招待論文

並列連想プロセッサIXM2

正　員　樋口　哲也[†]　　非会員　古谷　立美[†]
非会員　半田　剣一[†]　　非会員　高橋　直人[††]

IXM2: A Parallel Associative Processor

Tetsuya HIGUCHI[†], *Member*, Tatsumi FURUYA[†], Kenichi HANDA[†]
and Naoto TAKAHASHI[††], *Nonmembers*

あらまし　多数の計算機の集合ではなく，大容量連想記憶に基づいた新しい超並列処理のアプローチとして，256 K語の大容量連想記憶をもつ並列連想プロセッサIXM2のアーキテクチャについて述べている．IXM2は人工知能の知識表現形式である意味ネットワークの処理を主に設計され，最大で64 Kノードの意味ネットワークを並列処理できる．連想処理，交差演算，マーカ伝搬といった意味ネットワークの基本処理において，IXM2はSIMD素子としての連想メモリの並列性を生かし，データ数によらない一定時間の処理を達成できるため，大規模意味ネットワーク処理において生ずる計算量の急激な増大を抑制することができる．またIXM2を構成する73台のPEは完全結合に基づく接続方式で結合され，マーカ伝搬処理の高速化を図っている．他の超並列マシンとの比較も交え，基本処理や自然言語処理等の応用問題で評価することにより，本アプローチの有効性を確認した．

キーワード　連想メモリ，意味ネットワーク，超並列処理，人工知能

1. まえがき

意味ネットワークやフレームなど，ネットワーク型のデータによる知識表現形式は，自然言語処理や知識ベースなどの人工知能の諸分野において従来重要な位置を占めている．しかし，ネットワークが大きくなった場合に計算量が急激に増大するため，既存の逐次型計算機では取扱いが困難であり，それが今日まで実用規模の応用が開発されにくい一因となっていた．オブジェクト指向型など今後もネットワーク型データの利用は更に増えることが予想されるが，このような応用に対応できる計算機アーキテクチャの研究は，1978年のNETLマシン[(2)]の提案以後，十分に行われているとは言いがたい．

この計算量の問題解決のためには，単に計算機単体の性能の向上を図るのではなく，処理アルゴリズムのオーダを低減することが本質的である．そのためのアプローチとして最も適切と考えられるものに超並列処理がある．

超並列処理といった場合に一般に想起されるのは，MPP[(1)]やコネクションマシン[(6)]など，1ビットPEを多数集めた計算機集合体であるが，筆者らはネットワーク型データに対する処理を検討した結果，大容量連想メモリの有する超並列性に着目した．連想メモリは，意味ネットワーク処理に頻出する三つの基本処理（連想，マーカ伝搬，交差演算）において極めて効果的であり，新しい超並列処理のあり方を示唆している．しかし，これまでは大容量の連想プロセッサ自体がなく，またこの種の人工知能処理に適用しようという試みもなかった．

このため，大容量連想メモリに基づく超並列処理の有効性を示すことを目的として，筆者らは256 K語の大容量連想メモリを有する並列連想プロセッサIXM2を開発し[(5)]，その上で推論システムや自然言語処理の実験を行った．この結果，大容量連想メモリの導入によって，意味ネットワークをコンパクトに表現しながらも，その並列処理素子としての特徴を生かして高い処理能力を達成できることを確認した．また，その評価の過

† 電子技術総合研究所，つくば市
Electrotechnical Laboratory, Tsukuba-shi, 305 Japan
†† 筑波大学工学研究科，つくば市
Tsukuba University, Tsukuba-shi, 305 Japan

程において他のアーキテクチャとの比較を行い，意味ネットワークの処理における問題の性質とアーキテクチャ選択の関係についての考察も行った．

2. 意味ネットワーク処理

ここでは意味ネットワーク処理の基本的な性質を示し，並列処理の必要性について述べる．

意味ネットワークはノードとリンクを用いた図式的な知識表現形式であり，一般にノードは概念，リンクは概念間の関係を表すことが多い．この表現形式の最も重要な特徴は，isaリンクで結ばれた上位概念と下位概念との間で，上位概念の属性が下位概念へ継承されることである．これをインヘリタンスと呼ぶが，これを利用することによって，知識ベースの開発において記述量の削減を図ることが可能になる．

意味ネットワーク処理では，連想，マーカ伝搬，集合演算の三つが基本的で使用頻度が高いが，これらはマーカビットを用いて効率的に扱うことができる．マーカビットは処理結果を保持する1ビットのフラグであり，各ノードごとに複数個のマーカビットを用意する．

図1の意味ネットワークを例に，上述の基本処理と，そのマーカビットによる実現について以下に述べる．図1の意味ネットワークは，ノードA，およびBの二つの集合に属するノード群を示したものであるが，これら二つの集合の共通要素を求める場合を考える．まず連想処理によりノードAを求め，ノードAの中の，例えばマーカビット1番をセットする．次にAに属するメンバーを得るためにマーカ伝搬を行う．マーカ伝搬は，あるノードから特定のリンクをたどり，途中通過するノードごとにマーカビットをセットする処理である．この場合，Aからisaリンクを逆たどりし，到着するノードC，D，Eにマーカビット1番をセットする．同様にして，集合Bのメンバも，連想処理，マーカ伝搬によって求め，ノードD，Eにおいてマーカビット2番をセットする．

求めたいのは集合の共通要素であり，これは交差演算(set intersection)によって求めることができる．つまり，この場合，マーカビット1番と2番が同時にセットされているノードを探せばよい．この結果，ノードD，Eが解となる．

扱うネットワークが大きい場合に，これらの基本処理を逐次計算機上で行うことは，処理効率が悪く，これが実用的な規模の意味ネットワーク応用が十分にこれまで開発されなかった原因の一つとなっている．特に交差演算は人工知能の種々の応用において有用性が高いため，その回数を低減するために種々の工夫がなされているが，根本的な解決にはなっていない．

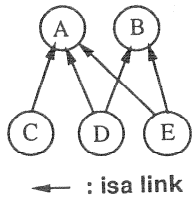


図1 意味ネットワーク
Fig. 1 A semantic network.

