分散環境におけるプロセス生成方式

5B－6

箱守 聰, 谷口 秀夫
NTT データ通信（株）開発本部

1　はじめに

プロセッサ間を高速通信路で結合した分散処理システムを用いて、トランザクション処理等の高負荷なサービスを実現する試みが行なわれている。このようなシステムでは、プロセス等の資源を効率的に制御することが求められる。特に、プロセス生成処理はディスクアクセス処理を伴うため処理時間が大きく、効率的な処理方式が望まれる。

本稿では、高負荷な処理を行なう分散処理システムにおいて、あるプロセッサから他のプロセッサ上へプロセスを生成するリモートプロセス生成機能の実現方式について検討する。

2　プロセス生成機能への要求

高負荷な処理を行なう分散処理システムでは、処理の並列化と負荷の分散化を目的として、各プロセッサに同一のプロセスを多数配置することが多い。

これらのプロセスを、リモートプロセス生成機能を用いて生成する場合について考える。リモートプロセス生成機能の実現例として、プロセス移送機能が報告されている [1][2]。しかし、この機能は一度にひとつのプロセスしか生成できないため、複数のプロセスを生成するときはプロセスの個数分の処理を繰り返す必要がある。したがって、多数のプロセスを生成しようとするには適当でない。このため、リモートプロセス生成機能に対して、多数のプロセスを効率的に生成する方式が求められる。

3　プロセス生成処理の流れとリモートプロセス生成機能の実現方式

プロセス生成処理の流れを図１に示す。プロセス生成処理は、大きく分類してコンテクスト作成処理とプログラムロード処理の２つの処理から構成される。リモートプロセス生成では、これらの処理をプロセス生成要求を発行するプロセッサ（以後、要求元プロセッサと呼ぶ）と、プロセス生成の対象となるプロセッサ（以後、要求先プロセッサと呼ぶ）とで分担して行う。なお、プログラムファイルは要求元プロセッサ側にあるものとする。このとき、各プロセッサの処理の分担により実現方式が考えられる。ここで、プログラムロード処理を行なうにはコンテクストが必要であることから、２つの処理は同一のプロセッサで行なう方がよい。したがって、以下の実現方式が考えられる。

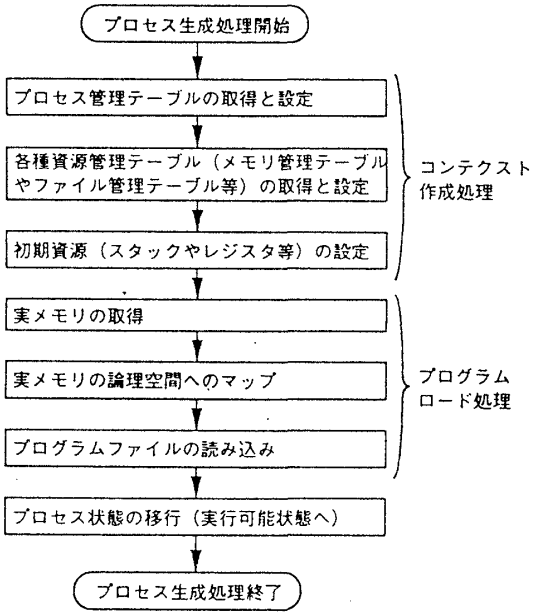

図１．プロセス生成処理の流れ

実現方式 A: 要求元プロセッサでコンテクスト作成とプログラムロードを行なった後、そのイメージを要求先プロセッサへ転送する。

実現方式 B: 要求先プロセッサで直接コンテクスト作成とプログラムロードを行なう。

4　多数プロセス生成時の課題と対処

リモートプロセス生成機能を用いて、多数の同一のプロセスを効率的に生成するためには、実現方式に対

Process Creation Mechanism in Distributed Environment
Satoshi HAKOMORI and Hideo TANIGUCHI
NTT DATA Communications Systems Corporation

表 1. リモートプロセス生成機能の実現方式の比較

	ディスクアクセス回数	プロセッサ間の転送データ量		プロセッサ上の主な処理	
		1回のデータ量	転送回数	要求元プロセッサ	要求先プロセッサ
実現方式A	1	コンテクスト+テキスト+データ	N	(1) コンテクストの作成 (2) プログラムのロード	(1) 転送されたデータを生成するプロセスの数だけコピー
実現方式B	1	テキスト+データ	N	—	(1) コンテクストの作成 (2) プログラムのロード (3) 転送されたデータを生成するプロセスの数だけコピー

して以下の課題がある。

課題 1 ディスクアクセス回数の削減
課題 2 プロセッサ間の転送データ量の削減
課題 3 同時に実行可能な処理の並列化

これらの課題への対処を以下に示す。

(1) 課題 1 への対処

ファイル内容をメモリ上に読み込み、これを利用することにより、ディスクへのアクセス回数を抑える。このため、実現方式 A は、プログラムロード処理を 1 回とし、作成したプログラムイメージは各要求先プロセッサへの転送処理で繰り返し利用する。実現方式 B では、プログラムファイルの内容をメモリ上にキャッシュしておき、要求先プロセッサからの読み込み要求に対しキャッシュ内容を用いて転送処理を行なう。

(2) 課題 2 への対処

重複した転送を行なわないようにすることにより、転送回数を抑える。このため、実現方式 A,B ともに各要求先プロセッサへの同一内容のデータ転送を 1 回とし、要求先プロセッサにおいて必要回数分のメモリ間コピーを行なう。この対処により、同一内容のデータの転送回数は要求先プロセッサの数となる。

(3) 課題 3 への対処

ディスクアクセス処理やデータ転送処理を並列に実行することにより、処理の並列化を行なう。実現方式 A では、要求元プロセッサにおけるプログラムロード処理とデータ転送処理を並列に行なう。実現方式 B では、要求元プロセッサにおいて、複数の要求先プロセッサから発行されるファイル読み込み要求を並列に行なう。

5 実現方式の比較

3 章の実現方式について、4 章で示した課題への対処を行なった際にどちらがより効率的に処理を行えるかを比較した。比較の観点として、ディスクアクセス回数、プロセッサ間の転送データ量、および要求元プロセッサと要求先プロセッサ上の主な処理を用いた。要求先プロセッサの数を N としたときの比較結果を、表 1 に示す。まずディスクアクセス回数について見ると、両方式とも 1 回で差はない。一方、プロセッサ間の転送データ量については、転送回数は同じであるが 1 回の転送データ量は実現方式 A の方が多いため、全体の転送データ量は実現方式 A の方が多い。また、プロセッサ上の処理について見ると、要求元プロセッサ上の処理は実現方式 A の方が多い。したがって、処理の並列性の観点からは実現方式 B の方がよい。

これらの比較結果により、実現方式 B の方がより効率的にプロセス生成処理を行なえるといえる。

6 おわりに

高負荷な処理を行う分散処理システムにおけるリモートプロセス生成機能に対する要求と、それに対する課題及び課題への対処を検討した。また、リモートプロセス生成機能の実現方式の比較を行った。現在は、DIROS[3] に対し検討結果をもとにしたリモートプロセス生成機能を実現中である。

参考文献

[1] D. Cheriton, *"The V Distributed System,"* CACM, Vol.31, NO.3, pp.314-333(1988).

[2] J. Ousterhout, et.al., *"The Sprite Network Operating System,"* IEEE Computer, Vol.21, No.2, pp.23-36(1988).

[3] 谷口, 遠城, 井村, 境: "分散型リアルタイムオペレーティングシステム:DIROS," 情処研報, 89-OS-45-9(1989).