計算機アーキテクチャ 59−7
オペレーティング・システム 29−7
(1985・10・24)

リスト処理向きデータフローマシン実験機DFMの試作とシミュレーションによるアーキテクチャ評価

Performance Evaluation of List-processing-oriented Deta Flow Machine Architecture

長谷川隆三　三上博英　武末勝　雨宮真人

Ryuzo HASEGAWA Hirohide MIKAMI Masaru TAKESUE Makoto AMAMIYA

NTT基礎研究所
NTT Research Laboratories

線形の速度向上を達成する高並列記号処理マシンを実現するには、通信・同期のオーバヘッド，共有資源に対するアクセス競合の他、リストの並列性の阻害要因となる線形リスト構造操作の問題を解決する必要がある。本論文では、記号処理用データフローマシンDFMの試作・評価を通じ、データフロー方式により、粒度の小さな関数を単位とした超多重処理，及び線形リスト操作のパイプライン処理化が達成され、記号処理において係数 0.8 以上の台数効果が得られることを示した。

1．まえがき

知識処理マシンの基礎を与える高並列記号処理マシンの方式としてリスト処理向きデータフローマシンの方式を提案し〔1，2〕、そのアーキテクチャ並びにそのプログラミング・言語の具体化〔3〕を図ると共にソフトウェアシミュレーション等により高並列記号処理の可能性を明らかにして来た〔4，5〕。この過程で、データフロー制御方式を用いることにより、

① プロセッサ内での多数の並列プロセスを同時に処理する超多重処理の実現，
② プロセッサ内およびプロセッサ間のパイプライン処理・並列処理制御の実現，
③ リストの先行評価機構によりリスト処理での高度並列処理性の抽出と実行，

が可能となることを示し、また再帰概念を基調とする関数型プログラミング言語とデータフロー型リスト処理方式の間に極めて良い親和性があることを明らかにしてきた。

しかし、このようなデータフロー型の並列リスト処理マシンが現実（経済的）に構成可能であるか、また現実に設計されたハードウェア上で実際に高並列記号処理が可能であるかを明らかにしなければ、それが真に（工学的に）意味のあるものであるかを示したことにはならない。

そこで、我々は、現在可能なハードウェア技術を用いてプロトタイプ機の設計・試作を行い、実現技術を具体化させると共に、設計・試作をベースにより詳細な、現実に即した評価を行った。具体的には、記号処理向きデータフローマシン・プロトタイプ（Data Flow Machine Prototype DFM）の設計・試作、及びDFMをクロックレベルで正確に模擬するRTL（レジスタトランスファレベル）シミュレータによる方式の評価である。

本稿では、DFMの試作、シミュレーションを通じて得られたデータをもとにデータフロー型高並列記号処理マシンの実現性に関して①超多重処理効果，②多プロセッサ台数効果，③リストの先行評価による並列処理化効果，を中心に具体的に評価する。そして、これらに関する評価の結果としてデータフロー型の高並列記号処理方式が実現上かなり有効であることを明らかにする。

2．DFMアーキテクチャ

本章では、並列記号処理に於ける問題点について考察し、これを解決する記号処理用データフローマシンDFMの設計思想並びにハードウェア構成について述べる。

2．1　並列記号処理の問題点

リニアな速度向上を達成する高並列記号処理マシンの実現のためには、以下の問題を解決しなければならない。

従来型マルチプロセッサ共通の問題は次の通りである。

① 通信・同期がＯＳ（ソフトウェア）の制御下で行われるため、スケジューリング等のオーバヘッドが大きく、プロセス切替え時の共通テーブルアクセスがボトルネックとなる。このため10^3個程度のプロセスを同時並行的に処理する超多重処理が困難である。

② 共有資源（メモリ）へのアクセス競合のため、メモリアクセス遅延の問題が顕在化する。

一方、リスト処理特有の問題は、次の通りである。

③ データ構造が不規則で、且つ動的に変化するリストを扱うため、事前スケジューリングが出来ず、動的負荷割付が必要となる。

④ 線形リスト構造の部分が多く、ポインタのたどり等、逐次的な処理を必要とする部分も多い。　また、配列・ベクトルのようには、リストの各要素を分解操作する処理が困難なため、従来型マルチプロセッサ方式では性能が上がらない。

⑤ 再帰構造データを扱うため、粒度の小さい関数単位のプロセス切替えが頻繁に起こる。

上記の問題を克服するため、データフロー実行制御方式をとり、以下に述べる方針に基づきＤＦＭの設計を行った。

2．2　設計思想

・データフロー方式では、命令の基本実行機構として通信・同期の機能を備えているため、ＰＥ内ではプロセス切替えのオーバヘッドがない。　従って、粒度の小さい関数を単位とした超多重処理が可能である。しかし、ＰＥ間通信の場合には、走行ＰＥ及び色（実行環境の識別情報）の割当てに伴う処理量の増加，通信路のトラヒックを考慮すると、命令レベルのＰＥ間通信は好ましくない。　そこでＤＦＭでは、関数を単位としてＰＥに割付け、ＰＥ内およびＰＥ間での超多重処理を行うこととした。さらに命令レベルにおいても、ＰＥ内でパイプライン処理することにより、ＰＥの効率向上を図っている。

