システムソフトウェアと
オペレーティング・システム 88－7
（２００１．７．２７）

# 複製を用いた大規模情報サーバの構築

宮澤 元

南山大学 数理情報学部
情報通信学科
〒 489-0863 愛知県 瀬戸市 せいれい町 27
miyazawa@it.nanzan-u.ac.jp

千葉 滋

東京工業大学大学院 情報理工学研究科
数理・計算科学専攻
〒 152-8552 東京都目黒区大岡山 2-12-1
chiba@is.titech.ac.jp

要旨

本稿では、多数のクライアントからの要求を処理するための情報サーバ構築技術として、複製を用いたサーバの並列化技法について述べる。情報サーバを複数のサーバ計算機を使って構成し、クライアントに提供する情報の複製を各サーバ計算機に持たせる。これにより各サーバ計算機がクライアントからの要求を分担して処理することができ、サーバ全体としての処理能力が向上することが期待される。プロトタイプシステムを用いた単純な読み書き要求の処理性能を測定する実験では、2 台のサーバ計算機を用いることによって、サーバ計算機が 1 台の場合に比べ、読み出し処理で 2 倍前後、書き込み処理で 1.2〜2 倍程度の性能向上が見られた。しかし、クライアントが大量の情報にアクセスした場合、サーバ計算機におけるファイルキャッシュの影響で処理性能そのものは低い値にとどまる場合があった。

# Construction of a Large-scale Information Server using Replications

Hajime Miyazawa

Dept. of Info. and Telecom. Eng.,
Nanzan University
27 Seirei-cho, Seto, Aichi, 489-0863
miyazawa@it.nanzan-u.ac.jp

Shigeru Chiba

Dept. of Math. and Computing Sciences,
Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo, 152-8552
chiba@is.titech.ac.jp

**Abstract**

This paper describes an idea of parallelizing an information server where multiple server hosts have their own replication of stored information for clients. As each request to the information server can be processed by any server host, total performance of the information server will be improved. Experiments with the prototype system showed that the server with two hosts performed around two times as good as that with only one hosts in read processing, and 1.2 to 2 times in write processing. On the other hand, accessing a vast amount of information by clients decreases the performance of the sever cluster under some experimental settings because of the low hit ratio of file caches.

# 1 はじめに

計算機ネットワークが普及するにつれて、Webサーバや分散ファイルシステムサーバのように、情報を集中管理し、クライアントからの要求に応じて情報の取り出し、更新を行うサーバを利用することが増えている。本稿ではこのようなサーバを総称して情報サーバと呼ぶことにする。

インターネットのような広域分散環境で動作する情報サーバには非常に多数のクライアントからの要求が集中することがある。これらのサーバが十分な処理能力を備えていない場合には、サーバがボトルネックとなり、システム全体としての性能が低下する。

従来、これらのサーバで大量の要求を処理するために、RAID(Redundant Array of Inexpensive Disks) [10] のような高速ディスク装置を備えた高性能なサーバ計算機を利用する方法がとられてきた。しかし、そのような計算機は非常に高価であるし、またいくら高性能といっても単一のサーバ計算機の処理能力には限界がある。そこで、比較的安価な計算機を並列に用いて高性能サーバを実現するアプローチが提案されている [8, 1, 3]。

我々はこのようなサーバ並列化技術のひとつとして、クライアントに提供する情報の複製を全てのサーバ計算機に持たせる方式 (以下、複製方式と呼ぶ) について検討してきた。複製方式では、全てのサーバ計算機が情報の複製をもつので、どのサーバ計算機が読み出し要求を処理しても問題ない。一方、複数のサーバ計算機で書き込み要求を並列に処理すると、更新される情報の一貫性の問題が生じる。そこで複製方式では、クライアントからの書き込み要求を受けた場合、該当する情報の複製を一時的に中断し、単一のサーバ計算機のみがこの書き込み要求を処理する。書き込み処理が終了した後、再度複製が再開される。ある情報の複製が中断されている場合でも、それ以外の情報の読み出し要求は各サーバ計算機で並列に処理することができる。

