並列ファイルシステム PF3 におけるファイル管理方式

5Q－7

大谷 寛之　大和 純一　相場 雄一　青木 久幸
NEC C&C メディア研究所

1 はじめに

近年、データベースの大規模化、マルチメディアサーバの発展に伴い、大容量、高スループットアクセスが可能なファイルシステムの需要が高まっている。

そこで我々は複数のワークステーションが高速ネットワークで接続されたクラスタシステム上で動作する並列ファイルシステム MFS を開発した [1][2]。

しかし、MFS の構成ではディスク管理ノードのデータ中継によるオーバーヘッドが発生するためレスポンスタイムが悪化し、またディスク管理ノードが必要になるためコストが上昇する問題がある。

このような問題を解決するため、我々は、各ノードからディスクに対してダイレクトにアクセス可能なディスク共有型クラスタシステム上で動作する並列ファイルシステム PF3 の研究を開始した [3]。

本稿では、PF3 のファイル管理方式について記述する。

2 ディスク共有型の課題

一般にファイルシステムでは、ディスク上に格納したデータの集合を論理的なファイルとして扱うため、ディスク上に配置したファイル管理データによってファイルを管理する。また、ファイル管理データを変更する場合、ディスク上では直接データを変更できないため、一旦ディスクから計算機のメモリ上にファイル管理データを読み出し、メモリ上で変更してからディスクに書き戻すという一連の操作が必要になる。

ところがディスク共有型クラスタシステムは、いずれのノード上で動作するクライアントからも共有ディスクに自由にアクセス可能であるため、それぞれのクライアントが勝手にファイル管理データを操作すると、ファイル管理データの一貫性が乱れる場合がある。

例えば、同一のファイル管理データを別々のノード上で動作する複数のクライアントがアクセスすることを想定する。一方のクライアントがディスクからファイル管理データを読み出して変更している間、もし他方のクライアントがディスクからファイル管理データを読み出すと、ファイル管理データは既に他のクライアントで変更されているにも関わらず、古いファイル管理データに変更を加えることになる。このため、ファイル管理データが乱れてしまう。

A Method of File Management in Parallel File System PF3
Hiroyuki Ohtani, Jun-ichi Yamato, Yuichi Aiba, Hisayuki Aoki
C&C Media Research Laboratories, NEC Corporation

3 従来の方式と問題点

3.1 ロック機構による一貫性制御

従来より、メインフレームなどのディスク共有型クラスタ構成のシステムでは、一般的にロック機構による排他制御方式を採用している。クライアントが、あるファイル管理データを操作しようとする場合、まず排他制御の管理モジュールに対してロック要求を発行し、他のクライアントからのアクセス要求を排他する。ファイル管理データの操作が終了した後、アンロック要求を発行し、他のクライアントに解放する。これにより、ファイル管理データの一貫性を維持する。

しかし、この方式では、ロック / アンロックの度に通信が発生する。もし、クライアントがファイル管理データを排他する期間を長くすれば通信の密度は減らせるが、他のクライアントがファイルにアクセスできない期間が長くなり、処理の多重度が低下する。一方、処理の多重度を上げるため、排他する期間を細かい単位で設定すると、ロック / アンロックの通信が頻繁に発生し、ネットワークのトラフィック増大を招くというトレードオフに陥る。

3.2 ファイルの集中管理による一貫性制御

我々の開発した MFS[1][2] をディスク共有型クラスタシステム上で動作させることも可能である。この構成では、各ディスクを分担して管理する複数のサーバが、担当するディスク上のファイルを集中管理する。各クライアントはそのファイルを担当するサーバに要求を出し、ファイルアクセスを行なう。

もし、複数のクライアントが、同一のファイル管理データを変更する要求をサーバに発行した場合、サーバ側ではファイル管理データの更新処理をしている間、そのデータに対する後続の処理要求を待たせる逐次的な処理を行なう。これにより、ファイル管理データの一貫性は維持できる。

しかし、このような構成では、ファイルのデータを一旦サーバが中継するオーバーヘッドが発生する。

4 PF3 の実現方式

PF3 は、ファイル管理データの操作とディスクへの I/O 処理を分離し、ある範囲のファイル管理を一つのサーバが集中制御する方式を採用した。これにより、複数のクライアントからの要求に伴うファイル管理データの操作を逐次的に処理可能となり、ファイル管理データの一貫性は維持できる。

さらに、各クライアントがファイルをアクセスする際は、

ファイル管理を行なうサーバにアドレス情報を問い合わせ、共有ディスクに対してダイレクトにアクセスを行なう。これにより、データ中継のオーバーヘッドを削減できる。

4.1 ファイルのデータ構造

PF3上の一つのファイルのデータは、ブロック単位に分割されて、複数のディスクにストライピング格納される。特に、分散格納されたデータの内、同一のディスクに格納されたブロックの集合を部分ファイルと呼ぶ。

