計算機アーキテクチャ　130－15
（１９９８．　８．　６）

受信メッセージ予測法によるノード間通信の高速化 ～予測方式の検討

岩本善行　澤田康雄　大津金光　吉永努　馬場敬信
宇都宮大学工学部

本稿では、メッセージパッシング処理を高速化するために、これまで提案してきた受信メッセージ予測法について、具体的な予測手法を提案・検討している。受信メッセージ予測法とは、並列計算機において、ノードプロセッサがアイドルとなったときに、その時間を利用して次に到着するメッセージを予測し、受信処理などを投機的に先行実行することによって高速化を図る手法である。この手法には、到着するメッセージを予測することが不可欠であり、その予測のための手法を具体的に６種類提案している。また、３種類のプログラムのメッセージ受信履歴を用いてそれぞれの予測手法を用いた場合の予測成功率を求めた結果を示して考察し、予測項目と予測手法の最適な組み合わせを提案する。

Receiving Message Prediction for High Performance Communication ～Prediction Methodologies

Yoshiyuki IWAMOTO　Yasuo SAWADA　Kanemitsu OOTSU
Tsutomu YOSHINAGA　Takanobu BABA
Department of Information Science, Faculty of Engineering
Utsunomiya University

This paper proposes and examines receiving message prediction for high performance message passing. In the receiving message prediction, when a node is idle, it predicts a message it will receive next, and speculatively executes the process of message reception. We propose six methods for message prediction. Using the execution trace of three programs, we calculate the success ratio for prediction using these methods. Results and discussion are also included in this paper.

1.はじめに

我々の研究室では、並列オブジェクト指向トータルアーキテクチャA-NETに関する研究を行っている[1]。A-NETでは並列オブジェクト指向言語A-NETLを設計し、これを用いてOSや応用プログラムの記述、さらにワークステーション上でネットワークワイドなシミュレーションを行ってきた。また、プロトタイプとして16ノードのA-NETマルチコンピュータを完成させた。現在、このプロトタイプ機上でメッセージ通信性能に関するさまざまな動的評価を行っている[2,3]。

同時に、商用並列計算機として富士通AP1000とNEC Cenju-3を取り上げ、並列言語処理系の実装とその評価を行ってきた。さらに、それぞれの並列ライブラリ、OSのソースプログラムの提供を受け、メッセージ送受信処理部分の分析を進めてきた。

この過程で、メッセージ転送処理において、受信側での受信処理に時間が多くかかること[4]や、アイドル時間を有効利用できることが明らかになってきた。

一方で、計算機アーキテクチャの分野では、機械命令の実行を高速化することを目的としたマルコフ連鎖によるプリフェッチの予測[5]や、分岐予測に対する予測手法の検討[6]、データベースに対する投機的な検索など、投機実行に関する多くの研究がなされており、その有効性も確認されている。

これらを背景として、我々はこれまでに、これから到着するであろうメッセージをアイドル時間を利用して予測し、その結果に基づいて受信後の処理を投機的に実行することによって、高速化を試みる受信メッセージ予測

法を提案している[7]。

本稿では、最初にA-NETLのメッセージ転送機能とその実現方法について簡単に述べる。次に、受信メッセージ予測法について説明した後、その予測方法の提案と、ベンチマークプログラムによる各予測方法の予測成功率について議論する。

2.A-NETのメッセージ転送機能

受信メッセージ予測法は計算機のアーキテクチャに依存しない高速化手法であり、A-NETマルチコンピュータ、AP1000 におけるネイティブなライブラリとMPIライブラリ、さらに、MPIを使用したワークステーションクラスタ上に実装中である。ここでは、今回受信メッセージ予測法を実装し、また、予測方法の評価のために、A-NETマルチコンピュータにおけるメッセージ転送機能を簡単に説明する。

2.1 メッセージ転送の概要

A-NETマルチコンピュータはメッセージパッシング型の並列計算機である。A-NETL言語では過去型／現在型／未来型のメッセージ送信と、現在型／未来型メッセージに対するリターンメッセージ送信命令の計4種を定義し、A-NETLプログラム内で各命令は明示的に記述する。