・リスト処理において並列性を抽出するためには、逐次操作の部分のパイプライン処理化が必須である。線形リストデータの分解・パイプライン処理を可能とするため、　ＤＦＭでは cons　の先行評価〔8〕を行うとともに、リストの各要素データ（セル）単位の同期機構を構造メモリ内で実現した。

・ボトルネックの要因となる共有資源としては、ＰＥ及び色の割付け機構，オペランドメモリ，リストを格納する構造メモリ，通信路がある。ＤＦＭではアクセス競合軽減のため、関数の生成・消滅に伴うＰＥ，色及びオペランドメモリの割付けは各ＰＥで行い、構造メモリを多バンクで構成した。さらに、新セル要求を各バンクに均等に分散させることで、ＳＭの負荷の均一化を図っている。

データフロー方式を採ったことによる問題は次の通りである。

① オペランドマッチングを行うため、メモリアクセス頻度が本質的に増加する。

② 有限の色の管理（幽霊トークン対策等）のためノード数が増加する。

③ 状態・時間依存処理の実現コストが高い。

上記問題の内、①については、マッチングに要する時間は演算時間に比べて小さく、パイプライン処理に隠されるため、実際上は問題とならない。②，③については、コンパイラの最適化及びハードウェアの支援により対処する方針である。

2．3　ハードウェア構成

上記の設計思想を具体化・実現するため、ハードウェアは以下の方針で設計・試作した〔6〕。

① 各種実験・計測を可能とする豊富な機能を盛り込む。

② プロセッサ台数，プロセッサ間結合方式は、研究室試作としての完動可能性を最重点に設定する。

③ 本格試作はＬＳＩ技術を駆使して行う。

試作したＤＦＭの構成を図１に示す。

ＤＦＭは８台のＰＥ（Processing Element），８台のＳＭＵ（Structure Memory Unit），及び１台のＣＣＵ（Cluster Control Unit）からなるクラスタ構成をとり、各々バスで結合されている。　ＰＥは単体でデータフロープロセッサとして動作し、データフロープログラムの格納・結果分配を行うＩＭＵ（Instruction Memory Unit），オペランドマッチング及び特殊命令の実行を行うＯＭＵ（Operand Matching Unit），制御／数値演算の実行と各バスのインタフェースを行うＦＵＵ（ Functional Unit ）を循環パイプライン型に接続した構成

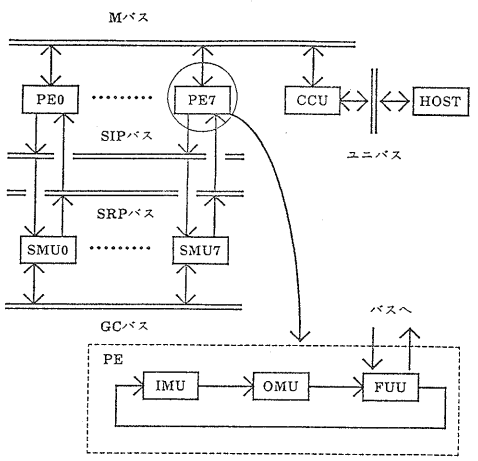

図1 DFMの構成

となっている。SMUはリストデータを格納・操作する一種の機能ユニットであり、リストデータの先行・遅延評価機構〔8〕及び並列ごみ集めの機構を有する。

3．RTLシミュレータ

DFM実験機を補完し、データフローマシン方式上の問題の分析とアーキテクチャのチューニングアップ，及び多数台構成の場合の性能評価を行うことを目的として、DFM実験機をクロックレベルで正確に模擬するRTL（レジスタトランスファレベル）シミュレータを作成した。本シミュレータは関数型言語 Valid〔7〕で記述され、DEC 2060上で稼働中である。

表1に主要命令及び構成要素の実行時間（クロック数）設定値を示す。1クロックはDFMの基本クロックに一致し、180 ns である。

本シミュレータでは、DFM及びそのバスをネットワークで置換えた構成の性能測定が可能である。32PE／32SMU迄の結合を前提として、ネットワークは5段のステージで構成されている。

4．関数の超多重処理特性

関数型言語の実行では、関数リンケージの処理効率が性能に大きな影響を及ぼす。DFMにおける関数の超多重処理能力を調べるため、関数リンケージが処理の大半を占め、通信の比重が高い条件下について評価を行った。評価に用いたプログラムは分割統治法によりnまでの総和を求める計算（sigma）である。sigma の場合、関数リンケージ関連の命令が静的に全体の53%を占める（6．表2参照）。

