ファイル自律配置方式を備えた仮想一元化 NAS システム X-NAS の実現と評価

川本 真一† 江端 淳† 沖津 潤† 保田 淑子† 樋口 達雄† 濱中 直樹†

†(株)日立製作所 中央研究所 〒185-8601 東京都国分寺市東恋ヶ窪 1-280

E-mail: †{skawamo, ebata, j-okitsu, yoshikoy, higuchi, hamanaka}@crl.hitachi.co.jp

あらまし 安価で容量スケーラブルなオフィス用途向け NAS を実現するため，複数のエントリ NAS を構成要素とし単一ファイルシステムビューを提供する仮想一元化 NAS システム X-NAS(eXpandable NAS)を提案する。仮想一元化 NAS システムには，ある要素 NAS に空きがなくなると他の要素 NAS に空きがあっても書き込みができなくなるディスク残量問題がある。従来は手作業でファイルを移動して対応していたが，X-NAS は使い勝手を向上するため，空きの少ない要素 NAS のファイルを空きの多い要素 NAS に自律的に移動する自律リバランス機能を備える。プロトタイプシステムによる評価から自律リバランス機能の有効性を検証した。

キーワード NAS, クラスタシステム， 単一ファイルシステムビュー，自律制御，ディスク残量問題，ファイルサイズ分布

Expandable NAS (X-NAS) and Its Autonomic File Redistribution Policy

Shinichi KAWAMOTO† Atsushi EBATA† Jun OKITSU† Yoshiko YASUDA†
Tatsuo HIGUCHI† Naoki HAMANAKA†

† Hitachi, Ltd., Central Research Laboratory 1-280 Higashi-koigakubo Kokubunji-shi, Tokyo 185-8601, Japan

E-mail: †{skawamo, ebata, j-okitsu, yoshikoy, higuchi, hamanaka}@crl.hitachi.co.jp

Abstract To achieve both reasonable cost of ownership and capacity scalability in office use NAS, we propose an expandable network attached storage (X-NAS) which is a cluster system of two or more entry-level NASes and offers an unified file system view. In such cluster system, the file size variation causes disk usage unbalancing problem. When one elemental NAS is filled up to its capacity, users can not write additional files into the system, even if the other elemental NASes have enough space. We solved this problem by adding an autonomic file redistribution policy to X-NAS. Experimental results indicate that this policy can solve disk usage unbalancing problem.

Keyword Network Attached Storage(NAS), Cluster System, Unified File System View, Autonomic Control, Disk Usage Unbalancing Problem, File Size Distribution

1. はじめに

安価で管理が容易であることから，企業の部門等がオフィスユースを目的としてエントリ NAS(Network Attached Storage)を導入するケースが増えている。インターネットコンテンツ市場の急速な拡大により，個々のデータ量は増大し，それと共に NAS に格納されるデータ量も増加していく。

エントリ NAS は安価だが容量拡張性に乏しい。容量が足りなくなると，新たに同種の NAS を購入することになるが，複数 NAS を使い分けなければならず使い勝手が悪い。拡張性の高いミッドレンジ NAS に移行することも考えられるが，価格がエントリ NAS に比べて大幅に高く，移行の手間もかかる。データの急増と現状 NAS 製品の価格体系を考慮すると，企業の部門等がオフィス用途に使用するには，安価で使い勝手が良く，容量拡張性の高い NAS へのニーズが高いと予測される。そこで，エントリ NAS を要素とするクラスタ構成により価格を抑え，複数要素 NAS を仮想化技術により単一ファイルシステムに見せることで使い勝手を向上し，要素 NAS の追加により容量拡張を実現した仮想一元化 NAS システム X-NAS(eXpandable NAS)を提案する。