過去型メッセージ転送命令は、コンパイラによって機械命令sendPにコンパイルされる。このとき、コンパイル時に得られたオブジェクトID、セレクタIDなどがオペランドとなる。この機械命令が実行されると、PE(Processing Element)のマイクロプログラムは、メッセージを転送するために必要な受信側のオブジェクト／セレクタID、引数等を含んだメッセージを生成する（図1）。これをルータとの共有メモリ領域に書き込み、ルータに通知後、次の機械命令の実行に移る。ルータは

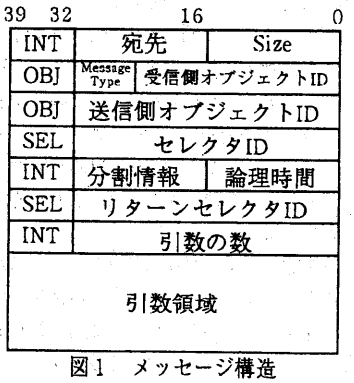


図1　メッセージ構造

この通知を受け、転送先オブジェクトに向けて、最短距離となるような隣接ノードのルータにメッセージを転送する。

受信側となったノードのルータは、このメッセージを受け取ると、マイクロプログラムに対する割り込みフラグをセットする。これを認識したマイクロプログラムは、あらかじめ設定されたOSのメッセージ処理プログラムを起動する。このOS内では、メッセージを置くための領域確保やルータとのやりとりを行ったり、メッセージを解読し、指定されたユーザメソッドを起動する。

現在型メッセージ送信(sendN)、未来型メッセージ送信(sendF)においても、過去型と同様なシーケンスでメッセージを送出する。ただし、現在型メッセージ送信では命令実行直後にフューチャトラップを発生し、返値メッセージが到着するまで他のメソッドを実行するか、キューに起動待ちのメソッドがない場合はアイドルとなる。未来型メッセージ送信では、返値が必要になった時にフューチャトラップが発生し、OSに処理が移行する。

リターンメッセージ送信においても、過去型メッセージ送信と同様のシーケンスであるが、メッセージ構造のリターンセレクタIDが不要となるため、メッセージサイズは1ワード少なくなるという違いがある。

2.2 メッセージ受信処理

ルータは、他ノードからのメッセージを受信すると、PEとの共有メモリに到着したメッセージのサイズを書き込むことによって、ハードウェア的にメッセージ到着フラグをセットし、マイクロプログラムに通知する。

マイクロプログラムは機械命令境界、または、アイドル状態にあるときに、このフラグをチェックし、メッセージが到着している場合にはOSを起動し、メッセージ受信処理を開始する。OSでは、

a) 到着したメッセージに対する処理
b) メッセージの認識
c) メソッドを起動するための処理
d) ユーザメソッドの実行

の4段階で、メッセージを処理する。

3.受信メッセージ予測

3.1 処理の流れ

通常のメッセージ通信時の流れを図2の最上段に、受信メッセージ予測を導入したときの流れを図2の中段に示す。通常実行時には一般的に、(1)ユーザプログラムが終了(terminate)した場合と、(2)他ノードへの現在型メッ

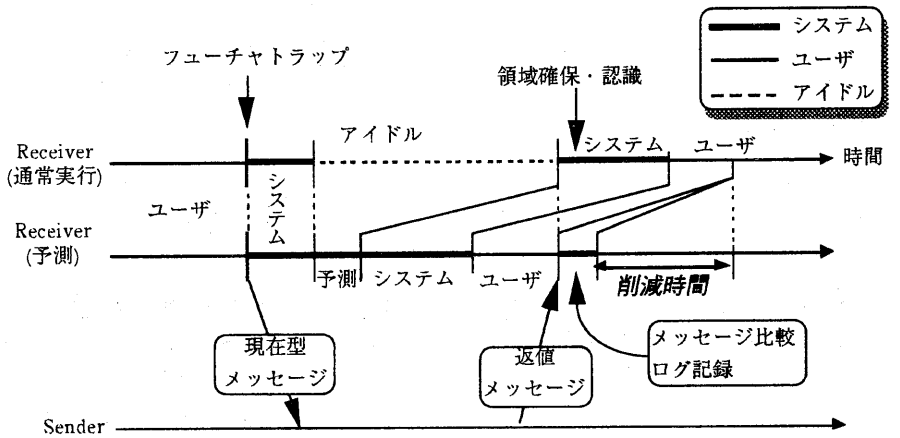


図2　受信メッセージ予測時の流れ

セージを送信しフューチャトラップが発生した場合とに、システム(OS)に制御が移り、その後処理を行う。この処理の終了後、次に実行されるべきコンテクストがない場合、アイドル状態(メッセージ待ち状態)となる。他ノードからメッセージが到着すると、2.2節で述べたように、そのメッセージに対する領域確保を行い、起動すべきメソッドを認識後、実際にユーザメソッドが実行される。