この複製方式を検討した理由としては以下のようなものがある。

- ハードディスクの大容量化と低価格化
  ハードディスクの大容量化と低価格化が急激に進んでおり、パーソナルコンピュータ (PC) 用でも 100GB 程度の容量を持つものが市販されている。こうした大容量ディスクの 1GB 当たり単価は 300 円を割り込んでおり、複数のサーバ計算機に情報の複製を持たせるために十分な容量を比較的安価に確保できる。

- 書き込みアクセスの少なさ
  Unix や WindowsNT におけるファイルシステムトレースによると、全てのファイルアクセス数に占める書き込みアクセス数の割合は 10〜20%と低い [4, 16]。また、World-wide Web (WWW) のサーバに対する要求は読み出し要求がほとんどであり、書き込み要求の割合はかなり低いと考えられる。従って、情報サーバに対してこのようなアクセスパターンを想定する限り、複数のサーバ計算機間で情報の一貫性を保持するコストは比較的低いと期待できる。

本稿では、複製方式による並列情報サーバについて述べるとともに、この並列情報サーバが単純な読み書き要求を処理する性能をプロトタイプシステムを用いた実験で測定する。

以下、2 節では複製方式の詳細について述べる。現在実装されている複製方式のプロトタイプシステムとこれを用いた実験について 3 節で示す。4 節では関連研究を紹介し、5 節を本稿のまとめとする。

# 2 複製を用いた並列情報サーバ

複製方式による情報サーバは、それぞれがハードディスクを持つ複数のサーバ計算機から構成される。各サーバ計算機は、クライアントに提供する情報の複製を持つ。

## 2.1 読み出し要求の処理

全サーバ計算機が同じ情報を持つので、どのサーバ計算機も、クライアントからの情報の読み出し

要求を処理できる (図 1)。従って、クライアントからの要求を各サーバ計算機にランダムに振り分けるなど、比較容易に負荷分散を実現できる。

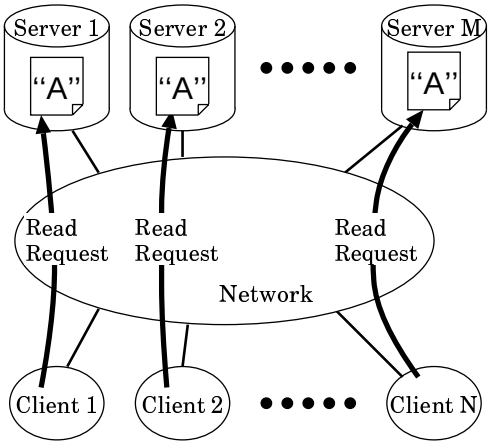


図 1: 読み出し要求の処理

## 2.2 書き込み要求処理時の複製の中断

クライアントからの書き込み要求を 各サーバ計算機が独立に処理すると、各サーバ計算機の持つ情報の一貫性を保つことができない。複製方式では、書き込み要求を受け付けたサーバ計算機が、他のサーバが持つ情報を一時的に無効化することによって情報の一貫性を保つ (図 2)。これを複製の中断と呼ぶ。複製の中断はファイル単位で行われ、あるファイルに書き込みが行われている場合でも、それ以外のファイルに対する読み出し要求を並列に処理することができる。

## 2.3 要求の転送

ある情報に対する書き込み要求によって複製が中断された場合、この情報の有効な複製を持つサーバ計算機は 1 台しかないので、これ以降クライアントからの要求を並列処理することができない。そのため有効な複製を持たないサーバ計算機は、クライアントからの要求を受け付けると、有効な複製を持つサーバ計算機に要求を出し直すようにクライアントに指示する。クライアントはこの指示に従って、有効な複製を持つサーバ計算機に要求