NAS に格納するファイルのサイズにはばらつきがあるため，仮想一元化 NAS の各要素 NAS のディスク使用量に偏りが生じ，特定の要素 NAS だけディスクが一杯になる場合がある。このとき，他の要素 NAS には空きがあるにも拘わらず，それ以上ファイルを書き込めないというディスク残量問題が発生する。この問題に対し，従来は一杯になった要素 NAS の特定のディレクトリ以下のファイル群を手動で他の要素 NAS に移動していたが，使い勝手が悪く，管理コストが大きかった。そこで，X-NAS では，システムが自律的にファイル群を移動してディスク使用量の偏りを是正する自律リバランス機能を搭載し，ディスク残量問題を予防する。

本稿では，まず X-NAS の基本方式について述べた後，自律リバランス機能の実現と評価について述べる。

2. X-NAS の基本方式と課題

X-NAS は UNIX や Windows クライアントに対しネットワークファイルシステムを提供する NAS である。サポートプロトコルは NFS と CIFS である。本章では，X-NAS の基本コンセプト，基本方式，および，課題について述べる。

2.1. X-NAS の基本コンセプト

企業の部門やワークグループではオフィス用途の NAS に(1)低価格，(2)使い勝手の良さ，(3)容量拡張性，の三点を期待している。これらすべてのニーズを満足するため X-NAS の基本コンセプトを以下の三点とする[1]。

(a) 低価格を実現するため，エントリ NAS を要素としたクラスタ構成を取る。
(b) ユーザの使い勝手を向上するため，クライアントに対し単一ファイルシステムビューを提供。
(c) 容量拡張・縮退は要素 NAS の追加・削除により対応。

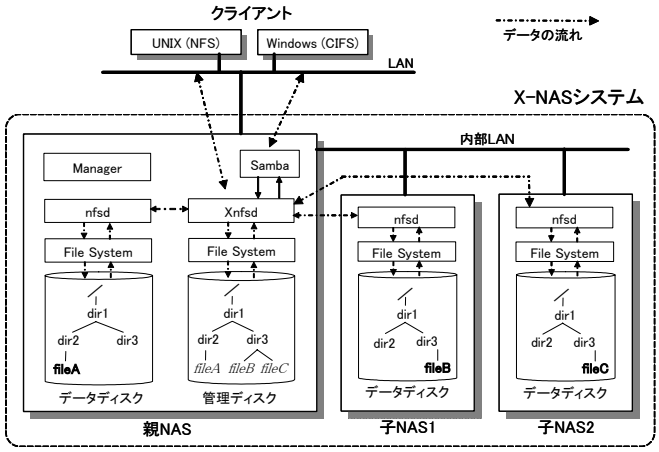

図 1: X-NAS の構成

2.2. 基本方式

本節では，X-NAS の構成，単一ファイルシステムビュー提供方式，および，容量拡張・縮退方式について述べる。

2.2.1. X-NAS の構成

X-NAS の構成を図 1 に示す。X-NAS は複数のエントリ NAS を構成要素としたクラスタシステムである。ただし，クライアントからの使い勝手を考慮すると，システム全体が一つの装置として扱えることが望ましい。そこで，一台の要素 NAS を X-NAS のエントリポイントとする。この要素 NAS を親 NAS と呼ぶ。クライアントが X-NAS にアクセスする場合には必ず親 NAS にアクセスするものとする。親 NAS 以外の要素 NAS を子 NAS と呼ぶ。X-NAS の唯一のエントリポイントである親 NAS には，クライアントからの全てのリクエストが集まる。従って，親 NAS はファイルアクセスに関するすべての情報を知ることができる。そこで，親 NAS に X-NAS のすべての機能を集約して持たせる。こうすることで，子 NAS には特別な機能を持たない通常の NAS を用いることができる。要素 NAS の台数に方式的な制限はないが，一般的な 100Mbit イーサネット利用時は性能的観点から 8 台が実用的な限界である。