AP1000の場合でも、l_arecv命令を使用したときに、この命令の引数としてタスクIDなどを指定し、そのメッセージが到着していない場合は、メッセージが到着するまでアイドルと同様な状態となる。

受信メッセージ予測法では、これらのアイドル状態を利用して、これから送られてくるであろうメッセージを過去の履歴等から予測し、OSでの受信処理や以降に続くユーザプログラムの中で実行可能な部分を先行実行することによって、予測ヒット時の高速化や、予測失敗時においても先行実行によるキャッシュの先読みなどによる高速化を目的としている。このために、これから到着するメッセージを予測しなければならないが、この予測方式としては、

1) 直前のメッセージのみを参照する
2) 過去のメッセージの平均や発生頻度を参照
3) メッセージ発生系列のマルコフ連鎖
4) 前回の実行が完結したときのトレース

などによる手法が考えられる。1)、2)、3)、4) の順で実装が複雑になり、予測に必要となる時間も長くなるが、的中率も上がることが期待される。

3.2予測方法の検討

受信メッセージ予測法を実現するために、必要となるメッセージの予測項目や予測方法について以下に述べる。

3.2.1 予測項目

2.1節で述べたように、A-NETマルチコンピュータにおけるメッセージパケットは、図1のような構成である。これらの項目の中から実際に予測するのは、1)メッセージサイズ(図中のSize)、2) 過去型／現在型などのメッセージの型(同じくMessage type)、3) メッセージの送信元(同送信側IDの下位16ビット)、4) 起動すべきメソッド(同セレクタID)の各項目とする。

1)のメッセージサイズを予測することによる効果としては、メッセージ到着前にあらかじめ格納領域を確保することが可能となることや、引数の数が計算可能となることなどが挙げられる。この値は7 words(1wordは5bytes)を最小のメッセージサイズとして、引数の数により1ワード単位で変化するが、実際に到着したメッセージのサイズが予測して確保した領域のサイズ以下であれば、予測失敗時でも再確保する必要はない。

3)のメッセージの送信元については、2)のメッセージの型と関連し、現在／未来型メッセージにおいて返値を返すノードを特定することが可能となる。さらに、4)の起動すべきメソッドを特定することにより、その起動準備として一時変数(自動変数)領域の確保や、受信側オブジェクトIDの特定が可能となる。

一方で、メッセージパケットに含まれるこれら以外の項目は、

i) 宛先は自ノード番号である
ii) 分割情報については、今回は分割済みの1メッセー

ジのみを予測するため必要ない

iii) 論理時間は評価用に使用される時間であり、通常の実行に影響しない

iv) 引数は、同期通信などのメッセージを除けば、A-NETではリストなどの構造体の転送もサポートしてるため、広範囲で変化し、予測は難しい

などの理由により予測は考えない。

3.2.2 予測方法

3.1節で述べた基本的な4種類の予測方法と上記の予測項目から、具体的に考えられる予測方法としてここでは、

a) 直前に到着したメッセージのみを参照する（直前）

b) これまでに到着したメッセージの最大値（最大値）

c) これまでに到着したメッセージの平均値（平均値）

d) 最も回数が多かったもの(最頻値)

