OS, PIMOSの開発

並列論理型言語KL1の分散処理系やPIMOSの開発は、お手本がなく、まさに、世界で始めての試みであった。

KL1の言語処理系は、並列ハードウェア上の並列分散処理系であり、極めて複雑な構造をもつものであった。特に、分散記憶を対象とするガーベージコレクション(並列GC)の実装や、データフロー方式に基づくプロセス間の自動同期機構などの設計や実装、さらに、プログラムのオブジェクトコードを要求発生時点で、特定の要素プロセッサにロードする動的なローディング機構の設計と実装など、これまで、論文レベルでしか扱われたことがない機構を、いかに効率よく実現するかが、大きな問題であった。また、システムの診断機構やユーザプログラムのデバッグ機能なども重要な問題であった。

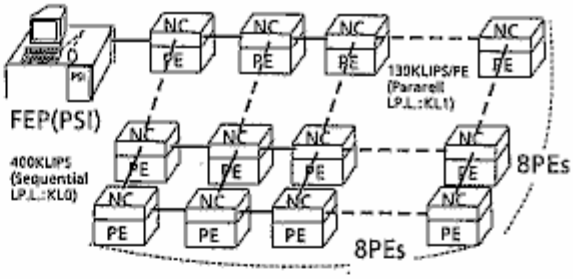
PIMOSの試作も、その資源管理やプログラムの実行管理などの各種機能についての、仕様決定や実装方法の工夫など、部分的な試作を行ないながら決定していく以外に、よい方法がないという、先端技術開発につきものの問題があった。

このため、第一段階は、Unix環境の上で、C言語を用い、KL1言語処理系とPIMOSのサブセットを開発して、仕様と実装方法を検討した。このシステムは、PDSSと呼ばれ、その後は、PIMOSの部分モジュール試作のワークベンチとして使われた。

第2段階では、KL1言語処理系は、PSI-IIのファームウェアで実装された。これは、KL1のフルセットをサポートしており、KL1の擬似並列処理系として、PIMOSの開発に広く用いられた。このKL1処理系は、後にマルチPSIに移され使用された。

PIMOSは、Unixマシン上の、PDSSシステムを用いて、部分モジュールを作成し、その後、PSI-II上のKL1疑似並列処理系上で、さらに開発が進められた。さらに、その後も、PSI-II上の、PIMOSを含むKL1の疑似並列処理系は、パーソナルなKL1のプログラミング環境として、おおいに用いられている。

マルチPSI上の、KL1の分散処理系は、並列GCや、データフロー方式に基づくプロセス間の同期機構をもち、すべて、ファームウェアで実装された。この結果、KL1の実行速度としては、要素プロセッサ単体あたり、150KLIPSを達成した。また、64台の要素プロセッサを用いることで、5ー10 MLIPSを実現した。



- Machine language: KL1-b
- Max. 64PEs and two FEPs (PSI-II) connected to LAN
- Architecture of PE:
  - Microprogram control (64 bits/word)
  - Machine cycle: 200ns, Reg.file: 64W
  - Cache: 4 KW, set associative/write-back
  - Data width: 40 bits/word
  - Memory capacity: 16MW (80MB)
- Network:
  - 2-dimensional mesh
  - 5MB/s x 2 directions/ch with 2 FIFO buffers/ch
  - Packet routing control function

図 3: マルチ PSI システム: 主な仕様と外観

この性能は、知識処理の並列応用実験ソフトウェアの試作用ツールとして、十分な性能であった。中期末の時点では、パズルや探索問題を解くベンチマークプログラムも作成され、KL1 と PIMOS の機能が、並列記号処理や知識処理の応用問題を記述するのに適していることが確認された。

4.2 並列推論システムの中核技術

4.2.1 汎用の並列プログラミング言語の実現

並列推論システムの開発にあたって、まず、第一に検討した問題は、知識処理を始めとする、多くの大規模で複雑な応用問題を自由にプログラムでき、かつ、効率良く並列実行させるためには、どのような機能を備えなければならないかということであった。

このプロジェクトの開始時点までの世界各地の並列処理の研究開発は、ハードウェアに重点をおいて進められていた。画像処理用や科学技術計算用の SIMD や MIMD の方式の並列マシンが作られ、実用に供されていた。しかし、これらの並列マシンにおいては、プログラム作成は、FORTRAN や C 言語などの、ノイマン型の逐次型言語か、あるいは、その拡張したものを用いることで行われていた。従って、その応用問題も、それらの言語でプログラムできる範囲に、おのずと限定されていた。

本プロジェクトの目指すような、汎用の並列プログラミング環境を必要とするような応用は、手付かずで残されていた。そして、この傾向は今日においてもそれほど変わっていない。従って、知識処理や記号処理用の汎用並列マシンを追及する立場からは、これらの並列マシンは、参考とはならなかった。

このような数ある並列マシンの研究の中で、関数型言語の並列実行を意図していた、データフローマシンは、汎用並列マシンとしての能力を備えていた。関数型言語は、ノイマン型言語と異なり、並列処理を根本においた言語である。

本プロジェクトで採用した並列論理型言語は、関数型言語よりは、言語の記述レベルが高く、やはり、本質的な並列処理の言語である。このような言語は、知識処理や記号処理用の応用問題を、自由にプログラムするために必要な機能を備えている。このような機能の実現は、容易ではないが、この実現なくしては、実際に人間が並列ソフトウェアを効率よく書くことができないと考えた。

このような機能は、次のような機構によって、実現される。従って、この実現技術が、並列推論システムの中核的な技術となっている。

1. 分散メモリーの自動管理機構 (並列ガーベージコレクション)
2. データフローモデルに基づく同期機構
3. 並列プログラムの分割とその負荷分散の最適化の支援機能

1と2は、KL1の言語処理系の中に組み込まれるものとし、3は、PIMOSのプログラミング環境や、その上の支援システムに組み込まれるものと考えた。

特に、3は、並列実行モデルに基づいて記述された応用プログラムを並列マシン上に、どのようにマッピングするかという問題であり、このような問題は、規則的な数値計算の問題なら自動化できるかもしれないが、汎用の知識処理や記号処理では、マシン性能を十分に引き出すような自動化は、容易ではないと考えた。そこで、まずは、プログラムを作成する人の責任で、陽に指定してもらうこととし、それを側面から支援する機能を準備しておく方針をとった。

4.2.2 汎用並列プログラミング言語機能の実現方法

上で述べた機構は、並列推論システムを構成する、次の主要な部分のいずれかで、実現する必要がある。

1. PIMのハードウェア、主に、要素プロセッサの機能に含める。
2. KL1言語処理系のファームウェア、もしくは、ソフトウェアの機能として実現。
3. PIMOS上のKL1プログラミング環境の機能として実現。コンパイラの最適化機能などがこの例。