意味ネットワーク処理におけるこのような逐次処理の問題点を解決するために，1979年にS. Fahlmanにより，NETLシステムが提案された．NETLでは意味ネットワークの各ノード，各リンクごとに小規模のPEを割り当てることで，連想処理と交差演算においてデータ数によらない常に一定時間の処理を実現し，またマーカ伝搬ではその処理時間がたどる木の深さに比例した時間で済むようにした．NETLは実際には作られなかったが，その影響を受け，MITでコネクションマシンが提案されることとなった．

3. IXM2のアーキテクチャ

3.1 意味ネットワーク処理向きアーキテクチャについての考察

1ノードにつき1PEを割り当てるような，NETLに代表される意味ネットワークの直接的なマッピングは，連想処理，および交差演算において$O(1)$の処理を実現できるため，MPPやコネクションマシンCM-2といった超並列のSIMD (Single Instruction Multiple Data stream) マシンは意味ネットワーク処理用アーキテクチャの有力な候補である．

しかしながら，もう一つの基本処理であるマーカ伝搬に対しては必ずしも有効な解決とはなり得ない．これには二つの理由が考えられる．

第1に，CM-2といった超並列マシンは計算機間の接続が転送能力の低いシリアルリンクを採用しているため，マーカ伝搬の処理時間自体が遅い．超並列マシンの適用は計算機間の交信が，局所的，かつ同時に多数の箇所で起きることが前提となっており，その点で意味ネットワーク処理はこの前提を満たしにくい．

第2は，他の多くのノードとの接続をもつノード(多ファンアウトのノード)の存在である．このようなノードは上位の概念に対応するノードに多く見られる．仮

に N 個のノードとの接続をもっているとし，もし，このようなノードに一つの PE が割り当てられ，このノードからマーカ伝搬が行われるとする場合，CM-2 などでは 1 ビット PE がマーカ伝搬を N 回逐次的に繰り返さなくてはならない．これがシリアル転送リンクの遅さとあいまって，マーカ伝搬を行う PE が同時に多数存在しない限り処理のボトルネックとなる．NETL の提案ではマーカ伝搬は常に木の深さに比例した時間でできるとしていたが，超並列マシン上でのマーカ伝搬の実現は単純にはそうならない．

3.2 IXM2 の設計方針

2.で示した意味ネットワーク処理の性質と，現在の超並列マシンの問題点を考慮した結果，IXM 2 の設計にあたっては次の方針をとることとした．

（a） 大容量連想メモリの採用　　意味ネットワークのノードを連想メモリ上で表現，処理することにより，連想処理および交差演算が，超並列マシンと全く同様に，データ数によらない一定時間で処理可能となる．しかも連想メモリはビット並列処理が行えるため，それらの処理では超並列マシンにおける 1 ビット PE の処理能力よりも高い．加えて，超並列マシンでの 1 ノード/1 PE による表現に比べ，連想メモリ上で意味ネットワークがよりコンパクトに表現・格納できる．そのような連想メモリを備えた PE を中規模台数用意し，処理すべき意味ネットワークをサブネットワークに分割し，各サブネットを各 PE の連想メモリに割り付ける．従って多ノード/1 PE の割付けとなる．またマーカ伝搬や算術演算において 1 ビット PE では並列度が低い場合にシステム全体の性能を低く抑えてしまうため，IXM2 では PE として高機能のマイクロプロセッサを用いる．

（b） 完全結合に基づく計算機間接続　　意味ネットワークを多ノード/1 PE で割り付ける場合，マーカ伝搬は PE 内，若しくは PE 間で起きる．マーカ伝搬が PE 間で起きる場合，メッセージ交信によって実行されるが，メッセージが途中の PE によって中継されるとマーカ伝搬が著しく遅くなる．このため PE 間のマーカ伝搬はできる限り直結する二つの PE 間で行われるべきである．そこで IXM 2 では完全結合に基づく計算機間接続方式をとり，またこの接続方式を生かすような，サブネットワークへの分割・配置プログラム（アロケータ）を開発した．

（c） 並列マーカ伝搬　　ファンアウトの多いノードからのマーカ伝搬の問題を解決するために，IXM2 ではファンアウト数によらず，常に一定時間でマーカ伝搬を実現する方式を開発した．これは連想メモリの検索，並列書込み機能を生かしたものである．

表 1　連想プロセッサと通信プロセッサの各構成要素

連想プロセッサ（AP）	T800 Transputer (17.5 MHz) 4 KB オンチップ RAM (57 ns) 32K X 32 bit SRAM (230 ns) シリアル転送リンク 4 本 (2.4 MB/sec) リンクアダプタ 4 個 (IMS C012) 4096 X 40 bit 連想メモリ (375 ns)
通信プロセッサ（NP）	T800 Transputer (17.5 MHz) 32K X 32 bit SRAM (230 ns) リンクアダプタ 16 個（放送機能） 2048 X 40 bit 連想メモリ (375 ns)

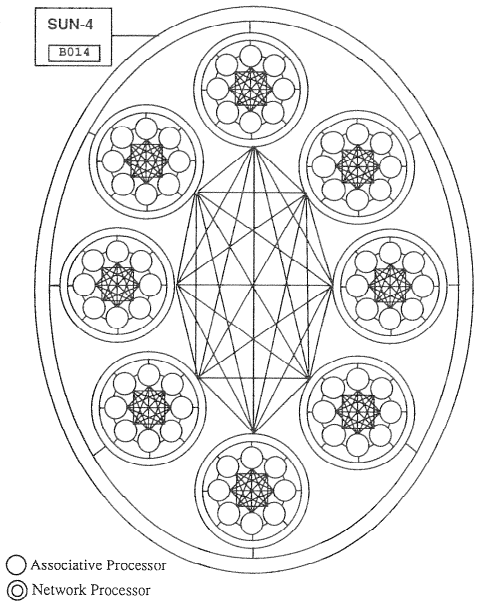


図 2　IXM 2 接続構成図
Fig. 2　Interconnection structure of IXM2.

