

MKng プロジェクトにおけるマルチコンピュータ環境: OS パーソナリティとしての並列処理サーバ†

1 Z-5

齊藤 鉄也¹ 安田 絹子¹ 喜多山 卓郎² 萩野 達也³ 徳田 英幸³¹慶應義塾大学大学院 政策・メディア研究科²慶應義塾大学 SFC 研究所³慶應義塾大学 環境情報学部

1 はじめに

ATM や Myrinet などの高速ネットワークの普及により, これらのネットワークを利用してワークステーションを接続したマルチコンピュータ環境の研究が盛んである。プロセッサやメモリ, ディスクといった, マルチコンピュータ環境の構成要素である各計算機の資源を共有資源と見做すことによる計算機資源の有効利用と, それぞれ独立した計算機から構成される自律性を生かした規模拡張性を実現することがマルチコンピュータ環境の研究の目的である。また, 超並列計算機を持つ計算機資源と, マルチコンピュータ環境を持つそれがほぼ等しいことから, この環境上での並列処理の研究も行なわれている [1]。

我々は上記の目的を実現するために, 次世代マイクロカーネル (MKng) 研究プロジェクト [5] のサブテーマのひとつとして, マルチコンピュータ環境の構築要素となる並列処理サーバを OS パーソナリティとして設計し, 実装を進めている。本論文では, 並列処理サーバの構造と機能について述べる。以下, 第2章ではマルチコンピュータ環境の構築方法とその利用方法を述べ, 第3章ではこの環境の構築要素となる並列処理サーバの構造と機能について述べる。第4章で関連研究を, 第5章で本論文をまとめるとする。

2 マルチコンピュータ環境へのアプローチ

2.1 OS パーソナリティ

マルチコンピュータ環境上で稼働するシステムソフトウェアを実現する方法としては, この環境に適した機能を持つオペレーティングシステム (以下 OS と略す) の開発または改良か, 同様の機能を提供するユーザレベルのサーバやライブラリを開発する, が考えられる。それぞれの方法には, 前者の場合, 目的に特化したサービスの提供やそれによる性能の向上が望める利点がある一方で, OS の開発が困難であるといった欠点がある。同様に後者の場合, ソフトウェア開発は容易であるが, 既存の OS (例えば UNIX) のインターフェイスに制約され, 新しい設計に基づいた実装が困難である。

上記の実装方法を検討し, 我々は, RT-Mach マイクロカーネル [4] 上に OS パーソナリティ (OS サーバ) として並列処理サーバを実現する方法を採用した。この方法は, アプリケーションの開発環境の利用が可能であることと, 柔軟な実装が可能であること等, 利点がある。OS パー

ソナリティはユーザレベルサーバとして実現できるため, マイクロカーネル上で稼働している 4.4BSD Lite サーバ上で開発が可能である。また, マイクロカーネルが提供するカーネルコールを直接利用することができ, 異なる概念や設計に基づく実装が相対的に容易である。

2.2 逐次処理と並列処理の共存

我々は, マルチコンピュータ環境を並列処理だけではなく逐次処理にも利用することを目指している。実際, 並列処理プログラムの数は逐次処理のプログラムに比べて相対的に少なく, 並列処理を行なう場合には並列処理プログラムをこの環境上に移植するか新しく開発する必要がある。このため, マルチコンピュータ環境の計算機資源を有効に利用しつつ, 段階的に並列処理を実現する方法を選択した。最初は逐次処理のソフトウェアを利用し, プロセスを並列度の単位とした粗粒度並列処理を実現する。これは複数のコマンドを同時に行なうといった既存のプログラムを単位とした並列処理である。次に, マルチコンピュータ環境の計算機資源を有効利用して既存のソフトウェアの高速化を行なう。これはマルチコンピュータ環境の資源の利用状況に応じてプロセスを実行する計算機を変更する遠隔実行や, 他の計算機のメモリを利用して高速化し負荷分散を行なう。

