ファイルシステム情報を利用する分散ストレージシステム

宮澤 元
南山大学 数理情報学部

E-mail: miyazawa@it.nanzan-u.ac.jp

本稿では、我々が現在実装中の分散ストレージシステムについて述べる。これは、ストレージ層とファイルシステム層の二層構造に基づく分散ファイルシステムのストレージ層にあたるもので、ファイルシステム層で管理する情報を用いてブロック管理を行う。これによりブロック配置を最適化してアクセス性能を向上し、ブロックの冗長性を持たせることができる。

1 はじめに

計算機ネットワークの普及に伴い、多様な特性を持つさまざまなネットワーク上で情報を共有する必要性が高まっている。計算機ネットワークを用いた情報共有という観点では、従来LAN (Local Area Network)上での情報共有が主流であったが、ADSL (Asynchronous Digital Subscriber Line) や FTTH (Fiber To The Home)、無線LANを用いたホットスポットサービスなどにより、インターネット接続が身近で一般的なものになるのに伴い、インターネットのような広域ネットワークを介した場合でも、現在電子メールやWebを使って行っている以上に密接な情報共有に対する要求が高まると考えられる。

また、情報共有といえば、従来は複数ユーザ間での情報共有のみが考えられることが多かったが、現在では一個人が複数の計算機を使い分けて作業する機会が増えており、これらの計算機の間で情報の一貫性を保つことも情報共有の一環として考慮すべきである。このような情報共有は、特にノート型計算機のように持ち運びのできる計算機を含むようなネットワークでは難しく、これらの計算機の間で適切な情報共有を行う必要がある。

計算機ネットワークを用いて計算機間でファイル共有を行う技術として発展してきたものに、分散ファイルシステムがある。従来からさまざまな研究が行われてきた他、Sun NFS[15, 12] のような商用システムも開発され、計算機ネットワークを用いた情報共有の枠組みとして実用的にも広く利用されている。

伝統的な分散ファイルシステムがネットワークを介して別の計算機にあるファイルを参照するための技術という色合いを持っていたのに対して、近年、分散ファイルシステムをネットワークによる分散性を吸収するためのストレージ層と、この上でさまざまなファイルサービスを提供するファイルシステム層とに分離して構成するアプローチが広く研究されている [8, 16, 14]。このような二層構造を取る理由としては以下のようなものが挙げられる。

- ネットワークの大規模化などに伴ってシステムが扱う情報量が増大し、従来の分散ファイルシステムのように単一のファイルサーバで情報を集中管理するのが困難である
- 複数のファイルサーバを用いる場合、管理・運用コストが増大する

- マルチメディア情報など、サイズが巨大なファイルをファイル単位で管理するコストは非常に大きい

われわれは、特性の異なるさまざまなネットワーク上で柔軟な情報共有を行うための枠組みを実現することを目的として研究を行っており、ストレージ層とファイルシステム層の二層構造に基づく分散ファイルシステムを新たに実装中である。本稿ではこの分散ファイルシステムの、特にストレージ層の設計について述べる。このストレージ層は、ネットワークに接続されたディスク装置群を単一の仮想ディスクとして見せるためのソフトウェア層で、分散性を吸収するとともに各ブロックについて複製を作ることによって冗長性をも確保する。これらストレージ層のブロック管理に、ファイルシステム層の情報を活用することにより、ブロック配置を最適化してアクセス性能を向上し、ブロックに冗長性を持たせることができる。

以下、2 節では本分散ファイルシステムのストレージ層の設計について述べる。システムの構成とともに、システムの動作についても説明する。ファイルシステム層との連携に関しては、現時点で検討中のファイルシステム層の設計とともに 3 節で示す。4 節で関連研究を紹介し、5 節を本稿のまとめとする。

2 ストレージ層の設計

この節では、本分散ファイルシステムのストレージ層の設計について、システム構成を述べる。分散性の吸収と冗長性の確保についてと、ブロックの読み出し・書き込み操作についても説明する。

2.1 システム構成

本分散ファイルシステムのストレージ層は、以下の構成要素からなる (図 1)。

- ストレージインターフェース部
  ファイルシステム層からのブロック読み出し・ブロック書き込み要求を受け付ける。ファイルシステム層で管理される情報を受け付けるためのインターフェースもここに含まれる。

- ブロック通信部
  ストレージインターフェース部で受け付けた要求が他のホストが持つブロックに対するものだった場合、そのホストとの間でブロックの送受信を行う。

- 論理ブロック管理部
  論理ブロック番号と、そのブロックが存在するホストと物理ブロック番号の対応付けを管理する。各ブロックは複製されているのでこの対応は一般に一対多の対応となり、冗長性を確保できる。