3.3 IXM2 の全体構成

IXM2 は，64 台の連想プロセッサ (Associative Processor ; 以下 AP) と 9 台の通信プロセッサ (Network Precessor ; 以下 NP) よりなり，ホストプロセッサである SUN-4/260 の制御下で動作する．AP と NP の各構成を表 1 に示す．使用した連想メモリは NTT の 20 K ビットチップで，通常の検索機能に加え，並列書込みやガーベージコレクションなど全部で 26 種の命令を備える[10]．

IXM2 の構成は階層的になっている．つまり，図 2 に示すように，8 台の AP と 1 台の NP が一つのモジュールを形成し，各モジュール内で 8 台の AP が完全結合される．そして，IXM2 にはそのような八つのモジュー

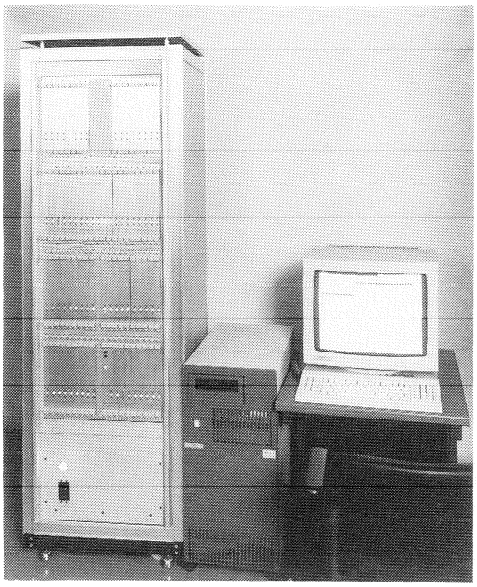


図3 IXM2の概観
Fig. 3 IXM2 overview.

表2 平均メッセージパス長

No. of PE	IXM2	hypercube	torus
4	1	1.33	1.33
8	1	1.71	—
16	2.06	2.13	2.13
32	2.54	2.61	—
36	—	—	3.08
64	2.77	3.04	4.03

ルがあるが，これらの間も完全結合されている．図3にIXM2とホストの外観を示す．

マーカ伝搬が計算機間で起きる場合，そのためのメッセージ交信は直結する2台のAP間で行われるのが一番高速であるが，64台のAP間のすべてに完全結合を実現することは困難である．しかしより少ないAP構成では可能であり，IXM2の場合，実装上の理由から8台のAP間の完全結合を採用している．

このような接続形式をとるのは，メッセージ路長(message path distance)を極力1に保つことによりマーカ伝搬の高速化をとるためである．その前提となっているのは意味ネットワークの性質である．つまり，Fahlmanは意味ネットワークの性質として，大きな意味ネットワークはサブネットワーク内は結合は密でも，サブネットワーク間は結合が比較的疎であるように意味ネットワークの分割がしやすいことを指摘している[2]．IXM2の接続方式はこれに沿うものであり，8台のAPが完全結合されたモジュールにサブネットワークを割り付けることにより，通信の局所性を完全結合された8台のAP間の中にとどめやすい．

最悪のケースとして，もしこのような局所性が意味ネットワークに見出せず，64台のAPの間でランダムにマーカ伝搬が起きる場合においても，表2に示すようにこの接続方式はハイパキューブやトーラスに比べ，平均メッセージ路長を低く保つことができる．

4. 連想メモリ上での並列処理

4.1 連想メモリ上でのデータ表現

IXM2での意味ネットワークのデータ表現はノードを単位としている．図4は図1の意味ネットワークを表現した例である．一つのノードについての情報は，連想メモリと通常のメモリの二つに分けて展開する．

まず連想メモリ上に置く情報について述べる．連想メモリは貴重な並列処理資源である．このため，情報は連想メモリの利用効率を高めるために簡潔にコード化されており，マーカビット領域(28ビット)，リンク領域(8ビット)，リテラル(16ビット)，並列マーカ伝搬情報(22ビット，PIDと略す)の四つの領域からなる．これらのビットはすべて並列処理の対象データとなる．このうちマーカビットは1ビット単位での書込みができることが必要である．そのようなビットは連想メモリの1語につき8ビットしかないので，28ビットのマーカビット領域を確保するために4語が必要である．このため，1ノードの表現に連想メモリの4語を用いている．従ってIXM2全体では連想メモリが256K語であるから，64Kノードの意味ネットワークを並列に処理することが可能となる．

マーカビット領域は1ビットのフラグの集まりであり，そのノードについての中間処理結果などを保持する，一種のレジスタ的役割を担う．

リンク領域は，このノードに接続する基本リンクの種類とその有無を示すものである．ここでいう基本リンクとは，IXM2がサポートする知識表現言語IXL[4]のもつ基本リンクに対応するもので，isa，instance_of(iso)，destination(des)，source(soc)の四つである．但し，方向も区別するのでリンク領域には計8ビットを用意し，各基本リンクの有無を示すようにしている．逆方向のリンクを示すのにrを付けることにすれば，このリンク領域の8ビットは，risa，isa，riso，iso，rdes，des，rsoc，socに対応する．例えば，図4においてノードCはノードAをisaリンクによって指しているので，連想メモリ上のノードCについてのリン