表1 実行時間の設定値

機能	命令	clock
関数呼び出し	call	7
	callr	5
パラメータ転送	link	4
	rlink	5
関数復帰	return	6
加算命令	add	3
制御命令	sw	4
PEからの受信	crpb	8
SMからの受信	srpb	8
SM命令送出	car/cons etc.	5
セルの読み出し	car / cdr	4~9
セルの確保	cons / gcell	6
セルへの書き込み	scar / scdr	6~9
Bus		3
Network（5段）		3/段

4．1 バス方式

図2に、バス方式の場合の台数効果（PE 1 台に対するスループットの向上比）を示す。バス方式では、PE 4 台以上で性能向上に飽和がみられる。又、台数効果も低く（8PEで4.7）、n＞256 では、インスタンスの増加のためバス競合が増し、性能が低下してくる。

図3に、PE 数8，n＝512 の場合におけるバス待ち（mbw），FUU のアイドル（idl）及びFUU の有効稼働率（efu）の時間推移を示す。図3に示す通り、関数の展開が進むにつれてバス待ちの比率が高くなり、FUU の有効稼働率が低下する。この場合、mbw，idl，efu の全実行時間に占める割合は、それぞれ 25 %，24%，51%である。

4．2 ネットワーク方式

図4に、ネットワーク方式の台数効果を示す。この場合、台数効果の飽和はみられず、nの増加に応じて性能が向上する。n＝1024では、PE数32の場合の性能は26.2を示し、係数0.8 の

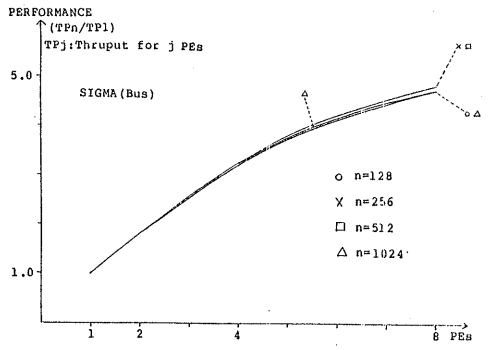

図2 バス方式の台数効果

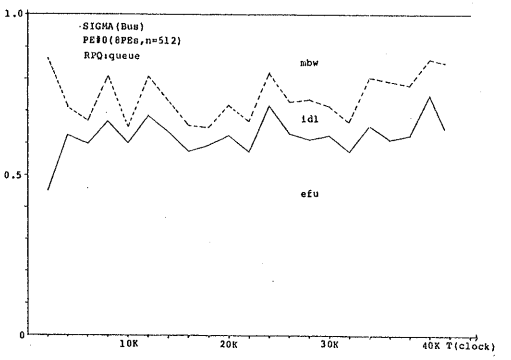

図3 mbw,idl,efuの時間推移

台数効果が得られている。図4には5．3で述べるn＝1024の場合の推定値も示した。シミュレーションの限界（メモリ容量不足）のためこれ以上のデータは得られていないが、nに対する性能向上の延び率と推定値から、nを大きくすると係数 0.9 以上の台数効果が得られると期待できる。

図5に、n＝512 の場合の全実行時間に対する mbw, idl, efu の比率を示す。ネットワーク方式では、バス方式に対して実行時間も1PEで23%，8PEで55%短縮される。バス方式に比べアイドルが増えているのは、FUU でのバス待ちのための無駄な処理がなくなり、プログラム特性としてのアイドルが顕在化してきたためと思われる。

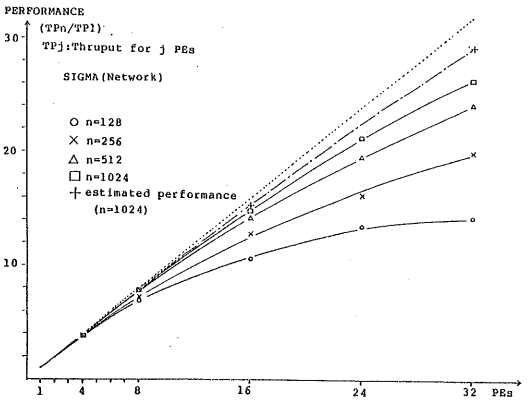

図4 ネットワーク方式の台数効果

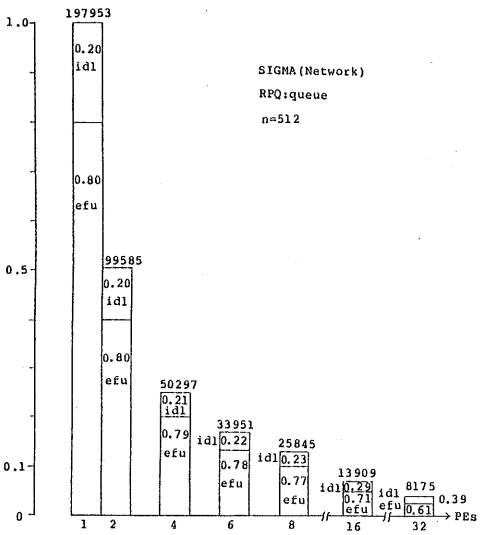

図5 mbw,idl,efuの全実行時間に占める割合

5．リスト処理特性

リストの並列処理に於いては、再帰関数の処理構造がマルチプロセッサ化の効果を大きく左右する。　本章では、この観点から、リストの並列処理構造と台数効果の関係について述べ、評価結果を示す。

