2J-7

プロセスを共有するためのシステムコール転送機構

三添 匠† 小鍛治 翔太†† 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，複数のスレッドを使用して動作するアプリケーションが増加している．しかし，一般的なシステムでマルチスレッドのプロセスを実行しても単一の計算機上でしかそのプロセスを動作させることができない．本研究の目的は，プロセスを複数の計算機で共有し，プロセスが持つスレッドを各々の計算機に分散させることが可能なプロセス共有機構を実現することである．これにより，プロセスは同時に複数の計算機の資源を利用することが可能となる．

現在，我々が開発を行っているプロセス共有機構では，単一のプロセスをネットワークに接続されている複数の計算機上で共有し，動作させることが可能である. これは, プロセスの持つアドレス空間を計算機間で共有することで実現される [1]．すなわち，ある計算機でメモリへの書き込みが行われると他の計算機でもそれを読み出すことができる．また，システムコールは，それを処理すべき計算機に転送され実行される．

本稿では，特に，各計算機上のスレッドから発行されたシステムコールをそれを実行すべき計算機に転送し実行するシステムコール転送機構について述べる．本機構により，スレッドはどの計算機からでもシステムコールを発行することが可能となり，位置透過にプロセスのコードを分散実行することが可能となる．

2 システムの構成

システムコール転送機構は，複数の計算機に分散した各スレッドから発行されるシステムコールを、それを実行すべき計算機に転送し実行する機能を持つ．システムコール転送機構の構成を図 1 に示す. システムコール転送機構はカーネル内にあり，共有プロセスのスレッドがシステムコールを発行したときに動作する．システムコール転送機構は以下の 2 つから構成される．

- 制御部
  システムコールの実行を行う
- 転送部
  システムコール転送に必要な通信を行う

System Call Forwarding for Distributed Shared Processes
Takumi Mizoe†, Shouta Kokaji††, Masahito Shiba†and Yoshihiro Okada†
†Faculty of Science and Technology, Ryukoku University
††Graduate School of Science and Technology, Ryukoku University

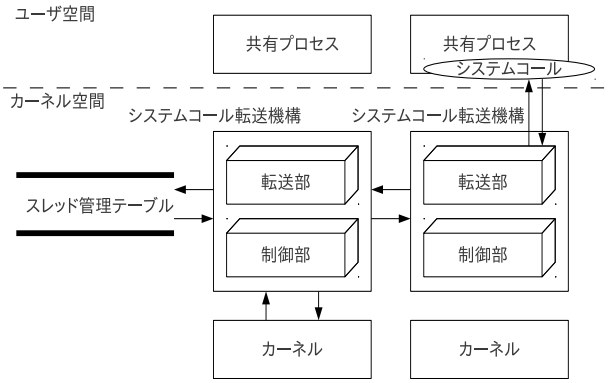

図 1 システム構成

制御部は，システムコール実行要求を受け取ると，そのシステムコールを実行する．また，共有プロセス内でスレッドが生成された際，各計算機で管理するプロセス番号とは別に，分散しているスレッドを一意に識別するスレッド番号 (pid_a) を割り当て，スレッド管理テーブルにより，pid_a，スレッドを実行している計算機のアドレスを管理する．

転送部は，共有プロセスが発行するすべてのシステムコールに対して，各システムコールの実際の処理が実行される前に転送部が実行される．また，他の計算機のシステムコール転送機構と以下の通信を行う．

- システムコール実行要求
- スレッド生成通知
- システムコール終了通知

システムコール実行要求では，pid_a と，システムコールの実行に必要なシステムコール番号，システムコールの引数を，システムコールを実行する計算機のシステムコール転送機構に送信する．スレッド生成通知では，スレッドが生成された際に，プロセス番号，アドレスを送信する．システムコール終了通知では，実行したシステムコールの戻り値を送信する．

3 システムコール実行

システムコールの実行は図 2 のように，実行スレッドにより処理される．実行スレッドは，共有プロセス内でスレッドが生成されると，カーネル内に生成されそのスレッドのシステムコールを処理する．この実行スレッドは共有プロセスのタスク構造体の情報を持つため，他の計算機で生成されたスレッドが発行したシ