- 物理ブロック管理部
  物理ディスクを管理し、ストレージインターフェース部やブロック通信部からの要求を受けてブロックを物理ディスクから読み書きする。

2.2 分散性の吸収

ストレージインターフェース部では、ファイルシステム層からのブロックアクセス要求を受け取ると、論理ブロック管理部に問い合わせ、該当ブロックがどのホストに存在するかを調べる。該当ブロックがローカルホストに存在している場合、ストレージインターフェース部から物理ブロック管理部にブロックに関する要求が伝えられる。それ以外の場合、ブロック通信部が該当ブロックが存在しているホストと通信し、該当ブロックに読み書きを行う。

このとき、ブロック通信部が受信したブロックをローカルディスクにキャッシュすることもできる。この場合、ブロック通信部は物理ブロック管理部に受信ブロックのキャッシュを依頼する。このキャッシュは、性能向上のために一時

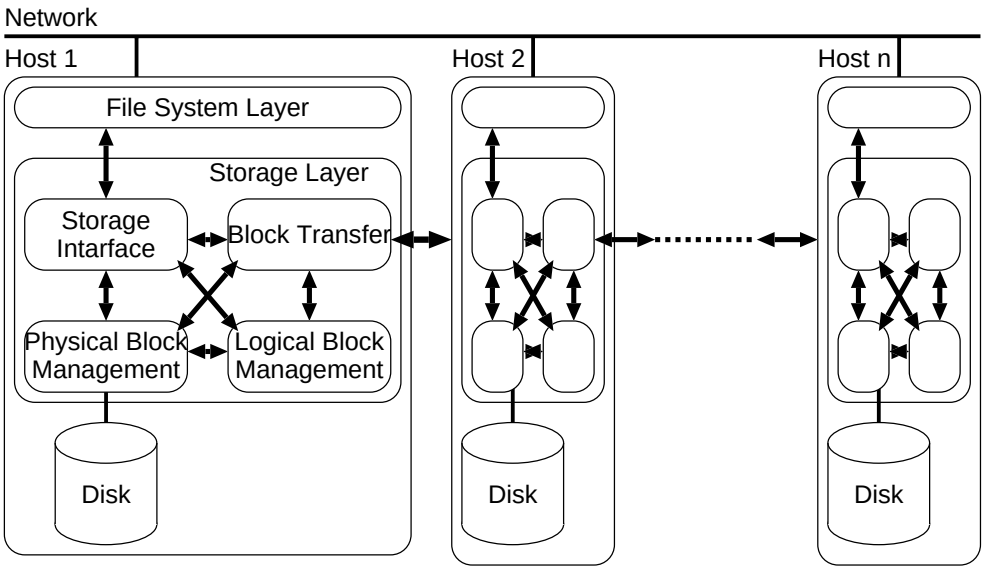

図 1: ストレージ層の構成

的に保持される場合と、次節で述べる冗長性の確保のための複製ブロックとして用いられ、長期的に保持される場合がある。

2.3 冗長性の確保

ある論理ブロックに対して複数の物理ブロックを対応させることにより、冗長性を確保する。通常、これらの物理ブロックの内容は全く同じであり、それぞれを異なるホストに配置することによって、あるホストが障害などでダウンした場合でもサービスの継続が可能となる。

性能向上の観点や、利用しているホストがネットワークから切断されるときのことなどを考えると、複製ブロックの一つは利用しているホスト上に保持されることが望ましい。それ以外の複製ブロックも、そのブロックを利用する可能性の高いホストに配置するべきである。このような複製ブロック配置を決定するにあたっては、ファイルシステムの利用状況やファイルの所有者状況など、ファイルシステム層から受け取る情報を利用する (3 節参照)。

2.4 ブロックの読み書き

ある論理ブロック読み出し要求に対して、複数の物理ブロックが対応する場合、利用しているホストからネットワーク的に最も近いホストに対して読み出し要求を出す。ホスト障害などでアクセスできない場合、他の複製ブロックを順次要求していく。また、モバイルホストなどがネットワークから切断されている場合、他のホストと通信することができず、必要なブロックを読み出せないことがありうる。このような場合には、Coda[6] の hoarding と同様の技術を用いて、必要なブロックをあらかじめモバイルホストのディスクに保持しておく必要がある。