5．1　並列処理構造の分類〔4〕

関数の再帰構造に注目し、実行過程で生ずる並列処理構造を以下のように分類する。一般には、並列処理構造はこれらの型が組合わされた混合型となる。

・木評価型構造

分割統治法の sigma 計算のように、関数を並列的に適用する過程で生じ、木状の並列処理構造が得られる。生起される関数が指数的に増加するため、多大の並列性が得られる。

・線型評価型構造

単純再帰構造の append 関数のように、関数を直列的に適用する過程で生じ、線形の並列処理構造が得られる。並列性は本来ないが、関数間にデータ依存関係がない場合、データフロー制御下では呼側の関数の実行と呼ばれる側の関数の実行がオーバラップして進むため、パイプライン効果が得られる。

データ依存関係がある場合には、従来方式ではマルチプロセッサ化の効果はないが、 Lenient cons 〔1〕のようにデータ同期機構を用いて先行評価を行うことにより、リストのパイプライン処理による並列性の抽出が可能となる。

5．2　関数の展開・収束時間の台数効果への影響

関数の展開・収束時の PE の遊びによる台数効果の減少を考慮に入れて、木評価型の sigma, 及び線型評価型の ssort（最大値選択法によるソート）を例に、台数効果を推定する。但し、関数の実行時間は一定で均一に展開されていくとし、PE の稼働率の変動は無視する。

・sigma 計算

データ数 $N=2^n$ ，プロセッサ数 $M=2^m$ の場合について考える。引数を受取り、配下の2個の sigma を call する迄の時間を a，結果を return するのに要す時間を b とする。展開の最終段（leaf）の sigma は return 処理のみを行うため、1PE の場合の全処理時間は $T1=aN+2bN$ 。PE 数 M の場合の理想処理時間 $IT=T1/M$ 。展開・収束過程でいずれかの PE の受持インスタンスが1未満（PEがアイドル）である時間を Te, Tc とすると、$Te=ma$, $Tc=mb$ である。

受持インスタンスが1以上となる場合の処理時間 Tf は $(T1-(a+b)M)/M$ 。従って、展開・収束時の遊びを考慮に入れた処理時間は $Tm=Te+Tf+Tc$ であるので、この時の台数効果係数 $k=IT/Tm$ は、

T1 ／ (T1 + M (m - 1) (a + b)) となる。

N = 1024, M = 32, a／b = 5／3 の場合、k = 0.92 となり、1台に対する性能向上比= 29.3 である。

・ssort 計算

ssort の計算では、与えられたn個のリストから最大値を取除いたn－1個のリストを生成し、これに対し再帰的に ssort を適用する。 Lenient cons を使用すれば結果のセルだけ先に返されるので、ある時間遅れ d で次の再帰レベルの処理に入る。この場合、 return の処理は隠されるので、 call の時間だけを考慮する。 sigma と同様に計算すると、

T1= (N+ 1) (N+ 2) ／2 。IT=T1／M 。又、Te=Tc=d M,

Tf= (T1 - 2d M (M- 1) ／2) ／M 。従って、

k = (n+ 1) (n+ 2) ／ ((n+ 1) (n+ 2) + 2d M (M+ 1)) となる。

5．3 並列処理構造と台数効果

本節では、データフローマシンの性能測定結果を示し、5．1で分類した処理構造に対する並列処理効果を具体的に示す。

評価プログラムとして、木評価型のマージソート (msort)，クイックソート (qsort)、線型評価型の ssort, 和集合計算 (union)，リストの短縮化計算 (compact) 及び混合型である算術式のコンパイル (arith) を用いた。

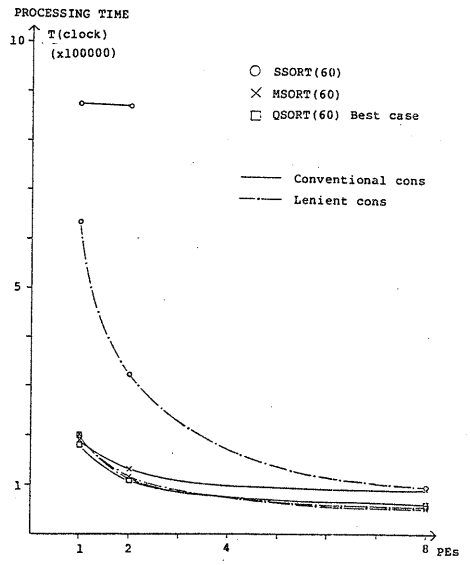

図6 実行時間(1)