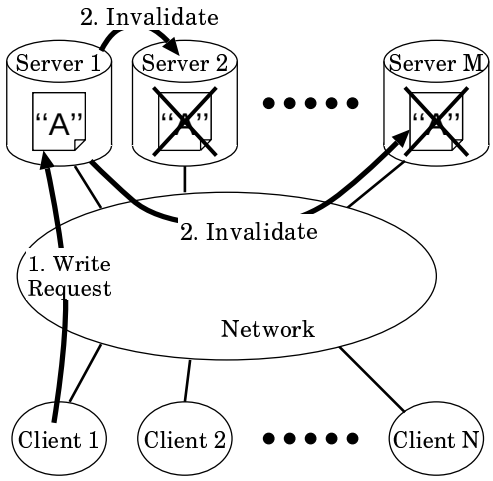


図 2: 複製の中断

を出し直す。例えば図 3 は、以下のように要求が転送される過程を示している。

1. クライアント 2 がある情報に対するアクセスをサーバ 2 に要求する
2. 該当する情報の複製は中断されているので、サーバ 2 はクライアント 2 にメッセージを送り、有効な複製を持つサーバ (ここではサーバ 1) に要求を出し直させる。
3. クライアント 2 はサーバ 1 に要求を出し直す、

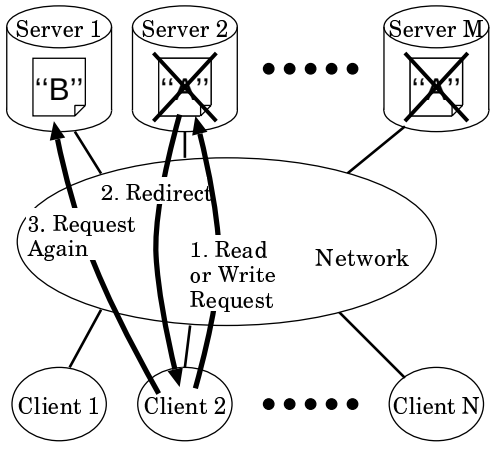


図 3: 要求の転送

## 2.4 複製の再開

複製が中断された情報は、適切なタイミングで再度各サーバ計算機に複製する必要がある。これ

を複製の再開という。現在は、複製が中断された情報に対して、異なったクライアント計算機からの読み出し要求が連続して出された場合に複製を再開する[1]。図 4 に、以下のような複製の再開の過程を示す。

1. クライアント 1 がサーバ 1 へ読み出し要求を出す。
2. クライアント 2 がサーバ 1 へ読み出し要求を出す。
3. サーバ 1 は、自分の持っている情報を複製し、他のサーバに送信する。

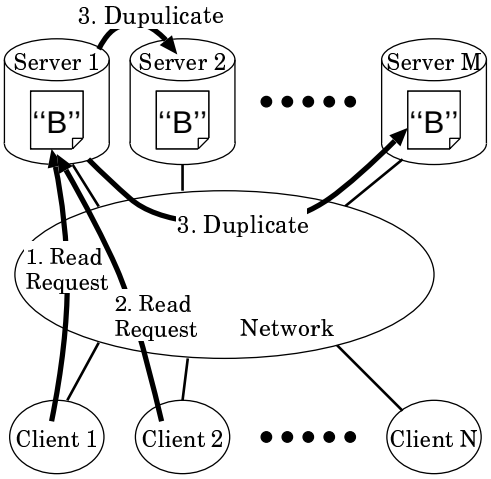


図 4: 複製の再開

# 3 実装および実験

本節では、2 節で述べた複製方式による並列情報サーバのプロトタイプシステムの実装と、これを用いた実験について述べる。

## 3.1 プロトタイプの実装

我々は、複製方式による並列情報サーバのプロトタイプシステムを Linux 上で動作するユーザレベルプロセスとして実装した。プロトタイプシステムは、クライアント側ライブラリ、サーバプロセス、シリアライザから構成される。

[1] 実際に複製の再開をどのタイミングで行うべきかは自明ではなく、今後検討の余地がある。

クライアント側ライブラリ サーバ計算機に要求を出すためにクライアントプログラムにリンクして使用される。サーバプロセスとのインターフェースには、ファイル全体を一気に読み出し・書き込みするための READ と WRITE を含む独自のプロトコルを使用している。

