●●●記念講演●●●

AI用コンピュータ・アーキテクチャ

田中　英彦*

1. はじめに

人工知能の研究や応用システム実現のために使われるコンピュータとはどんなものであろうか．コンピュータ・アーキテクチャの観点から見て，人工知能の処理に特有な点は何であろうか？　そして，より高度な人工知能の実現を支援することを目指した，現在のコンピュータ・アーキテクチャ研究の動向はどうなっているのであろうか？　これらの問題について触れるのが本論の目的である．

一般に人工知能で現われる処理の特質は，まず数値計算よりも記号処理が多用されること，そして，処理で出現するデータは，複雑なデータ構造，それも動的に変化するものが多いということ，さらに，指数的に増大する膨大な選択のトリー内の探索を必要とするように，処理速度，記憶容量の点で大容量なものを必要とすることなどであろう．このような特性を持った処理をアーキテクチャから支援するのがこの分野におけるコンピュータ・アーキテクチャ技術の役目である．いわゆる汎用機上でもこれらの処理が可能なのは当然であるが，処理の大容量性からして，できる限りの高速性を実現したいし，また，ソフトウェアの高度化に伴う，ハードウェアのセマンティックギャップも狭めたい．したがって，より人工知能に向いた専用機が求められるところとなる．ただし，汎用機か専用機かの論争は，その時代の素子技術との対比において評価されるべきものであろう．素子自体の速度が年々非常な勢いで向上するときは，汎用機を志向すべきであろうし，速度が頭打ちを見せ始めれば，並列性や特殊構造などのアーキテクチャ技術が求められるところとなる．もっとも，性能が格段に改善されるような革新的なアーキテクチャは常に意味があることは当然であろう．

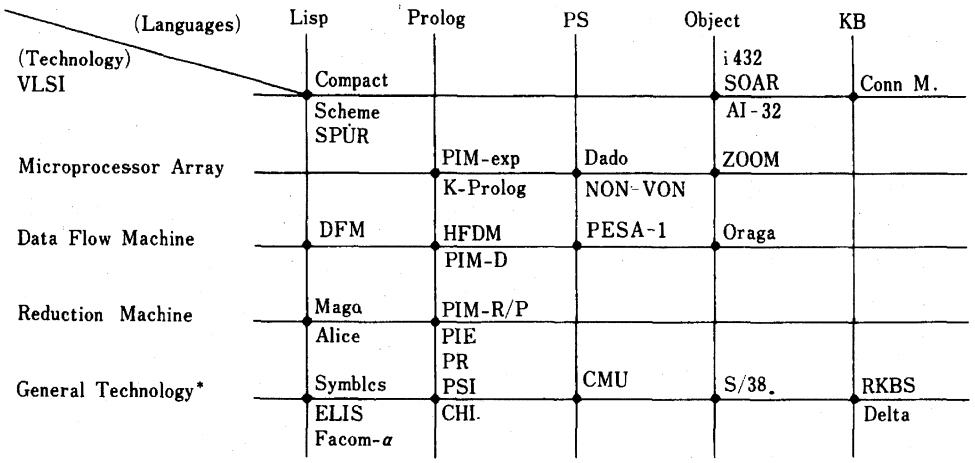
実際の応用プログラムがマシンの上で動作する場合，マシン内の活動はさまざまなレベルの表現形態をとる．それを図示したのが図1である．真ん中の列が各表現形に対応し，その右(:○○…)が，そのレベルにおけるキーワード，左がそのレベルにおける並列性の名前である．コンピュータ・アーキテクチャからみると計算モデルから下がその関連範囲であって，そのモデルで表わされた活動を，物理的なアーキテクチャ上での実行モデルに変換し，それを実際のハードウェアの上で走らせる．ハードウェアは，VLSIなどのデバイスをもとに，実装技術を用いて実現される．したがって，コンピュータ・アーキテクチャ技術は，これらの総合技術である．

LEVEL OF PARALLELISM	LEVEL OF REPRESENTATION	KEY WORDS
problems	applications	: included parallelism
task	algorithm	: parallel programming paradigm
	program	— languages, debugging-aid
		— compile
primitives	computation model	: data-flow, reduction
		— schedule, allocation
machine code	execution model	: physically mapped code
		— control, resource managt.
logical operation	hardware organization	: pipeline, switch, shared memory
	implementation technology	: high density, cooling
logic circuit	device technology	: VLSI, high speed device,

