並列I/O SPFSの実装[1]

3G－2

藤崎 直哉，　村上 岳生，　新開 慶武
（株）富士通研究所

1　はじめに

NOW(Network Of Workstations)および並列計算機のような並列処理システムにおいて，プロセッサやプロセッサ間ネットワークの性能向上およびメーッセージパッシングなどの並列化技術の進展に伴ってI/Oの性能向上が重要な課題としてクローズアップされている。また，このような並列処理環境において，ファイル内のデータを分散配置して並列計算を行なう並列アプリケーションには，各計算ノードがそのファイルに対してストライドアクセスをする特徴[1]がある。SPFS(Scalable Parallel File System)[2]は，複数I/Oノードへのデータストライピング，及びストライドアクセスとコレクティブI/Oをサポートすることによって，スケーラブルなI/O性能を可能にするポータブルなI/Oライブラリである。本稿はSPFSの実装について述べる。

2　実装

2.1　オープンとクローズ

クライアントは，処理の違いによって1つのマスタとその他のスレーブに分かれる。ファイルのオープンは，全クライアントが同期をとった後，マスタクライアントが代表してネーム情報（ストライピングの単位サイズ，I/Oノード名，サブファイル名）よりサーバ生成リクエストを作成し，コーディネータを起動（fork使用）する。そのコーディネータがサーバ生成リクエストを受け，各I/Oノードのデーモンにサーバプロセスの生成を依頼する。起動したサーバはサブファイルのオープン結果をマスタクライアントに通知する。マスタクライアントはその結果を全スレーブクライアントに広報する。

ファイルのクローズは，全クライアントが同期をとった後，マスタクライアントが直接各I/Oノードのサーバプロセスに終了を指令する。マスタクライアントはその終了結果を全スレーブクライアントに広報する。

[1]Implementation of Parallel I/O SPFS
Naoya FUJISAKI, Takeo MURAKAMI and Yoshitake SHINKAI
Multimedia Systems Laboratory, FUJITSU LABORATORIES LTD.

2.2　ユーザ認証

コーディネータは，ユーザ認証のために，一時的にオープン時に生成されるプロセスである。ルート権限で動作（setuid使用）し，予約済ポート番号を使ってデーモンと通信する。デーモンに送信するデータは，クライアントのポート番号とアプリケーションのユーザIDとグループIDである。デーモンは，サーバプロセス生成リクエストが予約済みポート番号から要求された時にのみサーバを起動する。クライアントとサーバは互いにポート番号を有することでユーザ認証は実現されている。また，SPFSが使用している通信はUDPで実現されている。

2.3　I/Oノードストライピング

I/Oノードストライピングは，図1のようにファイル内データがI/Oノードのディスクに単位サイズ毎にラウンドロビン様式で分散配置されている。図1のようにある計算ノードがファイルをアクセスする場合，当該クライアントは，アクセスの範囲を計算して該当するI/Oノードにアクセスリクエストを行なう。該当I/Oノードは並行にそのリクエストに従ってディスクをアクセスする。I/Oノードストライピングは，ネットワークがボトルネックとならない限り，I/Oノードの台数に比例してスケーラブルな性能を得ることができる。また，クライアントがキャッシュを持たずWriteリクエストを直ちにサーバに通知することにより，複数の計算ノードがファイルを共用しても，シーケンシャルコンシステンシは保証されている。