サーバプロセス 各サーバ計算機上で動作し、クライアントからの要求を処理する。クライアントがアクセスする情報は、サーバ計算機のディスク上に Unix ファイルとして格納されており、サーバプロセスは通常の Unix のシステムコールインターフェースを介してこれらの情報にアクセスする。

シリアライザ 情報に対する排他書き込み権を管理しており、必要に応じてサーバプロセスと通信し、情報の一貫性を保持する。サーバ計算機のうちの 1 台で動作する。

### 3.1.1 読み出し要求の処理

クライアントからの読み出し要求を受けたサーバプロセスは、該当の情報の複製の状態 (複製が行われているかどうかと、複製が中断されていれば有効な複製を持つサーバのヒントなど) を制御ファイルに保持しており、これに応じて以下のように要求を処理する。

- 複製が行われている場合
  そのままディスク上のファイルを読み出し、クライアントにファイル内容を返す。

- 複製が中断されている場合
  制御ファイルに保持されたヒントに基づき、クライアントに要求を出し直すように指示する。この後、該当の情報の複製再開・複製中断が連続して起こってからクライアントからの再要求がサーバ計算機に届いた場合、ヒントが間違っていることもありうるが、この場合でも再要求先のサーバ計算機が再度要求の出し直しを指示する。従って、最終的にはクライ

アントの要求をいずれかのサーバ計算機が処理することになる。

#### 3.1.2 書き込み要求の処理

有効な複製を持っている場合、サーバ計算機はシリアライザと通信して排他書き込み権を得てから、書き込み処理を行う。シリアライザと通信したとき、同じ情報の排他書き込み権を既に得ているサーバプロセスが他にいた場合、シリアライザは排他書き込み権の代わりに排他書き込み権を持つサーバのアドレスを返す。サーバプロセスは、この情報に基づいてクライアントに要求を出し直すように指示する。クライアントが要求を出し直すまでに、該当の情報の複製が再開され、また中断された場合には、ヒントが間違っていることもありうるが、この場合でも再要求先のサーバ計算機が再度要求を出し直すように指示する。従って、最終的にはクライアントの要求をいずれかのサーバ計算機で処理することになる。サーバ計算機が書込み要求を受け付けた時に既に有効な複製を持っていなかった場合、制御ファイルのヒントに従ってクライアントに再要求を指示する。

### 3.2 実験

ネットワークスイッチに接続されたPC上で、プロトタイプシステムを用いて実験を行った。表1に実験環境の詳細を示す。

#### 3.2.1 ネットワークによるサーバ処理性能の違い

サーバ計算機を接続するネットワークインターフェースによる処理性能の違いを見るために、利用するサーバ計算機を1台のみとし、ネットワークインターフェースを100Base-TXと1000Base-SXと変えて、以下の実験を行った。

1. ファイルサイズ $s$ のファイル50個を1セットとしてサーバに用意する。
2. クライアントは1.のファイルセットをサーバから連続して読み出す。または、このファイルセットに対して連続して書込を行う。
3. 20台のクライアント計算機が以上の操作を各クライアントとも同一のファイルセットに対して行う。

全クライアントが読み出しを行う場合と書き込みを行う場合について、ファイルサイズ $s$ を1KB, 4KB, 16KB, 64KB, 1024KB(1MB), 4096KB(4MB)と変化させて、サーバ計算機が1秒あたりに処理するファイル転送量を計測した。測定は同一の条件についてファイルキャッシュをクリアせずに連続して8回行い、その平均値をとった。図5に結果を示す。

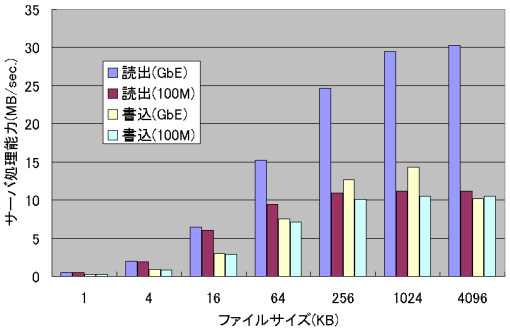


図5: ネットワークによるサーバ処理性能の違い

読み出し性能を見てみると、100Base-TXではファイルサイズが256KB程度以上でネットワークの理論性能(12.0MB/sec.)に近い11MB/sec.程度となっている。一方、1000Base-SXではファイルサイズが1MB以上で30MB/sec.前後となっており、サーバ計算機1台ではネットワーク帯域を使いきれないことを示している。