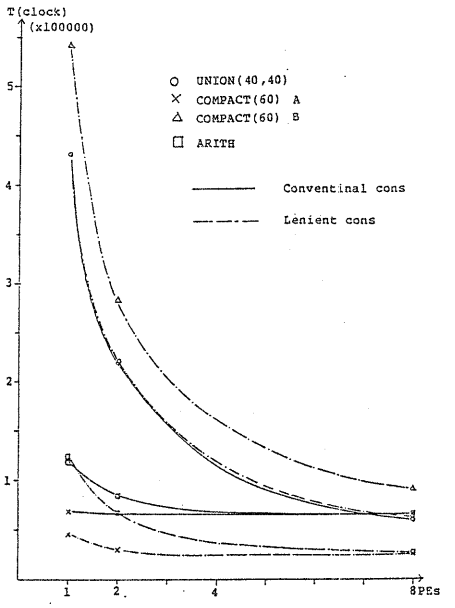

図7 実行時間(2)

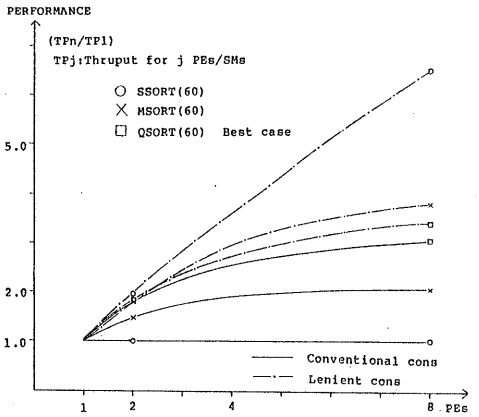

図8 台数効果(1)

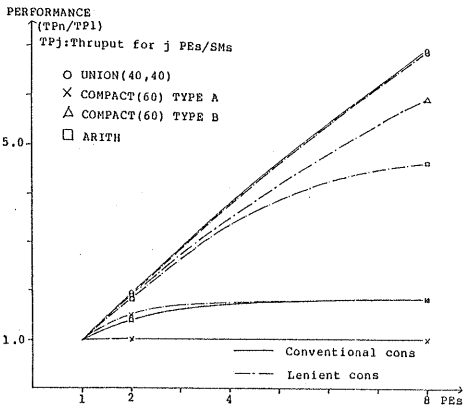

図9 台数効果(2)

各プログラムの実行時間を図6，7に示す。図8，9はこの時の台数効果である。

・木評価型構造の台数効果

木評価型の場合、分割統治アルゴリズムによるパラレル効果が主要なファクタとなる。しかし、Lenient cons では関数展開と並行して結果が返されるので関数の収束が速くなり、通常 cons 使用の場合より性能が向上している（PE数8の場合のmsort，qsortの性能向上比は、Lenient cons ではそれぞれ3.8，3.4、通常 cons ではそれぞれ 2.1，3.1 である）。PE台数効果が sigma に比べて低い理由については5．4で述べる。図6に示す通り、PE1台では Lenient cons を gcell，scar，scdr，gate ノードに分解したことによるオーバヘッドが現れ、実行時間が延びるが、僅か5 ～6 %と殆ど問題とならない程度である。

・線型評価型構造の台数効果

データ依存関係のない union 計算では、関数間のオーバラップ実行によりパイプライン化が達成されているため、通常 cons と Lenient cons の差異は殆どなく、8PEで約7倍の性能向上がある（図9）。一方、データ依存関係のある ssort では、通常 cons を使用した場合、台数効果は全くない。しかし、Lenient cons を用いると PE 数の増加に比例して性能が向上し8PEで6.5 の性能が得られている（図8）。

ssort と同様の処理構造を持つ compact については、タイプA（4種類の要素からなるリスト）及びタイプB（全て異なる要素からなるリスト）の2種のデータに対する結果を示した。タイプAでの台数効果が低い（8PEで1.8 の性能向上）理由は、再帰が4段迄しか深まらず論理的な並列性が小さいためである。タイプBでは ssort と同様の傾向を示し、8PEで5.9 の性能が得られた（図9）。

・混合型の台数効果

混合型の arith の場合にも Lenient cons の効果が現れ、PE数8での性能は、通常 cons の場合1.8，Lenient cons の場合4.7 となっている（図9）。

5．4 考察

・データ数の増加と台数効果

分割統治アルゴリズムを用いた msort，qsort と、同様のアルゴリズムを用いた sigma を比較すると、前者の台数効果の係数が低くなっている。この理由は、msort，qsort のデータ数（n＝60）が、sigma のデータ数（n＝1024）に比べて小さく、call の発行回数も少ない（sigma の1／3程度）ため、PE内で処理される関数インスタンス数が減少し、パイプラインの詰まりの度合が低いことによる。このことは、sigma（n ＝1024）と同じ位の数の call が発行される ssort（n ＝60）では、Lenient cons 使用の場合パイプライン効果が高まり、その時の台数効果係数が sigma と同程度に高くなっていることからも言える（図8）。

