6J-6

複数の計算機でプロセスを分散実行するための システムコール転送機構

玉井 智規† 深尾 拓司†† 芝 公仁† 岡田 至弘†
†龍谷大学理工学部 ††龍谷大学大学院理工学研究科

1 はじめに

近年，科学技術計算の分野やシュミレーションの分野，超高精細画像の処理など，多くの演算能力を必要とするアプリケーションやデータが増加してきている．これに伴い，大規模な計算を行う際，ネットワークに接続された計算機を多数使用し，複数の計算機を協調させ，処理を行う分散並列処理システムの活用が多くなってきている．しかし，多くの場合，これらのシステムを利用するには，分散処理を行うために計算機が持つ環境や情報の把握することや，また分散環境に適したアプリケーションを作成することなどが必要になる．そのため，多くの分散システムでは，アプリケーションをその分散システムに適した形に変更することが必要になる．このような問題に対し，我々はアプリケーションへの変更を必要とせず，マルチスレッドで動作するアプリケーションが，ネットワークに接続された複数の計算機を同時に用いて動作することが可能となるプロセス共有機構を開発している．

分散システムに最適化されたアプリケーションは少ない．分散共有メモリで代表的な ThreadMarks は [2]，ネットワークに接続されている複数の計算機でデータ共有を行う．また，プロセスを分散するマイグレーション機構を UNIX 上に実装したものとして，MOSIX[1] がある．MOSIX では，プロセスが移動した際，代理プロセスという特殊なプロセスが，元のホストに残される．移動後のプロセスが移動先のホストでシステムコールを発行すると，そのシステムコールはカーネルによって移動元のホストへと転送される．転送されたシステムコールは待機している代理プロセスによって実行され，実行結果をシステムコールを発行したホストへと返す．プロセス共有機構では，プロセスのアドレス空間を共有し，スレッドの移送を行うことを実現している．そのため，共有メモリはアプリケーションの分散化を行うことを目的とし，分散された各スレッドが発行するシステムコールの発行に関してはカーネル空間での通信を利用し行う．また，スレッドと計算機とのテーブルを作成し対応付けを行っておくことで，複数の計算機からの要求に対応させることを可能としている．

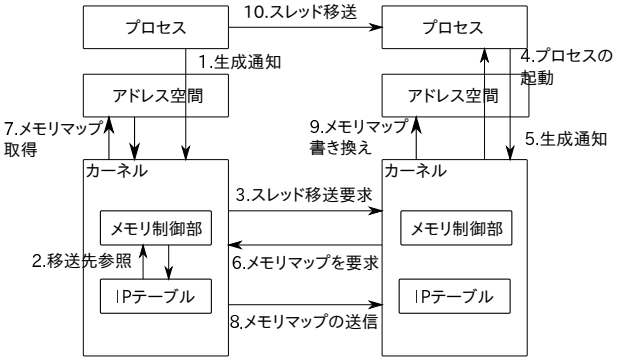

図 1 メモリ制御とスレッド移送

2 プロセス共有機構

プロセス共有機構では，既存のアプリケーションのプロセスに変更を加えることなく，複数の計算機でプロセスを共有し動作させることができる．プロセス共有機構では，プロセスが持つアドレス空間を計算機間で共有する．これによって，共有されるプロセスの持つスレッドを，任意の計算機上で動作させることが可能となる．このとき，各スレッドから発行されるシステムコールを，プそれを実行すべき計算機へ転送し処理する．プロセス共有機構は，メモリ制御，スレッド管理，システムコール転送機構から構成され，プロセスの位置透過な動作を実現する．

2.1 メモリ制御

プロセスを実行し，スレッドを分散させるまでの処理の流れを図 1 に示す．

メモリ制御部では，アドレス空間の共有を実現するために，各計算機でメモリの一貫性制御を行う．一貫性制御はページを単位として行われ，順序一貫性のメモリモデルが実現される．これにより，共有プロセスはメモリの一貫性制御を意識することなく動作することが可能となる．

プロセスを起動されると，それがメモリ制御部へと通知され，メモリ制御部は，分散環境を構築している各計算機へアドレス空間の情報を送信し，メモリの一貫性制御を開始する．アドレス空間の共有により共有プロセスが持つスレッドは，どの計算機で動作していても，起動しているプロセスのデータやコードにアクセスすることが可能となる．

Forwading system calls for destributed threads on shared memory

Tomonori Tamai†, Takuji Hukao††, Masahito Shiba†and Yoshihiro Okada†
†Faculty of Science and Technology, Ryukoku University
††Graduate School of Science and Technology, Ryukoku University