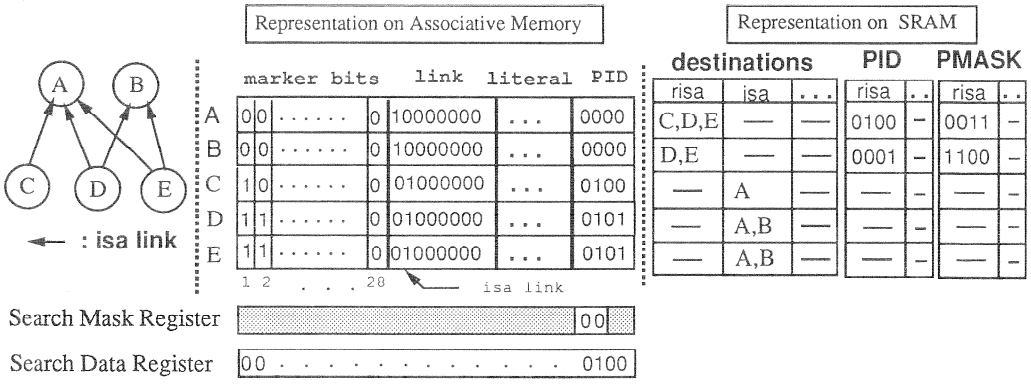


図4　意味ネットワークのデータ表現
Fig. 4　Data representation of semantic network.

ク領域は'01000000'となり，isaリンクの存在を示すビットが1になっている．

リテラルは，ノード自身が数値を表す場合の領域で，連想メモリのためのビット直列演算アルゴリズムを用いて並列処理される．

一方，通常のメモリ上に置く情報には次の三つがある．(1)そのノードと接続する，目的ノード名(destination)と接続リンクの種類，(2)並列マーカ伝搬情報(上記PIDと同一)，(3)並列マーカ伝搬マスク(PMASKと略す)．例えば図4において，ノードAは，isaの逆方向リンクによってノードC，D，Eと接続するので，目的ノード名の表のrisaの欄に，C，D，Eと記載する．

4.2　マーカ伝搬

IXM2のマーカ伝搬には2種類ある．一つは，各APにより一時に一つずつ行われる逐次的なマーカ伝搬である．これは，通常のメモリ上におかれた目的ノード名の情報を参照し，そのノードを含むAPに対し，マーカを更新するメッセージパケットを送るものである．もし目的ノードが同一のAP内にあるときは，そのAP内でマーカ伝搬が起きる．

もう一つは，各AP内において連想メモリの並列性を用いて，一つのノードから一時に複数のノードのビットをセットする並列マーカ伝搬である．但しこれらのノードはすべて同じAP内になくてはならない．以下では，並列マーカ伝搬の実現手法について述べる．

図4の意味ネットを例にとると，ノードAのもとにノードC，D，Eの三つがある．ノードAからこれら三つに対して並列マーカ伝搬を行う場合を考える．この場合Aを親ノード，C，D，Eを子孫ノードと呼ぶ．

並列マーカ伝搬の基本アイデアは次のとおりである．

(1)　すべての子孫ノードに，ある識別子(並列マーカ伝搬情報PID)を割り付けておく．

(2)　同じ識別子を親ノードにももたせておく．

(3)　親ノードから並列マーカ伝搬を行おうとするとき，親ノードはまずPIDと検索用マスクであるPMASKを用いて子孫ノードを1度に検索し，続いてすぐに並列書込み機能(連想メモリの機能)を用いて，ヒットしたノード(子孫ノード)の中の特定のマーカビットをセットする．

ノードAからの並列マーカ伝搬の場合，親ノードであるAは検索を行うために，PIDとして'0100'，PMASKとして'0011'をA自身のノード情報から得る．これらはそれぞれ連想メモリ用の検索データレジスタ(SDR)，検索マスクレジスタ(SMR)に図4下部に示すようにセットされる．SMRの中で灰色の部分はすべて1で，マスクビットが1だとこれらに対応する連想メモリ内のデータビットは検索対象とならない．この設定で検索を行うと，ヒットするのはC，D，Eである．この後すぐに，ヒットした語に対してのみ有効な並列書込み操作を行うことにより，これらの子孫ノードに共通のマーカビットがセットされ，並列マーカ伝搬が実現される．

並列マーカ伝搬に必要なPIDやPMASKなどの情報は，意味ネットワークをIXM2へ分割・配置するためのアロケータによって生成され，IXM2での意味ネット処理に先だって連想メモリにロードされる．アロケータは，{親ノード，リンク種類，子孫ノード}の組を与えられた意味ネットの中から識別し，その各々の組にユニークなPIDとPMASKとを割り付ける．

このような組の識別は，アロケータへのパラメータとして与える，親ノードからのファンアウト数Nに基づいており，N以上の子孫ノードをもつ親ノードのみが，このような組としてアロケータに認識される．例

えば図4でN=2とすれば，アロケータに認識される組は，{A,risa,(C,D,E)} と {B,risa,(D,E)} の二つとなる．

また，検索マスクPMASKを用意しなくてはならないのは，あるノードのもつPIDが複数の並列マーカ伝搬のための識別子を含む場合があるためである．

例えば図4でノードDとEは，親ノードAとBからの共通の並列マーカ伝搬先となっている．このような場合，DとEのもつPIDを，親ノードの違いに応じて使い分ける必要があり，そのためにPMASKがある．例えばノードAからの並列マーカ伝搬の場合に必要なPIDの情報は右から3，4ビット目だけなので，1，2ビット目を検索時にマスクするため，PMASKとして '0011' を用意している．一般に，あるノードがM通りの並列マーカ伝搬先となっている場合，そのPIDはM個の領域に分けて利用される．

4.3 交差演算

IXM2では，処理の途中結果をマーカビットに保存することで，交差演算などの集合演算をデータ数にかかわらず一定時間で行うことが可能である．連想メモリを用いた交差演算の実現法を以下に示す．

図5(a)は，図4の意味ネットにおいて，ノードAに属するものとしてC，D，E，ノードBに属するものとしてDとEが並列マーカ伝搬によって見つかり，それぞれマーカビット1番，2番がセットされた状態を示している．ここでノードAとBに共通する要素を求める交差演算は，マーカビット1番と2番がともに立つノードを見つけることに対応する．

このようなノードを見つけるには，図5(b)のように

	marker bits			link	literal	PID
A	0	0	.....	0 10000000	...	0000
B	0	0	.....	0 10000000	...	0000
C	1	0	.....	0 01000000	...	0100
D	1	1	.....	0 01000000	...	0101
E	1	1	.....	0 01000000	...	0101
	1	2	. . .	28		

(a) Associative memory status after marker propagation

SMR 00

SDR 1100 000

(b) Contents of SMR and SDR for the intersection

図5 連想メモリを用いた交差演算
Fig. 5 Intersection on associative memory.