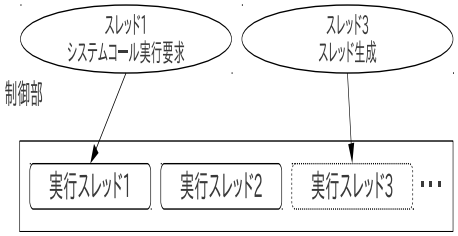

図 2　実行スレッド

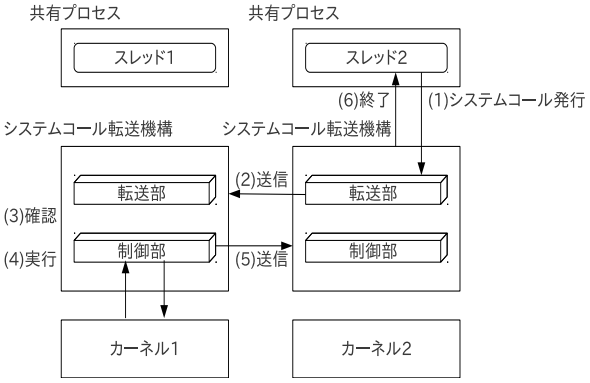

図 3　システムコール転送

ステムコールを共有プロセスから発行されたかのように処理することができる．システムコール実行要求の際，受信した pid_a から処理する実行スレッドを制御部で判断する．

スレッド 2 から発行されたシステムコールをカーネル 1 に転送および，実行は以下のように行われる (図 3 参照)．

(1) 分散されたスレッドからシステムコールが発行される
(2) 転送部が発行されたシステムコールを実行する計算機を調べ，その計算機にシステムコール実行要求を送信する
(3) システムコール番号が有効なものか確認する
(4) 処理する実行スレッドを判断し，システムコールを実行する
(5) 戻り値を送信する
(6) 戻り値をレジスタに格納し，ユーザモードの共有プロセスに復帰する

(2) においてシステムコールを実行する計算機を調べた際，システムコールを転送する場合はシステムコール実行要求を送信し，転送しない場合はシステムコールが発行された計算機で実行する．また，システムコール実行要求を送信したあとは，システムコールを発行したスレッドの状態を実行状態から待ち状態に移行し，戻り値を受信するまで停止させる．

表 1　システムコール処理時間

	A	B	C
open	0.275ms	0.279ms	4.304ms
mkdir	1.355ms	1.365ms	1.385ms
rmdir	0.985ms	0.992ms	1.029ms
symlink	1.755ms	1.756ms	1.772ms

(4) においてシステムコールを実行する際は，受信した pid_a から実行スレッドを判断し，当該スレッドを用いて，システムコールを実行する．

4　性能評価

基本性能の評価として，システムコール転送機構を Linux カーネル 2.6.33.7 に実装し，Core i7 2.8GHz のプロセッサ，2GB のメモリを搭載した計算機 2 台を 1000Mbps のイーサネットで接続した環境で，以下のような条件でシステムコールを実行して処理時間を測定した．

A 本機構を使用せず，システムコールを発行した計算機で実行
B 本機構を使用し，システムコールを発行した計算機で実行
C 本機構を使用し，システムコールを他の計算機に転送し実行

発行するシステムコールは open，mkdir，rmdir，symlink の処理時間を測定する．また，ファイルシステムには NFS を使用し，第 5 階層のディレクトリ内のファイルを操作の対象とした．処理に要した時間を表 1 に示す．結果より，A と B の処理時間の差は，システムコールを実行すべき計算機を調べる際に起きるオーバヘッドであると考えられる．システムコールを転送するのに要する時間は B と C の差である．open の転送処理時間の増加は，アドレス空間の送受信が伴ったためであると考えられる．

5　おわりに

本稿では，分散されたスレッドが発行したシステムコールを適切な計算機に転送し実行するシステムコール転送機構について述べた．本機構により，任意の計算機からシステムコールを発行することが可能になり，位置透過なスレッドの分散実行が実現される．

参考文献

[1] 小鍛治 翔太，芝公仁，岡田至弘: プロセスを共有するためのソフトウェア分散共有メモリの実現，第 72 回情報処理学会全国大会論文集，Vol72，No1，pp83-84(2010)．