3 並列処理サーバ

3.1 構造と機能

並列処理サーバはマルチコンピュータ環境の構築要素であり, その目的はこの環境の計算機資源の有効利用と規模拡張性の実現である。最初に並列処理サーバの概略について述べ, 次にその構成要素であるサーバとエミュレータについて述べる。

並列処理サーバはサーバとエミュレータからなる。サーバはユーザレベルのプロセスである。エミュレータはアプリケーションのアドレス空間内にマップされ, アプリケーションの実行する並列処理サーバへのサービス要求をサーバへ転送する。

サーバの目的はアプリケーションが利用するマルチコンピュータ環境の情報の提供である。特定のサーバが特定のサービスを提供し, そのサービスを利用するすべてのアプリケーションがそのサーバに対してサービスを要求する集中サーバ方式では, 特定のサーバの負荷の増加やサーバのサービスの停止といった状況に対する脆弱性が増加する。この現象を回避するために, どのサーバも同等なサービスを提供し, マルチコンピュータ環境の情報, 例えばプロセスの実行状況の情報は並列サーバ同士が一定時間ごとに交換することにより, 特定のサーバだけが特定の情報を提供する方式を採用しない。これによりアプリケーションは同一ノード上のサーバと通信することで環境全体の情報を利用することができる。サーバの提供する情報はクライアントが更新する必要があるの読み出しのみのアクセス権を設定した共有メモリを通してアプリケーションに提供される。

Multicomputer Environment in the MKng Project:
The Design of a Parallel Processing Server as an OS Personality Module
Tetsuya SAITO
Graduate School of Media and Governance, Keio University
5322 Endo Fujisawa 252 Japan
E Mail: <saimune@mag.keio.ac.jp>

†この研究は, 情報処理振興事業協会 (IPA) が実施している創造的ソフトウェア育成事業「次世代マイクロカーネル研究プロジェクト」のもとに行われた。

エミュレータの目的はサーバが提供するマルチコンピュータ環境の情報を利用して、アプリケーションからの要求に応じた処理をすることである。エミュレータ自身はマルチコンピュータの特定の利用方法を強制することはない。特定の利用方法は、エミュレータに対してマルチコンピュータ環境上の資源の利用方法を指定する機構を利用してアプリケーションまたはユーザが指定する。例えば負荷分散を行なう場合、どのノード上に新しくプロセスを作成するかはユーザまたはアプリケーションによって負荷分散する方法が異なる。あるアプリケーションは負荷の少ないところでのプロセスの実行を要求し、異なるアプリケーションは特定の複数のノードの中でプロセスの実行を要求する可能性がある。それゆえ、特定の利用方法を強制することは望ましくない。

並列処理サーバの構造と機能を図1に示す。

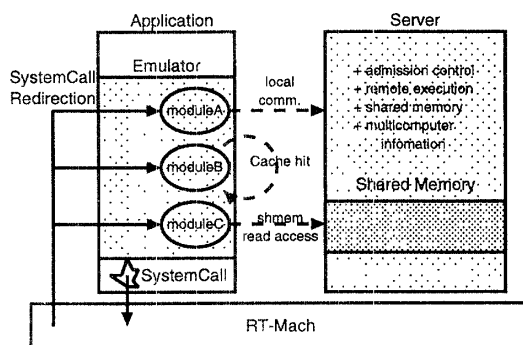


図1: 並列処理サーバの構造と機能

処理は次のように行なわれる。アプリケーションが並列処理サーバのシステムコールを呼び出す。RT-Machのシステムコールリダイレクション機能によりエミュレータに制御が移る。エミュレータはユーザまたはアプリケーションによって指定された処理方法により処理を行なう。処理によってサーバと通信を行なう場合、エミュレータの持つキャッシュを利用する場合、サーバの提供する共有メモリを読み出す場合があり、それぞれの処理方法でサービスを提供する。

3.2 実装方針

並列処理サーバはUNIXシステムコールインターフェイスを持ち、プロセス間通信(以下IPCと略す)の削減、サーバ機能の削減、高速ネットワークの実現、を満たすことを目標とする。それぞれについて以下に述べる。