SMR，SDRを設定すればよく，検索を実行した後に直ちに並列書込み操作を行うことにより，交差演算がデータ数にかかわらず一定時間で行える．

5. IXM2のプログラミング

IXM2はホスト計算機の制御下で動作する付加型プロセッサであり，IXM2本体のプログラム記述はoccam2で行う[8]．IXM2はホスト計算機から発せられる知識表現言語IXLのコマンドを実行する形態が主である．これに加えホスト計算機上で実行されるCプログラムからの呼出しをIXM2が実行することもできる．以下ではIXLの概要とその実行方式について述べる．

5.1 知識表現言語IXL

IXLはprologのスーパーセットであり，通常のprologに意味ネットワーク処理用の述語を付加したものである．これらの述語を図6に示す．大別して意味ネットワークの記述用の述語と，記述した意味ネットワークに対する検索用の述語の二つに分類される．またこれらの述語をIXLコマンドと呼ぶ．

IXLのインタプリタはQuintus prologで記述されてホスト計算機上に置かれる．IXLのプログラムもホスト上に置かれる．プログラム中の通常のprologの述語はホスト計算機で実行されるが，IXLコマンドはホストで認識されてIXM2へ伝達される．そしてIXM2での実行後，そのIXLコマンドの解がホストに送られ，IXLプログラムの実行はそのまま継続する．バックトラックも通常のprologプログラムと同じように扱われる．次の文において，isa(x,bird) はIXLコマンドの例である．

?－isa(x,bird), write(x), fail.

この文がホスト計算機に入力されると，IXLコマンド

To construct a relation:
assertion(R, X, Y).
property(R, X, Y).

To inquire a link:
isa(X, Y).
instance(X, Y).
ako(X, Y).
source(R, X).
destination(R, Y).

To inquire a relation:
asst(R, X, Y).
prop(R, X, Y).

To connect nodes by a link:
link(is_a, X,Y).
link(not_isa, X,Y).
link (instance_of, X,Y).
link(not_instance_of, X,Y).
link(a_kind_of, [X, Y, . . .], Z).
link(source, R, X).
link(destination, R, Y).
link(rule, X, ((
asst(R, X, Y) :- . . .
prop(R, X, Y) :- . . .
isa(X, Y) :- . . .
instance(X, Y) :- . . .)).

図6 IXLコマンド
Fig. 6 IXL command.

が認識され，このコマンドは bird の下位概念をすべて求めるために IXM 2 に送られる．その解が IXM 2 からホスト計算機へ転送された後，表示される．

5.2 意味ネットワークの処理形態

意味ネットワークの処理にあたって，データとプログラムの配置は次のとおりである．まず処理データは，扱う意味ネットワークをサブ意味ネットワークに分割し，各々を AP の連想メモリに格納する．プログラムは各 IXL コマンドごとに occam 2 で記述し，AP のローカルメモリに置く．プログラムは全 AP に共通である．

NP はその下に接続する 8 台の AP に対してブロードカスト機能をもつため，これを用いてホスト計算機から IXL コマンドが各 AP へと伝達される．

ホストの SUN-4 の中にはトランスピュータボード B014 が組み込まれ，これを介して，IXM 2 へのプログラムとデータのロード，IXM 2 の出力する解データの回収，および例外処理が行われる．

5.3 IXL コマンドの実行

IXL コマンドはホスト計算機から一時に一つずつ与えられ，従ってコマンドのレベルでは SIMD 型の並列処理である．しかし各コマンドの内部の実行はマーカ騒動と呼ぶ MIMD (Multiple Instruction Multiple Data stream) 型の並列処理で行われる．すなわち，各 IXL コマンドは IXM 機械語命令の列からなっており，IXM 機械語命令の実行はマーカ騒動型の非同期処理である．ほとんどの IXM 機械語命令の実行は，開始条件マーカとしてその機械語内で指定するマーカビットが，他の命令の実行によってセットされたときにのみ非同期的に開始される．

例えば **5.1** に挙げた IXL コマンドの isa(x, bird) は次の三つの IXM 機械語命令からなっている．

ASSOC(bird, 1)
MARK(1, risa, 1)
REPORT(1)

1 番目の命令では，もし AP が bird ノードを含んでいる場合，bird ノード内のマーカビットの 1 番をセットする．2 番目の命令では，もしあるノードのマーカビットの 1 番がセットされ，そのノードから逆方向に isa リンクが出ているならば (つまりノードが risa リンクをもつ) そのリンクでつながるノードのマーカビット 1 番をセットする．3 番目の命令ではもしあるノードにマーカビット 1 番がセットされたならそのノードの識別子をホストに送る．実行形態の例を示す．

今図 7 (a) の意味ネットワークがあって，これらが図

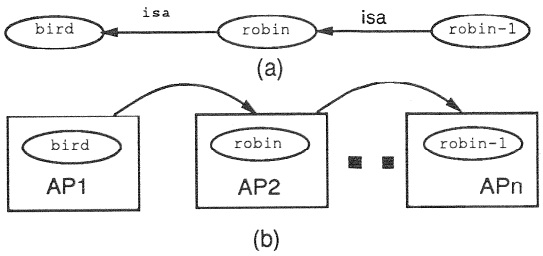


図 7　IXL コマンドの実行
Fig. 7　Execution of IXL command on IXM2.