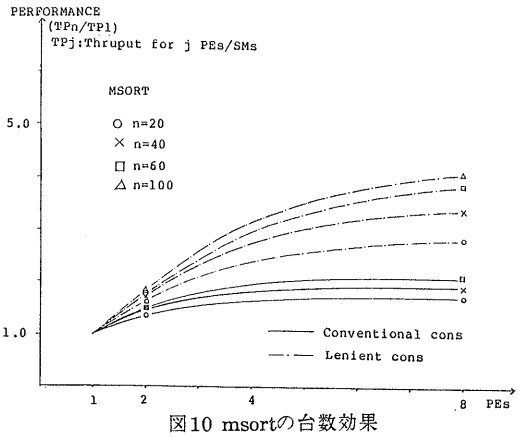

図10 msortの台数効果

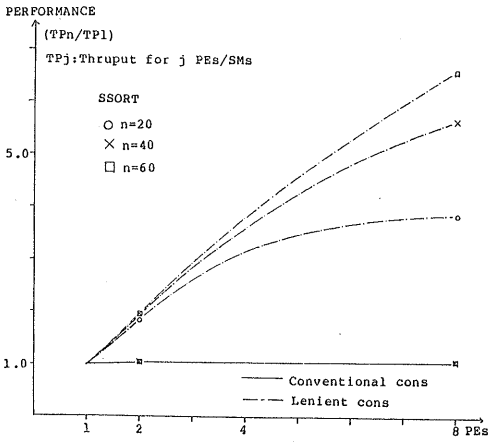

図11 ssortの台数効果

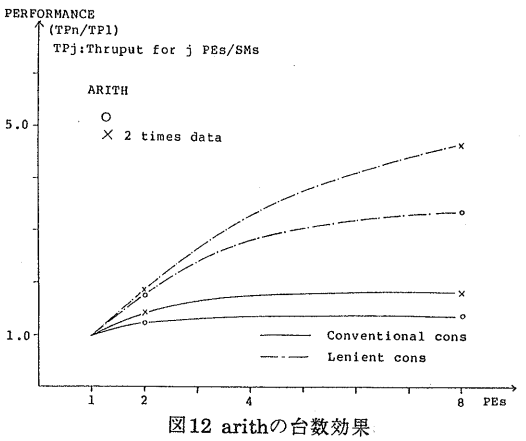

図12 arithの台数効果

参考のため、msort, ssort, arith について、データ数の増加と台数効果の関係を図１０～１２に示す。 いずれに於いても、データ数の増加に応じて台数効果は上昇し、飽和減少は見られない。したがって、データ数を大きくするとさらに台数効果の係数は大きくなると予想できる。

・関数の超多重処理能力

並列処理構造の内、並列性を損なう構造はデータ依存関係のある線型評価型構造であり、木評価型及びデータ依存関係のない処理構造は、論理的には高い並列度を有している。
しかし、これを効果的に引出すためには粒度の小さな関数単位での超多重処理が必要である。 従来のマルチプロセッサ方式では、プロセス切替えのオーバヘッドがあるため、この単位での超多重処理が困難である。これを軽減するには粒度を高めなければならず、処理構造に内在する並列性を十分に引出すことが困難となる。

これに対しデータフロー制御では、プロセス切替えのオーバヘッドがないため、関数レベルでの超多重処理が可能となり、関数間の並列・パイプライン処理が達成される。 この結果、union の台数効果（図9）で示したように、リスト処理に於いても、係数 0.9 に近いリニアリティが得られている。

・ Lenient cons 効果

線形リスト構造の逐次操作の問題を克服するには、リストを各要素毎に分解して操作するパイプライン処理が必須である。このためには、関数の超多重処理に加え、cons 時にセルだけ先に返す先行評価，及びデータ（セル）に対して同期をとる機構が必要となる。ノイマン方式では、コントロールフローに基づいて実行が制御されるので、この機構の実現は実際上は困難であり、オーバヘッドを伴う。データフロー方式では、同期の機能を実行機構中に備えているため、５．３木評価型構造と台数効果の節で述べたように、オーバヘッドが殆どなく実現される。 Lenient cons の機構は、ssort, compact の例で示した通り、並列処理の阻害要因とされていたデータ依存関係のある線型評価型構造に対して、顕著な効果を示しており、並列リスト処理の問題を解決する有効な手段であると言える。

６．従来型マシンとの性能比較

単一プロセッサとしてのデータフローマシンの性能を評価するため、PE／SM 各１台構成のデータフローマシンと従来型マシン上の Lisp （コンパイラ版）との実行時間比較を行った。表２に、評価用プログラムの静的特性を示す。いずれも関数本体のサイズは小さく、関数リンケージ関連の命令が４～５割を占めている。 SM 命令の割合は１割程度である。Lenient cons の場合、 cons を基本ノードに分解するため、通常 cons 使用に比し、１０～２０%程度の命令数の増加がある。その他の命令は、大半が TF スイッチ等のフロー制御用命令である。

６．１ 実行時間比較

VAX 750 上の VAXLISP (Common Lisp) 及び DEC 2060 上の Maclisp の各コンパイラ版とデータフローマシンとの実行時間比較結果を表３に示す。