これらの並列推論システムを構成する各階層の役割分担についての決定は、中期の始め頃から、徐々に行われた。そして、それは、このプロジェクトにおける、研究開発の技術的な大方針を決定することでもあった。

取り得る選択は、いくつかあり、そのうちの一つは、データフローマシンの初期の論文にみられるような、機能の高い要素プロセッサの開発を狙う方法である。VLSI技術を多用した、新しいアーキテクチャを探求するという方針は、極めて魅力的であった。

しかし、そのようなプロセッサの開発は、VLSI技術の急速な進歩があったとしても、リスクが大きいものであった。特に、並列応用ソフトウェアや並列OSが、存在せず、その挙動が未解明であったことから、そこで設計、試作されるプロセッサチップは、多くの設計変更が加わると想定された。いかにCADシステムが、発達しようと、ハードウェアの設計変更は、ソフトウェアに比べて、ずっと大変であり、ターンラウンド時間も長い。また、その要素プロセッサを、数百、数千と並べることを考えると、信頼性や保守性も考慮する必要がある。このようなことから、要素プロセッサのハードウェアの機能を多くするのは、リスクが大きいと判断した。

並列推論システムのユーザの立場から、考えるとプログラム負荷の分割と割り当ての自動化や最適化のような機能までもが、並列OSの機能として求められる。しかし、このような高度な機能を除いたとしても、並列OS、もしくは、KL1の言語処理系は、数百、数千台の要素プロセッサを対象とする資源管理やプログラムの実行管理を行わねばならない。このようなソフトウェア技術は、研究としては行われていたが、本格的に実装されたことはない。このことから、KL1の並列言語処理系やPIMOSの開発も、極めてリスクの大きなものと考えられた。

このような考察をもとに、要素プロセッサよりは、KL1言語処理系と並列OSに、より多くのリスクをかけることとした。要素プロセッサのハードウェアについては、性能対コスト比が大きく、要素プロセッサのハードウェアの量を、それほど増やさないタグアーキテクチャの利用や、命令実行のパイプライン化などに留めることとした。しかし、要素プロセッサの命令セットを、KL1の実行に最適化したことによる効果は、かなり大きなものとなった。

このような方針に従い、記憶の自動管理機構やデータフロー方式に基づくプロセス間同期機構は、タグアーキテクチャーによるサポートを前提として、KL1の言語処理系ファームウェア、もしくは、ソフトウェア(実行時ルーチン)によって実現することとした。また、プログラム負荷の分割と割り当ての支援機能は、KL1言語処理系とPIMOSで、分担することとした。

プログラム負荷の分割と割り当ての問題(負荷分散問題)の解決策としては、次のようにしている。すなわち、並列OSは、まず、要素プロセッサ数や要素プロセッサ間の結合方式が変わっても、それらを抽象化してみせる機能を提供する。その上で、並列プログラムの分割と負荷分散の指定は、プラグマと呼ばれる記述を、KL1言語と独立に追加できるように工夫した。

これにより、並列プログラムの記述は、まず、負荷分散などの指定を除外して、アルゴリズムのみを念頭において、問題を記述することから始められる。そして、あとから、負荷分散の指定を追加でき、作業の手順を、2段階に分けてに進めることができる。このような方法をとることで、PIMOS自身のプログラム開発も、能率良く進めることができた。

以上のように、このプロジェクトは、汎用並列処理の実現に必要な機能のうち、仕様や実装方法が未解明なものについては、KL1言語処理系やPIMOSなどのソフトウェアに分担させ、そこで、試行錯誤を行いながら、技術を確立して行く方針をとった。

ハードウェアの役割は、ソフトウェアの開発に適したプラットフォーム、もしくは、ワークベンチとしての要求を迅速に満たすことに重点をおいた。具体的には、並列ソフトウェアの試作実験をするに十分な、性能、メモリ容量、プロセッサ台数、安定性、信頼性を確保する大規模並列ハードウェアを開発することとした。しかし、数百台の要素プロセッサを並べた並列ハードウェアは、従来の汎用大型コンピュータなどとくらべて、規模的に桁違いであり、信頼性確保は、もちろんのこと、ハードウェアの動作監視や、システムの初期化など、多くの研

究すべき問題を解決しなければならなかった。

5 後期における研究開発

5.1 研究開発内容

中期末までに、並列ハードウェアとしてマルチPSI、その上のKL1の分散処理系、および、PIMOSから成る、並列推論システムが完成し、ペントミノや最適経路問題、簡単な構文解析、つめ碁などの並列ベンチマークプログラムが作成された。これらのベンチマークプログラムは、小規模ではあったが、記号処理の問題が、KL1で記述でき、並列処理による効果も、十分あることが示された。

これをうけて、後期に取り組むべき、研究開発の詳細計画と目標が定められた。その骨子は、要素プロセッサが64台のマルチPSIから、要素プロセッサが数百台規模の並列推論マシンPIMへと大きく飛躍をすることである。このためには、PIMのハードウェアの開発と共に、より大規模、複雑な知識処理、記号処理の応用問題を設定し、KL1によるプログラム作成を行ない、これまで本格的に試みられなかった知識処理の問題の並列処理に挑戦することが必要でである。このような大規模な知識処理の応用問題に挑戦できるためには、効率的な並列ソフトウェア作成手法をも、同時に、開発してゆかねばならない。

このような観点から、研究開発の重要課題は次のように定められた。

1. 効率的な大規模並列ソフトウェア作成技術

   (a) 並列プログラミング技術
   - 並列モデルに基づく問題のモデル化技術
   - 並列プログラミングのパラダイムの考案と収集
   - 並列アルゴリズムの開発

   (b) 並列プロセスの並列プロセッサへの割り当て技術
   - 動的な負荷の最適割当て技術
   - 稼働状況データの収集と提示技術
   - パーフォマンスデバッギング技術

2. 並列推論システムの処理能力を利用した知的システム構成技術

   (a) より高い機能を持つ推論エンジンや記述レベルの高いプログラミング言語の研究

   (b) 知識記述の方法(知識プログラミング言語)と記述された知識の管理方法、及び、知的システム構成技術

以上のような並列推論システムの本格的な活用に必要な基礎技術と、その強力な記号処理能力を知的システムの知的レベルの向上につなげる技術の開発を目指して、次のような後期の研究開発テーマを設定した。

1. 大規模な並列ハードウェア、PIMの試作

   要素プロセッサ数としては、1,000台が、目標であった。しかし、応用問題のモデルと、PIMアーキテクチャー間の、マッピングの研究を推進することを意図して、1,000台を単一の アーキテクチャーを持つ並列マシンとせずに、いくつかの異なるアーキテクチャーを持つPIMのハードウェアモジュールに分けて試作することとした。これは、いろいろなアーキテクチャの比較検討を行うこと、並列応用ソフトウェアのモデルとのマッピングの良否の評価を行なうことをねらったことによる。