論理ブロックを書き込む場合、このブロックに対応する複製ブロックを全て更新する必要がある。しかし、複製ブロックを保持しているホストが障害などでダウンしている場合、その複製ブロックを更新することができない。障害でダウンしている場合、障害を復旧して再起動する際にディスクをチェックして、複製ブロックを最新のものに更新することができる。モバイルホストなどがネットワークから切断されてい

る場合にも、ホスト障害と同様、他のホストと通信できなくなるのでモバイルホストの持つ複製ブロックを更新できない。この場合にも、障害から復旧したときと同様、ネットワークに再接続した時に複製ブロックを更新する必要がある他、モバイルホスト自身が更新したブロックを他のホストに複製する必要がある。このとき、あるブロックが 2 台以上のホストで更新され、更新が衝突することがありうる。このような衝突をある程度自動的に解決する方法も提案されている [7] が、何らかの形でユーザの判断をあおぐような機構も用意すべきであろう。

2.5 システムの実装状況

現在、上記の設計に基づき、Intel CPU ベースの PC(表 1) を用いて本分散ファイルシステムのストレージ層の実装中である。実装は Redhat Linux 8.0 (kernel 2.4.18) を改造する形で行っている。

3 ファイルシステム層とストレージ層の連携

本システムのストレージ層はネットワーク上で仮想ディスクを提供するものなので、通常のディスク装置上で運用できるファイルシステムならば、原理的には全て本システムのストレージ層の上で利用することができる。しかし、ストレージ層の性質を踏まえた上で、専用のファイルシステム層を設計することにより、全体としてより効率的な動作が期待できる。

ファイルシステム層については現在まだ詳細は決定していないが、ブロックの複製によって、書き込み時のコストが大きいことを考えると、書き込みコストを比較的低減できる Log-structured ファイルシステム [11, 13] をベースにしたものを検討中である。一ユーザが複数の計算機を用いる場合の情報共有を支援するという視点から、システム全体で単一のログを用意するのではなく、ファイルの利用者ごとにログを分割し、ログ単位で必要な計算機に複製を用意する。

これを可能にするために、ストレージ層はブロックの複製を作成するべきホストのヒントを得るためのインターフェースを備える必要がある。ファイルシステム層は、このインターフェースを通じて、ファイルの所有者・利用者情報を元に複製を設ける計算機のヒントを与える。

将来的には、ユーザの同一性を決定するための仕組みを用意する必要もあると考えられる。現在、Unix ベースのオペレーティングシステム (OS) では、ユーザ ID(UID) を用いてこれを行っているが、一人の実ユーザが複数の計算機を用いる場合、これらの全てでこのユーザの UID が一致しているとは限らない。また、別人が故意に同じ UID を用いてファイルにアクセスしようとするなどの問題も考えられる。

同様に、計算機の同一性を調べるマシン ID のようなものを導入する必要もあるかもしれない。現在、計算機の一意性はネットワークアドレス (IP アドレス) を使って調べることが多いが、そもそもネットワークゲートウェイのように複数の IP アドレスを持つ計算機も多い。また、IPv4 のプライベートアドレスのように、一意性を保証できないアドレスも存在する。更に、モバイルホストのようにネットワークアドレスが変化する場合もある。これに関しては、Mobile IP[9] のような技術を用いれば解決できるかもしれないが、さらに検討する必要がある。

4 関連研究

ネットワークに接続された複数の計算機に接続されたディスク装置を用いて、全体として単一のファイルシステムとして動作させるようなシステムは、これまで数多く提案されている。

Zebra[3, 4, 5] は、ネットワークに接続された計算機のディスク装置を RAID (Redundant Arrays of Inexpensive Disks)[10] と同様に用いてストライピングを行うことにより、ファイルアクセス性能を向上する分散ファイルシステムで

	CPU	メモリ	ディスク	ネットワーク
デスクトップ PC	Pentium-4 2.4GHz	1GBytes	120GBytes	1000Base-SX
ノート PC	Celeron 300MHz	64MBytes	4GBytes	100Base-TX

表 1: 実装用 PC の仕様

ある。Log-structured ファイルシステム [11, 13] の手法を用いることによって、サイズの小さなファイルの書き込みにおいても性能低下を押さえるような工夫がなされている。xFS[1, 2] は、Zebra の手法を改良し、分散処理をさらに進めたシステムである。

Petal[8] と Frangipani[16] はファイルシステム層とストレージ層の二層構造をとる分散ファイルシステムである。前者が分散ストレージシステムであり、後者が前者の上で動作するファイルシステムにあたる。分散性の吸収やブロックの複製による冗長性の確保は Petal で行われているが、ファイルシステムである Frangipani の管理情報を利用するようなことは行われていない。

