汎用超並列オペレーティングシステム SSS-CORE の ユーザレベル通信同期機構

A High-Speed User-Level Communication Mechanism for the General-Purpose Massively-Parallel OS: SSS–CORE

松本 尚† 平木 敬†
Takashi MATSUMOTO Kei HIRAKI

†東京大学 大学院理学系研究科 情報科学専攻
Department of Information Science, University of Tokyo

概 要

本稿では、ワークステーションクラスタ上で開発中の汎用超並列オペレーティングシステム SSS–CORE の構成を述べる。次に、効率の良い汎用並列実行環境の実現に不可欠な、ノード間の高速かつ保護され仮想化されたユーザレベル通信および同期であるソフトウェアメモリベース通信の高速実装技術を説明する。最後に、100BaseTX を用いて実装したメモリベース通信と高並列計算機のユーザレベル通信の基本性能について比較する。

1 はじめに

並列計算機システムへの汎用性（マルチユーザ・マルチジョブ）導入による保護機能の実現や実資源の仮想化は並列処理性能の大幅な低下の原因となるため、現状の並列計算環境および並列計算機は基本的にバックエンドプロセッサとしてバッチ処理で使用されることが多い。つまり、ソフトウェアによる様々な最適化方式の適用および軽い通信同期手法がシステムの独占的使用を要求し、システムの柔軟かつ汎用的な使用形態と相容れないのである。

このような状況を改善するため、我々は 1989 年よりすべての CPU から等距離でアクセス可能な集中共有メモリを持つ Uniform Memory Access（UMA）型並列計算機を対象にして、ユーザによる最適化を支援する汎用オペレーティングシステム核 SS–CORE [1]を研究開発してきた。SS–CORE の研究においては議論を単純化するために、UMA 型並列計算機に対象を限定し、実プロセッサの資源管理に的を絞った研究を行った。しかし、UMA 型並列計算機よりも多くのプロセッサを接続可能であり、構成のスケーラビリティが高い NUMA（Non-Uniform Memory Access）型並列計算機*が現在実用化されつつある。そこで、汎用性を持った使用形態においても NUMA 型並列計算機を高い効率で使用するためのキーとなるオペレーティングシステム **SSS–CORE**（**Scalable SS–CORE**）[2]の研究開発を 1994 年より開始した。本稿では、NOW 版 SSS–CORE Ver.1.0 の構成と SSS–CORE において高速かつ仮想化され保護されたユーザレベル通信同期を可能とするソフトウェアメモリベース通信の高速実装技術を説明する。最後に、100BaseTX を用いて実装したメモリベース通信と高並列計算機のユーザレベル通信の基本性能について比較する。

2 SS–CORE

SSS–CORE に関して述べる前に、SSS–CORE のベースとなった SS–CORE に関する研究について簡単に紹介する。UMA 型並列計算機用の OS 核である SS–CORE はユーザによる並列実行の最適化の

* ここではワークステーションクラスタ（Network of Workstations: NOW）、分散メモリ型並列計算機、分散共有メモリ型並列計算機等の総称

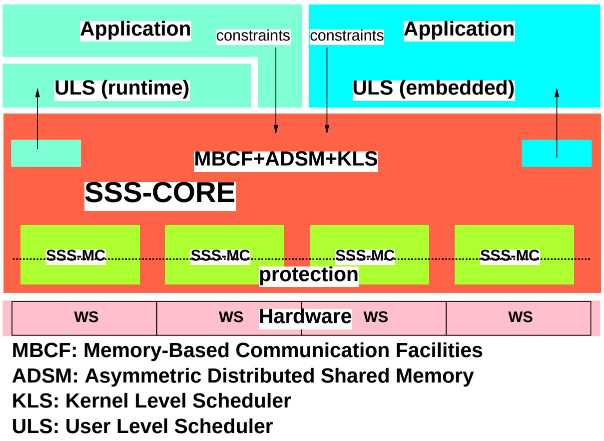

図 1 **NOW 版 SSS–CORE** の構成

ために、

- 「時分割とパーティショニングの併用によるマルチジョブ」
- 「資源のグループ単位管理」
- 「保護の下でユーザへ資源を最大限に解放」
- 「資源割り当て情報をユーザへ開示」
- 「カーネルレベルスケジューリングへのユーザ制約／ヒントの採用」