e) マルコフモデルに基づく連鎖（マ連鎖）

f) 同プログラムを前回実行したときのログ（ログ）

の6種類を考える。

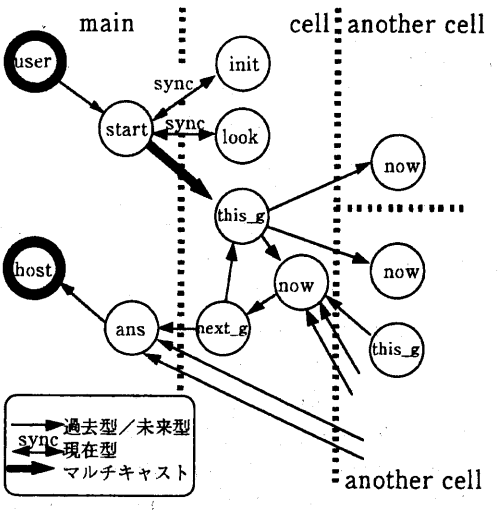
a)の直前参照はあらゆる項目について使用可能であり、もっとも容易に実装可能であるが、複雑なプログラムになった場合はその成功率は低下する。b) は、メッセージサイズを予測するために使用可能であり、予測値以下であれば予測成功となるため、その成功率は高いと考えられる。また実装も容易であり、予測するために必要となるメモリ量も1 wordで十分である。c) も実装が容易であり、メモリも少なくて済む。引数の内容によっては使用可能となるが、今回のようなメッセージヘッダを予測しようとした場合には、応用範囲は非常に限られる。d)の最頻値(mode)は、a)と同様にあらゆる項目に使用可能であるが、予測対象となる項目の出現回数を記録するために、予測対象項目数に比例したメモリが必要であり、予測時にはそのメモリから出現回数が最多の項目を検索すればよい。e)もあらゆる項目に対して使用可能であり、統計的な手法としてa)～d)に比べて最も予測成功率が高いと考えられる。しかし、n回前に到着したメッセージまでを参照するマルコフモデルを使用した場合、予測対象項目数mに対して、$O(m^{n+1})$のメモリ容量が必要となり、さらにその検索にも若干の時間が必要となるため、現実的には$n=2$程度が限度となる。最後にf) の手法は、同じプログラムの引数／データを変えて何度も実行する場合に使用可能となる手法であり、その予測成功率は、変更されたデータによってメッセージの発生に変化がなければ非常に高くなる。また、実行時に与えられるデータによってメッセージが発生しなくなる可能性がある場合には、先の実行で記録されたログとのずれが生じるため、これを補正できれば予測成功率を上げることができる。メモリに関しては、少なくともプログラム実行1回分のメッセージをすべて保存しておく必要があるため、特に粒度の小さなプログラムでは、使用メモリ量が非常に多くなる可能性がある。

4. 予測方法の定量的評価

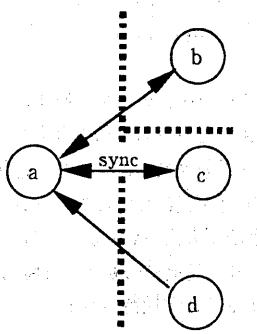
先に述べた6種類の予測方式の予測成功率を調べるために、ここでは3種類のプログラムを取り上げ、ある特定のノードに到着するメッセージ系列から、どの程度予測が成功するかを計算し、その結果について考察した。

4.1 メッセージマップ

A-NETマルチコンピュータでは1つのノード上に複数



(a) Life Gameにおけるメッセージマップ



(b) 予測受信で想定するメッセージマップ

図3　メッセージマップ

のオブジェクト／メソッドが載ることを前提としている。ここで使用したプログラムの例として、LifeGameにおけるノード間／オブジェクト間の通信関係をメッセージマップとして図3(a)に、予測受信において想定しているメッセージマップを図3(b)に示す。予測受信では、ノードのアイドル時間を使用するため、図3(b)に示したような、外部のノードとの通信を想定している。これは、内部で発生したメッセージ(selfcall)では、ノードはアイドル状態とならないため、予測／先行実行を行うことができないことによる。よって、図3(a)に示したメッセージの中で予測対象となるのは、外部からのメッセージとなるnowとansのみである。

4.2 予測成功率の評価と考察

すでにハードウェアのプロトタイプとして動作しているA-NETマルチコンピュータでは、ソフトウェアのシミュレーションではなく、ルータなどのタイミングやネットワーク内でのメッセージ同士の衝突を考慮した動的なメッセージ到着記録を取得することが可能である。これを利用して、ノードの外部から到着するメッセージを実際にプログラムを実行して複数回記録し、それらのメッセージ系列に先の6通りの予測方式を適用したときの予測成功率を計算した（図4～8）。ただし、メソッドおよびメッセージタイプの予測には最大値と平均値は使用できないため計算していない。また、使用したノード数は14～15ノードであるが、計算にはメッセージがより多く(今回のプログラムでは70回以上)到着する特定のノードを使用している。