線形評価型の ssort, compact は通常 cons では並列度は上がらないが、 Lenient cons を使用した場合、実行時間は３０%程度短縮し、対 VAXLISP 比及び対 Maclisp 比も３～４割向上する。 一方、 Lenient cons 効果の薄い木評価型プログ

表2 静的プログラム特性

program		Inst size	inst type			
		I	F(%)	L(%)	S(%)	O(%)
ssort	C	73	48	0	12	40
	L	82	43	16	7	34
msort	C	104	44	0	14	42
	L	118	39	14	11	36
qsort	C	121	45	0	11	44
	L	136	40	15	6	39
union	C	68	38	0	10	52
	L	71	37	6	8	49
compact	C	67	42	0	12	46
	L	73	38	11	8	43
arith	C	107	41	0	21	38
	L	130	34	25	1	31
sigma		35	54	0	0	46

F：関数リンケージ関連
L：Lenient cons 関連
S：SM命令
O：その他（制御／数値演算）

表3 実行時間比較 (DFM : PE / SM = 1)

program	V (msec)	M (msec)	DFM		V / DFM		M / DFM	
			C (msec)	L (msec)	V/C	V/L	M/C	M/L
ssort (60)	776,860	88,200	121,188	97,147	6.41	8.00	0.73	0.91
msort (60)	196,833	11,233	27,677	29,932	7.11	6.58	0.41	0.38
qsort (40) 逆順	595,247	33,090	84,788	86,808	7.02	6.86	0.39	0.38
qsort (60) Best	214,800	12,500	26,435	29,678	8.13	7.24	0.47	0.42
union (40,40)	509,223	21,923	45,667	45,948	11.2	11.1	0.48	0.48
compact (60) A	25,823	1,530	8,376	6,064	3.08	4.26	0.18	0.25
compact (60) B	415,133	54,710	118,717	87,820	3.50	4.73	0.46	0.62
arith	29,193	2,118	4,469	4,820	6.53	6.06	0.47	0.44

V : VAXLISP(on VAX / 750)
M : MACLISP(on DEC 2060)
C : Conventional cons
L : Lenient cons

ラムに対しては、 Lenient cons の方の実行時間がやや長くなる。しかし、実行時間の延び率は数～10%程度であり、 cons を分解したことによるオーバヘッドの影響は少ない。

表3に示す通り、単一プロセッサとしてのデータフローマシンのリスト処理性能は、対 VAXLISP 比で約7倍，対 Maclisp 比で 0.4 ～0.5 倍である。データフローマシンには市販の TTL ロジックを使用している点を考慮すると、方式・ハードウェア技術に見合った性能を達成していると言えよう。

6．2 動作特性

表4は、評価プログラム実行時の FUU, SMU の稼働率 (efu, smu)，FUU のアイドル率 (idl) 及び call ／ SM 命令の発行回数を示したものである。表中、cdr-w は cdr 命令の実行が値待ちのため保留される回数を示す。 Lenient cons では、先行制御が可能となるため、パイプライン効果により全般に FUU の有効稼働率が向上する傾向にあり、この効果は ssort 及び compact に顕著に現れている。この場合、 cdr-w の回数が示す通り、殆どの cdr 命令の実行が待ちに入るが、実行時間への影響は少ない。 SM アクセスのない sigma 計算の場合、稼働率が80%に達しているが、その他のプログラムでは稼働率が低下している。この理由としては、

・ SM ネットワークを経由するため、 SM 命令のターンアランドが長い（ car 命令のターンアランドはバス方式で34クロック，ネットワーク方式では56クロックかかる），

・SM命令の頻度が他の命令に比べて低く、パイプが十分に詰まっていない，

ことが主な要因と考えられる。

6．3 DFM実験機の性能〔9〕

n個迄の素数を求めるプログラム（ループで記述したもの）を実験機上で走行させた時の実測時間を図13に示す。

n=50000 の場合、DFM ／VAXLISP ／Maclisp の実行時間はそれぞれ、31.3, 108.5, 9.8秒であり、DFMの性能は VAXLISP の3倍，Maclisp の 1／3 程度を示している。表4に示した値よりも低くなっている理由は、ループ本体中の命令の並列度が低いこと、及びループの入り口で毎回同期をとっていることによる。 ループの展開 (unfolding)，コンパイラの最適化，PE-SM 間結合方式の改善等により、さらに性能は上がるものと思われる。

7．あとがき

高並列記号処理マシン・プロトタイプ実験機DFMの試作及びそのRTLシミュレータによる評価を通じ、以下の点を明らかにした。

① データフロー制御により、粒度の小さな関数を単位とした超多重処理が可能であり、1PEでも循環パイプライン処理による処理の高速化が達成される。。

② マルチプロセッサ化の効果は再帰関数の並列処理構造に依存するが、木評価型・線型評価型のいずれの構造に於いても関数間の並列処理とパイプライン処理による超多重処理効果により、高い台数効果が得られる。

