リレー解説

海外の並列処理研究動向

# スウェーデン SICS の並列処理研究†

ニルソン・マテン††

## 1. はじめに

並列処理は，Swedish Institute of Computer Science (SICS) における主要な研究分野の一つであり，特に，並列プログラミング言語，並列コンピュータアーキテクチャ，並列処理応用に重点をおいて研究が進められている．本稿では，SICS の組織を簡単に紹介したあと，SICS の並列処理関連プロジェクトについて解説する．

SICS はストックホルムの都心から地下鉄で 15 分ぐらいの距離にある Kista（チースタ）* という郊外にある（図-1）．

ストックホルム国際空港アーランダから SICS までは簡単に行くことができる．たとえば，先日，筆者の友人の日本人研究者が初めて SICS に訪ねてくることになったとき，筆者は彼が道順を間違えるのではないかと心配していた．彼の飛行機の到着時刻の少しあと，筆者はアーランダ空港の案内所に電話して，「日本人が道順を尋ねに来ませんでしたか？」と問い合わせた．係員は「いや，こちらは毎日数万人の人が通るんですから」と答えた．しかし筆者が「この男の人は長髪を三つ編みにしていますが」と付け加えると，すぐに「あっ，彼ですか！ はい，道がよく分からなかったようでしたが，チースタ行きバスに乗せましたよ！」という返事をもらえた．筆者がチースタのバス停へ迎えに行くと到着したバスから友人が降りてきた．

## 2. SICS の概要

SICS は 1985 年に設立された研究組織であり，その研究活動が基礎研究 (Basic Research) と契約研究 (Research Contracts) という二つのプログラムに分けられている．年間予算はおおよそ 10 億円である．基礎研究プログラムの予算のうち，4 割は国費であり，残りの 6 割はスウェーデンの主要企業が出資している．一方，契約研究の契約先は，予算ベースで 5 割は国であり，5 割は個別の企業や Esprit*，Eureka**，Drive*** のようなヨーロッパの協同研究プロジェクトである．約 60 人が SICS に勤めていて，そのうち 50 人が研究者である．研究者の中には博士課程の大学院生も多く含まれている．

SICS の役割は，コンピュータ・サイエンスの研究を行い，その成果をおもにスウェーデンの企業が利用できるようにすることである．ただし，研究レポートは通常オープンであり，広く公開されている．また，SICS のもう一つの重要な仕事は大学院生の教育である．

国際協力や共同研究にも重点をおいている．現在，SICS が参加している国際プロジェクトは，6 つの Esprit 関連プロジェクト，Eureka プロジェクト Prometheus, Drive プログラムである．共同研究は日本の ICOT と英国ケンブリッジの SRI と協定している．

SICS は次の三研究室に分かれている．LPS (Logic Programming and Parallel Systems) 研究室では論理型プログラミングと並列処理が，DS (Distributed Systems) 研究室では分散システムが，KBS (Knowledge-Based Systems) 研究室では知識ベースシステムが研究されている．

## 3. SICS の並列処理研究

並列処理研究の中心は LPS 研究室であり，並列処理に関するプロジェクトとしては：Andorra 並列言語，DDM 並列コンピュータ，実時間処理と応用，Muse 並列 Prolog，の 4 つがある（図-2）．

† Parallel Processing Research at SICS by NILSSON Martin
†† Swedish Institute of Computer Science

* スウェーデン語では，「kista」という別の言葉は「棺」を意味するが，その場合は「チスタ」と発音する．

* EG の precompetitive 協同研究プログラム
** (西) ヨーロッパ全体の協同研究プログラム
*** EG の道路運送についての協同研究プログラム