7 (b) のように各 AP に配置されており，isa(x, bird) がホストからブロードカストされたとする．まず全 AP が ASSOC 命令を実行するが，成功するのは AP1 だけであり，bird ノードのマーカビット 1 番がセットされる．bird ノードは逆方向の isa リンクをもっているため，マーカ駆動により MARK 命令の実行条件が整い，この実行の結果，AP2 にある robin ノードのマーカビット 1 番がセットされる．このビットのセットは，AP1 から AP2 へのマーカ伝搬によって行われる．robin ノードのマーカビット 1 番がセットされたことにより，次に APn の robin-1 ノードのマーカビット 1 番がセットされることになる．このように MARK 命令の実行により，isa 階層のたどりが実現される．REPORT 命令では，マーカビット 1 番がセットされたノードがその識別子をホストへ返すが，これにより bird ノードの下位概念が見つかる．

6. IXM2 の性能評価

本章では，コネクションマシン CM-2 など他のアーキテクチャとの比較を交えながら，基本処理と応用の二つの面に分けて，IXM2 の性能について評価する．

基本処理では，連想，マーカ伝搬，交差演算，算術論理演算を挙げるが，これらは一般的，かつ頻繁に用いられるため，IXM2の性能を見る上での基本的尺度となる．応用では，知識ベースに対する問合せ処理と自然言語処理について述べる．

実行時間の測定条件は次のとおりである．IXM2 は，AP 内のトランスピュータ[7]のクロックを 17.5 MHz とし，プログラムは occam 2 で記述している．実行時間は T800 内部のタイマで測定した．クレイおよび SUN-4 のプログラムは C で記述し，O4 で最適化した．CM-2 のクロックは 7 MHz であり，プログラムは C* で記述，最適化した．CM-2 の実行時間は CM-2 上のタイマーを使用した．また IXM2，CM-2 とも，ホストとのオーバヘッドは除き，並列処理部上でのみ計測し

た．

6.1 基本処理

6.1.1 連想処理と交差演算

IXM2での連想処理は，連想メモリ上でのビット並列動作により64Kデータまで12マイクロ秒で実行できる．逐次計算機上でもハッシュによってデータ数によらない一定時間での処理は可能である．

しかし交差演算では，IXM2などのSIMD計算機が一定時間での処理が可能であるのに対し，逐次計算機はハッシュを用いてもデータ数の増加の影響を受ける．表3は，二つの同じ大きさの集合が3通りあり(1k,10k,64k)，二つの集合の共通要素を求める交差演算を行ったときの実行時間を示す．逐次計算機の場合，データ数をNとすると，ハッシュを用いて交差演算時間が$O(N)$となるように工夫しているが，それでもたとえスーパコンピュータであるCray X-MPを用いても，64KデータではIXM2やCM-2などのSIMDマシンと著しい差が生じていることがわかる．

この処理では配列を多用しているため，CrayはSUNより有利となっている．またIXM2がCM-2より高速なのは，連想メモリのビット並列処理による．

6.1.2 マーカ伝搬

(a) 逐次的マーカ伝搬の性能　1台のPEによって，ある一つのノードから1本のリンクをたどり，接続先のノードに付随するマーカビットをセットするまでの時間をアーキテクチャ別に表4に示す．これはリンクを1本1本たどる逐次的なマーカ伝搬であり，SUN-4などの逐次計算機の場合はこれ1種類のマーカ伝搬時間しか存在しない．

これに対し表4のIXM2の数値は，64台のAPのうちの1台のAPが行うマーカ伝搬時間を示しており，従ってこの処理が最大64箇所で並列に起こり得るので実効上はこれより速くなり得る．またCM-2の場合は，CM-2の中の1台の1ビットPEによるマーカ伝搬時間である．従って表4は本来マーカ伝搬の並列処理が可能なIXM2やCM-2といったシステムには不利な比較であるが，構成計算機単体での能力を見るために掲げた．

表4にあるように，SUN-4が最も速い．IXM2の場合は，主記憶の遅さ(サイクル時間230ナノ秒)のために若干遅くなっているものの，ほぼ逐次計算機と同程度の範囲に収まっている．しかも，これは1台のAPによるものであり，64台のAPにおいて同時にこの処理が並列処理され得る．これに対し，CM-2ではシリアルリンクの転送能力の低さが直接現れて著しく遅い．この時間は，CM-2の機械語であるPARIS命令[11]の中から，マーカ伝搬を実行する三つの命令の実行時間の和として求めたものである．このことから，CM-2の利用はマーカ伝搬の並列性が低い場合，相当のボトルネックとなることがわかる．

(b) 並列マーカ伝搬　IXM2とCM-2ではマーカ伝搬において並列処理が可能であり，その効果について考察する．

IXM2の場合は連想メモリを用いて並列マーカ伝搬を行うことができる．つまり，複数のファンアウトをもつノードから同時に複数のノードにマーカビットをセットすることができ，この操作が常に35マイクロ秒である．従って，これを表4の逐次計算機のマーカ伝搬時間と比較すると，ファンアウトが二つ以上であれば，IXM2の方が常に速いことがわかる．

表5は，CM-2においてマーカ伝搬が複数箇所で並列に起きる場合に，一つのマーカ伝搬当りの平均のマーカ伝搬時間が短縮される様子を5進木，10進木のデータを使って示したものである．参考にSUN-4のデータ

表3 交差演算の処理時間(μs)

set size	1K	10K	64K
IXM2	18	18	18
CM2	103	103	103
Cray X-MP	1829	7372	39823
SUN-4/330	28998	44332	142827

表4 逐次的マーカ伝搬の処理時間(μs)