表4 動作特性 (PE/SM = 1)

program	operation rate (%)			frequencies			
	idl	efu	smu	call	cons (C) / gcell (L)	car / cdr	cdr-w
ssort (60) C	52.0	48.0	4.6	1891	1830 / 0	1830 / 1830	0
L	28.2	71.8	1.3		0 / 1830		1711
msort (60) C	33.8	66.2	7.2	706	641 / 0	1235 / 772	0
L	25.3	74.7	12.9		0 / 641		315
qsort (60) C	43.5	56.5	4.8	720	479 / 0	479 / 479	0
Best L	33.0	67.0	8.2		0 / 479		18
union C	25.1	74.9	7.7	81	40 / 0	1640 / 1640	0
(40,40) L	32.7	67.3	5.4		0 / 40		0
compact C	53.7	46.3	4.8	1891	1830 / 0	1830 / 1830	0
(60) B L	26.4	73.6	12.8		0 / 1830		1711
arith C	37.4	62.6	7.5	543	733 / 0	480 / 449	0
L	24.1	75.9	13.3		0 / 733		38
sigma (1024)	19.5	80.5	0.0	2047	0	0	0

C : Conventional cons
L : Lenient cons

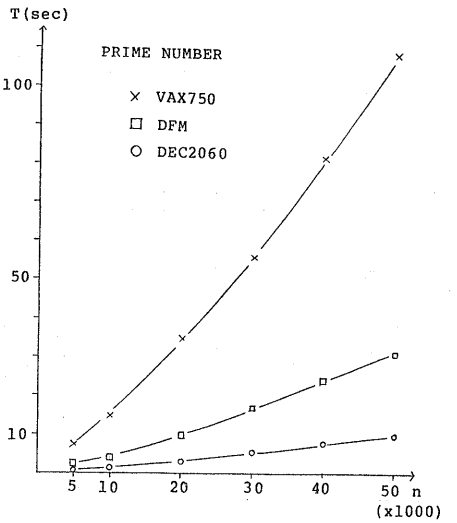

図13 n迄の素数計算

③ 特に、cons の先行評価・リストデータに対する同期機構の導入により、従来型マルチプロセッサでは困難であるリストの分解パイプライン処理が可能となる。この効果は、並列性の阻害要因とされていた線形リスト構造操作に対して顕著に現れ、性能を飛躍的に向上する。

本評価では、プログラム規模・データ数共に小さなものしか測定できなかったが、大規模なプログラムの場合には、少なくとも１００台規模のＰＥに対して係数 0.8 以上の台数効果を達成しうる見通しを得た。

現在、ＤＦＭは１クラスタ内の２ＰＥs ／２ＳＭUs ／１ＣＣＵが稼働しており、実験環境の整備, Valid コンパイラの開発と並行して、評価を更に具体化している段階である。

今後、規模の大きなプログラムに対して、

・ＰＥの割付け方式

・ネットワークを含めたＳＭ方式

等について評価・改良を進め、これらの結果に基づいて６４台規模のＬＳＩ版ＤＦＭを試作し、知識処理マシンを目指した本格的評価を行う予定である。

謝辞

終わりに、ＤＦＭ実験機の試作を担当して頂いた、ＮＴＴ基礎研究所情基部成瀬調査員, 吉田研究主任、並びに、本論文作成にあたり御助力頂いた、情基部第２研究室の小川社員に深謝します。

〔参考文献〕

〔1〕 M.Amamiya, R.Hasegawa, S.Nakamura and H.Mikami : A List-processing-oriented Data Flow Machine Architecture, AFIPS NCC', pp.143-151, 1982.

〔2〕 M.Amamiya, R.Hasegawa and H.Mikami : List Processing with A Data Flow Machine, Lecture Notes in Computer Science, No.147, Springer-Verlag, 1982.

〔3〕 雨宮, 長谷川："リスト構造を並列処理するデータフロー・マシンの方式とその言語, "日経エレクトロニクス" 1984.10.8, pp.219-258.

〔4〕 長谷川, 雨宮："データフローマシンによる並列リスト処理"信学論 (D), J66-D, 12, PP.1400-1407 (昭58-12)

〔5〕 長谷川, 三上, 雨宮："リスト処理向きデータフローマシンアーキテクチャの評価"信学論 (D), J67-D, 9, PP.957-964, (昭59-9)

〔6〕 武末, 三上, 成瀬, 吉田, 雨宮："データフローマシン・プロトタイプ DFM の設計・試作", 東北大通研シンポジウム「新しい計算機アーキテクチャ」 1985 年 7月

〔7〕 雨宮, 長谷川, 小野："データフローマシン用高級言語Valid "「研究実用化報告」, vol.33, no.6, pp.1167-1181, 1984年 6月

〔8〕 Amamiya,M. and Hasegawa,R., "Data Flow Computing and Eager and Lazy Evaluation," New Generation Computing, vol.2, no.2. 1984.

〔9〕 三上："データフロー計算機DFM の性能評価"情処学会大31回 (昭和60年後期) 全国大会