既存のソフトウェア資産を有効利用するためにUNIXシステムコールインターフェイスを提供する。並列処理用の独自のシステムコールインターフェイスだけを並列処理サーバが提供する選択肢も存在するが、独自インターフェイスだけでは、ソフトウェアを新しく開発する必要がある。そのため、これはマルチコンピュータ環境の計算機資源を有効利用する機能が実現できた後に実装する。

アプリケーションはエミュレータを通して並列処理サーバと通信をする。そのためサーバの持つ機能を削減し、削減した機能をエミュレータ上で実現することにより通信によるオーバーヘッドを削減することができる。また、必

要な情報の通信も共有メモリを利用する。

マルチコンピュータ環境の情報の提供や並列処理に特化したシステムコールの処理などサーバの機能追加要求は多い。しかし、これらの機能追加を行なうと、特定のサービスだけの提供、サーバ機能の増加による通信の増加や、機能の複雑化による段階的な機能追加が困難になる、といった問題が発生する。機能の追加要求に対応するために、追加する場合であってもRT-Machのシステムコールリダイレクションを利用しエミュレータに対して機能を追加するか、またはInterposition Agent[2]に基づき、ツールキット化を計る。これにより様々な機能が交換可能になる。

高速ネットワークはマルチコンピュータ環境を構成するための基盤である。これを実現するため、資源の割り当てや権限の要求を必要とする要求はサーバと通信するが、データの転送要求はサーバと通信せずに、直接デバイスに対して通信を行なう。これによってネットワークを介した通信の高速化を計る。

4 関連研究

Berkeley NOW[1]はマルチコンピュータ環境を構成する計算機のメモリを利用したファイルシステムの改良や高速な通信方法の開発を行ない、計算機資源の有効利用と規模拡張性を実現する方法を採用している。SHRIMP[3]では既存のOSと独自のネットワークインターフェイスを採用しマルチコンピュータ環境の構築を行なっている。これらの研究に対して本研究は、マイクロカーネルを利用しユーザレベルのサーバでマルチコンピュータ環境を構築する点が異なっている。また、集中サーバ方式を採用せず、遍在するサーバ機能の提供、交換可能な追加機能の提供を目指している点でも異なっている。

5 まとめと今後の課題

本論文では、マルチコンピュータ環境の計算機資源の有効利用と規模拡張性の実現を目的とし、これを実現するマイクロカーネル上のOSパーソナリティである並列処理サーバの構造と機能について述べた。並列処理サーバは、サーバがマルチコンピュータ環境の情報を提供し、エミュレータがその情報を利用して計算機資源を有効利用することである。機能の提供方法の特徴には、集中サーバ方式による実現を行わずそれぞれのノード上のサーバが同等の機能を持つこと、サーバの機能の一部をエミュレータ側で実現し通信オーバーヘッドを削減すること、Interposition Agentを利用して様々な利用方法を提供すること、がある。今後の課題としては並列処理サーバの実装を進め、その機能の実現と検証を行なうことである。

参考文献

- [1] T. Anderson, D. Culler, D. Patterson, and the NOW team. A Case for NOW(Networks of Workstations), *IEEE Micro*, pages 54-64, February, 1995.
- [2] Michael B. Jones. Interposition Agents: Transparently Interposing User Code at the system Interface, *Proceedings of the 14th ACM Symposium on Operating Systems Principles*, pages 1-14, December 1993.
- [3] The SHRIMP Project. <http://www.CS.Princeton.EDU/shrimp/>
- [4] Hideyuki Tokuda, Tatsuo Nakajima, and Prithvi Rao: "Real-Time Mach: Towards a Predictable Real-Time System," In *Proceedings of USENIX Mach Workshop*, October, 1990.
- [5] 徳田 他: "MKng: 次世代マイクロカーネル研究プロジェクト," 第53回情報処大論文集, 5B-4, pp. 1-39-1-40 (1996).