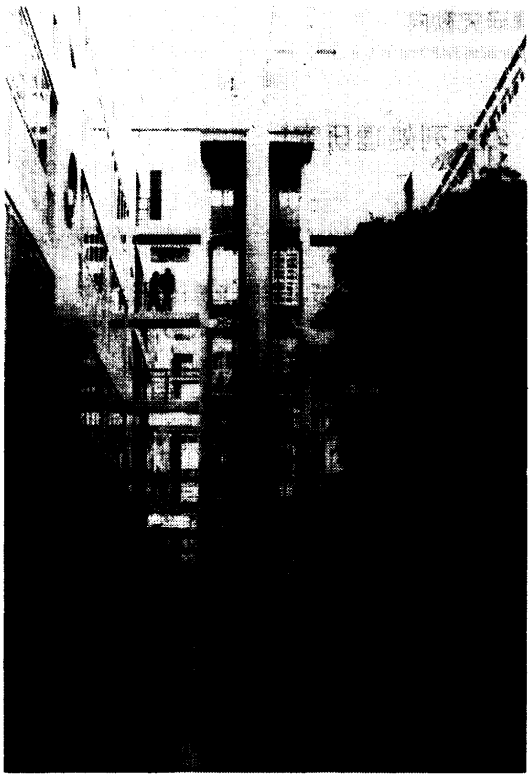


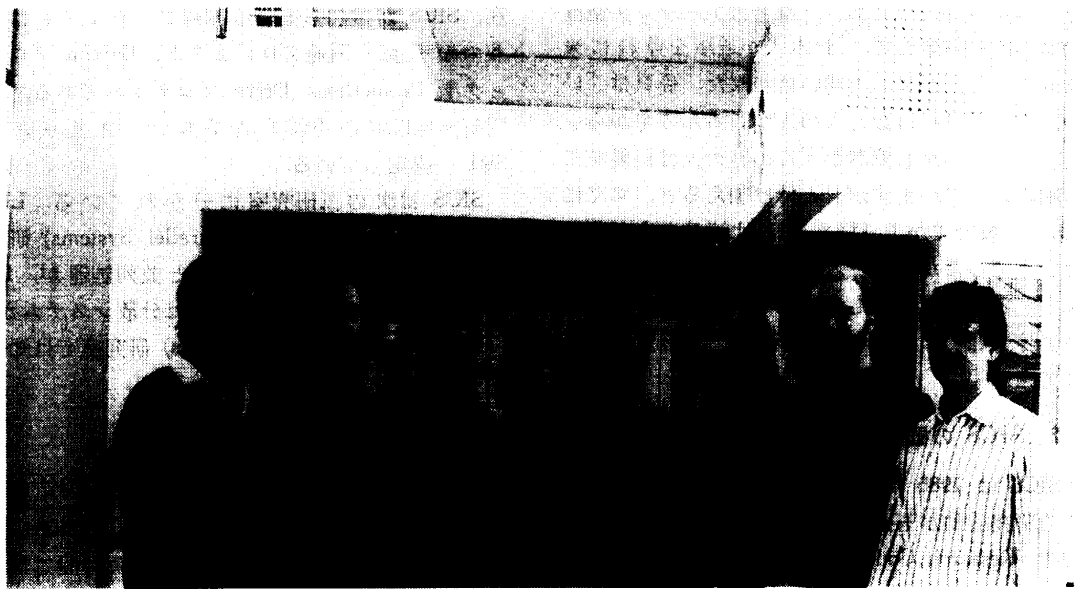
図-1 SICS がある建物の内部

Andorra プロジェクトの目標は，Prolog と，GHC のような committed-choice 言語とを組み合わせた新しい言語を開発することである．DDM プロジェクトは，Cache-only-memory の汎用並列計算機アーキテクチャの開発を目指している．実時間処理と応用のプロジェクトでは，ロボットなどの実時間システム向きの並列論理型プログラミングが研究されている．Muse プロジェクトでは，多種のマルチプロセッサ・アーキテクチャの上で OR 並列 Prolog プログラムを実行する方法が検討されている．

## 4. Andorra 言語

Andorra は，SICS において研究されている代表的な並列言語であり，その名前は並列論理型言語の AND 並列と OR 並列の両方を意味する．Andorra Kernel Language (AKL) は，Prolog のような探索指向の非決定的な言語と GHC のようなプロセス指向の committed-choice 言語の両者を組み合わせた言語である．AKL は，次のポイントを考えながらデザインされている：

• AKL は，Prolog と GHC の両方のプログラミング・パラダイムを提供すること．ほとんどの



左から Khayri Ali, Erik Hagersten, Anders Landin, Martin Nilsson, Seif Haridi, Peter Magnusson, Sverker Jansson, Roland Karlsson.

図-2 SICS の並列処理研究者

GHC と Prolog プログラムを AKL に書き換え可能であること.

● AKL は，単一プロセッサ・アーキテクチャ上に効率的に実装可能であること．ここで「効率的」とは，同様の手法を用いる Prolog や GHC の処理系と同程度の効率を意味する.

● AKL 言語とその計算モデルは形式的に定義されること.

● AKL は制約を基礎とすること.

● AKL をマルチプロセッサ・アーキテクチャ上に実装することにより主要な形態の並列性が抽出できること.

AKL は GHC とよく似ているため，AKL の GHC サブセットで書かれたプログラムは，GHC プログラムとほとんど同じように実行される．ただし，次のような点で両者は異なる.