PF3のファイルのデータ構造は、図1のように、一つのファイルを代表するfnode、各ディスクに格納された部分ファイルを代表するpnode、各部分ファイルのアドレス変換情報を格納するBlockMapから構成される。このようにPF3では、ファイル管理データが各ディスクに分散した構成であるため、それぞれが独立に管理され、ファイル管理データに対するアクセスの負荷が分散される。

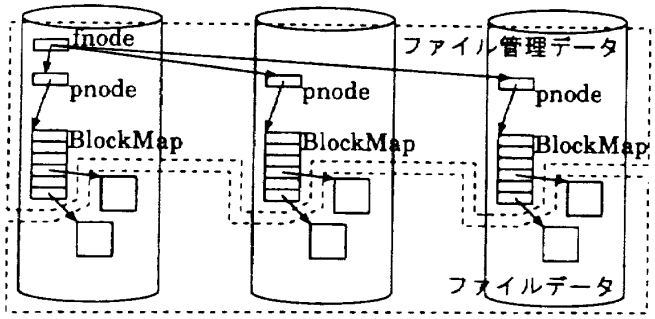

図 1: ファイルのデータ構造

4.2 PF3のモジュール構成

PF3のファイル管理に関係するモジュール構成を図2に示す。FileControllerはファイル管理を行なうモジュールで、システム中に複数存在可能であり、共有ディスクを分担して管理する。FileControllerにはファイルを管理する機能(FnodeManager)と部分ファイルを管理する機能(PnodeManager)がある。FnodeManagerは、オープンされたファイルのfnodeをキャッシングし、ファイル属性情報の管理を司る。PnodeManagerは、オープンされたファイルのpnodeやBlockMapをキャッシングし、部分ファイルのアドレス変換処理を司る。

4.3 ファイルのオープンとクローズ

アプリケーションプログラム(AP)はファイルをオープンすることにより、ファイルへのアクセスが可能となる。

APはファイルをオープンする際、そのファイルを担当するFnodeManagerにオープン要求を発行する。FnodeManagerは、各部分ファイルのオープン処理と、要求されたファイルのfnodeを読み出して、ファイルのアクセス準備を整える。一方、ファイルのクローズに関しても、FnodeManagerがクローズ要求を受け取り、各部分ファイルのクローズ処理を司る。

4.4 ファイルアクセス

APがファイルアクセスを行なうには、ファイル内オフセットアドレスを指定してFALを呼び出す。呼び出されたFALはファイル内オフセットアドレスから、そのアドレスに対応する部分ファイルを特定し、その部分ファイルを管理するPnodeManagerに対して、アドレス変換要求を発行する。この要求を受けたPnodeManagerは、必要に応じてBlockMapをディスクから読み出し、部分ファイル内オフセットアドレスから、ディスク上のオフセットアドレスへ変換して、要求元へ返却する。その後FALは、返却されたディスク上のオフセットアドレスを指定して、ディスクに対してダイレクトにI/O要求を発行する。

この制御方式により、各クライアントはファイルのデータを他の計算機を経由せずに、ディスクに対してダイレクトにアクセス可能となる。また、各PnodeManagerが、担当する部分ファイルのアドレス変換を集中管理することにより、ファイル管理データの一貫性は維持される。

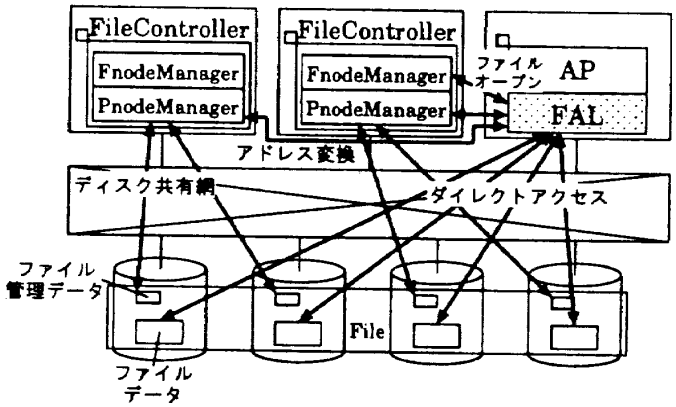

図 2: PF3のファイルアクセス

5 おわりに

本稿では、ディスク共有型クラスタシステム上で動作する並列ファイルシステムPF3において、ファイル管理データ一貫性維持の問題を解決するファイル管理方式について記述した。

PF3は現在実装中であり、今後はPF3の性能評価を行なう予定である。

参考文献

[1] 青木, 他, "並列ファイルシステム MFS", 情処研報, Vol.96, No.79, pp31-36, 1996

[2] 相場, 他, "並列ファイルシステムにおける運用管理機能", コンピュータシステムシンポジウム論文集, Vol.97, No.8, pp125-132, 1997

[3] 大和, 他, "並列ファイルシステムPF3の概要", 第57回情処全大, 5Q-6, 1998