2. PIM用のKL1言語処理系開発

   要素プロセッサのアーキテクチャやプロセッサ間結合方式の違いを考慮にいれた、効率のよいKL1言語の処理系の実装を行なう。また、異なるアーキテクチャーのPIMに、適したKL1処理型を開発する。

3. PIMOSの拡張、改良

   資源管理、実行管理などの機能の拡張、多数のディスクを並列に利用できるファイルシステムの開発、および、KL1の上位言語の開発や並列プログラミングやパーフォマンスデバッギングを支援するツールを備えた生産性の高い並列プログラミング環境の開発。また、KL1の上に、オブジェクト指向言語、AYAを開発する。

4. 並列知識ベース管理システムの開発

   中期までに試作を完了した非正規型関係データベース管理システム、Kappaの仕様をもとに、その並列版を開発する。PIMの多くの要素プロセッサと多数の磁気ディスクを用いて、分散処理と並列処理を行う、高速なデータベース管理システムを開発する。また、その上に、演繹オブジェクト指向データベースにもとづく知識表現言語とその知識ベース管理システムを開発する。

5. 知識プログラミングソフトウェアの研究

   知識処理をより効率的に実現するための、制約論理プログラミング言語処理系などの、より記述レベルの高いプログラミング言語や、定理証明システムなどの、より機能の高い推論エンジンなどの開発。そ

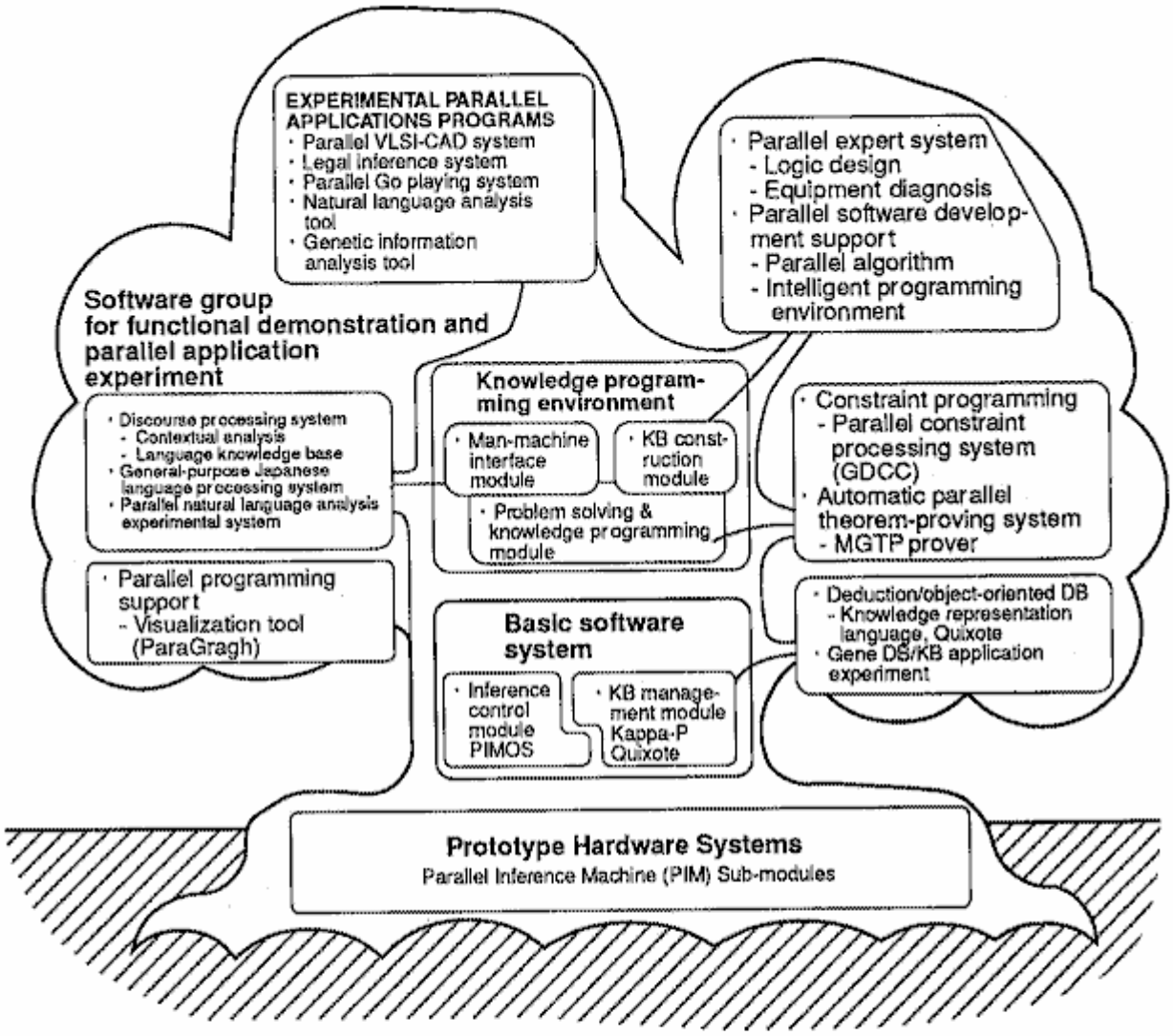


図 4: 後期の研究開発テーマ

のほか、自然言語を用いる 対話用インターフェースソフトウェアの開発などを行う。

6. 機能実証と並列応用実験ソフトウェア

並列推論システムや知識プログラミングソフトウェアの評価のための 実際的なベンチマークプログラムの開発、および、 並列プログラミング技術や知識の表現や管理、高度な推論機能を 探求するための例題となる並列応用の実験ソフトウェアを開発する。

5.2 後期における研究開発成果

後期において、実施された個々の研究テーマの狙いと実施内容は、テーマの性質によって異なるものとなっている。並列推論システムの開発においては、PIM、KL1言語処理系、PIMOSを一体化し、大規模な記号処理や知識処理の応用問題に適用することに重点を置いた。

一方、知識ベース管理ソフトウェアや知識プログラミングソフトウェアの研究開発においては、理論の確立や、まとまりのつく要素技術のソフトウェアツール化に重点をおいた。また、並列応用実験ソフトウェアについては、新しい学際的な応用問題を見出すことに重点をおいた。

5.2.1 PIMのハードウェアとKL1言語処理系

PIMのハードウェアの役割は、PIMOSなどの基本ソフトウェアと共に、その上で大規模な知識処理、記号処理の並列応用実験ソフトウェアを稼働させ、並列知識処理のソフトウェア技術開発の土台となることである。