PersonalRAID[14] は、分散ストレージシステムである。単に複数の計算機にブロックを複製するだけではなく、持ち運び可能なリムーバブルディスクにも書き込みを行ったブロックをログとして記録する。このディスクを持ち運ぶことにより、互いにネットワークで接続されていない計算機同士の間でもファイル共有を透明に行うことができる。しかし、ネットワーク接続があるような場合にこれを積極的に利用してファイル共有を行うようなことは考慮されていない。

5 まとめ

本稿では、我々が現在実装中の分散ストレージシステムについて述べた。これは、ストレージ層とファイルシステム層の二層構造に基づく分散ファイルシステムのストレージ層にあたるもので、ファイルシステム層で管理する情報を用いてブロック管理を行う。これによりブロック配置を最適化してアクセス性能を向上し、ブロックの冗長性を持たせることができる。具体的には、ファイルの所有者や利用統計などを元に、ブロックの複製を配置するホストをヒントとして与えるようなインターフェースを設けることを検討している。

謝辞

この研究は、2002 年度 南山大学パッヘ研究奨励金 (Pache Research Subsidy)I-A-1 の助成を受けています。

参考文献

[1] Thomas E. Anderson, Michael D. Dahlin, Jeanna M. Neefe, David A. Patterson, Drew S. Roselli, and Randolph Y. Wang. Serverless Network File Systems. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, pages 109–126, December 1995.

[2] Thomas E. Anderson, Michael D. Dahlin, Jeanna M. Neefe, David A. Patterson, Drew S. Roselli, and Randolph Y. Wang. Serverless network file systems. *ACM Transactions on Computer Systems*, 14(1):41–79, February 1996.

[3] John H. Hartman and John K. Ousterhout. Zebra: A striped network file system. In *Proceedings of the Usenix*

File Systems Workshop, pages 71–78, May 1992.

[4] John H. Hartman and John K. Ousterhout. The Zebra striped network file system. In *Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles*, pages 29–43, 1993.

[5] John H. Hartman and John K. Ousterhout. The Zebra striped network file system. *ACM Transactions on Computer Systems*, 13(3):274–310, August 1995.

[6] James J. Kistler and M. Satyanarayanan. Disconnected operation in the coda file system. In *Proceedings of 13th ACM Symposium on Operating Systems Principles*, pages 213–225, October 1991.

[7] Puneet Kumar and M. Satyanarayanan. Flexible and Safe Resolution of File Conflicts. In *Proceedings of the Usenix Winter 1995 Technical Conference on Unix and Advanced Computing Systems*, January 1995. available via ftp (CMU-CS-94-214).

[8] Edward K. Lee and Chandramohan A. Thekkath. Petal: Distributed virtual disks. In *Proceedings of the 7th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPROS VII)*, pages 84–92, October 1996.

[9] Ip routing for wireless/mobile hosts (mobileip). URL. http://www.ietf.org/html.charters/mobileip-charter.html.

[10] David A. Patterson, Garth Gibson, and Randy H. Katz. A case for redundant arrays of inexpensive disks(RAID). In *ACM SIGMOD*, pages 109–116, June 1988.

[11] Mendel Rosenblum and John K. Ousterhout. The design and implementation of a log-structured file system. In *Proceedings of 13th ACM Symposium on Operating Systems Principles*, pages 1–15, October 1991.

[12] Russel Sandberg, David Goldberg, Steve Kleiman, Dan Walsh, and Bob Lyon. Design and Implementation of the Sun Network Filesystem. In *Proceeding of the Summer 1985 USENIX Conference*, pages 119–130, Portland OR (USA), June 1985. USENIX.

[13] M. Seltzer, K. Bostic, M. K. McKusick, and C. Staelin. An implementation of a log-structured file system for UNIX. In *Proceedings of the Winter 1993 USENIX Technical Conference*, pages 307–326. USENIX Association, 1993.

[14] Sumeet Sobti, Nitin Garg, Chi Zhang, and Xiang Yu. Personalraid: Mobile storage for distributed and disconnected computers. In *Proceedings of the FAST 2002 Conference on File and Storage Technologies*, January 2002.

[15] Sun Microsystems, Inc. NFS: Network file system protocol specification. RFC 1094, Network Information Center, SRI International, March 1989.

[16] Chandoramohan A. Thekkath, Timothy Mann, and Edward K. Lee. Frangipani: A scalable distributed file system. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, pages 224–237. ACM, October 1997.