書き込み性能は、100Base-TXの場合ファイルサイズが大きくなっても10MB/sec.前後で頭打ちとなっている。1000Base-SXの場合、ファイルサイズが1MBのとき最も処理性能が高く、15MB/sec.弱となっているが、ファイルサイズが4MBのときはかえって性能が悪化している。これは、ファイルサイズの合計が大きすぎて(4MB×50個=200MB)、ファイルキャッシュのミスを引き起こしていることと、共通のファイルセットに対して書き込みを行うので、サーバ計算機でクライアントからの要求を並行に処理できないことが原因だと考えられる。

| | クライアント計算機 | サーバ計算機 |
|---|---|---|
| PC | IBM-PC/AT 互換機 (富士通 FM-V 6500CL4c) | |
| CPU | Intel Celeron 500MHz | |
| メモリ | 128MB | |
| ディスク | UltraATA66, 10.2GB (5,400rpm) | |
| ネットワーク | 100Base-TX (Intel EtherExpressPRO/100 相当) | 1000Base-SX (Planex GN-1000SX) |
| 台数 | 20 | 2 |
| OS | Vine Linux 2.1 | Vine Linux 2.1.5 with kernel-2.2.19 |
| ネットワークスイッチ | Planex GS-2124NW (100Base-TX 24 ポート、1000Base-SX 2 ポート) | |

表 1: 実験環境

### 3.2.2 複製方式の読み出し性能

プロトタイプシステムにおいて複数のサーバ計算機を使った場合のサーバの読み出し性能を調べるため、サーバ計算機の台数を 1 台と 2 台と変えて以下のような実験を行った。サーバ計算機のネットワークインターフェースは 1000Base-SX とした。

1. ファイルサイズ $s$ のファイル 50 個を 1 セットとして、クライアント計算機の台数分 20 セットをサーバに用意する。
2. 1 台のクライアントは 1. の 20 セットのうち自分用の 1 セットをサーバから連続して読み出す。
3. 20 台のクライアント計算機が以上の操作を同時に行う。

さらに以下のような実験を行った。

1. ファイルサイズ $s$ のファイルを 50 個サーバに用意する。
2. クライアントは 1. のファイル 50 個をサーバから連続して読み出す。
3. 20 台のクライアント計算機が以上の操作を共通のファイルセットに対して行う。

それぞれの実験で、ファイルサイズ $s$ を 1KB, 4KB, 16KB, 64KB, 1024KB, 4096KB(共通のファイルセットの場合のみ) と変化させ、サーバ計算機が 1 秒あたりに処理するファイル転送量を測定した。測定は同一条件でファイルキャッシュをクリアせずに

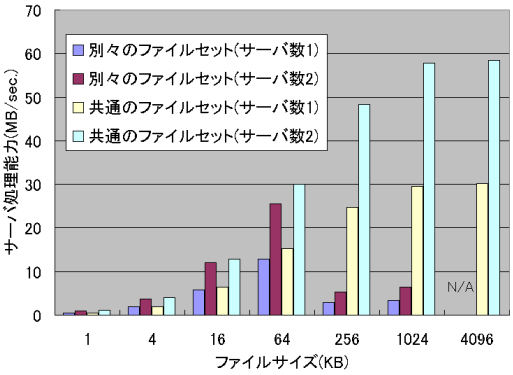


図 6: 並列サーバの読み出し性能

連続して 8 回行い、その平均値をとった。実験結果を図 6 に示す。

サーバ計算機を 2 台用いた場合、全般に 2 倍近くの性能向上が見られる。しかし、別々のファイルセットを読み出す場合、ファイルサイズが 256KB 以上ではファイルセット全体のサイズが大きすぎ (256KB×50 個 ×20 クライアント =256MB)、サーバ計算機のファイルキャッシュのミスを起こすので極端に性能が落ちることが確認できる。共通のファイルセットを読み出す場合は、このような極端な性能の落ち込みは見られないが、更に大きなファイルサイズで実験を行った場合には別々のファイルセットを読み出す場合と同様の結果になることが予想される。