図1　コンピュータ内諸レベルの活動表現形態

現在，人工知能向きのコンピュータ・アーキテクチャは，各所で多くの研究があり，部分的には商用化されている．しかし，一口に人工知能向きコンピュータ・アーキテクチャといっても，そのスペクトルは広い．支援対象とする言語や，依って立つ技術などによってさまざまである．したがって，これらのコンピュータ・アーキテクチャを，言語と技術の二つの軸に沿って分類してみると図2のようになる．図中の軸の交点には，その位置を代表するマシンの例が示されている．言語の

＊　東京大学工学部電気工学科助教授



* cache, stack, multi-register, hash, tag, pipeline, GC, microprogramming, etc.

図2 AI向きコンピュータ・アーキテクチャの分類

軸としては，Lisp (関数型の諸言語を含む)，Prolog，プロダクションシステム (PS)，オブジェクト指向言語，および知識ベースを取り上げた．知識ベースは，言語ではないが，意味ネットワークなどさまざまな知識表現言語を代表するものとして，ここにあげてある．技術の分類としては，VLSI，マイクロプロセッサ・アレイ，データフローマシン，リダクションマシン，一般技術の五つに分けた．現在，VLSIは，あらゆるコンピュータ・アーキテクチャで使われる技術であるが，よりVLSI向きアーキテクチャを指向したものという意味で取り上げた，マイクロプロセッサ・アレイについても同様である．一般技術のところは，これらに入らないさまざまなアーキテクチャ技術として，キャッシュ，ハードウェア・スタック，多レジスタ，ハッシング，ダグアーキテクチャ，パイプライン方式，ガーベッジ・コレクション，マイクロプログラミングなどをひとまとめにしたものである．以下，これらの分類に沿って，アーキテクチャ技術の動向を述べる．

2. コンピュータ・アーキテクチャ技術からのAIマシン

優れたコンピュータ・アーキテクチャを実現する汎用の技術として，データフローマシン，リダクションマシンなどのいわゆる計算モデルからコンピュータ・アーキテクチャを設計していこうとするものと，素子技術を代表してVLSI技術とをここでは取り上げる．図2のマイクロプロセッサ・アレイもVLSI技術に含めることとし，一般技術については，次章の各論で触れる．

2・1 データフローマシン

データフローマシンは，オペランドが揃ったときにその演算を開始するという，いわゆるデータ駆動原理に基づいたマシンで，データの流れに着目したプログラミングをすることによって，容易に並列性を抽出することができる．いわば並列処理の基本原理となる計算モデルを与えるものである．その基本構成は，発火機構，演算器，オペランド記憶などがループ状のパイプラインを形成した形をとるが，最近の動向としては，基本的なデータ駆動原理に加えて要求駆動を加味した遅延評価，構造データを分離して貯え，そのポインタのみを流す構造記憶，Lenient consなどの機能拡張が行われ，より融通性のある汎用並列マシンとしての体裁を整えつつある．応用も数値処理のみならず，記号処理を目指した研究が行われ，実験機が作られている．たとえば，NTT基礎研のDFM，ICOTのPIM-Dなどがそうであり，十数台規模のマシンであって，いずれも1台に比べたときの並列利得は，応用問題に内在する並列性が高いときは，優れた特性を示している．応用問題の処理形態が行列演算のように定形的な場合は，マシンのアーキテクチャをそれに合わせることによって通常のマイクロプロセッサを複数並べても並列利得が得られるが，処理の形が規則的でない場合，特に人工知能の応用などでは，データフローマシンは，その優れた汎用性を発揮する可能性がある．

大容量の実験機としては電総研のSIGMA-1があり，百数十台のプロセッサからなる本格的なマシンが開発されつつある．これらのほか，データフロー処理用のLSIやマシンも商用化が始まっている．他の動

向としては，プロジェクト Cedar に見られるように，大きな処理単位間の並列処理をデータフロー的に行うというマクロデータフローの検討や，制御フロー方式との融合によって，より優れたアーキテクチャを見いだそうという方向の研究がある．