図4は8 Queensの結果である。このプログラムでは、到着するメッセージが、引数を除くと、毎回非常に類似度が高いため、サイズ・メソッド・タイプの予測成功率はどの方法を用いても80～100%と高くなっている。しかし、この同じようなメッセージが複数のノードからほぼ同時に発生するため、同じノードから連続して到着することは少なく、さらにネットワークやルータなどのタイミングにより到着順番が変動するため、送信元の予測は非常に当たりにくくなっている。このことから、各ノード間でやり取りするメッセージの内容が毎回同じである場合は予測は容易であり、さらに、その発生／受信タイミングも同じような場合は、ほぼ同時に到着したメッセージのうち、予測した送信元と一致するメッセージを優先的に処理するような方法をとることによって予測の成功率を上げることが可能となる。

図5は、台形公式による積分のプログラムであり、数値計算によるマスター-スレーブ型のリダクション処理の

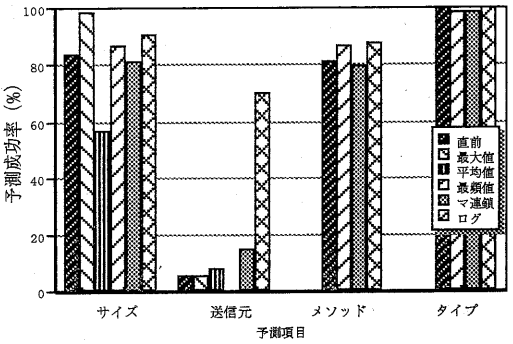


図4　8Qに対する予測方式とその成功率

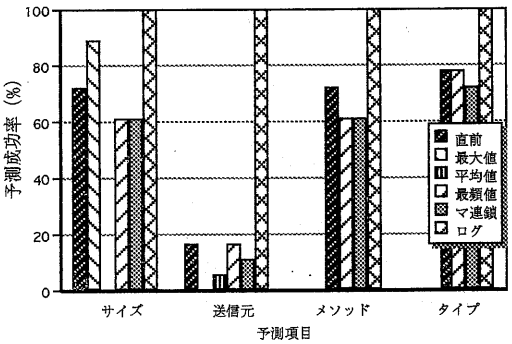


図5　台形公式による積分に対する予測方式とその成功率

例である。この例では、コンパイル時にオブジェクトをノードに割り振るときに使われるアロケータの結果をそのまま使用し、マスタとスレーブのオブジェクトが同一ノード上に割り振られて実行したため、到着するメッセージの規則性が大きく失われてしまい、予測成功率も7割以下になっている。これは、アロケータを、従来のノード間通信時間が最小になるようなアルゴリズムから、性質の異なるマスタとスレーブを別のノードにのせるような条件を追加することによって大きく成功率を上げることが可能となる。

図6と図7は、LifeGameのプログラムである。両者の違いは、図6ではループ回数を4世代、図7ではループ回数を倍の8世代まで行った結果である。図6から図7への変化を見てみると、直前参照予測や前回実行ログ予測では予測成功率がほとんど変化していないが、最頻値を使用した場合やマルコフ連鎖を利用した場合では予測成功率が上昇している。これは、似たようなメッセージ系列が何度も繰り返されることによって、連鎖が強化された結果である。

最後に、これまでに使用した4種類の結果を平均したものが図8である。ここで使用しているベンチマークプ

ログラムは、複数回実行したときもほぼ決定的な動作をするため、前回実行のログを使用した予測が非常に高い成功率となっている。ただし、予測に必要なメモリ容量等を考慮した場合、メッセージサイズの予測には最大値、送信元にはマルコフ連鎖、メソッドとメッセージタイプには直前参照または最頻値の各予測方式を使用するのが適するといえる。

5.おわりに

本稿では、メッセージ転送処理を高速化するための手法として受信メッセージ予測法について説明し、メッセージの予測方式として6種類の手法を提案した。さらに、各手法を予測内容と関連させて、実際のプログラムのメッセージ受信記録を元にした、定量的評価を示した。

