並列Ｉ／Ｏ　ＳＰＦＳの概要

3Ｇ－1

新開慶武、村上岳生、藤崎直哉
（株）富士通研究所

1. はじめに

近年のプロセッサ処理能力の向上、インタコネクト技術の進展に伴い、種々の並列コンピュータが商用化され、広く利用されるようになってきた。一方、入出力装置単体の性能はプロセッサ処理能力ほど向上しておらず、並列コンピュータの適応分野の拡大に伴い、入出力がボトルネックとなる場合が増えている。このため、複数のノード上のディスクを並列に動かし高性能を実現する並列Ｉ／Ｏが盛んに研究されている[1,2,3]。本稿では当社の並列コンピュータAP3000 を主ターゲットとした並列Ｉ／Ｏ機構ＳＰＦＳ（Scalable Parallel File System）の概要を報告する。

2. 並列アプリのＩ／Ｏ特性

近年の研究[4]の結果、UNIX の API に準拠した第一世代のパラレルファイルシステム[7]の限界が明らかになってきた。並列アプリが発行するＩ／Ｏのパターンが第一世代のパラレルファイルシステムが想定していたものと大きく異なっていたため、実環境での性能が期待値以下になっていた訳である。並列アプリのＩ／Ｏ特性を表１に示す。

	write シェア	ｱｸｾｽﾊﾟﾀｰﾝ	要求長
並列アプリ	一般的	離散ｼｰｸｴﾝｼｬﾙ	小
一般アプリ	まれ	連続ｼｰｸｴﾝｼｬﾙ	小
ｽﾊﾟｺﾝｱﾌﾟﾘ	まれ	連続ｼｰｸｴﾝｼｬﾙ	中～大

表 1: 並列アプリのＩ／Ｏ特性

複数のノードが同じファイルを同時にアクセスし、しかも一つ以上のノードが書き込みを行う所謂 write シェアが一般的に行われるのが、並列アプリの第一の特徴である。このため、UNIX の分散ファイルシステムで一般に用いられているクライアントキャッシュ[6]はコンシステンシ制御のオーバヘッド増大を招き、有効に機能しない。このことは昔からよく知られていた点であり、パラレルファイルシステムは普通クライアントキャッシュをサポートしていない。

離散シークエンシャルとは、ファイルをアドレスの昇順に飛び飛びにアクセスする最近明らかになったパターンで、ディスク上に格納される配列を各ノード上のメモリに分散配置した上で計算を行うする並列アプリに特有のものである。たとえば、図１の例の場合、各計算ノードはディスク上に配置された２Ｌ×２Ｌの配列の担当部分を参照するため、２Ｌバイト毎にＬバイトアクセスする。後述するストライド転送をもたず、しかもクライアントキャッシュがない第一世代のパラレルファイルシステムでは、データ転送長の小さいＩ／Ｏ要求が多発し、性能が思ったほど出ないという結果になっていた。

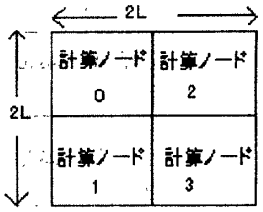

図1 Block-Block分割

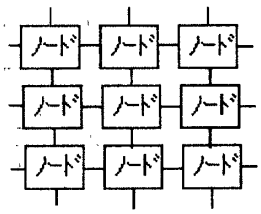

図2:AP3000の構成

3. ＡＰ３０００の構成

最大１０２４台の UltraSPARC ワークステーション（ノード）を高速ネットワークＡＰ－Ｎｅｔで接続した並列コンピュータである。各ノードにディスクがつき Solaris が動作する。

4. ＳＰＦＳの構成

ＳＰＦＳはディスクをもつノード(Ｉ／Ｏノード)

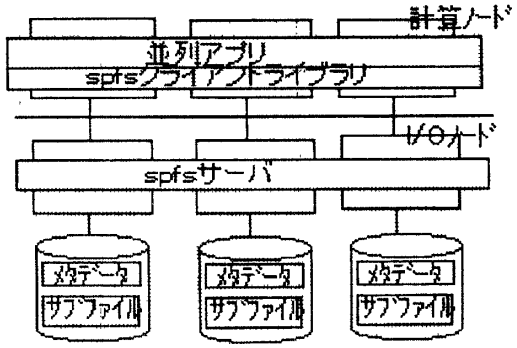

図3 SPFSの構成

Overview of the SPFS(Scalable Parallel File System)
Yoshitake Shinkai , Takeo Murakami and Naoya Fujisaki
Multimedia Systems Labaratory、 Fujitsu Laboratories Ltd.