AKL では，GHC のコミットオペレータ（|）に加えて，カットオペレータ（!）とウェイトオペレータ（:）がある．カットは Prolog のそれと似ているが，AKL のカットは「静か (quiet)」で，カットの左側では外部変数への代入ができない*．このため，GHC と同じように出力は節のボディ部で行われなければならない.

「静か」なカットへの制限は，カットの効果がゴールの並列実行順序に依存しないようにするために必要である．ウェイトオペレータは，そのウェイトオペレータの属する節の呼び出し側のゴールに対して適用可能かもしれない他の節がすべて失敗するまでサスペンドする．ウェイトの節も Prolog と同じように非決定的 (don't-know-nondeterminism) に実行できる.

現在，AKL から抽象機械語へのコンパイラがある．この機械語はCで書かれたインタプリタによって解釈実行される．単一プロセッサでは，決定的なコードの速度が SICStus Prolog のコードより4倍ぐらい遅い．AKL はまだマルチプロセッサ上には実装されていない．AKL の詳細は，参考文献の [HJ 90] や [JH 91] にある.

## 5. The Data Diffusion Machine

Data Diffusion Machine プロジェクトは並列（大規模）アーキテクチャと並列（大規模）OS の二つに主眼をおいて進められている．短期的な目標は，汎用 OS を実行する Data Diffusion Machine (DDM) という新しいアーキテクチャのプロトタイプを試作することである（**図-3**）．長期的には，スウェーデンの企業が知識ベースを作り上げるために必要な並列処理の技術を確立することである.

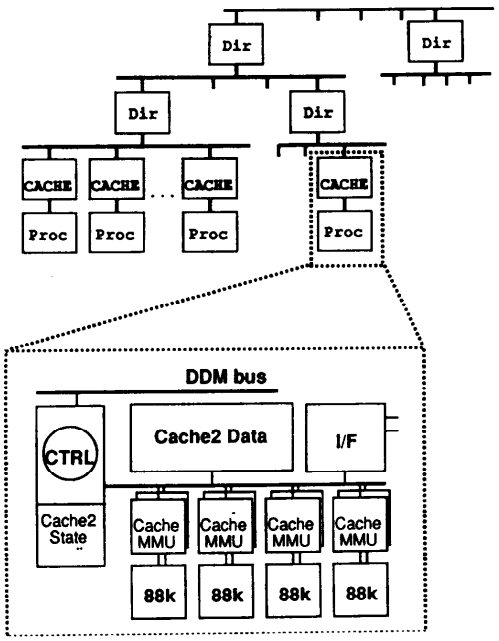



**図-3** DDM のプロトタイプアーキテクチャ

DDM は，Cache-Only Memory Architectures (COMA) という新しいクラスに属する．システム内の全てのメモリはプロセッサごとにある大きい一貫性キャッシュ (coherent cache) を階層ネットワークで結合することによって構成される．データは，どのプロセッサにも属さず，必要な場所に移動されたりコピーされたりする．近日中に，Motorola 88100 プロセッサを用いたプロトタイプが完成する予定である．このプロトタイプの設計は，16 プロセッサからなるレジスタトランスファレベル (RTL) レベル・シミュレータ上でCプログラムを実行* することにより検証されている.

実際の並列応用プログラムを用いた実験をするために大型システム（128 台以上）を扱うことができる execution-driven シミュレータも開発されている．これを用いて OR 並列 Prolog や数値計

* 代入しようとするとサスペンドする.

* 一種のトレース命令を生成するようにCコンパイラが改造されている．トレース命令は実行されるたびにシミュレータを呼び出し，実システムの処理時間を計算する.

算において良好なベンチマーク結果が得られている.

アーキテクチャ面の研究は一貫性 (coherent) 共有メモリに焦点を当てて進められている. 新しいアイデアや理論は, とりあえずシミュレーション環境で検討されてから, DDM プロトタイプに導入されている.

大規模オペレーティングシステムの研究は CMU の Mach OS を始点としており, 現在, 生の Mach を DDM プロトタイプへ移植しているところである. 将来的には, どのようにして OS をスケーラブルにするかが研究課題である. ここで「スケーラブル」とは, マルチプロセッサシステムのプロセッサ台数を増やしても効率的に動くことを意味する. 面白いテーマとしては, タスク移動, 並列入出力, 動的なワークスペース, 並列プリミティブ, 同期, TLB の一貫性などがある. DDM の詳細は, 参考文献の [HLH 91] や [WH 88] にある.

## 6. 実時間処理と応用

今後は, 長時間独立動作のできる自律的な実時間システムの重要性が増すと思われる. 応用としては, 毒素や放射能で汚染されている場所の検分や掃除, 近づきにくいところや危険な環境の探索, 消防, 宇宙探査, 悪天候時の救援活動などが考えられる.