同時に、要素プロセッサのアーキテクチャや、要素プロセッサ間の結合方法についても、できるだけ多くの知見が得られることが望ましい。また、並列応用ソフトウェアのモデルと並列ハードウェアの物理的な構造(アーキテクチャ)との、適合性(マッピングの良否)についても、評価できることが望ましい。このように、PIMのハードウェアは、並列応用ソフトウェア研究用のツールとして役割を与えられており、この目的にあった設計を行なうことがもとめられる。

実際に、PIMを設計するにあたり、その要素プロセッサのアーキテクチャやプロセッサ間のネットワークの構造を検討した。しかし、これらは多くの選択枝がある。まず、要素プロセッサについては、マイクロプログラム制御を用いるCISC型か、バイト命令を持つRISC型かといった、命令セットの選択がある。 プロセッサ間ネットワークについては、マルチPSIで用いた2次元メッシュ構造の他に、8台のプロセッサを共通バスで密結合したクラスタを構成し、これをパケット交換網で接続する方式などがある。

これらの選択枝の良否は、そのマシンの上で実行される並列応用プログラムのモデルやアルゴリズムに依存する。つまり、このようなマッピングの良否によって評価が異なる。 しかし、そのようなマッピングの良否は、これから研究すべきもので、PIMは、そのツールとしての役割を負っていると考えた。

そこで、要素プロセッサのアーキテクチャーやネットワークの構造など、いくつかの重要と思われる選択枝を、まとめて、5種類のハードウェアモジュールを試作することとした。それぞれには、Table 2 のような特徴がある。試作したモジュールの規模は、大規模なソフトウェアの実験に使うものでは、256から512台の要素プロセッサを接続することとし、アーキテクチャの実験用のモジュールは、16から20台の小規模のものとした。

これらのPIMのハードウェアモジュールの要素プロセッサ数は、1000台以上となる。並列の基本ソフトウェア、特に、PIMOSにおいては、物理的な要素プロセッサ数は、100万台規模の要素プロセッサ数を管理し得る能力をもっているが、PIMの要素プロセッサ数は、数百台の規模であれば、機能の評価には、十分と判断した。

これらハードウェアモジュールは、すべて、KL1とPIMOSをサポートし、いろいろな応用ソフトウェアを実行させて、プログラムのモデル化の手法とマシンのアーキテクチャの間のマッピングの良、否の比較実験ができるようにした。すなわち、並列ソフトウェア技術の開発の実験台となる並列ハードウェアを目指した。

PIMの5種のモジュールは、それぞれ異なる命令セットを持つことから、それらのKL1言語処理系は、新たに開発する必要があった。マルチPSIと同一のアーキテクチャを用いた、PIM/mでは、マルチPSIのファームウェアで実装された処理系を、新しい要素プロセッサの命令に合せて修正することでよかった。ほかの4種は、クラスタ構造を持つことから、KL1言語処理系は、新たに設計し、試作する必要があった。

4種のPIMは、そのプロセッサの命令の詳細は、KL1の高速実行をめざした設計となっている点では、共通点も多い。また、別々の処理系を、まったく独立に作成すると、作業の重複が多く、非能率的である。

そこで、4種のPIMの命令セットであるPIM機械語の上位に、抽象PIM機械語、PSLを設定し、これにより、KL1言語処理系のうちのPIMに登載される部分を記述した。この抽象PIM機械語、PSLを持つPIMを、仮想PIM(VPIM)と呼んでいる。

KL1言語は、まず、コンパイラにより、WAMレベルの中間言語、KL1-Bに、変換される。この時に、コードの最適化も行われる。KL1-Bの実行の仕方は、各PIMで異なり、ファームウェアのインタプリタで実行するモジュールや、ソフトウェアのランタイムライブラリーによるものなどがある。このKL1-Bを実行する処理系の仕様は、PSLで記述されている。PSLは、C言語に類似した形式で定義されており、KL1-Bのインタプリタの仕様記述言語ともなっている。PSLで記述されたKL1-B言語処理系は、それぞれのPIMのマシン命令セット、PIM機械語に、半自動的に変換できる。

表 2: PIM の各モジュールの特徴

Item	PIM/p	PIM/c	PIM/m	PIM/i	PIM/k
Machine instructions	RISC-type + macro instructions	Horizontal microinstructions	Horizontal microinstructions	RISC-type	RISC-type
Target cycle time	60 nsec	65 nsec	50 nsec	100 nsec	100 nsec
LSI devices	Standard cell	Gate array	Cell base	Standard cell	Custom
Process Technology (line width)	0.96 μm	0.8 μm	0.8 μm	1.2 μm	1.2 μm
Machine configuration	Multicluster connections (8 PEs linked to a shared memory) in a hypercube network	Multicluster connections (8 PEs + CC linked to a shared memory) in a crossbar network	Two-dimensional mesh network connections	Shared memory connections through a parallel cache	Two-level parallel cache connections
Number of PEs connected	512 PEs	256 PEs	256 PEs	16 PEs	16 PEs

このような VPIM 処理系の構成は、将来、KL1 処理系を他機種に移植するときにも有効なものとなっている。

この KL1 処理系の構造は、複雑であり、その性能向上のためには、各部のチューンアップが不可欠である。また、この処理系では、クラスタ内の自動負荷分散機能を組み込むことを試みている。

PIM のハードウェアモジュールの役割は、それぞれ異なり、PIM/m、PIM/p、および、PIM/c が、並列知識処理ソフトウェア研究開発ツールとして予定されており、PIM/k, PIM/i は、アーキテクチャの研究用との位置づけである。

製造規模は、要素プロセッサ数で、PIM/p が 512 台、PIM/m が 384 台、PIM/c が 256 台、　　と計画された。最大構成は、PIM/p が 512 台、PIM/m が 256 台、PIM/c が 256 台の設計である。これらは、1992 年の 6 月ころを目標に、小規模の構成から、大規模な構成へと徐々にテストしながら拡張していく予定である。

処理速度は、モデルごとに異なるが、要素プロセッサ単体で、KL1 に対して、200 から 500KLIPS を目標とした。プロジェクトの目指す目標値は、100MIPS 以上であり、ピーク値では、これを容易に満足する。しかし、実際の応用プログラムでは、コンパイラの最適化の効果や、要素プロセッサの稼働率の良否により、その実効性能が変化する。このため、実際に、ベンチマークプログラムを作成して、計測し評価する必要がある。

5.2.2 並列推論マシンのオペレーティングシステム、PIMOS

PIMOS は、汎用の知識処理や記号処理を目的とする大規模な並列マシンの OS の標準系となることを目指して開発した。そのために、独立して動作する自己充足型の OS として設計されており、KL1 のプログラミング環境を含み、OS として必要な機能を網羅している。

そのハードウェアの資源管理や応用プログラムの実行管理の機能は、マシンのアーキテクチャの細部からは、独立しており、徹底した分散管理方式が採用されている。この結果、制御可能な要素プロセッサ数は、100 万程度まで拡張可能なものとなっている。