2・2 リダクションマシン

与えられた文字列に対し，種々の書換え規則を適用して，次々と書き換えてゆく操作をリダクションと呼ぶが，この操作によってあらゆる処理を実現しようというのがリダクションマシンである．これは，もともと GMD の K. Berkling によって提案されたものであるが，動作のわかりやすさ，およびλ計算との高い親和性によって新しい計算モデルとして期待されている．その実現形としては一般に，コピーを用いるストリングリダクションと，ポインタ操作を中心とするグラフリダクションとに分けられるが，文字列のどの部分を書き換えるかでさまざまな戦略が考えられている．また，書換えを組合せ子 (combinator) によって実現するという理論も検討が行われている．

マシンの研究例としては，先ほどの GMD-RM のほか，ユタ大学の AMPS，ノースカロライナ大学の Mago マシン，英国インペリアルカレッジの ALICE，ケンブリッジ大学の SKIM，日電の ARM などがある．このうち，複数プロセッサマシンとして実際に実現したものは ALICE だけであるが，これらはいずれも FP，HOPE など関数型言語を対象としており，述語論理型言語でも同様の実現が可能である．それらについては，次章で述べる．

2・3 VLSI 指向アーキテクチャ

最近のコンピュータはほとんどが VLSI を用いて構成されてはいるが，ここで述べるのはそのようなものではなく，最初から VLSI に向いたアーキテクチャを目指す一連の研究である．たとえばシストリック・アレイがそうであるが，個々の要素プロセッサをより汎用化した Warp マシンが CMU で作られている (図 3 参照)．

また，単純なプロセッサを 2 次元アレイ状に並べて大規模な並列プロセッサを実現したものとして，Goodyear Aerospace の MPP (16,000 台)，NTT 厚木通研の AAP-1 および AAP-2 (64,000 台)，意味ネットワークを直接ハードウェアで実現する Thinking Machine 社の Connection Machine (64,000 台) などがある．

このくらいの規模以上になると，もはや VLSI の利用なしの実現は考えられなくなってくる．汎用のマイクロプロセッサを多数，適当なトポロジーで接続した形のシステムは，それ自体 VLSI 指向アーキテクチャ自身ではないが，それに至る途中段階のマシンとして位置づけられる．この種のマシンとしては，カリフォルニア工科大学の Cosmic Cube (図 4 参照)，intel 社の iSPC，筑波大の PAX などがあるほか，多くの商用機が続々と現われ始めた．当初は，並列マシンの実験機としての需要を満たすのに使われ，そのうちに，個々の具体的な応用にしぼった利用が多く出てくるものと思われる．

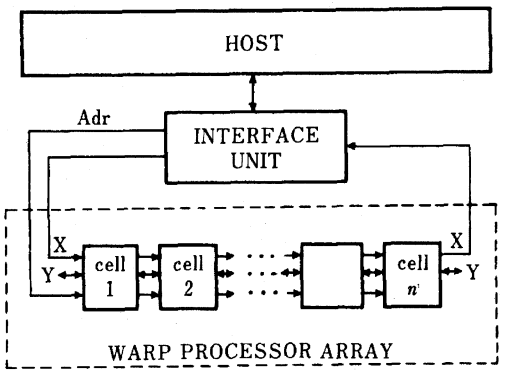


図 3 Warp マシン

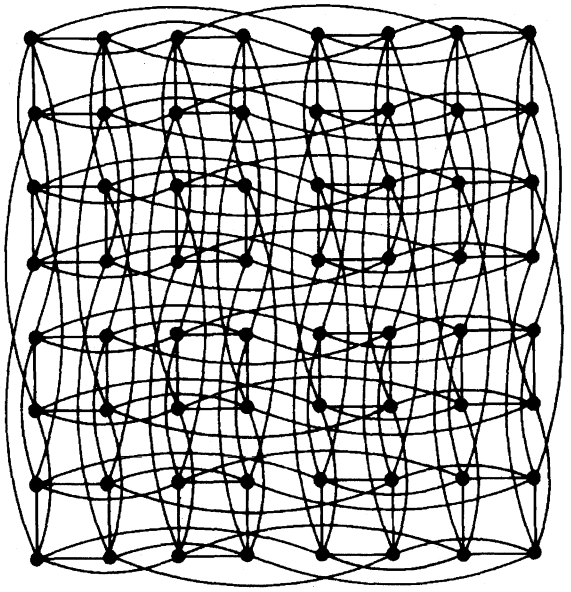


図 4 ハイパキューブ (6-Cube)