このようなシステムには, オペレータがいないため, 全てのレベルにおける制御を内部にもつ必要がある. センサとアクチュエータ (サーボ) といった低いレベルから, 戦略計画をたてたり制約処理をするような抽象的な仕事を管理する高いレベルまで, 制御を統合しなければならない. このようなシステムでは, 信頼性, モジュラリティ, 実時間パーフォーマンスとセンサからの流れ込む大量のデータの処理の応答など, 厳しい要求があるため, 分散並列プロセッサが必須となる. 同時に, システムの重量と消費電力はできるだけ小さくならなければならない.

実時間システムとその応用に関するプロジェクトでは, 高級並列 committed-choice 言語と分散制御システムの組み合わせが解決策の一つとなると考えている.

他の言語と比較して, GHC や AKL のような committed-choice 言語は, 生まれつき, プロセス概念, 記号処理能力, および豊かな表現能力をもっており, 効率的な処理系もエレガントに実装できる. また, これらの言語では, メッセージパッシングで通信するオブジェクトとしてセンサとアクチュエータを表現することも容易である.

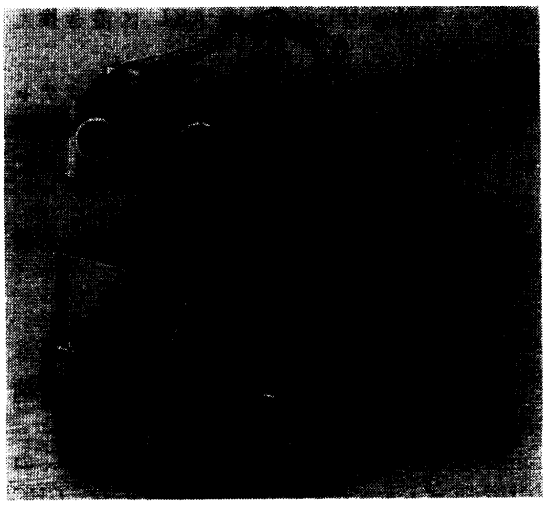


図-4 ターミネータ-I

このアプローチを確かめるために, ターミネータ-I と呼ぶ, センサとアクチュエータをもつ小さいリモコン車のようなロボットが試作された (図-4). このロボットを用いて, 並列言語のどのような機能が実時間処理応用に必要となるかの研究が行われた.

現在は, ターミネータ-I の発展形としてターミネータ-II と呼ぶ次バージョンの開発がスタートしている (図-5). ターミネータ-II には直径 30 cm ぐらいの可動プラットフォームがあり, その台の上には, 6 自由度の腕と制御用のノート・コンピュータ (386 sx) が乗っている. ロボットの重さは 40 kg で, 電池を一回充電すれば, おおよそ一時間ぐらいの動作時間が得られる. 実時間処理と応用のプロジェクトについての詳細は, 参考文献の [N 90] と [N 91] にある.

## 7. 並列 Prolog システム Muse

近年, マルチプロセッサが多く商用化されだした. しかし, それらがもつプロセッサ能力を効率的に利用することはきわめて難しい. プログラマには, できれば, 並列プロセッサを逐次プロセッサと同じようにブラック・ボックスとして考えら

れ，システムが自動的にプログラムを分析して，並列実行可能な処理を並列に実行してくれることが望ましい．さらに，並列プロセッサ上での実行は一台のプロセッサの実行より速くならなければならない．このような要求に応えることは，記号処理や知識ベースの応用においては難しいが，Muse システムでは，Prolog 応用プログラムに関して上の目標に近づいてきている．

図-5　ターミネータ-II

Muse は，Prolog 言語のフルセットの OR 並列処理系である．インクリメンタルなスタック・コピー方式に基づいており，現在，Sequent Symmetry, Sun Galaxy, BBN Butterfly などの共有メモリマルチプロセッサ上に実装されている．逐次版の SICStus Prolog を OR 並列実行用に改造したものであるが，並列実行のオーバヘッドはほんのわずかである．このため，多くのプログラムにおいて，並列処理による速度向上比は，プロセッサの数に近い．

Muse の応用の一つは，Sequent Symmetry や BBN Butterfly I および II の上で走る大きな知識ベースシステムである（**表-1**, **図-6**）．性能比は，単一プロセッサ上の SICStus Prolog に対するものである．評価に用いた知識ベースシステムは，企業と SICS が一緒に開発した回路基板デザインルールチェッカであり，並列実行をまったく考慮せずに書かれている．Muse の詳細は，参考文献の [KK 90] や [KK 91] にある．