今後は、今回の評価結果を実験的に評価するために各予測手法をA-NETマルチコンピュータやAP1000などのアーキテクチャに実装し、今回の評価の正当性の検証と、実際に予測に必要となる時間・メモリサイズなどの評価を行う予定である。

謝辞

本研究は、一部文部省科学研究費（基盤(C)課題番号09680324, 基盤(B)課題番号10558039, 奨励(A)課題番号09780237）、並列・分散処理研究推進機構の援助による。

参考文献

[1] 吉永努 馬場敬信:並列オブジェクト指向トータルアーキテクチャA-NET－マルチコンピュータ開発と言語実装の現状－,情報処理学会第52回全国大会,6-97(1996).

[2] 澤田東, 阿部大輝, 広田守, 岩本善行, 吉永努, 馬場敬信: A-NETマルチコンピュータの通信性能, JSPP'97, pp13-20 (1997).

[3] 岩本善行,吉永努,馬場敬信:言語レベルからみたA-NETマルチコンピュータのメッセージパッシング性能, 情報処理学会研究報告(SWoPP '96), 96-ARC-119, pp.1-6(1996).

[4] Y.Iwamoto, K.Ooguri, T.Yoshinaga and T.Baba: A Comparison of Communication Performance in the NEC Cenju-3 and FUJITSU AP1000, Proc. of the FIRST CENJU WORKSHOP, pp60-64(1997).

[5] D.Joseph and D.Grunwald: Prefetching using Markov Predictors, Proceedings of the International Symposium on Computer Architecture (ISCA'97), pp.252-263, June (1997).

[6] 児島彰, 弘中哲夫, 高山毅, 藤野清次: 複数分岐での投機的実行の有効性, 情報処理学会研究報告,97-ARC-123-10, pp55-60(1997).

[7] 岩本善行, 澤田東, 阿部大輝, 澤田康雄, 大津金光, 吉永努, 馬場敬信: メッセージ転送処理の高速化法とその評価, 情報処理学会論文誌, Vol 39, No.6, pp.1663-1671(1998).

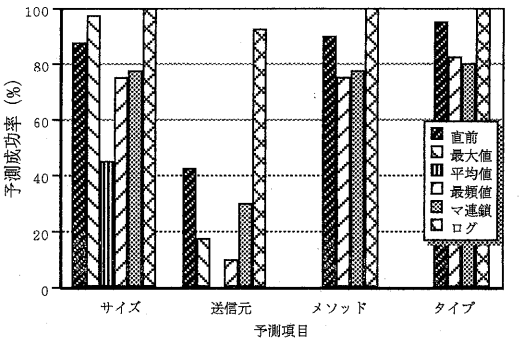


図6　LGに対する予測方式とその成功率(4世代)

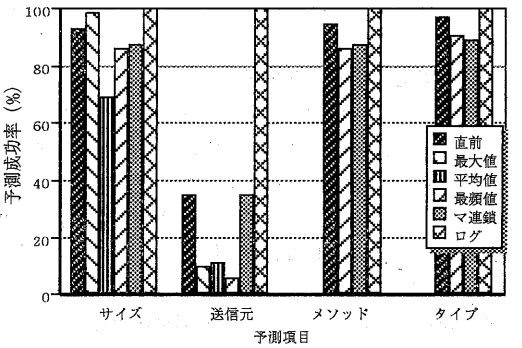


図7　LGに対する予測方式とその成功率(8世代)

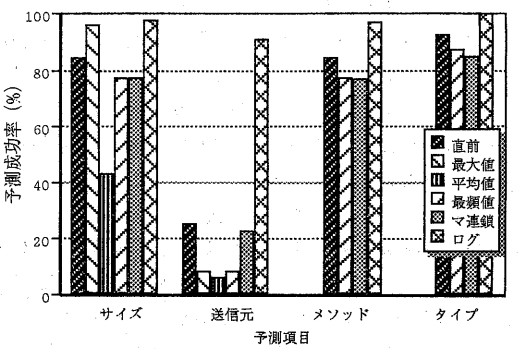


図8　予測方式とその成功率の全結果の平均