B-41

並列分散オペレーティングシステム CEFOS における ジョブ実行方式と分散共有メモリ機構
Job Execution System and Distributed Shared memory System of CEFOS

副島 裕司† 日下部 茂‡ 谷口 秀夫‡ 雨宮 真人‡
Yuji SOEJIMA Shigeru KUSAKABE Hideo TANIGUCHI Makoto AMAMIYA

1．はじめに

複数の計算機を高速なネットワークで結合した分散処理環境の開発が進んでいる。この分散処理環境は、複数の計算機資源を利用した高性能な処理環境である。しかし、これらの計算機資源を有効に利用するためには、プロセス数やプロセス間の通信を強く意識したアプリケーションを作成する必要がある。また、プロセスの配置を工夫し効率よく実行する機構が必要である。

本論文では並列分散オペレーティングシステム CEFOS (Communication-Execution Fusion Operating System)[1][2][3] におけるジョブ実行方式と、その実行方式に適応した分散共有メモリ機構について述べる。

2．ジョブ実行方式

2.1 CEFOS

CEFOS は細粒度マルチスレッド実行方式を基本計算方式とし、細粒度処理と非同期通信処理を効率よく実行する、スループットを重視したオペレーティングシステムである。CEFOS では複数のジョブを並行に実行する。ジョブは一つ以上のプロセスで構成されサービスを提供する。プロセスは一つ以上のスレッドで構成され一つの計算機内で実行される。スレッドの粒度が小さいためスレッド切り替え操作が多発する事が予測される。そこで、切り替え操作のオーバヘッドを削減するために、スレッドを OS 核外で管理する。ユーザレベルで動く OS 核外の管理部を External Kernel と呼ぶ。一方、カーネルレベルで動く従来の OS 核を Internal Kernel と呼ぶ。プロセスは、Internal Kernel により管理される。

スレッドはデータの依存関係により関連づけられ、先行して実行されるスレッドの生成するデータを待つ。スレッドは同期が取れると、走行可能になる。ここで同期が取れるという事は、先行スレッドがデータの生成を完了した状態である。同期が取れたスレッドはスレッドスケジューラにより選択され実行される。走り出したスレッドは中断する事なく走り切り、自身の生成するデータを必要とする継続スレッドに対して同期操作を行う。一つのプロセス内でのスレッドの同期は External Kernel の同期処理機構によりユーザレベルで高速に実行される。複数プロセス間でのスレッドの同期は、別の計算機に相手が存在する場合であっても相手側の External Kernel に通知され同期操作が行われる。図１に CEFOS における実行環境を示す。

2.2 基本方式

CEFOS はプロセス自体が内部にスレッドスケジューラ

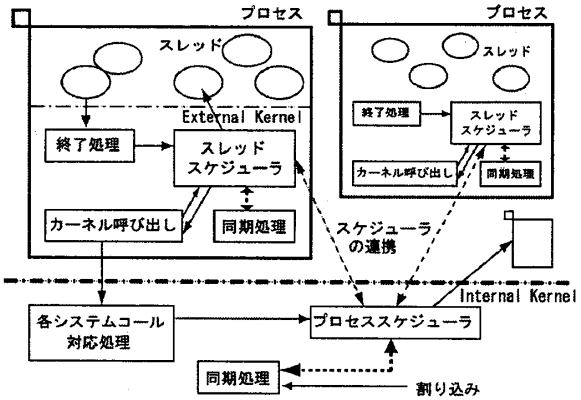

図１：CEFOS における実行環境

を持ち複数のスレッドが並行に実行する機構を持つ。プロセス内で走行可能になっているスレッドはお互いに依存関係がないので、並列に実行することが可能である。

そこで、従来は複数プロセスを個別に作成してジョブを構成していたが、CEFOS では同一の処理を行う事ができるプロセスを複数作成しジョブを構成する。つまり、ある処理を実行したい場合にジョブ内のプロセスはどれでもその処理を行うスレッドを生成できる。

スレッドの実行が終了し同期操作を行う場合、通常は同一プロセス内で同期を待っている継続スレッドに同期操作を行う。しかし、ジョブ内のすべてのプロセスが同一の処理を行うスレッドを生成できるので、可能ならば同一プロセス内のスレッドで処理するのではなく、他のプロセスで同一の処理を行うスレッド生成し実行する事もできる。そこで、一つのプロセス内でのみ処理を行うのではなく、一部の処理を他のプロセスでリモート処理として実行する。つまり、External Kernel が自動的にプロセス内の処理をリモート処理に変更すれば、プログラマが分散処理を意識していなくても、分散処理を行う事ができる。リモート処理に変更する場合は、External Kernel はそのリモート処理の継続スレッドを指定する。リモート側のスレッドの処理が終了した時は、元のプロセスのスレッドに対して同期操作が行われる。本実行方式は RPC(Remote Procedure Call)形式に基づいている。

今回提案した実行方式では、プログラマは分散処理を意識しなくても、CEFOS のスレッドモデルに基づいてアプリケーションを作成する事で、External Kernel が自動で分散処理を行うという利点がある。その反面、プログラマは計算機資源を最も効率よく利用するように最適化したアプリケーションを作成する事ができない。しかし、通常のシステムでは単一のサービスのみを提供するのではなく、複数のサービスを提供する場合が多い。CEFOS のプロセスは内部に存在するスレッドを効率よく実行して無駄な処理を削減する事で、スループットを重視した設計となっている。これを拡張した本実行方式も無駄な処理は削減されている。

† 九州大学大学院システム情報科学府
‡ 九州大学大学院システム情報科学研究院

ここでいう無駄な処理とはデータを待つ処理で計算機資源を消費する事である。従って複数のジョブが起動し負荷が高い場合は、計算機資源を効率よく利用するために最適化したものに比べ、本実行方式の方がシステム全体としては高いスループットが期待できる。