**表-1**　Muse の性能

| コンピュータ | プロセッサの数 | スピードアップ |
|---|---|---|
| Symmetry | 26 | 25.6 |
| Butterfly I | 77 | 50.7 |
| Butterfly II | 38 | 32.0 |

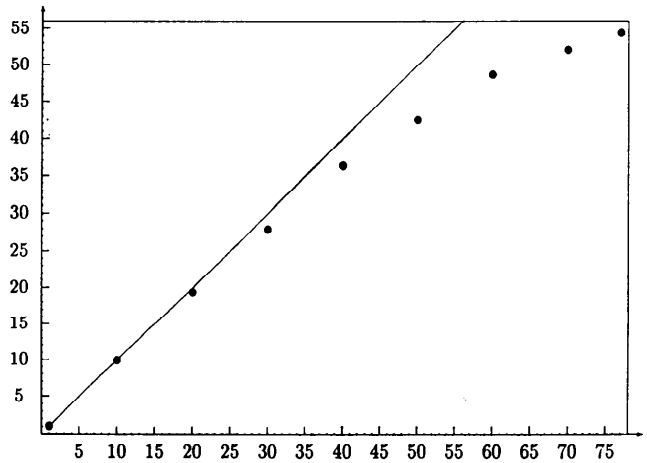



**図-6**　Muse のプロセッサ台数(横)/スピードアップ(縦)関係
システムは Butterfly I (GP 1000)

## 8. ま と め

SICS では今後も並列処理研究を続け，さらに強化していくつもりである．特に，ニューラルネット研究の拡大を現在検討中である．また，国際協力は研究を成功させる上で重要な条件の一つであり，SICS は訪問者や研究成果の交換を歓迎している．SICS では，その創立時点から多くの日本人研究者に訪問していただくことにより，有益な研究交換ができてきた．この伝統が将来も続くことを願っている．

おわりに，筆者が日本にいた間にお目にかかった皆さま，特に東京大学の田中英彦教授に深謝する．筆者の下手な日本語を直してくださったNTTの後藤厚宏博士に感謝する．

## 参 考 文 献

[HLH 91] Hagersten, E., Landin, A. and Haridi, S.: DDM—A Cache-Only Memory Architecture, SICS Research Report R 91 : 19. Forthcoming in IEEE Computer.

[HJ 90] Haridi, S. and Jansson, S.: Kernel Andorra Prolog and Its Computational Model, In Proc. of the 7 th Int. Conf. on Logic Programming, ICLP '90. MIT Press (1990).

[JH 91] Jansson, S. and Haridi, S.: Programming Paradigms of the Andorra Kernel Language, SICS Research Report R 91 : 08. ISSN 0283-3638 1991. In Proc. of the 1991 Int. Logic Programming Symp., ILPS '91. MIT Press (1991).

[KK 90] Ali, K. A. M., and Karlsson, R.: The Muse Approach to OR-parallel Prolog, Int. J. of Parallel Programming, Vol. 19, No. 2, pp. 129-162. (Apr. 1990)

[KK 91] Ali, K. A. M. and Karlsson, R.: Scheduling OR-parallelism in Muse, In Proc. of the 8th Int. Conf. on Logic Programming, ICLP '91, MIT Press, pp. 807-821 (1991).

[N 90] Nilsson, M.: Mobile Robot Control with Concurrent Logic Languages, In Distante, F. (ed.): Euromicro '90 Workshop on Real-Time. IEEE Computer Press pp. 42-46 (June 1990).

[N 91] Nilsson, M.: Parallel Dynamic Map Construction and Navigation in Real-Time for Autonomous Robots, In Proc. Japanese Symp. on Parallel Processing, JSPP '91, Kobe, pp. 397-404 (May 1991).

[WH 88] Warren, D. H. D. and Haridi, S.: Data Diffusion Machine—A Scalable Shared Virtual Memory Multiprocessor, In Proc. Int. Conf. on Fifth Generation Computer Systems 1988, Ohmsha Tokyo, pp. 943-952 (1988).

(平成4年1月 16 日受付)

**Nilsson Martin** (正会員)

1959 年生．1983 年スウェーデン王立工科大学コンピュータサイエンス専攻修士課程修了．1989年東京大学大学院工学系研究科情報工学専攻博士課程修了．同年 SICS (Swedish Institute of Computer Science) に勤務．並列処理，実時間処理，制約，ロボットなどに興味を持つ．ACM，日本ソフトウェア科学会各会員．