PIMOS は、KL1 で記述されており、その管理機構は、KL1 のメタコール機能 (荘園機能) を用いて、実装されている。KL1 の言語処理系が、GC を含む自動メモリ管理機能とデータフロー型の自動同期機構をもっていることから、PIMOS の資源管理や実行管理機能は、これより上のレベルでの管理、制御を行う。

すなわち、資源管理では、ユーザプロセスに与えるメモリ資源やプロセッサ資源の割り当て管理や、入出力機器の管理などを行なう。実行管理では、ユーザのプロセスの実行、停止や、実行順序についての優先度制御などを行なう。また、マルチユーザのサポート、ネットワーク経由のアクセスのサポート機能なども、備えている。

また、PIMOS は、KL1 のプログラミング環境を持っている。これは、コンパイラ、リンカー、デバッガなどの通常のプログラミングツールのほかに、プログラムの負荷分散を最適化するための支援ツールを備えている。これには、パーフォマンスデバッグを支援するための、負荷分散状況をグラフィク表示するツールなどを含んでいる。

5.2.3 知識ベース管理システム

知識ベース管理システム、KBMS は、2 つの階層から構成されている。その下層は、並列データベース管理システム、Kappa-P である。Kappa-P は、非正規型の関係データベースに基づく管理システムであり、自然言語辞書や遺伝子情報のデータベースなど、格納するデータ

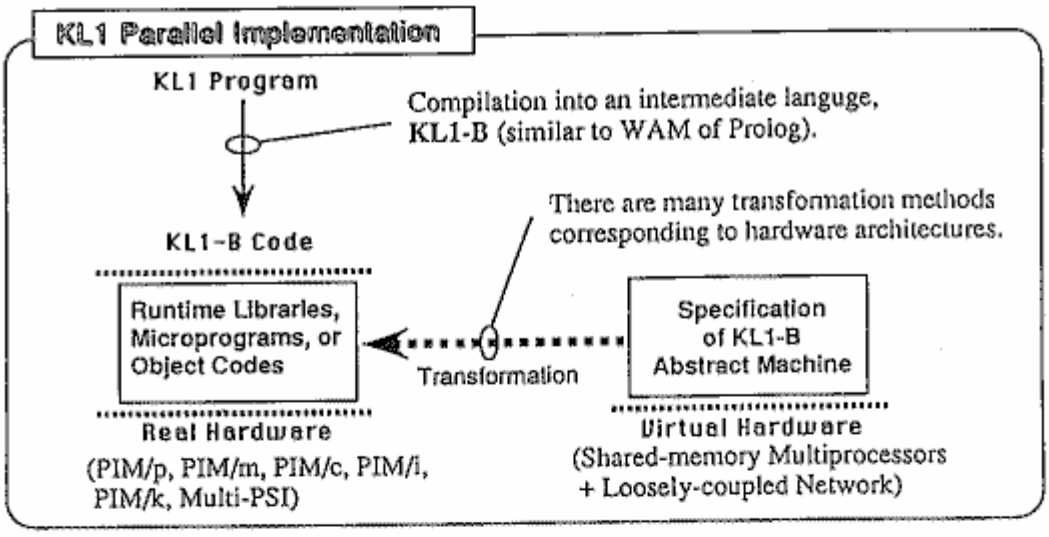
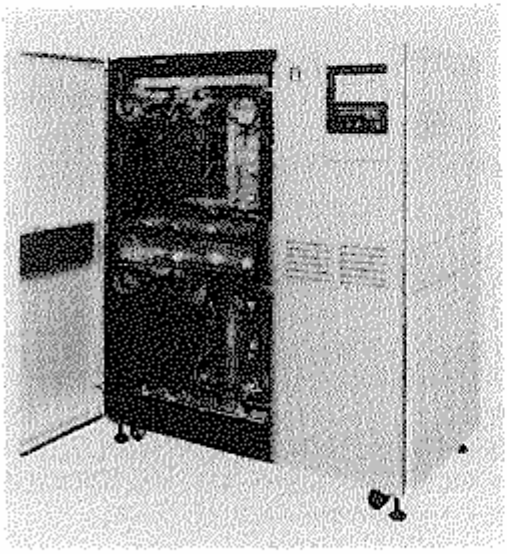
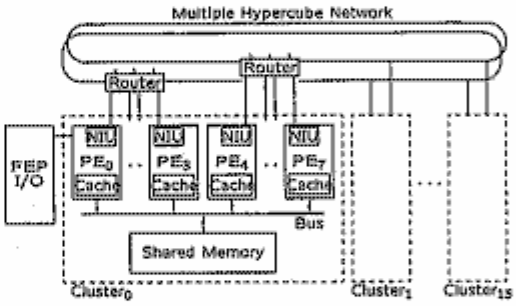


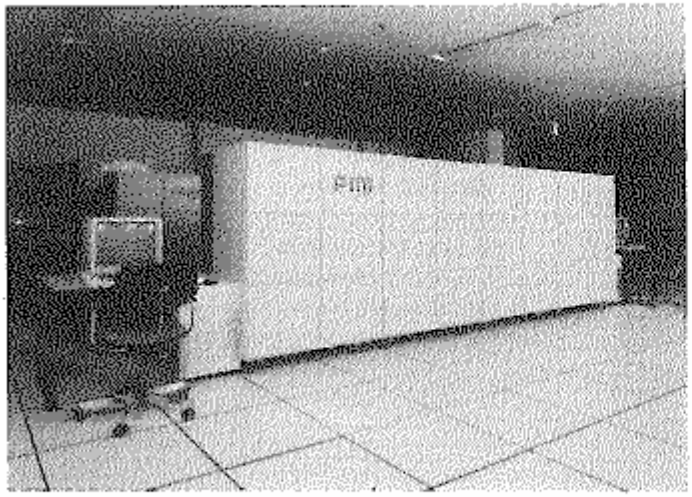
図 5: KL1 言語処理系、 VPIM の構造

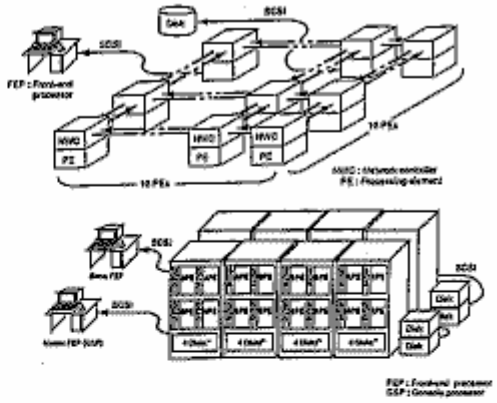




- Machine language: KL1-b
- Architecture of PE and cluster
  - RISC + HLIC(Microprogrammed)
  - Machine cycle: 60ns, Reg.file: 40bits x 32W
  - 4 stage pipeline for RISC inst.
  - Internal Inst. Mem: 50 bits x 8 KW
  - Cache: 64 KB, 256 column, 4 sets, 32B/block
  - Protocol: Write-back, Invalidation
  - Data width: 40 bits/word
  - Shared Memory capacity: 256 MB
- Max. 512 PEs, 8 PE/cluster and 4 clusters/cabinet
- Network:
  - Double hyper-cube (Max 6 dimensions)
  - Max. 20MB/sec in each link