といった基本方針 [3] [4]を持つ。この基本方針に基づいた研究の結果、プロセッサ資源管理に関して、世界に先駆けて以下の方式／機構を考案した。

- カーネル／ユーザの資源割当状況や実行状況によるスケジューリングオプションつきのスピンウェイト: Snoopy Spin Wait (SS-wait†) [5]
- カーネルにプリエンプトされたプロセッサコンテクストのユーザレベルの復旧 [5]
- 実プロセッサグループをユーザ（プロセス）に割り当て、グループ内のプロセッサスケジューリングはユーザに一任するユーザ／カーネルの二階層スケジューリング方式 [5]
- 並列プログラムの性質による分類とそれを利用したカーネルレベルスケジューリング [1]

3 SSS–CORE

SSS–CORE の資源管理方針は SS–CORE の基本方針を受け継ぎ、 SS–CORE の研究において考案された機構／方式に関しては対象計算機システムを NUMA 型に変更したことに伴う拡張を行っている [2] [6]。また、システム全体としての資源管理を階層構造を持つ管理構造体で行うことにより、プロセッサ台数（ノード数）が増大しても、管理コストがリニアに増大しないようにしている。

† SS–CORE の名前の由来となる同期方式

SS–CORE では集中共有メモリを対象計算機側に仮定していたため、ユーザレベルの通信同期はこの共有メモリを介して行うことが可能である。また、メモリにはメモリ保護，論理／物理のアドレス変換，仮想記憶サポートがあるために、 UMA 型並列計算機における共有メモリを介した通信同期はオペレーティングシステムの直接的な仲介なしに保護と仮想化を実現できる。さらに、ハードウェアで実現された集中共有メモリのアクセス速度は速いため、ユーザレベルの軽い仮想化され保護された通信同期が集中共有メモリを用いるだけで実現できる。

SSS–CORE では、対象計算機システムをスケーラビリティが高く低コストな NUMA 型並列計算機に設定している。さらに、 SSS–CORE では通信同期用の特別なハードウェアを仮定しない方針を立てており、今回実装した NOW 版 SSS–CORE Ver.1.0 では市販の量産ワークステーションを LAN 用のネットワークで接続したシステムを対象としている。このため、軽いユーザレベルの保護され仮想化された通信同期を実現する方法が大きな問題となる。この問題に対して、 SSS–CORE ではソフトウェアメモリベース通信（Memory-Based Communication Facilities: **MBCF**） [8] [9] [10]で解決を図っている。

MBCF はシステム全体をカバーする仮想化され保護された共有メモリ空間をユーザに提供するが、遠隔ノードのメモリに対するキャッシュシステムは提供しない。分散共有メモリ用の専用ハードウェアを持たない環境において、キャッシュシステムを含む効率の良いソフトウェア分散共有メモリ（ソフトウェア DSM）を実現するために、松本は新しいソフトウェア DSM の枠組である非対称分散共有メモリ（Asymmetric Distributed Shared Memory: **ADSM**） [8] [9]を考案した。

図 1に NOW 版 SSS–CORE の構成を示す。各ワークステーション上に SSS–MC(Micro Core) と呼ばれるオペレーティングシステム核が常駐しており、ユーザ／カーネル共にノード間の通信同期を MBCF を用いて行う。なお、 SSS–MC には TCP/IP および UDP/IP も実装されているため、 SSS–CORE の各ノードは既存の計算機システムと通信を行うことができる。カーネルレベルスケジューラ（**KLS**）はノードを跨りユーザ制約に基づいたスケジューリングを行う。ユーザがカーネルの資源割り当て状況を低コストで把握できるように階層化情報収集開示機構が MBCF を使って実装されている。ワークス

テーション間には ADSM により分散共有メモリが実現される。

4 MBCF の高速実装技術

分散メモリ実装の並列計算機や NOW においても、集中共有メモリを持つ並列計算機と同様にメモリアクセスによって高速かつ保護され仮想化されたユーザレベル通信同期が可能になるように、松本は従来のページ管理機構を遠隔メモリアクセスに拡張した Memory-Based Processor (**MBP**) [7]を考案した。

しかし、 SSS–CORE の開発に当たって、通常の通信ハードウェアしか持たない分散メモリ実装の並列計算機環境（NOW を含む）において実現可能な、高速かつ保護され仮想化されたユーザ通信／ユーザ同期を考案する必要にせまられた。これに対して松本が出した解答が MBCF である。