に配置される複数のサーバと計算ノード上のアプリに組み込まれたクライアントライブラリが会話することにより、複数のノード上のディスクを並列に動作させるライブラリタイプのパラレルファイルシステムである。ファイルのデータは各ノードに存在するサブファイルにストライプ配置される。

5. ＳＰＦＳの特徴

(1) スケーラブル

サブファイルに割り当てたディスクブロックを示すメタデータは各ディスクに分散配置される。したがって、各サーバは自分の担当のサブファイルをアクセスする時、他のサーバに問い合せることなく、各々独立に動作できる。このため、サーバ数に比例したスケーラブルな性能を発揮できる。さらに、ファイルの名前情報もディレクトリツリー単位に分散配置できるので、名前検索処理で特定のノードがボトルネックになることもない。

(2) 高性能

ストライドアクセスおよびコレクティブＩ／Ｏをサポートしているので、並列アプリに特徴的なデータ長の短い離散シークエンシャルアクセスに対しても高い性能を発揮する。勿論、複数ノードから同時に同じファイルをアクセスした場合にもシークエンシャルコンシステンシを保証する。

(3) 標準への準拠

ＳＰＦＳはライブラリ方式のため、ＵＮＩＸのＡＰＩとは異なるインタフェース採用している。しかし両インタフェースは非常によく似ており、既存のＵＮＩＸプログラムをＳＰＦＳ用に書き直すのは容易である。また、ストライド転送、コレクティブＩ／Ｏといった並列アプリ向け新規インタフェースは並列アプリ作成上の実質標準に育つと期待されている現在仕様制定作業中の MPI-IO インタフェース[5]に沿って設定した。ＳＰＦＳは MPI とは独立に動作するので、MPI プログラム以外でも同等のインタフェースが使える表２にインタフェース名をあげる。

基本仕様	コレクティブ I/O
spfs_pvm_open	spfs_read_all
spfs_mpi_open	spfs_write_all
spfs_seek	非同期
spfs_read	spfs_aread
spfs_write	spfs_awrite
spfs_fstat	spfs_aread_all
spfs_fsync	spfs_awrite_all
spfs_close	ストライド転送
	spfs_fcntl

表2: SPFSのインタフェース

(4) ユーザ負担の軽減

ユーザはファイルをどのサーバにストライプするかを意識する必要はない。ファイル創生時に指定されたストライプ幅とストライプ数から、使用するＩ／ＯノードをＳＰＦＳが自動的に決定する。勿論、既存ファイルをオープンする場合はファイル名を指定するだけでよい。

(5) ポータブル

ＳＰＦＳはユーザプログラムとして作成されており、ＯＳの修正は不要である。このため、ＡＰ３０００以外のＮＯＷ (Network Of Workstation) やＶＰＰ７００に適用するのが容易である。

6. おわりに

並列Ｉ／Ｏ ＳＰＦＳの概要を述べた。ＳＰＦＳは並列アプリのＩ／Ｏ特性にマッチしたインタフェースをサポートすることにより、ノード数に比例したスケーラブルなＩ／Ｏ性能を提供するライブラリタイプのパラレルファイルシステムである。またインタフェース及び制御方式の工夫によりユーザの負担を極力低減している。なお、実装については[8]を性能については[9]を参照されたい。

参考文献

[1] J. Huber, C. L. Elford, D. A.. Reed, A. A. Chien, and D. S. Blumenthal,"PPFS:A high performance portable parallel file system," Proc. 9th ACM Int. Conf. on Supercomputing.,Barcelona, pp.385-394, July 1995

[2] K. E. Seamons, Y. Chen, P. Jones, P.Jozwiak, and M. Winslett,"Server-directed collective I/O in Panda",Proc. Supercomputing'95, Dec. 1995

[3] R. Thakur, R. Bordawekar, and Choudhary, "PASSION Runtime Library for parallel I/O," Proc. Scalable Prallel Libraries Conf. Oct. 1994

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

[5] The MPI-IO Committee, "MPI-IO:A Parallel File I/O Interface for MPI",April 1996, Version 0.5

[6] M. G. Baker, J. H. Hartman, M. D. Kufer, K. W. Shriff, and J. K. Ousterhout, "Measurements of Distributed File System," Proc. 13th ACM Symp. on Operating System Principles 1991

[7] Intel corp., "Paragon Users Guide," 1994

[8] 藤崎直哉、村上岳生、新開慶武, "並列I/O SPFS の実装"、情報処理学会第５４回全国大会 Mar. 1997

[9] 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回全国大会論文集、Mar. 1997