2J－2

スレッドレベル動的再構成システムにおけるスレッド間通信の拡張

鈴木信雄　脇　英世
東京電機大学　工学部

1．はじめに

スレッド機構は、高い並列性、自然な並列の論理性、そしてCPU資源の有効活用という利点を活かして様々なアプリケーションにおいて利用されている。

一方、分散配置されたオブジェクトの動的負荷分散手法の一つに動的再構成がある[1][2]。この手法では、協調動作するオブジェクトの組織において、任意の時点に任意の要素を別な組織へ自由に組み入れることを可能としている。従来の動的再構成においては、オブジェクトの単位がプロセスを前提に検討されてきた。これに対し、本研究では再構成の単位をスレッドで実現することに着目し、より柔軟な動的再構成の可能性を検討してきた[3][4][5]。

これまでの検討により、スレッドレベルでの動的再構成の基本構成は実現できたが、スレッド間通信についての取り扱いが課題となっていた。本稿では、スレッド間通信の種類と移送時の対処方法について整理し、グローバルデータ領域を介した通信時においてメモリマップドファイルを使用することを提案する。

2．スレッド間通信の移送

（1）再構成の対象となる組織

再構成の対象となる組織は通常モジュール間の結合を持っている。具体的に結合とはモジュール間通信を意味している。動的再構成においては、モジュールの移動と共に移動後の結合の継続性も保証する必要がある。

（2）スレッドにおける結合

スレッドにおける結合とはスレッド間通信であるが、これには一般的に以下のような手法が存在する。

①同期変数
②グローバルデータ領域（グローバル変数、ヒープ領域等）
③プロセス間通信（ソケット、共有メモリ等）

①については同期変数がグローバルデータ領域を用いて実現されていることから、グローバルデータ領域の問題に帰着できる。また、③についてはプロセス移送時の通信路変更法として、すでに既存の他研究にて提案されている[6]。

したがって、スレッド間通信の移送後の対処としては、グローバルデータ領域に対する処理を検討することにより実現できることとなる。

（3）グローバルデータ領域を介したスレッド間通信の移送

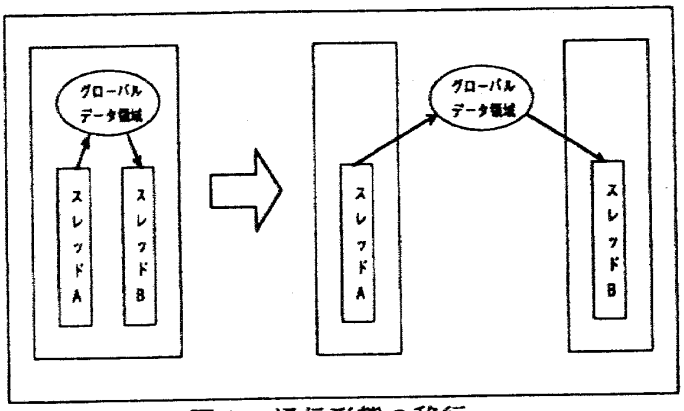

図1　通信形態の移行

スレッドレベル動的再構成におけるスレッド移送時には、グローバルデータ領域を介したスレッド間通信をソケットや共有メモリなどのプロセス間通信にマッピングする必要がある(図1)。これを検討するために、分散配置されたモジュールから共有データを参照する手法である分散共有メモリの手法と対比させて検討する[7]。

まず、分散共有メモリは以下のように分類される。

①分散共有データ機構による分類
・replication：同一データ項目の複数コピーが異なるローカルメモリに存在し、異なるサイトから同じデータを同じようにアクセスが可能となる。
・migration：一時に一つのコピーのみが存在し、排他制御のためにデータを要求サイトへ移動する。
②アクセスアルゴリズムによる分類
・単一リーダ/単一ライタ・アルゴリズム(SRSW)

A Expansion of Interthreads Communication for Thread Level Dynamic Reconfiguration
Nobuo SUZUKI and Hideyo WAKI
Tokyo Denki University

