Litesに対する並列分散ファイル入出力機能拡張の設計†

4Z－2 平川 泰之[1] 安田 絹子[1] 遠山 緑生[1] 斎藤 鉄也[1] 服部 隆志[2]
[1]慶應義塾大学 大学院政策・メディア研究科 [2]慶應義塾大学 環境情報学部

1 はじめに

複数マシンから成る並列分散システム上でアプリケーションを開発するためには，複数の計算機資源を有効に扱うための使い易いインタフェースが求められる．本稿では，複数のノード上でそれぞれ動作するMachマイクロカーネル上においてアプリケーションの開発・実行環境を提供するLites (4.4BSD Lite Server) [1] の並列分散拡張のうち，特に並列分散ファイル入出力機能を持つ並列分散ファイルシステムの設計について述べる．

2 並列分散ファイル入出力機能

2.1 機能概要

本機能は，並列分散Lites上で協調動作するプロセス群が並列にデータを読み書きすることを可能とする．そのため大量のデータからなるファイルは仮想的に単一のファイルとして実現され，ファイルの異なる領域は並列分散コンピュータシステムを構成する各ノードのローカルディスク上に分割して格納されている．

この機能は全ての入出力が利用するのではなく，

- 並列分散ファイルの異なる領域に対して重複なく独立に並列読み書きし，かつ
- アプリケーション同士がそのファイルフォーマットを既知のものとして参照する

場合に有効である．なおここで言うファイルフォーマットとは，コマンド **grep** のアクセスの単位は行である，のようなアクセス単位を指す．

2.2 基本設計

本機能は，並列分散LitesのVirtual File Systemインタフェース [2] に並列分散ファイル入出力層を挿入し，実現する (図1)．これにより，各ノードのローカルディスク上に物理的に分散配置されているファイルの断片を統合し，仮想的に単一のファイルとして見せることが可能となる．

並列分散システム上の各ノードのローカルディスク上に物理的に分散配置され，仮想的に単一のファイルとして見えている並列読み書き可能なファイルを並列分散Lites上の新しい型のファイルとして定義する．これを並列ファイルと呼ぶ．並列ファイルはディレクトリファイルやシンボリックリンクファイル同様，システム側でのみ解釈可能な特殊ファイルである．並列ファイルは後述する断片ファイルに関する管理データを保持している．一方並列分散システム上の各ノードのローカルディスク上に物理

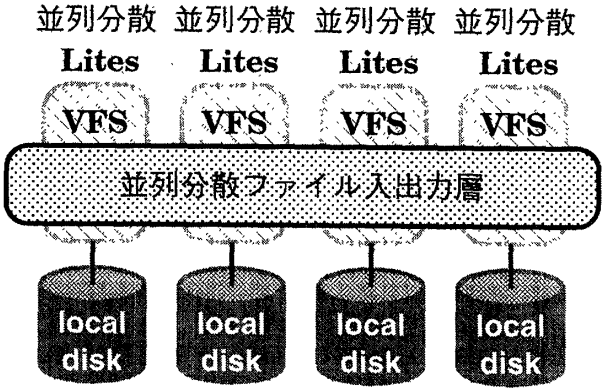

図 1: 並列分散ファイル入出力層

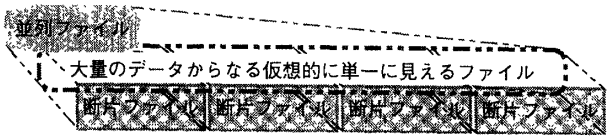

図 2: 並列ファイルと断片ファイル

的に分散配置されているファイルの断片を，断片ファイルと呼ぶ．断片ファイルはユーザに対して並列ファイルが仮想的に単一に見せている大量のデータからなるファイルの実体を格納しているレギュラーファイルである (図2)．

ユーザはこの並列ファイルをオープンすることによって大量のデータからなる単一ファイルの並列読み書きが可能となる．なお実際のファイルの読み書きは該当する断片ファイルのそれに対応づけられる．ただし単一の大量なファイルデータをどのように分割するかについてはアプリケーション依存となる．これはファイルがどの程度の粒度で並列に読み書き出来るかはアプリケーションしか知り得ないためである．

ユーザが実際に並列ファイルにアクセスする場合は次のようになる．並列分散システム上でその断片ファイルを実際に格納しているローカルディスクの存在するノードにプロセスを生成することによって実際のファイルアクセスを行う．また並列ファイルを新たに生成する場合は，アプリケーション側で並列読み書きする粒度に合わせてプロセスを並列分散システム上のローカルディスクの存在するノードに生成し，それぞれのプロセスが処理しただけの断片ファイルをそのローカルディスクに格納することによって行う．

どちらの場合もファイルデータをそれを必要としているプロセスが生成されたノードに転送するのではなく，ファイルデータのあるノードにそれぞれ並列処理のためのプロセスを生成することによってアクセスが可能となる．このときそれぞれの並列処理は独立していることが必要で

Design of Parallel and Distributed File I/O Extensions for Lites

Yasuyuki Hirakawa[1], Kinuko Yasuda[1], Norio Touyama[1], Tetsuya Saitou[1], and Takashi Hattori[2]

[1]Graduate School of Media and Governance, Keio University
[2]Faculty of Environmental Information, Keio University
E-Mail: <chibao@sfc.keio.ac.jp>

†本研究は情報処理振興事業協会 (IPA) の創造的ソフトウェア育成事業「並列・分散処理基盤ソフトウェアの開発」の一部として実施している．

ある．なお並列分散システム上でのプロセス生成については並列分散 Lites の一機能となる並列分散プロセス管理機構に委ねられる．

なお並列ファイルをレギュラーファイルとしてアクセスすることも可能である．この場合，ファイルデータは NFS と同等の機能を利用して各ノードからその並列ファイルをオープンしたプロセスが存在するノードに転送される．