さらに，VLSI も単に平面内の密度を高めるのにとどまらず 3 次元実装の研究が行われている．3 次元実装のメリットは，素子間の配線長が短くなることで，それによって速度向上が期待できるほか，より複雑な構成も可能となる．問題は，作成技術のほか，冷却技術であるが，だんだんと改良されていこう．当初は，目の構造に似た形でシステムを作る画像処理応用など

が中心であろうが，3次元 VLSI 技術が進むにつれて，汎用の並列処理マシンの検討も進むであろう．実際，すでに各所でそのような提案がなされている．Hughes 社の 3D コンピュータもその一例である．

3．高級言語マシンとしての人工知能向コンピュータ・アーキテクチャ

本章では，ある人工知能用の言語を支援するという形で設計の行われたコンピュータ・アーキテクチャについて述べる．

3・1 Lisp マシン

人工知能用言語として最も歴史が古く，かつ最も広く使われているのは Lisp である．Lisp を高速に実行するコンピュータとしての Lisp マシンの研究は古く，すでに 10 年以上になる．そのアーキテクチャとしては，語の上位何ビットかをタグとして用いるタグアーキテクチャ，使用頻度の高いスタックの高速化をはかるためのハードウェア・スタック，大容量の記憶空間，複雑な操作を効率よく実現するために大量のマイクロプログラム記憶，さまざまな目的に使う多くのレジスタなどが特徴であり，ガーベッジ・コレクションの支援も考えられている．マシンとしてはすでに商用化されたものが多く出まわっている．Symbolics 3600, LAMBDA, Explorer, Facom α などがそれである．そのほか，ELIS, FLATS, SPUR のようなマシンもある．最近のものでは TI 社の Compact Lisp Machine がある．これは，Lisp マシンの VLSI 化をはかったものである．アーキテクチャ上の動向としては，Lisp マシンのマルチプロセッサ化，および単なる記号処理のみならず数値計算の機能をも高速に行うための工夫もなされるようになってきた．これは多量に，多くの分野で Lisp マシンが有効に使われるための条件でもあろう．

3・2 推論マシン

Prolog に代表される述語論理型言語を能率良く処理するためのマシンである．この種の処理の特徴は，単一化とバックトラックである．並列マシンの場合はバックトラックを伴わず，代りに OR 並列が用いられることもある．推論マシンのアーキテクチャとしては，まず逐次型マシンの場合，タグアーキテクチャ，ハードウェアスタックなどがよく用いられるのは Lisp マシンと同様であるが，推論マシンでは，スタックの深い所へのアクセスもかなり存在するため Cache の並用が有効である．さらに，多くの領域を目的別に分けて使うためのベースレジスタを多く設けるとか，ファームウェアの多重分岐，多重仮想アドレスなどが特徴である．代表的なマシンとしては ICOT の PSI, CHI などがあり，能率的なマシンコードとしてよく引合いに出されるものに Warren's Abstract Machine (WAM) がある．