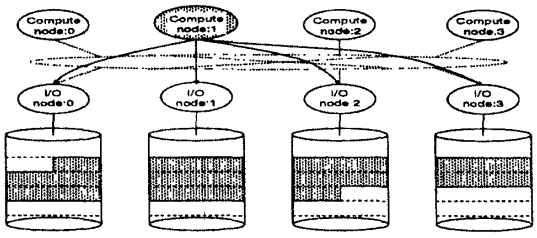

図 1: I/Oノードストライピング

2.4　ストライドアクセス

ストライドアクセスは，図2中node:0の場合，L長アクセスしてL長ストライドを繰り返すというアクセスが行なわれ，通常UNIXインタフェースでn回の

I/Oリクエストを発行する必要があるところを，1回のストライドアクセスリクエストで実現させる機能である。

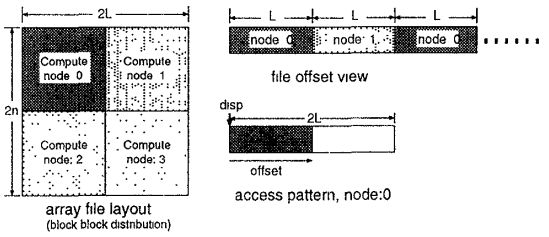

図 2: データレイアウトとアクセスパターン

クライアントがストライドアクセスを行なうためには，事前にアクセスパターンをサーバに指定する必要がある。アクセスパターン情報を図2 node:0を例に表1に示す。オフセットの組数(tile)は，図2中アクセスパターンの影部の数である。同様にnode:1の場合，定義開始オフセットからのオフセットはSoffset=L，Eoffset=2Lとなる。node:2とnode:3の場合，アクセスパターンの定義開始オフセットはdisp=2L×nとなる。

表 1: アクセスパターン情報

disp=0	ファイル先頭からの定義位置オフセット
length=2L	パターン長
tile=1	オフセット組数
Soffset=0	定義位置からの開始オフセット
Eoffset=L	開始オフセットから連続する終端オフセット

図3のように各計算ノードは各自のアクセスパターンをもっている。ある計算ノードがストライドアクセスリクエストを行なうと，当該クライアントは，ストライドアクセスリクエストを代表して受けとるマスタサーバに通知する。マスタサーバは受信したリクエストをその他のスレーブサーバに広報する。全サーバは，並行してリクエスト情報から各自のアクセスする範囲を計算し，ディスクへ連続にアクセスする。更に，アクセスパターンを辿りながら離散アクセスデータを束ねることで通信コストの低減を図り，クライアントと通信を行なう。計算ノードとI/Oノード間のデータ転送は，リクエスト元のクライアントがスケジューリングしてサーバへデータ転送をリクエストするのではなく，サーバ主導のスケジューリングでクライアントとデータ転送を行なうディスクディレクティッド方式[3]により行なわれている。ファイルのアトミック性は，マスタサーバのみがリクエストを代表に受けとることによって保証されている。

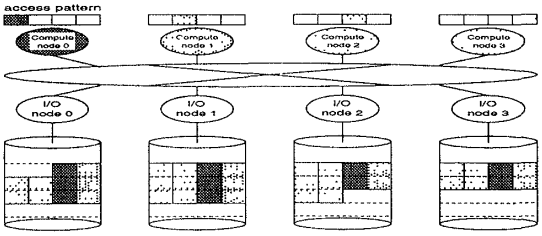

図 3: ストライドアクセスとコレクティブI/O

2.5 コレクティブI/O

コレクティブI/Oは，図3のようにnode:0からnode:3までのストライドアクセスリクエストを一括すると，サーバが一連の領域をシーケンシャルにアクセスすることができ，ディスクと通信路の最適化を図ることができる。

マスタサーバは，数が揃うまでクライアントが通知したコレクティブリクエストを待つので，クライアント同士の同期を取る。全クライアントからのリクエストが揃うと，マスタサーバは，それらのリクエストを一括してスレーブサーバに通知する。全サーバは，一連のリクエスト情報とアクセスパターンを用いてストライドアクセスと同様に計算し，ディスクへ連続アクセスを行なう。更にサーバは，データ転送リクエストが一つのクライアントに集中せずうまく分散するように，通信相手の計算ノードを求めながら，計算ノードとI/Oノード間のデータ転送を行なう。

3 おわりに

並列I/O SPFSについて，I/Oノードストライピング，ストライドアクセス，コレクティブI/Oの実装を述べた。SPFSの性能については[4]で述べる。

今後，ライブラリによるポータブルかつフレキシブルなSPFSの特徴を活かし，様々なプラットホームにおける並列アプリケーションのファイルアクセスの特徴を調べ，並列I/Oスケジューリングの最適化に努めたい。

参考文献

[1] D. Kotz and N. Nieuwejaar, "File-system workload on a scientific multiprocessor", IEEE Parallel and Distributed Technology, pp.51-60, Spring 1995.

[2] 新開 慶武, 村上 岳生, 藤崎 直哉. 並列I/O SPFSの概要. 情報処理学会第54回全国大会. 情報処理学会, 1997.

[3] David Kotz,"Disk-directed I/O for MIMD Multiprocessors", Proc. 1994 Symp. on Operating Systems Design and Implementation,pp.61-74, 1994.

[4] 村上 岳生, 藤崎 直哉, 新開 慶武. 並列I/O SPFSの性能評価. 情報処理学会第54回全国大会. 情報処理学会, 1997.