表 1　計算機テーブル

NAME	DESCRIPTION
id	計算機を一意に識別するための番号
ip_addr	計算機の IP アドレス
port	通信を行うポート番号

表 2　スレッドテーブル

NAME	DESCRIPTION
th_id	スレッドを一意に識別する番号
th_pid	各計算機が発行するスレッド ID
th_mm	アドレス空間識別番号

2.2　スレッド管理

プロセス共有機構では，既存のアプリケーションを対象とし，そのアプリケーションに変更を加えず実行できること，ユーザは分散環境を意識せず位置透過に処理できることを目指している．プロセス共有機構では，マルチスレッドアプリケーションを対象としており，分散対象はスレッドである．

プロセス共有機構は Linux カーネルに実装している．カーネルにリンクする際，プロセスを実行する計算機へと通信を行う．プロセスを実行する計算機では計算機テーブルを作成し，通信を行う各計算機へと計算機テーブルを返送，もしくは更新を行う．また，分散対象とするスレッドや，各計算機へ送信したアドレス空間に対して，一意に識別可能な対応付けを行っておく．スレッドはどこに分散したかを確認するために必要であり，アドレス空間は，スレッドの分散要求を行う際，分散先の計算機がメモリマップした計算機であるかどうかという判断を行うために必要となる．各データのテーブルを表 1 と表 2 に示す．

スレッド移送要求を受け取った後，プロセス共有機構に付随した，空プロセスを起動するアプリケーションを実行する．これは，カーネルが持つ API を利用し，カーネル空間からユーザ空間のアプリケーションを起動する．この API を利用することで，空プロセスを起動し，分散先となるスレッドを生成する．また，生成するスレッドは，実行対象としているプロセスが持つスレッド数と常に同数保持しておく．これは，参照対象となるアドレス位置のずれを回避するためであり，また，別の計算機から，スレッドの移送を任意に行えるようにするためである．

2.3　システムコール転送機構

システムコール転送機構は，スレッドの処理を円滑に行えるようにするための機構である．アプリケーションは本来一台の計算機で処理されることを想定されている．システムコール転送機構は，各スレッドが発行するシステムコールを，本来実行すべき計算機へと転送し実行処理するシステムである．

システムコール転送機構は二つの機能を有する．システムコールの転送に必要な通信を行う転送部と，実際の処理を行う処理部である．処理のおおまかな流れを含む構成を図 2 に示す．各計算機に分散されたスレッドから発行されるシステムコールは，自身の計算機に

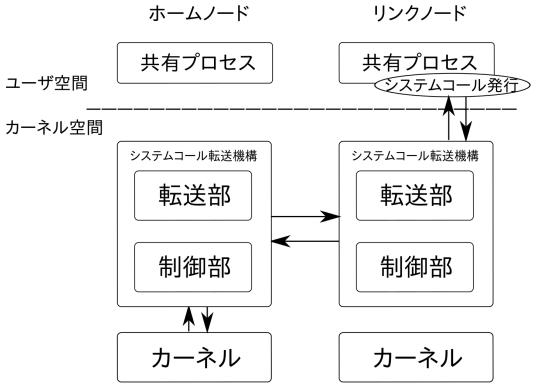

図 2　システムコール転送機構の構成

対して発行を行う．例えば，open システムコールの場合，これを処理すべき計算機上のファイルを対象にして処理しなければならない．このようなことから，すべてのシステムコールは，プロセスを起動した計算機へと転送を行い，処理を行う．

また，プロセス共有機構は分散環境を構築することを目指しており，システムコール転送機構も，分散環境に適応したシステムとして機能できなければならない．システムコール転送機構内の制御部では，実際にシステムコールを実行する機能を持つ．この制御部では，共有プロセスによって生成されたスレッド数に応じた数の実行スレッドを，プロセスを実行した計算機上に生成する．共有プロセスが持つスレッドに対応付けされた実行スレッドを用意することで，複数の計算機からのシステムコール実行要求に対応することを可能とする．

参考文献

[1] A.Barak and R.Wheeler. MOSIX: An Integrated Multiprocessor UNIX. In Process migration. ACM Computing Surveys, Vol. 32, No. 3, pp. 241-299，2000.

[2] Christina Amza, Alan L. Cox, Hya Dwarkadas, Pete Keleher, Honghui Lu, Weimin Yu RamakrishnanRajamony, and Williy Zwaenepoel. ThreadMarks:Shared Memory Computing on Networks of Workstations. IEEE Computer, Vol.29, No2, pp. 18-28(1996).