並列推論マシンは，述語論理の処理を，AND 並列，OR 並列，ストリーム並列などの並列性（図 5 参照）

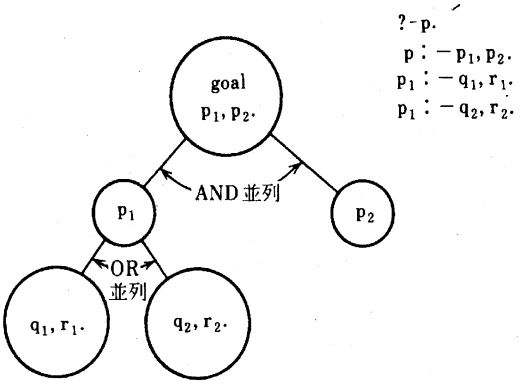


図 5 推論処理の並列性

を利用して高速に実行することを目的とするほか，並列プログラミング・パラダイムの優れている点を実証するための支援アーキテクチャでもある．現在，並列推論用の言語としては，明示的に並列性を記述する行き方と，GHC のように Guard を基本とする行き方とがある．これらはまだ研究段階にあるが，OR 並列については，十分それを引き出すアーキテクチャの有効性が立証されており，現在は AND 並列やストリーム並列について検討が進められている．マシンとしては，ICOT の PIM-R, PIM-P, PIM-D, 東大の PIE, 神戸大の PARK, 京大の PR, シュランベルジェの FAIM-1, RIT の OR 並列トークンマシンなどがある．これらのうち幾つかは実際に実装されたが，その並列実装規模自体はあまり大きくなく，高並列マシンの実装は今後の課題である．

3・3 プロダクションシステムマシン

if-then-else の構文で表わされるようなルール（それぞれをプロダクションと呼ぶ）の集合によって処理を記述するプロダクションシステムは，各種のエキスパートシステムを実現するのに使われている（図 6 参照）．この処理は，現在の状態で条件を満たすルー

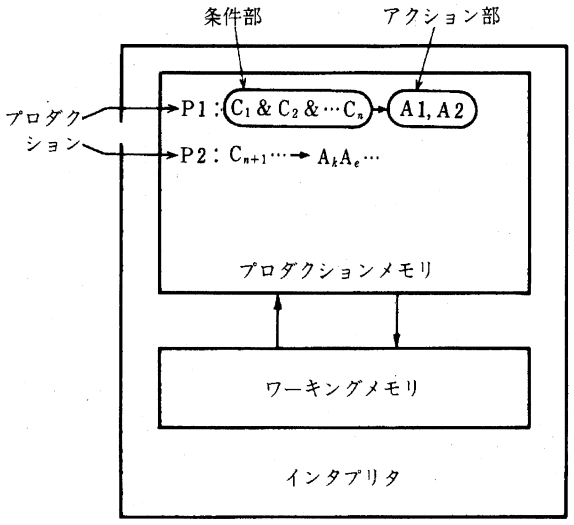


図 6　プロダクションシステム

ルの探索と，その中から一つを選択する処理と，そのルールを実行することの3フェーズの繰返しからなるが，一般に探索フェーズにはかなりの処理を要するため，それをいかに効率良く処理するかがポイントとなる．一つは，木状に接続された何千台もの数多くのプロセッサの上で並列に探索する行き方で，たとえばコロンビア大学の Dado がその例である．

もう一つは，探索をアルゴリズム的に効率化する行き方で，RETE アルゴリズムなどがその代表であるが，その結果として処理の並列性はあまり高くないと言われており，比較的少数の高速マシンでの処理が提案されている．しかし，現状のプロダクションシステムの処理速度は数ワーキングメモリ変化 / 秒程度であり，実用的なエキスパートシステムを実現するには，もっともっと高速化する必要があり (100倍以上)，この方面での高速マシンアーキテクチャの研究が望まれる．

3・4　オブジェクト指向マシン

プログラミングをするときのモジュールをオブジェクトとしてとらえ，それらが通信し合うことを処理とみなす，いわゆるオブジェクト指向プログラミングは，プログラミングの生産性を著しく向上させる可能性があるということで注目を集めている．言語例としては，Smalltalk-80, Objective-C, LOOPS, ABCL, ORIENT84/K, Dinner Bell, Flavor などがあるが，このパラダイムは，人工知能用言語にも取込みが行われているほか，多くの知識からなる知識ベースを組織化する手段として期待がかけられている．

しかし，この種の言語の実行は，メソッド探索などのためにかなりの手間がかかって一般に遅いのが難点である．したがって，それをアーキテクチャ側から支援することを目的として研究が行われている．ポイントは，高速手続き呼出し，アドレス境界チェック，コンテキスト割付の高速化，プロセス間通信の高速化，記憶の高速クリア機構，キャッシュなどである．オブジェクト指向マシンとしては，intel 432, IBM System 38, AI-32, Sword, SOAR などがある．動向としては，各種言語にこのパラダイムを持ち込む試みが行われているほか，言語に並列性の記述能力を付与することと，それを並列アーキテクチャで支援することの研究がある．もともと，オブジェクト指向プログラミングは，オブジェクトがメッセージを交わし合うことをモデルとしているので並列処理との対応が付きやすいが，それは，あくまで論理的なモデル上のことであって，並列処理を能率良く行うためには，物理的なオブジェクト間の近接性との対応をよく考慮する必要があり，このあたりの研究が行われている．