図 6: PIM/p の主な仕様と外観





- Machine language: KL1-b
- Architecture of PE:
  - Microprogram control (64 bits/word x 32 KW)
  - Data width: 40 bits/word
  - Machine cycle: 60ns, Reg.file: 40 bits x 64W
  - 5 stage pipeline
  - Cache: 1 KW for Inst., 4 KW for Data
  - Memory capacity: 16MW x 40 bits (80 MB)
- Max. 256 PEs, 32 PE/cabinet
- Network:
  - 2-dimensional mesh
  - 4.2MB/s x 2 directions/ch

図 7: PIM/m の主な仕様と外観

のサイズや構造が不揃いのものに対しても、能率良く対応できることを特徴としている。

その上層部は、演繹オブジェクト指向データベースに基づく知識ベース管理システムで、知識表現言語、Quixote を提供している。これらは、ともに、KL1 で記述されており、PIMOS 上で動作する。

これらの知識ベース管理システムや、データベース管理システムは、中期の始めから、開発された成果を発展させたものである。

非正規型関係データベース管理システム、Kappa の開発は、自然言語辞書などの人間社会に蓄積された「自然データベース」ともいうべきものの管理を目指して、中期の始めから開始された。非正規型のデータベースは、正規型に比べて、レコード長が不規則で構造がネストするようなデータに対しても有効である。このため、自然言語辞書の他にも、遺伝子データベースや、法律や契約条件など、社会システムの種々のルールの管理にも、適用範囲が広がりつつある。

このデータベース管理システム、Kappa の試作は、まず、PSI マシンの上において、逐次論理型言語の ESP を用いて行われ、中期末に完成した。これが、Kappa-II である。後期より、その並列版の Kappa-P を、KL1 を用いて PIMOS 上で開発し、後期末に、その第一版が完成した。Kappa-P は、多数の要素プロセッサと多数の磁気ディスクを利用した分散処理と並列処理の双方を用いており、この種のデータベース管理システムとしては、世界でもはじめてのものとなっている。

このようなデータベース管理システムの開発と並行して、知識表現言語や知識ベース管理についての研究も、中期の始めから行われた。知識の表現については、いろいろな試みが行なわれたが、最終的には、演繹データベースを基礎にして、それにオブジェクト指向のモジュール化機能を付加したものに発展した。この知識表現言語は、Quixote と呼ばれている。その言語処理系を含む管理システムは、Kappa-P の上に構築されている。このことから、Quixote には、集合演算などの関係データベース処理の演算が組込まれている。

従って、応用に際しては、通常のデータベース作成にあたる、受身的な事実データの集積からはじめ、後から、能動的な規則データを、随時付加していくことが容易に可能となる。 すなわち、データベースシステムから、連続的に、本格的な知識ベースシステムに、拡張していくことが可能となるわけである。このような本格的な知識ベース管理システムの開発は、世界でもあまり例がなく、全体としてどのような、挙動を示すかを見ることは極めて興味深い。

5.2.4 知識プログラミングソフトウェア

このソフトウェアは、人間が用いている種々の知識をより直接的に記述し、利用することを目指して、知識処理の要素技術の開発や理論研究のために試作されたものである。これらの多くは、KL1で記述されており、並列推論システムの応用ソフトウェアともみることも可能である。代表的なものをあげると、次のようなものがある。

1. 制約論理プログラミングシステム

   制約論理プログラミング言語、GDCCの処理系を、KL1で実装している。どのように解くかという手続きを記述するのではなく、なにを解くかという解を得るための条件を記述すると、処理系制約を解消して、自動的に解を求めることができる高級言語である。ロボットの腕の制御プログラム作成支援などへの応用を題材に研究を行っている。

2. 定理証明システム

   モデル生成型の定理証明プログラム、MGTPをKL1で実装した。並列実行による効果が大きく、要素プロセッサの台数に、ほぼ比例した高速化が達成されている。また、この定理証明システムは、法的推論システムのルールベース推論エンジンとして使われており、法律知識の記述の高レベル化を可能にしている。これは、並列推論システムの高い処理能力が応用システムの知的レベルを高めた典型的な例となっている。

3. 自然言語処理システム

   談話理解のための言語情報データベースや、自然言語の入力に必要な構文解析などと、文の生成のためのソフトウェアライブラリーを開発している。また、特定の問題に関する知識を与えて、それを論じる立論システムなどの実験システムが試作されている。

5.2.5 機能実証、および、並列応用実験ソフトウェア

並列推論システムや知識プログラミングソフトウェアの評価のための実際的なベンチマークプログラムの開発を目指すとともに、並列プログラミング技術や、知識の表現や管理、高度な推論機能を探求を目指す。この目的に適した、応用問題を設定して、機能実証や並列応用の実験的なソフトウェアの開発を行っている。

VLSIのCADや電子装置の診断システムなど、工学的な分野のエキスパートシステムのほかに、大規模な知識処理、記号処理の応用問題を求めて、遺伝子情報処理や法律エキスパートシステムなどの異なる分野にも、応用分野を拡大した。この研究の中には、次のような並列記号処理や知識処理の研究テーマがある。

1. VLSIのCADシステム

   VLSIの論理シミュレーション、配線、配置を行なうプログラムを作成し、並列処理のアルゴリズムや負荷分散の研究を行っている。プログラム作成の工夫により、大きな台数効果が得られている。

2. 遺伝子情報処理

   蛋白質の配列解析を行なうシステムや、蛋白質の折り畳みのシミュレーションを行なうプログラムを作成している。配列解析を行うシステムは、解析の速度とともに、その解析の品質が高いことも重要であり、双方の条件において優れたシステムが作成されている。また、負荷分散の面においても、十分な台数効果が得られるまでになっている。

3. 法的推論システム

   法律の条文と、これまでの判例を蓄えた知識ベースを用いて、事件の被告人が、どのような罪に該当するかを推論するシステムであり、人間の社会システムの実際例を扱っている。並列定理証明器、MGTPをルールベース推論のエンジンとして使用することで、法律知識の記述が自然なものとなり、システムの知的レベルを高めている。

4. 電子装置や発電プラントなどの診断システム

   電子装置や発電プラントの機器や、それらの動作、機能、故障の性質などを知識として蓄えておき、故障や異常が発生した時に、その原因を推論して、表示したり、回復処置をとるシステム。並列処理により、複雑な推論を短時間に行うことができる。