MBCF は MBP の動作と機能を付加ハードウェアなしにソフトウェアのみで実現している。このため、通信同期の端緒は単なる load/store 命令ではなく、通信相手（タスク／プロセス）の論理的な識別子と通信相手における操作対象論理アドレスを含んだ通信パケットをユーザレベルで構成し、 MBCF 送信用のシステムコールを実行する。システムコール内で汎用の通信ハードウェア（100BaseTX インタフェース等）を利用して送信する。受信側は受信割り込みルーチン内でパケット内の情報を利用して通信先タスクの操作対象アドレスを直接操作する（返答が必要であれば返答を送信する）。 MBCF は以下のような最新技術やソフトウェア技巧を用いて、この動作を高速に実現している。なお、以下の記述における数値は Sun Microsystems 社 の SS20（SuperSPARC 85MHz）を使用して実装した MBCF 上での実測値である。

- 論理アドレスによる通信相手空間の直接操作
  固定されたキューを介したデータ通信を行わないため、通信データのコピー回数ならびにキュー操作を大幅に削減できる。逆に、キュー構造が必要な場合はメモリの任意の位置にキューを設定することができる（Memory-Based FIFO）。
- 高性能プロセッサのローカル処理の高速性
  キャッシュミスが少なくなるように最適化されたプログラムを用意して MBP の機能をメインプロセッサによって実現すれば、処理オーバヘッドは大きくない。
- 軽い送信専用システムコール
  既存の OS（UNIX 等）の通信用システムコールはオーバヘッドが大きい。その理由は、通信パケットのコピー回数の多さと、通信と無関係な処理を行っているからである。 SSS–CORE の MBCF 実装では ether の通信保証に必要な最低限の一回のコピーで実装されており、通信と無関係な処理はまったくなされていない。また、専用システムコールでノンブロッキングであることが保証し、プロセッサコンテクストの退避／復旧も最低量（システムコール内で使用する資源のみ）で済ませている。 4byte の遠隔書き込みの送信システムコールのオーバヘッド（ether の順序保証、到着保証のプロトコルを含む）は **3.4**μsec である。
- 軽い専用受信割り込みルーチン
  送信専用システムコールと同様に、余分な処理を一切せず MBCF に特化した受信割り込みルーチンを用意した。また、割り込み処理ルーチンの共通化といった高速化を阻害するプログラミング手法は一切排除した。 4byte の遠隔書き込みの受信側での受信割り込みルーチンのオーバヘッドは **6.4**μsec である。
- 複数コンテクストの混在できる **TLB**
  最近の高性能プロセッサの TLB はコンテクスト識別子を含んでおり、複数のコンテクストが混在できる。このため、異なるアドレス空間（コンテクスト）を操作する MBCF コマンドが割り込みで実行される際に、多くとも MBCF で操作するアドレスの TLB をセットするだけで済み、 TLB の内容はほとんど影響を受けないため割り込み後の実行速度も低下しない。
- 軽いアドレス空間切替えハードウェア
  現在のコンテクスト識別子を更新する CPU 命令は 1 命令であり、実行クロック数も小さい。
- ページエイリアス機能
  ページエイリアス機能と前出の複数コンテクストの混在できる TLB を活用することにより、完全なアドレス範囲チェックをソフトウェアによって行わなくても、 MBP と同レベルのメモリ保護を行うことができる [10]。
- 物理アドレスタグを持つプロセッサキャッシュ
  ページエリアスを利用する場合に、コンシステンシを保つために必要な条件である。また、各ノードのワークステーションがマルチプロセッサ構成になっていてもキャッシュコンシステンシを保つことができる。

表1　通信性能比較（MBCF vs MPPs）

Machine	Peak band (Mbytes/s)	Round-trip latency(μs)
SP–1 + MPL/p	8.3	56
Paragon + NX	7.3	44
CM–5 + Active Message	10.0	12
SS20 cluster + SSAM	7.5	52
SS20 cluster + MBCF	**11.2**	**49**

- 多機能かつ機能固定の受信割り込みルーチン
  受信割り込みルーチン内つまりカーネルモード内で処理を行うために、ユーザプログラムを受信ルーチンとして使用することを許していない。保護と仮想化の下でユーザプログラムを受信ルーチンとして使うためには余分なコピーの発生が避けられない。

100BaseTX 版および 10BaseT 版の MBCF の実装に関する詳細は文献 [10]を参照されたい。

5　MBCF と MPP の基本通信性能比較

以下に、100BaseTX を使って実装された MBCF と高並列計算機（MPP）のソフトウェアを含んだ通信性能の比較を表 1に掲げる。表中の MBCF 以外の性能値は文献 [11]から引用した。なお、ここに掲げた MPP のユーザレベル通信は保護および仮想化の度合が MBCF と比べて低く[‡]、機能的にも大幅に劣るものであり、本来は定性的側面から比較対象から除外されるべきものである。