3・5　知識ベースマシン

人工知能では知識を一般に fact と rule とに分けることがよく行われるが，より優れたシステムを開発していけば，これらの量も当然大きくなることが予想される．専門領域を限定すれば，ある程度の量でその知識をカバーすることもできようが，“常識”を取り込んで，より人間にとって使いやすいシステムとしていくためには膨大な知識が必要となろう．そのような場合に必要となるのが知識の入れ物としての知識ベースマシンである．処理の各局面において必要となる知識を迅速に引き出したり，また知識間の関係を調べるという機能を持ったマシンが必要である．

現在，この構成としては，関係データベースマシンを中核にして，それに推論マシンを付加する形のものや，関係代数を拡張して単一化 (unification) 機能を包含したような拡張関係代数を支援するマシンといった形での検討が行われている．従来の関係データベースでは，一つ一つのデータを固定長かつ構造がないものとして扱っていたが，rule を関係データベースの枠組で扱うためには少し工夫が必要で，rule 表現方式の工夫や，関係データベースモデル自体の拡張などが望まれる．

マシンの研究例としては，ICOT の Delta, RKBS, MCC の LDL, 東大の Kelly, 阪大の DB-Prolog, ECRC の EDUCE などがある．動向としては，初期の検討が一段落した結果，単に関係データベースマシンと推論マシンを接続し fact をデータベースマシン

に，ruleを推論マシンに置いて処理をすすめる形ではオーバヘッドが非常に大きくなるため，両マシンを融合した形での構成を考えるとか，もっと疎な形で処理を分離するなどの工夫が必要なことがわかってきている．さらに，推論も，単なる演繹にとどまらず帰納型推論をもっと持ち込むことや，学習機能を持たせることなどが今後の課題であるが，そうなると知識ベースの多重世界化，更新機能などが必要になり，知識のコンシステンシィチェック，信頼性維持など，知識管理のシステム化技術が必要となってこよう．

4. おわりに

1960年代より，コンピュータ・アーキテクチャの研究は，高速化を目指してさまざまな工夫がなされてきた．演算器の高速化，パイプライン処理方式，並列処理方式，マイクロプログラミング，多レジスタセット，キャッシュ，チャネルなどである．これらの開発に素子技術の発達がからみ合って，さまざまなコンピュータが開発されてきた．しかし，基本的なアーキテクチャはあまり変わってこなかった．レベルが少々異なるとはいえ，ADD, MOVEなどの機械語セットをハードウェアで実現し，ソフトウェアで高級言語との境界を埋めてきたわけである．1970年代の多くの並列プロセッサの研究もその例にもれず，行きつく先は「問題に内在する並列性の抽出」と，その並列性を物理的なハードウェアの上で「能率良く動かすための制御方式」がポイントであるという，今から見れば至極当然の結論が出ている．

したがって，1980年代のコンピュータ・アーキテクチャの研究は，まず計算モデルが重視されなければならないというコンセンサスがあるように思われる．このモデルの重視によって，応用問題に含まれる並列性や処理の動きと，コンピュータ・アーキテクチャとの結びつきがより緊密となった．ソフトウェア駆動によるアーキテクチャ研究の動きである．この方向の一つがデータフローマシンであり，また推論マシンである．最上位の応用問題から，最下位の物理素子に至る各所に存在する高速化/能率化の種を拾い集めて総合的な高速化をはからなければならない．

コンピュータ・アーキテクチャ技術は，そういう意味で総合技術である．従来までの研究によって多くの基本手法が集まってきている．さらにまた，素子技術の発展によって装置の複雑さからの限界というくびきから逃れることが可能となりつつある．アーキテクチャ研究者にとっては，今や，その組合わせの妙を楽しむ時代に入ってきたのであろう．その成功は，コンピュータ処理能力の著しい向上（1桁どころではなく，数桁もの）を意味し，それは，ソフトウェアの構成法の変革や，従来不可能と思われていた応用分野の拡大につながるインパクトを持ち得るし，またそうであることを期待したい．