5. 囲碁システム

   囲碁の対局を行うシステムで、碁盤上の石の位置を、抽象度を順次増した記述レベルにより表現して、これをもとに、定石を記述して、知識ベースとしている。人間の初心者なみの強さに達している。

以上の研究テーマは、いずれも、記号処理、知識処理の応用であるが、KL1を用いることで、容易に並列プログラムが開発されている。また、CADや遺伝子情報処理などでは、その負荷分散の最適化により、処理速度の大幅な向上が実現されている。

しかし、囲碁のように、高度に知的なシステムでは、また、処理能力の向上が知的レベルを向上させるまでには、至っていないものも多い。これは、まだ、コンピュータが、人間の知恵を知識ベースとして、有効に利用できないでいないことを意味している。

しかし、いろいろな知識言語やを知識記述手法を開発し、知識ベースとしての整理、蓄積、利用の各技術が開発されていくことにより、徐々に、強力な計算能力を,知的レベルの向上に結び付けていくような技術が、確立して行けるものと思われる。

6 並列推論システムの評価と今後の展開

6.1 KL1による汎用並列プログラミング環境

知識処理と記号処理の応用問題をKL1で記述し、実際に、数百台の要素プロセッサからなるPIMで、解くことができた。KL1のよる並列ソフトウェアの生産性は極めて高く、並列OSから、法的推論システムのような応用まで、大規模、複雑、かつ、非定型で不均質の処理が記述できた。これは、従来のプログラミング言語では、実現できなかったことであり、大きなブレークスルーである。この実現は、メモリの自動管理機構とプロセス間の自動同期機構の効果と考えられ、これらが、汎用並列ソフトウェアの生産性向上に不可欠の技術であることが証明されたと考えられる。

また、負荷分散の指定の仕方などの記述手法も、合理的であることが明らかになった。KL1では、問題を解くためのアルゴリズムをプログラムとして記述し、デバッグした後、プラグマと呼ばれる記法により、負荷分散の指定を付加する。この2段階の分離が成功している。

KL1の言語仕様については、試験的な実装では、十分実用的だが、利用者がコンピュータの非専門家となる場合は、より容易な言語仕様のものを準備することが望ましい。これについては、オブジェクト指向言語、AYA開発し、提供することとしているが、この他にも、用途別にいくつか言語の種類があってよいと考えられる。

6.2 PIMOSとKL1の制御管理機能

PIMOSとKL1の機能として実現されている制御管理機能は、効果的な並列ハードウェアの利用を可能としており、プロセッサや主記憶などの資源管理機能や、実行の優先度制御などを含むプログラムの実行管理機能ともに、有効に働いている。一例として挙げれば、ユーザプロセスは、約4000レベルの実行優先度を用いることで、いろいろな探索アルゴリズムやスペキュレイティーブな計算を容易に記述している。プログラミング環境についても、並列ハードウェア上の、負荷分散の状況のユーザへの提示などが、効果的に利用されている。このように、KL1の機能と共に、PIMOS資源管理、実行管理機能も、有効に機能おり、プログラマの負担は大幅に軽減されている。プログラマは、PIMの要素プロセッサ数の増減などのハードウェア仕様の変化による影響を、ほとんど受けずに、応用問題のモデル化とアルゴリズムの記述、さらに、負荷分散の最適化に集中できている。このような結果から判断して、PIMOSとKL1は、汎用並列処理用のOSとしてもっとも進歩したものといえる。

6.3 並列言語機能のハードウェアサポートについての評価

PIMの、要素プロセッサの機能については、PIMのハードウェアの役割を、多数の要素プロセッサを安定に動作させ、ソフトウェア開発実験のプラットフォームにすることを第一の要件と考えたことから、専用のアーキテクチャ的なサポートは、タグアーキテクチャや命令実行パイプラインの強化、共通バス上のキャッシュプロトコルの最適化などに留めた。すなわち、データフローマシンのような命令同期機構(発火機構、マッチングストア)までハードウェアで持つ方法は、採用しなかった。

これは、VLSI技術とCAD技術、特に、ランダムロジックを設計し、VLSI化する技術の進歩予測が、方針決定のポイントであった。現状で見る限り、ランダムロジックを自由にVLSI化する技術は、データフローマシンのハードウェアを、どんどんチップに詰め込めるほどに進んでいない。

データフローマシンの研究においても、最近は、コンパイラによる最適化など、ソフトウェアによる役割分担を増やしており、PIMの試作に対するこのような判断は妥当であったと考えている。

6.4 現在の商用技術との比較

最近は、RISCを用いた汎用マイクロプロセッサの高速化がめざましい。これは、命令形式を単純化してクロック速度を上げる手法をもちいている。現在は、チップ内での演算の並列化など、回路が複雑化しているものの、50MIPSから、100MIPSのチップが開発されている。プロセッサ性能を向上させる手段としては、PIMで用いたような、タグアーキテクチャの利用のように、専用ハードウェアを用いる手法もある。クロック速度が同じなら、専用ハードウェアを用いる方が常に高速だが、量産効果を考慮すると、現時点では、RISCチップの方が有利な場合もある。

KL1のような高級言語は、ノイマン型の言語の実行にくらべて、メモリアクセスの局所性が少なく、ワーキングセットが大きい。また、RISCの場合、コンパイル後のオブジェクトコード長が、タグアーキテクチャのそれに比べ、かなり長くなることが予想される。従って、キャッシュサイズや、主記憶サイズの大きいことが必要となる。

しかし、近い将来このような点が改善され、100MIPS位の性能のRISCチップを用いたMIMD型の並列マシンが入手可能となると、KL1の処理系を移植しても、十分な性能を得ることできると考えられる。KL1の言語処理系は、専用化ハードウェアに対する処理系の依存度が、多くないことから、移植は比較的容易であると考えられる。

現在、市場にでてきた、RISCプロセッサを要素プロセッサとして持つ、MIMD型の並列マシンは、従来の応用の延長として、科学技術計算などの、定型的、規則的な処理の分野に的を絞っていることもあり、KL1やPIMOSのような汎用並列プログラミング環境を持っていない。

KL1やPIMOSは、このようなMIMD型の並列マシ

ンの応用分野を、非定型的、不均質な科学技術計算の分野や、知識処理や記号処理の分野にも、拡大させることが可能であろう。このような移植は、既存の逐次型の汎用OSや、C言語で書かれたソフトウェアとの結合も含めて考えれば実用的な意義は大きいと思われる。

6.5 高級言語によるプログラミングのオーバヘッド

数百台の要素プロセッサを用いる並列処理プログラムの作成において、KL1は、従来の逐次型言語の拡張版とは比較にならないほど、生産性が高い。また、負荷分散の最適化もいろいろ工夫でき、プロセッサの稼働率も、高くできる。