表中の SSAM と MBCF 以外は専用通信ハードウェアを持つ並列計算機システムであり、通信ハードウェア自体の性能は今回の MBCF 実装が使用した 100BaseTX よりも大幅に高い。SSAM [11]は MBCF と同様にワークステーションクラスタベース（ただしネットワークは 156Mbps ATM）のソフトウェアによる通信機構である。ただし、SSAM はパケットの到着保証と順序保証のプロトコルを省略しているため、実用化時にはこの値よりも悪くなることが予想される。

我々の　MBCF は CM–5 上の Active　Message (AM) に Round-trip タイムで劣っているが、MBCF が一切特殊ハードウェアを必要としないこと、AM/CM–5が仮想化や保護に十分に対応していないことを考慮すれば、我々の MBCF の方式および 100BaseTX 上の MBCF の実装が非常に優れていることが判る。現在、MBCF の通信能力（Peak bandwidth）は 100BaseTX の性能によって上限が規定されているため、通信ハードウェアとして他の MPP と同程度に高速なものを使用すれば、さらに大幅な性能向上が期待できる。

‡　ノードを跨るギャングスケジューリングが強制されたり、通信ネットワークが一つのアプリケーションに占有されたりする。

6　結　　論

保護と仮想化の徹底した高速ユーザレベル通信であるソフトウェアメモリベース通信（MBCF）を、そのサポートオペレーティングシステム SSS–CORE と共に実装した。100BaseTX 版 MBCF の基本性能は、MBCF の現実装より大幅に転送能力の高い通信ハードウェアを持ち仮想化や保護のレベルが低い高並列計算機のユーザレベル通信能力と比べて、優るとも劣らないものである。

謝　　辞

本研究は情報処理振興事業会（IPA）が実施している独創的情報技術育成事業の一環として行なった。

参考文献

[1] 松本, 根岸, 渦原, 森山: 粒度に基づいた並列計算の分類法とマルチプロセッサの資源管理法について. 日本ソフトウェア科学会第 7 回大会論文集, pp.133-136 (October 1990).

[2] 松本 尚, 平木 敬: 汎用並列オペレーティングシステム SSS–CORE の資源管理方式. 日本ソフトウェア科学会第 11 回大会論文集, pp.13–16 (October 1994).

[3] 松本 尚: 細粒度並列実行支援マルチプロセッサの検討. 信技報, CPSY89-37, pp.37–42 (August 1989).

[4] 松本 尚: 細粒度並列実行支援マルチプロセッサの検討. 情報処理学会論文誌 Vol.31 No.12, pp.1840–1851 (December 1990).

[5] 松本 尚: マルチプロセッサ上の同期機構とプロセッサスケジューリングに関する考察. 計算機アーキテクチャ研究会報告 No.79-1, 情報処理学会, pp.1–8 (November 1989).

[6] 信国, 松本, 平木: 汎用超並列 OS *SSS–CORE* における資源管理機構 — スケジューリング方式とメモリ置換方式の性能評価 —. 並列処理シンポジウム JSPP '97 論文集, pp.21–28 (May 1997).

[7] 松本 尚, 平木 敬: Memory-Based Processor による分散共有メモリ. 並列処理シンポジウム JSPP '93 論文集, pp.245–252 (May 1993).

[8] 松本 尚, 平木 敬: 汎用超並列オペレーティングシステム: SSS–CORE — ワークステーションクラスタにおける実現 —. 情報処理学会研究報告 96–OS–73, 情報処理学会, Vol.96, No.79, pp.115–120 (August 1996).

[9] 松本, 駒嵐, 渦原, 平木: メモリベース通信による非対称分散共有メモリ. コンピュータシステムシンポジウム論文集, 情報処理学会 pp.37–44 (November 1996).

[10] 松本 尚, 平木 敬: 汎用並列オペレーティングシステムにおける資源保護と仮想化. 情報処理学会研究報告 97–OS–75, 情報処理学会, Vol.97, No.56, pp.37–42 (June 1997).

[11] T. von Eicken, A. Basu, and V. Buch: Low-Latency Communication Over ATM Networks Using Active Messages. *IEEE Micro*, pp.46–53 (February 1995).