IXM2 within an AP	IXM2 between two APs	CM-2	SUN-4 /330	Cray X-MP
25	122	1295	20	22

表5 CM-2の並列マーカ伝搬

木の種類	木の中のノード数	最大並列度	CM-2 (μs)	SUN-4 (μs)
5進木:				
深さ4	156	25	92	20
深さ5	781	125	27	20
深さ6	3905	625	6	20
10進木:				
深さ3	111	10	156	22
深さ4	1111	100	23	22
深さ5	11111	1000	2	22

も付す．これからわかるようにマーカ伝搬が数百から千箇所以上で同時に起きるのであれば，一つのマーカ伝搬あたりの平均時間がかなり短縮可能になる．但し，これらはあくまでも平均で数百から千箇所以上でマーカ伝搬が起きることが条件であって，平均でこれだけの並列度が常に維持できる応用は現在意味ネットワーク処理では見出しにくいと考えられる．

6.1.3 算術論理演算

連想メモリを用いた算術論理演算アルゴリズムは1960年代より盛んに研究されてきた[3]．その特徴はビット直列にデータを1ビットずつ処理していくが，すべてのデータに対して演算を並列に進められることである．従って連想メモリ内のデータ数が多いほど，1データ当りの演算時間は短くなる．IXM2では連想メモリ上での意味ネットワークの表現においてリテラルを指定することができるため，それに対してビット直列な並列演算を施すことができる．

IXM2での大小比較演算は，32ビットデータに対して常に36マイクロ秒である．1台のT800トランスピュータがoccam2で書いた大小比較処理に，1データあたり1.25マイクロ秒かかるのに比べて一見遅い．しかし連想メモリを用いた大小比較演算はデータ数によらず常に一定時間であるため，64Kデータの場合には1データ当り0.56ナノ秒と極めて高速である．また16ビットデータの加算では115マイクロ秒を要する．また速度に加え，算術論理演算ではデータを連想メモリからフェッチせずに演算ができるので，トラフィックの増加を抑える利点がある．

6.2 応用における評価

（a） 知識ベース処理　　まずワインについての5層のisa階層からなる知識ベースに対する問合せについて述べる．知識ベースの大きさは277ノード，632リンクと比較的小規模である．これに対して，ボルドーワインでかつ四つ星のワインはどれかを求める問合せを発する．行われる処理としては，マーカ伝搬による二つの集合の要素の取出しと，集合の交差演算であり，これらは意味ネットワークの典型的な操作である．

その処理結果を表6に示す．知識ベースが小さく，従ってデータ数によらない一定時間の処理というIXM2の利点が出にくいにもかかわらず，IXM2はSUN 4より高速である．一方CM-2の結果はかなり遅く，その主たる理由はマーカ伝搬の並列度が最大でも87しかなく，従ってシリアルリンク転送の弱点が直接的に現れてしまったためである．ちなみにCM-2の処理時間のうち95％がマーカ伝搬である．

（b） 自然言語処理　　実時間入力音声を対象とした高速な機械翻訳システムの実現のための手法として，メモリベーストパージングと呼ぶ手法が開発されている[9]．入力文に対して文法規則を繰り返し適用して解析する従来の方法ではなく，図8のように最下層が音素，次の層が音素列，次が単語の層というように，階層的に構成された意味ネットワークを用いて入力文の実時間解析を目指す．

入力として音素列を与え，それらから認識される概念単位が得られるまでのパージング部分をIXM2を使って実験した．入力する音素の数と，それらの認識に要する時間を表7に示す．扱うネットワークの大きさは443ノード，645リンクである．

この処理では認識すべき音素数が増えるにつれ，パターンマッチに要する交差処理の回数が増える特徴が

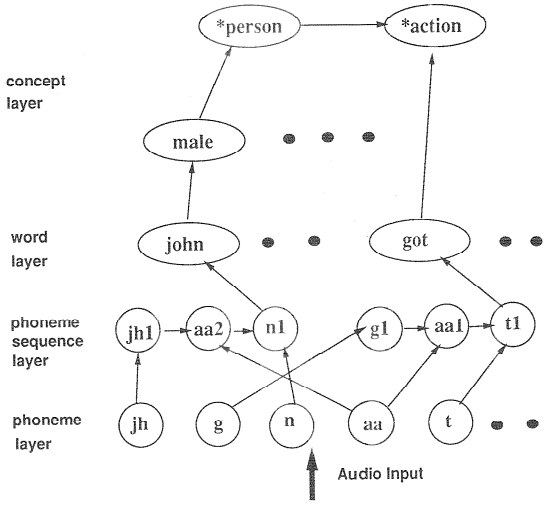


図8　メモリベーストパージングでの意味ネットワーク
Fig. 8　A semantic network in memory-based parsing.

表6　問合せ処理時間（milli sec.）

IXM2	CM-2	SUN-4/330
0.8	28.5	3.0

表7　パージング処理時間（milli sec.）

入力音素数	IXM2	SUN-4/330	CM-2
5	0.5	2.5	18.6
10	1.0	4.8	35.7
14	1.4	7.6	46.2
22	2.3	13.2	70.2

ある．従って**6.1**で述べたように，逐次計算機には不得手な処理であるが，実験データが小規模のため，あまりその影響は見られない．またCM-2は交差演算では本来有利のはずであるのに，マーカ伝搬の並列性が低いためSUN-4よりも遅い結果となっている．

ここで示した二つの応用での評価に共通して言えることは，用いた実験データが小規模であるため，IXM2の方が高い性能を示してはいるものの，IXM2には不利な評価の仕方であるということである．つまりIXM2のようなSIMD並列マシンは，大規模のデータを扱うときにその効果が明らかになる．例えば，表3に示した交差演算の処理時間は集合の大きさが千の場合で，IXM2とSUN-4の時間差が3けたにも達する．しかし，表6，および表7の場合は集合の大きさは最大でも数十にとどまり，IXM2には不利な問題規模である．このため，SUN-4に比べて圧倒的な性能差を示すまでには至っていない．しかし，現実の問題規模は数万といった意味ネットワークに拡大するため，SIMDマシンとしてのIXM2の効果がより顕著に現れることは確実と考えられる．