・複数リーダ/単一ライタ・アルゴリズム(MRSW)
・複数リーダ/複数ライタ・アルゴリズム(MRMW)
スレッドレベル動的再構成の場合を考えると、スレッド移送は1対1の移動であるため、共有データ領域はmigrationにより実現し、データのアクセスはSRSWを選択する。

3．メモリマップドファイルを使ったスレッド間通信の移送

グローバルデータ領域を介したスレッド間通信を行っているスレッドを移送した場合、移送元プロセス内の通信相手のスレッドと移送後のスレッドとの間で引き続き通信を行う必要がある。

本機能を実現するために、NFSファイルに対するmmap()を利用する。図2に示すように、移送開始時に移送対象スレッドが他のスレッドと通信を行うグローバルデータ領域を、mmap()によりファイルに関連づける。移送後はそのファイルを双方のノードから参照することにより移送前と同じようにグローバルデータ領域による通信を継続することができる。

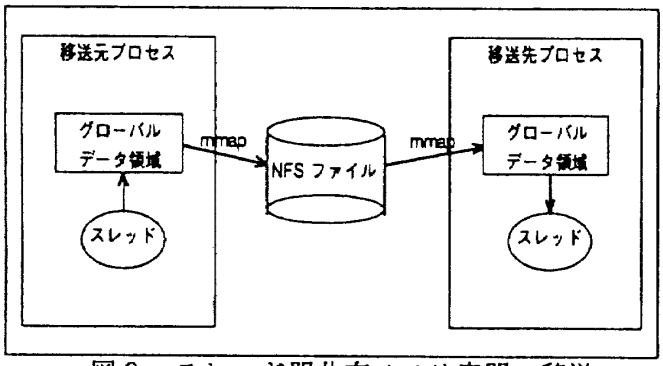

図２　スレッド間共有メモリ空間の移送

本方式では、ThreadMigGetShm()というライブラリコールを用意し、以下のような手順でグローバルデータ領域による通信を移送する。

（１）初期化時：移送対象スレッドでは、他のスレッドと通信を行うためのグローバルデータ領域を確保するためにThreadMigGetShm()をコールする。ThreadMigGetShm()は、移送開始時にmmap()を行うために必要なグローバルデータ領域の開始アドレスと長さの情報を保持する。また、移送対象スレッドと通信する他のスレッドでも同様にThreadMigGetShm()をコールし、移送開始時のためにグローバルデータ領域の情報を保持する。

（２）移送開始時：スレッドとグローバルデータ領域との対応関係を基に、移送元と移送先プロセスの双方においてNFSファイルに対してmmap()を実行し移送対象スレッドからグローバルデータ領域をアクセス可能とする。

（３）移送前後：移送前後において変化することなく移送元プロセス上のスレッドと移送スレッドとの間でグローバルデータ領域による通信が可能となる。

4．おわりに

本稿では、スレッドレベル動的再構成において必要となるスレッド間通信の移送方法について検討した。

今後は、本検討に基づく実装を進める予定である。

参考文献

[1]Christine Hofmeister,et al:"Dynamic Reconfiguration in Disributed Systems: Adapting Software Modules for Replacement",IEEE 13th Int. Conf. on Disributed Computing Systems,pp.101-110,May 1993

[2]Michel Wermelinger:"A Hierarchic Architecture Model for Dynamic Reconfiguration",Proc. of the 2nd Int. Workshop on Software Engineering for Parallel and Distributed Systems, 1997

[3]鈴木,脇:"SunOSにおけるマルチスレッドプロセス移送方式の提案",情処全大5H-5,1995.3

[4]鈴木,脇:"スレッドレベル動的再構成のためのスレッド移送方式の検討",通信学会総大D-56,1996.3

[5]鈴木,脇:"スレッド移送の実装とスレッドレベル動的再構成システムの検討",情処全大,1997.3

[6]白木原,金井:"通信を行うプロセスの移送機能の設計と実装",情処論文誌Vol.34,No.6,1993.6

[7]Julica Protic et. al:"Distributed Shared Memory: Concepts and Systems",IEEE Parallel & Distributed Technology, Summer 1996