### 3.2.3 複製方式の書き込み性能

プロトタイプシステムにおいて、複数のサーバ計算機を使った場合のサーバの書き込み性能を調べるため、サーバ計算機の台数を 1 台と 2 台と変

えて以下のような実験を行った。サーバ計算機のネットワークインターフェースは1000Base-SXとした。

1. 20台のクライアント計算機がサイズ$s$のファイル50個をサーバに対して連続して書き込む。
2. サーバ計算機は1.のファイルをそれぞれ別々のファイルセットとしてディスクに書き込みむ

さらに以下のような実験を行った。

1. 20台のクライアント計算機がサイズ$s$のファイル50個をサーバに対して連続して書き込む。
2. サーバ計算機は1.のファイルを共通のファイルセットとしてディスクに書き込みむ。

それぞれの実験で、ファイルサイズ$s$を1KB, 4KB, 16KB, 64KB, 1024KB, 4096KB(共通のファイルセットの場合のみ)と変化させ、サーバ計算機が1秒あたりに処理するファイル転送量を測定した。測定は同一条件でファイルキャッシュをクリアせずに連続して8回行い、その平均値をとった。実験結果を図7に示す。

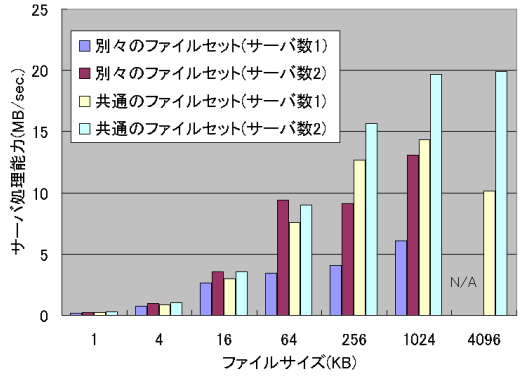


図 7: 複製方式の書き込み性能

別々のファイルセットに書き込む場合、サーバ計算機を2台使用すると1台の場合のほぼ1.3〜2倍の性能向上が見られる。特に、ファイルサイズが64KB、256KBのときは、サーバ計算機1台の場合の性能がファイルキャッシュのミスにより低下している関係で2倍以上の性能向上が見られる。共通のファイルセットに書き込む場合、書き込み要求を転送することによって若干性能が落ちるが、複製の中断はファイルごとに行われているので並列処理の効果が出ており、1.2〜2倍性能が向上していることが分かる。

# 4 関連研究

Andrew File System (AFS)は大規模分散環境を指向したファイルシステムとして開発された[12, 9, 15]。AFSでは、ファイルサーバの負荷を低減するために、クライアントディスクに対するキャッシュや共有セマンティクスに関する工夫を行っている。AFSの後継システムのCodaでは、サーバの利用可能性の向上のために複数のサーバ計算機に複製を持たせるなどの拡張が行われている[13]。

高速LANに対応した分散ファイルシステムに関する研究レベルのシステムとしてZebra [6, 7, 8]やBerkeley NOW Project [1]のxFS [2, 3]がある。これらは高速LANに接続された複数のサーバ計算機をファイルサーバとして用いるシステムである。Log-structured File System(LFS) [11, 14]を採用し、各サーバ計算機が持つディスク装置をLAN上に分散したRAIDと見なしてファイルを格納する(ストライピング)。そのため、ファイルに対するアクセス要求は、各サーバ計算機で並列処理され、高速LANの広い帯域を生かした高いスループットでのファイルアクセスが実現できる。