3. 分散共有メモリ機構

3.1 概要

同一のプロセス内での処理を他のプロセスでリモート処理として実行する場合、データのやりとりに分散共有メモリを利用する。External Kernel がプロセス内処理をリモート処理に変更して分散処理を行うので、分散共有メモリ機構は External Kernel に対して提供される機構である。主な特徴を以下にあげる。

- Home[4]に基づいた方式である。
- 共有メモリ領域をロックする必要が無い。

CEFOS では、Home という概念を用いて一貫性保持を行う方式を採用する。これは、最新データを持つ Home 領域と、Home のコピーを持つリモート領域を用意して、リモート領域は Home から最新データを取得し、リモート領域で更新されたデータは Home に転送する事で Home を最新に保つ方式である。CEFOS はデータがそろった時にシグナルを受け処理を開始するという設計であるため、リモート側でデータを取得するために共有メモリ領域を更新しない。リモート側にデータがそろった時にシグナルを送り、処理が開始される。一般の Home を用いた方式ではリモート側でデータが必要になった時に Home から最新データを取得するが、CEFOS では自発的に Home 側からリモート側にデータを送信する方が自然であり効率の良い処理ができる。

通常の分散共有メモリの一貫性保持はロック操作、ロック解除操作を工夫することにより性能を向上させている。CEFOS のスレッドモデルでは、先行スレッドがデータを生成するまでそのデータを使用する継続スレッドは実行されない。同期が取れてスレッドが実行可能になった状態は、ロック操作を行い共有メモリがロックされた状態とみなせる。一方、先行スレッドの終了時に行われる継続スレッドへの同期操作は、共有メモリ領域のロック解除操作とみなせる。従って、共有メモリ領域にロックをする必要がない。

3.2 処理内容

スレッドの実行が終了し同期操作が行われる。この時、External Kernal は同一プロセス内のスレッドへの同期操作を横取りし、リモート処理に変更する。そして、External Kernel は継続スレッドが使用するはずだったデータ領域をリモート側に転送する。リモート側プロセスの External Kernel は、共有メモリ領域を使用する継続スレッドを生成する。このスレッドは元のプロセスの継続スレッドと同一の処理を行うスレッドである。リモート側でスレッドの処理が終了すると、External Kernel は共有メモリ領域のデータを Home 側に転送する。Home 側ではデータが更新されると、それを契機として External Kernel がそのデータを使用する継続スレッドの同期操作を行う。

図 2 に実際の処理の様子を示し、以下に説明する。

（1）External Kernel は同一プロセス内のスレッドへの同期処理を横取りする。

（2）External Kernel は継続スレッドが使用するはずだったデータ領域をリモート側にコピーする。

（3）リモート側で継続スレッドを生成し同期操作を行う。

（4）リモート側のスレッドは共有メモリ領域を使い処理する。

（5）共有メモリに書き込まれたデータはリモート側のスレッドの終了を契機に Home に反映する。

（6）Home のデータが更新されると、それを契機として External Kernel はそのデータを待っていた継続スレッドに対して同期操作を行う。

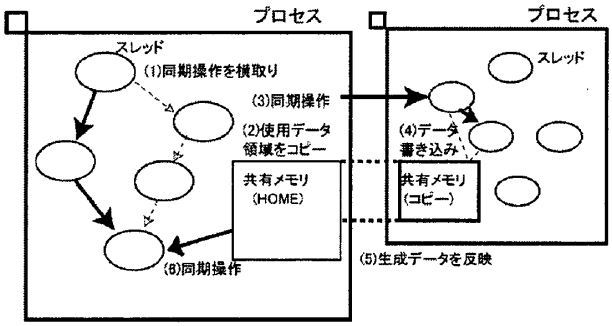

図２：ジョブ実行の様子

4. おわりに

本論文では、並列分散オペレーティングシステム CEFOS におけるジョブ実行方式と、その実行方式に適応した分散共有メモリ機構について述べた。本実行方式ではプログラマが分散処理を意識する事なしにアプリケーションを作成しても、システムが自動的にプロセス内の処理を他のプロセスへのリモート処理に変更し分散処理を行う事が可能である。本実行方式は一つのジョブでは計算機資源を最も効率よく利用することができないが、複数のジョブを起動した状態では全体的なスループットの向上を望むことができる。また、本実行方式の特徴と CEFOS のスレッドモデルを活かして、分散共有メモリ機構ではロック操作を無くし一貫性保持のオーバヘッドを削減した。

今後は、今回提案した実行方式をサポートする機構を実現し、その性能を測定する予定である。

参考文献

[1]日下部茂，富安洋史，村上和彰，谷口秀夫，雨宮真人，“並列分散オペレーティングシステム CEFOS(Communication-Execution Fusion OS)”，信学技報，Vol.99, No.251, pp.25-32 (1999).

[2] 谷口 秀夫, 日下部 茂, 棚林 拓也, 中山 大士, 雨宮 真人，“CEFOS オペレーティングシステムのスレッド管理機構”，情処研報，2000-OS-83, Vol.2000, No.21, pp.7-12 (2000).

[3]Makoto Amamiya, Hideo Taniguchi, Takanori Matsuzaki, “An Architecure of Fusing Communication and Exccution for Global Distributed Processing,” Parallel Processing LettersVol.11, No.1, pp.7-24,2001

[4]Yuanyuan Zhou, Liviu Iftode Kai Li, “Performance Evaluation of Two Home-Based Lazy Release Consistency Protocols for Shared Virtual Memory Systems,“ Proceedings of the 2nd Symposium on Operating Systems Design and Implementation, Oct. 1996.