しかし、要素プロセッサ単体ごとの実行速度は、ノイマン型言語を用いて、記憶の自動管理機構やプロセス間の同期機構のない、裸のハードを直接用いる場合と比べると、数倍から、一桁くらい、おそいことがあり得る。これは、高級言語によるプログラミングのオーバヘッドの問題としては、常に議論されるものである。

しかし、これは同時に、コンパイラによる最適化技術の開発が、この点に関しては、まだ、未熟であることを示している。並列論理型言語に限らず、並列言語の最適化は、まだ、端緒についたばかりであり、今後の研究をすすめるべき分野である。

7 おわりに

知識処理や記号処理の応用問題を並列処理により解くためには、非定型、不均質な並列処理を、自由に、効率良く記述できる汎用の並列プログラミング言語と環境が必要である。ノイマン型言語の単純な拡張は、定型的、均質な科学技術計算を行なうのがせいぜいである。

論理型や関数型の高級言語は、記憶管理の自動化機構やプロセス間の同期の自動化機構を持ち、このような汎用並列プログラミング言語としての性質を備えており、有力な候補である。

しかし、これらの機構は、複雑かつ精密な実装を必要とし、実際に大規模並列ハードウェア上に実装することは容易ではなく、かつ、その実行上のオーバヘッド大きいことが予想されていた。しかし、本プロジェクトにおける並列推論システムの開発によって、その実装が行われ、かつまた、そのオーバヘットは、多数のプロセッサによる大規模並列処理の台数効果と、プログラムの生産性の高さにより、十分、補われることが証明された。

また、論理を共通の基盤として、ハードウェア技術、ソフトウェア技術は、もちろんのこと、知識処理の応用分野のモデル化やそこで知識記述にまで、一貫して開発したことにより、プロジェクトの成果が強くむすびついたものとなった。

この結果、並列推論システムの上に構築された、いくつもの並列応用実験ソフトウェアは、知識処理の手法を開拓すると同時に、汎用並列処理のアルゴリズムやパラダイムを生み出した。この新たに生み出された並列ソフトウェア技術があったことで、知識処理や記号処理の問題には十分な並列性があること、アルゴリズムやプログラムのパラダイムが集まってくれば、並列化の効果は、まだまだ、大きくできることが確信された。

このような結果から判断して、本プロジェクトの目標は、並列推論システムに関しては、特に、十二分に達成されたと考えている。また、RISC型の汎用マイクロプロセッサを用いたMIMDマシンの登場は本プロジェクトで開発された、汎用並列処理の成果が、PIMというKL1に最適化したハードウェアを離れて、ひろく広まる可能性を示している。

このような状況を考慮して、現在、KL1やPIMOSなどの主要なソフトウェアの成果を、パブリックドメインソフトウェア、もしくは、フリーソフトウェアとして、広く公開し、国際公共財とできるよう準備中である。このような努力によって、本プロジェクトの成果が、世界中に広まり、さらに高度な知識情報処理の研究の共通基盤となることを念願してやまない。

参考文献

[Uchida 1987] S. Uchida. "Inference Machines in FGCS Project", TR 278, ICOT, 1987.

[Uchida et al. 1988] S. Uchida, K. Taki, K. Nakajima, A. Goto and T. Chikayama, "Research and Development of The Parallel Inference System in The Intermediate Stage of The project", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Nov.28-Dec.2, 1988.

[Goto et al. 1988] A. Goto, M. Sato, K. Nakajima, K. Taki, and A. Matsumoto. " Overview of the Parallel Inference Machine Architecture (PIM)", In Proc. of the International Conference on Fifth Generation Computing Systems 1988, Tokyo, Japan, November 1988.

[Taki 1992] K. Taki, "Parallel Inference Machine, PIM", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Jul.1-5, 1992.

[Chikayama 1984] T. Chikayama, "Unique Features of ESP", In Proc. Int. Conf. on Fifth Generation Computer Systems 1984, ICOT, 1984, pp. 292-298.

[Warren 1983] D.H.D. Warren, "An Abstract Prolog Instruction Set", Technical Note 309, Artificial Intelligence Center, SRI, 1983.

[Clark adn Gregory 1983] Keith L. Clark and Steve Gregory, "Parlog: A parallel logic programming language",

Research Report TR-83-5, Imperial College, March 1983.

[Clark and Gregory 1984] K. L. Clark and S. Gregory, "Notes on Systems Programming in PARLOG", In Proc. Int. Conf. on Fifth Generation Computer Systems 1984, ICOT, 1984, pp. 299-306.

[Shapiro 1983] E. Y. Shapiro, "A subset of Concurrent Prolog and Its Interpreter", TR 003, ICOT, 1987.

[Ueda 1986] K. Ueda. Guarded Horn Clauses, "In Logic Programming", '85, E. Wada (ed.), Lecture Notes in Computer Science 221, Springer-Verlag, 1986, pp.168-179.

[Ueda 1986] K. Ueda, "Introduction to Guarded Horn Clauses", TR 209, ICOT, 1986.

[Chikayama and Kimura 1985] T. Chikayama and Y. Kimura, "Multiple Reference Management in Flat GHC", In Proc. Fourth Int. Conf. on Logic Programming, MIT Press, 1987, pp. 276-293.

[Chikayama el al. 1988] T. Chikayama, H. Sato and T. Miyazaki, "Overview of the Parallel Inference Machine Operating System (PIMOS)", In Proc. Int. Conf. on Fifth Generation Computer Systems 1988, ICOT, 1988, pp. 230-251.

[Chikayama 1992] T. Chikayama, "Operating System PIMOS and Kernel Language KL1", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Jul.1-5, 1992.

[Uchida et al. 1988] S. Uchida,"The Research and Development of Natural Language Processing Systems in the Intermediate Stage of the FGCS Project", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Nov.28-Dec.2, 1988.

[Yokota et al. 1988] K. Yokota, M. Kawamura, and A. Kanaegami, "Overview of the Knowledge Base Management System (KAPPA)", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Nov.28-Dec.2, 1988.

[Yokota and Nishio 1989] K. Yokota and S. Nishio, "Towards Integration of Deductive Databases and Object-Oriented Databases–A Limited Survey", Proc. Advanced Database System Symposium, Kyoto, Dec., 1989.

[Yokota and Yasukawa 1992] K.Yokota and H. Yasukawa, "Towards an Integrated Knowledge-Base Management System", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Jul.1-5, 1992.

[Aiba and Hasegawa 1992] A. Aiba and R. Hasegawa, "Constraint Logic Programming System", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Jul.1-5, 1992.

[Nitta 1992] K. Nitta, K. Taki, and N. Ichiyoshi, "Development of Parallel Application Programs of the Parallel Inference Machine", Proc. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Jul.1-5, 1992.