ネットワークで接続された複数の計算機を利用してHTTP要求処理の負荷分散を行うことによってWebサーバの性能を向上する研究も数多く行われている[5]。

# 5 まとめ

多数のクライアントからの要求を処理するための情報サーバ構築技術として、複製方式による並列情報サーバについて述べた。情報サーバを構成する複数のサーバ計算機のそれぞれが、クライアントに提供する情報の複製を持ち、クライアントからの要求を分担して処理することによってサーバ全体としての要求処理性能が向上することが期待される。複製方式による並列情報サーバのプロトタイプシステムをLinux上のユーザレベルプロ

セスとして実装し、実験により単純な読み書き要求を処理する性能を測定した。2 台のサーバ計算機を利用した場合、 1 台の場合と比べて読み出しでは 2 倍前後、書き込みでは 1.2〜2 倍程度の処理能力向上が見られた。一方、クライアントが大量の情報にアクセスした場合には、サーバ計算機におけるファイルキャッシュのミスを引き起こし、性能が極端に低下する場合も見られた。今後は、本システムのより現実に近いワークロードでの振舞を明らかにするとともに、サーバ計算機におけるファイルキャッシュに関する検討を進めることが必要である。

## 謝辞

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

## 参考文献

[1] Thomas E. Anderson, David E. Culler, and David A. Patterson. A case for now (networks of workstations). *IEEE Micro*, Vol. 15, No. 1, pp. 54–64, February 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. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, pp. 109–126, December 1995.

[3] 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*, Vol. 14, No. 1, pp. 41–79, February 1996.

[4] Mary G. Baker, John H. Hartman, Michael D. Kupfer, Ken W. Shirriff, and John K. Ousterhout. Measurements of a Distributed File System. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles*, Operating Systems Review, pp. 198–212, Pacific-Grove, October 1991.

[5] Valeria Cardellini, Michele Colajanni, and Philip S. Yu. Dynamic load balancing on web-server systems. *IEEE Internet Computing*, Vol. 3, No. 3, pp. 28–39, May/June 1999.

[6] John H. Hartman and John K. Ousterhout. Zebra: A striped network file system. In *Proceedings of the Usenix File Systems Workshop*, pp. 71–78, May 1992.

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

[8] John H. Hartman and John K. Ousterhout. The Zebra striped network file system. *ACM Transactions on Computer Systems*, Vol. 13, No. 3, pp. 274–310, August 1995.

[9] John H. Howard, Michael L. Kazar, Sherri G. Menees, David A. Nichols, M. Satyanarayanan, Robert N. Sidebotham, and Michael J. West. Scale and performance in a distributed file system. *ACM Transactions on Computer Systems*, Vol. 6, No. 1, pp. 51–81, February 1988.

[10] David A. Patterson, Garth Gibson, and Randy H. Katz. A case for redundant arrays of inexpensive disks(RAID). In *ACM SIGMOD*, pp. 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*, pp. 1–15, October 1991.

[12] M. Satyanarayanan, John H. Howard, David A. Nichols, Robert N. Sidebotham, Alfred Z. Spector, and Michael J. West. The ITC Distributed File System: Principles and Design. In *Proceedings of the 10th ACM Symposium on Operating Systems Principles*, pp. 35–50, December 1985.

[13] Mahadev Satyanarayanan, James J. Kistler, Puneet Kumar, Maria E. Okasaki, Ellen H. Siegel, and David C. Steere. Coda: A Highly Available File System for a Distributed Workstation Environment. *IEEE Transactions on Computers*, Vol. 39, No. 4, pp. 447–459, April 1990.

[14] 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*, pp. 307–326. USENIX Association, 1993.

[15] Mirjana Spasojevic and M. Satyanarayanan. An Empirical Study of a Wide-Area Distributed File System. *ACM Transactions on Computer Systems*, Vol. 14, No. 2, pp. 200–222, May 1996.

[16] Werner Vogels. File system usage in windowsnt 4.0. In *Proceedings of 17th ACM Symposium on Operating Systems Principles (SOSP'99)*, pp. 93–109, December 1999.