7. むすび

本論文では，大容量連想メモリに基づく新しい超並列マシンのアプローチとして並列連想プロセッサIXM2のアーキテクチャと，その意味ネットワーク処理への適用の有効性を示した．使用頻度が高い基本処理である連想処理と交差演算において，IXM2は最大64Kの並列性を有するSIMDマシンとしての特徴を生かし，データ数に依存しない一定時間の処理を実現した．また処理の隘路となる多ファンアウトノードからのマーカ伝搬において，連想メモリを生かした並列マーカ伝搬を実現した．アーキテクチャ的には大容量連想メモリと，意味ネットワークの性質を生かした完全結合に基づく接続方式が特徴である．

IXM2の性能評価の過程で，逐次計算機による交差演算が大規模な意味ネットワーク処理では大きな隘路になることを示し，データレベルの並列性を生かすことのできるSIMDマシンの必要性を明らかにした．

また本論文では，コネクションマシンCM-2に代表されるSIMD型超並列マシンとの比較も行い，1ビットPEとシリアルリンク接続を主体とした現在の超並列マシンでは，マーカ伝搬の並列性が平均で千近くなれば，シリアル転送の遅さがボトルネックとなり，意味ネットワークなどネットワーク型の大規模データの処理には適さないことを示した．

連想処理，交差演算についてはIXM2はCM-2と同程度の性能を示す．しかしマーカ伝搬においては，IXM2の採用する完全結合に基づく接続網と，多ファンアウトノードにおける並列マーカ伝搬により，マーカ伝搬の平均並列度が高くとれない通常の意味ネットワーク処理ではCM-2よりも高い性能を期待できる．

今後の課題としては，意味ネットワークをAP間に最適に配置することにより，AP間でのメッセージ交信によるマーカ伝搬のオーバヘッドを低減することのできるアロケータソフトウェアの開発が重要である．データの最適配置の問題はNP完全の問題であり，遺伝的アルゴリズムなどの最適化手法を導入したアロケータがIXM2の性能を向上させる上で必須である．

またIXM2のようなSIMDマシンはデータ並列度の高い大規模データほど効果が出やすい．本論文での評価に用いた意味ネットワークは小規模であるにもかかわらず，IXM2の優位性を示した．しかし意味ネットワーク自体の構造やそれに対する推論アルゴリズムは応用によって大きく異なるため，大規模で，かつ多様性に富んだベンチマーク応用を開発し，更にアーキテクチャの検討を進めることが必要である．

謝辞 本研究の機会を与えられた電総研の柏木寛所長，弓場敏嗣情報アーキテクチャ部長，国分明男前室長，内堀義信氏，連想メモリで御援助を頂いたNTTの小倉武氏，研究の支援と有益な助言を頂いたカーネギーメロン大学の富田勝教授，カーボネル教授，北野宏明氏，S. ファールマン氏に感謝する．

文献

(1) Batcher K. : "Design of a Massively parallel processor", IEEE Trans. Comput., **29**, 9 (Sept. 1980)
(2) Fahlman S. E. : "NETL : a system for representing and using real-world knowledge", MIT Press, (1979).
(3) Foster C. C. : "Content Addressable parallel processors", Van Nostrand Reinhold Company., New York (1976).
(4) Handa K., Higuchi T., Kokubu A. and Furuya T. : "Flexible Semantic Network for knowledge Representation", Journal of Information Japan., **10**, 1 (1986).
(5) Higuchi T., Furuya T., Handa K., Takahashi N., Nishiyama H. and Kokubu A. : "IXM2 : a Parallel Associative Processor", Intl. Symp. on Computer Architecture (May 1991).
(6) Hillis D. : "Connection Machine", MIT press (1985).
(7) Inmos : "IMS T800 Transputer" (April 1987).
(8) Inmos : "Occam 2 reference manual", Prentice Hall

(1988).
(9) Kitano H. and Higuchi T. : "High performance Memory-Based Translation on IXM2 Massively Parallel Associative Memory Processor", AAAI-91 (July 1991).
(10) Ogura T., Yamada J., Yamada S. and Tanno M. : "A 20-kbit Associative Memory LSI for Artificial Intelligence Machines", IEEE J. Solid-State Circuits, **24**, 4 (Aug. 1989).
(11) Thinking Machines Corporation : "Paris release note", ver. 5.2 (1989).

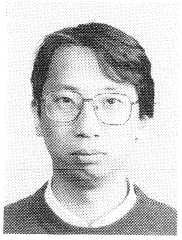
(平成4年3月17日受付，4月14日再受付)

樋口　哲也

昭57慶大大学院工学研究科博士課程了．電子技術総合研究所入所後，人工知能処理向き並列計算機システムの研究に従事．情報処理学会，AAAI各会員．

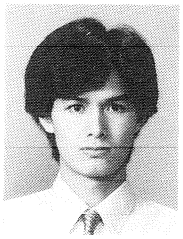
古谷　立美

昭48成蹊大大学院修士課程了．同年電子技術総合研究所入所．現在同所計算機構研究室長．工博．IEEE会員．



半田　剣一

昭56東大・工・電気工学科卒．昭58同大大学院電子工学修士課程了．同年，電子技術総合研究所入所．現在，知能情報部推論研究室ならびに協調アーキテクチャ計画室所属．人工知能の基礎研究に従事．情報処理学会，日本ソフトウェア科学会各会員．



高橋　直人

昭62筑波大第三学群情報学類卒．同年同大大学院博士課程工学研究科に入学，自然言語処理に関する研究に従事．情報処理学会，人工知能学会各会員．