2.3 処理メカニズム

本機能において必要となる基本的な処理は，次の３通りとなる．

- 既存の並列ファイルを並列ファイルとしてオープンする
- 既存の並列ファイルをレギュラーファイルとしてオープンする
- 新たに並列ファイルを生成する

以下それぞれの処理について述べる．

2.3.1 既存の並列ファイルを並列ファイルとしてオープンする場合

プロセスが既存の並列ファイルを並列ファイルとしてオープンすると仮のファイルディスクリプタを得る．得られたファイルディスクリプタはプロセスのファイルディスクリプタテーブルにエントリされ，最終的に vnode [6] を参照する．この vnode は，並列分散ファイル入出力機能用に新たに用意される並列分散ファイル構造体を参照しており，並列ファイル内に記述されている断片ファイルリストなどを含むデータを保持する．

その後並列分散プロセス管理機構により並列処理のためのプロセスが断片ファイルの存在するノードに生成される際に，この vnode 内の並列分散ファイル構造体を参照して生成先ノードを決定する．そして生成されたプロセスが処理を開始する前までに実際に読み書きされる断片ファイルのローカルノードにおける具体的なファイルディスクリプタを得ることにより，並列なファイルアクセスが可能となる．

2.3.2 既存の並列ファイルをレギュラーファイルとしてオープンする場合

プロセスが既存の並列ファイルをレギュラーファイルとしてオープンすることも出来る．この場合も並列ファイルを並列ファイルとしてオープンするのと同様に vnode が用意され，並列分散ファイル構造体を参照するようになる．異なる点は並列分散プロセス管理機構によるデータの存在するノードへのプロセス生成が行われない点と，並列分散ファイル構造体中のフラグ設定がレギュラーファイルとしてアクセスすることを示す点と，それにより各断片ファイルへのアクセスが vnode 以下で吸収され，並列ファイルにおける断片化が隠蔽される点である．

2.3.3 新たに並列ファイルを生成する場合

プロセスが新たに並列ファイルを生成するには並列分散 Lites 上に新たに用意されるシステムコールを用いる．これにより新たに用意されたファイルディスクリプタから最終的に参照される vnode はレギュラーファイルのものではなく，並列ファイル用のものとなる．またこのシステムコールで指定されただけの断片ファイルが予め確保され，確保されたノード名とそのノードにおける断片ファイルのパスなどのデータがその vnode から参照可能となる並列分散ファイル構造体に格納される．

その後は既存の並列ファイルを並列ファイルとしてオープンした場合と同様に並列分散プロセス管理機構により並列処理のためのプロセスが断片ファイルの存在するノードに生成され，実際に読み書きする断片ファイルのローカルノード上でのファイルディスクリプタを得ることにより，並列なファイルアクセスが可能となる．

3 関連研究

これまでにも分散ファイルシステムという観点においては様々なファイルシステムが研究されてきた．特に Network Filesystem [2], [3] や Sprite Network File System [4] などはキャッシュを有効に利用することでサーバのボトルネックを小さくし，効率の良い遠隔ファイルアクセスを提供した．また Serverless Network File Systems [5] では，システムを構成する複数ノード上のローカルディスクを RAID として用い，log-structured なファイルデータを格納している．

しかし NFS や Sprite ではファイルは常に分割不可能なバイトの配列として抽象化されており，並列なファイルアクセスは不可能である．xFS においてはシステムを構成する複数ノード上のローカルディスクを効率良く使用する点においては本研究と通ずるところがあるが，ファイルが常に分割不可能なバイトの配列として抽象化されていることには変わりはない．

並列分散 Lites の最大の特徴は，並列分散システム内に分散するデータの位置に従って並列分散プロセスを生成する点である．そのため，並列分散システムを構成するノードのローカルディスクを最大限に利用し，分割可能なバイトの配列としてファイルの概念を拡張している．

4 まとめ

本稿では，並列分散ファイル入出力機能拡張の設計について述べた．並列分散ファイルは各ノードのローカルディスク上に分割して格納されている．この拡張により並列分散 Lites 上の協調動作するプロセス群は，大量のデータからなる仮想的に単一のファイルの異なる領域に対して並列的にファイルアクセスすることが可能である．

本機能は UNIX のファイルの概念を拡張し，並列分散 Lites の一機能である並列分散プロセス管理機構による遠隔プロセス生成を組み合わせている．この結果，並列ファイルに対する並列アクセスを，各断片ファイルの存在するノード上のローカルディスクへのアクセスへと置き換え，効率的な並列ファイルアクセスが可能である．

参考文献

[1] J. Helander, "Unix under Mach – The Lites Server," Master Thesis, Helsinki University, Dec 30, 1994.

[2] Marshall Kirk McKusick, Keith Bostic, Micheal J. karels, John S. Quarterman, "The Design and Implementation of the 4.4 BSD Operating System," Addison-Wesley publishing, 1996.

[3] R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, B. Lyon, "Design and Implementation of the SUN Network Filesystem," Proc. USENIX Summer Conf., June 1985, pp. 119-130.

[4] Michael N. Nelson, Brent B. Welch, John K. Ousterhout, "Caching in the Sprite network file system," ACM Transactions on Computer Systems, Vol. 6, No. 1 (Feb. 1988), pp. 134-154.

[5] Thomas E. Anderson, Michael D. Dahlin, Jeanna M. Neefe, David A. Patterson, Drew S. Roselli, Randolph Y. Wang, "Serverless Network File Systems," SOSP, December, 1995.

[6] S. R. Kleiman, "Vnodes: An Architecture for Multiple File System Types in Sun UNIX," Proc. Summer USENIX Conf., June 1986, pp. 238-247, Atlanta, GA.