親 NAS の Xnfsd は X-NAS のために開発した NFS サーバ機能を提供するデーモンプログラムである。Xnfsd は管理ディスクを用いながら要素 NAS を仮想化し，クライアントに単一ファイルシステムビューを提供する。親 NAS が UNIX クライアントから NFS リクエストを受けると，Xnfsd は要素 NAS に NFS リクエストを発行してデータディスクにアクセスし処理を行う。親 NAS が Windows クライアントから CIFS リクエストを受けると，フリーソフトウェアである Samba と親 NAS のファイルシステムが CIFS リクエストを NFS リクエストに変換し，それを Xnfsd が処理する。これにより，UNIX と Windows との間でファイルの共有ができる。Manager は新規開発したデーモンプログラムで，子 NAS の追加・削除機能を提供する。

2.2.2. 単一ファイルシステムビュー提供方式

(1) 仮想ファイルツリー

クライアントに単一ファイルシステムビューを提供するため，親 NAS はシステム全体のファイルツリーを管理ディスクに保持する。このファイルツリーを仮想ファイルツリーと呼ぶ。管

理ディスクには通常のUNIXファイルシステムが構築され，その上に仮想ファイルツリーの全てのディレクトリが作成される。また仮想ファイルツリーのすべてのファイルもこのファイルシステム上に作成されるが，これらは中身のデータを持たない**ダミーファイル**である。ファイルの実体は，要素 NAS のデータディスクに格納される。ファイルを細分化して各要素 NAS に分散配置する方法(ストライピング)も考えられるが[2][4]，要素NASの追加・削除による容量拡張の実現が困難となる。そこで，各ファイルを細分化せず要素 NAS のいずれか一つに配置する。ディレクトリ情報を冗長化するために，データディスクにも管理ディスクと同様にすべてのディレクトリを作る。

クライアントが X-NAS に三つのディレクトリ dir1～dir3 と，ファイル fileA～fileC を格納した場合の例を図 1 に示す。管理ディスクには仮想ファイルツリーが格納される。dir1～dir3 はディレクトリであり，*fileA*～*fileC*はダミーファイルである。ファイルの実体である **fileA**，**fileB**，**fileC** は，それぞれ親 NAS，子 NAS1，子 NAS2 のデータディスクに格納される。Xnfsd はクライアントに仮想ファイルツリーをアクセスさせる。クライアントが仮想ファイルツリーのディレクトリを辿ってファイルに到達し，そのファイルに 読み出しや書き込みを行うと，Xnfsd はファイルの実体が格納されている要素 NAS にアクセスし，ファイルの実体に対して読み出しや書き込みを行う。これにより，ファイルの実体はファイルごとに各要素 NAS に分散して配置されるが，クライアントには単一ファイルシステムビューを提供できる。

(2) ファイルの配置

親 NAS はファイルの実体がどの要素 NAS に格納されているか記録している必要がある。管理テーブルを用いファイルごとに情報を管理すると，テーブルサイズが膨大になる。一方，ファイルの識別子にハッシュ関数を適用して格納先要素 NAS を決定すると，管理テーブルは不要だが，ファイルと格納先要素 NAS の関係を変更することが難しく，容量拡張・縮退の実現が困難となる。テーブルサイズをコンパクト化し，かつ，ファイルと格納先要素 NAS の対応を柔軟に管理できるよう，ファイルの識別子にハッシュ関数を適用してファイルをいくつかのファイルグループに分け，ファイルグループと格納先要素 NAS との関係をテーブルで管理する。

ファイル識別子としてはファイルのフルパス名を使用する方法が考えられる。しかし，ファイル名が変更されると識別子が変わるので，ファイルの実体の格納先要素 NAS を変更しなければならないという問題がある。X-NAS に格納されている全てのファイルはダミーファイルという形で仮想ファイルツリー上に存在し，すべてのダミーファイルは固有の inode 番号を持つ。ファイル名が変更されてもその inode 番号は変更されない。そこでダミーファイルの inode 番号をファイル識別子として用いる。ハッシュ関数としてはファイルグループ数を除数とするモジュロ演算を用いる。ファイル f のダミーファイル f の inode 番号を i_f，ファイルグループ数を N とすると，ファイル f が所属するファイルグループ G_f は $G_f = i_f \bmod N$ となる。ファイルグループと格納先要素 NAS との関係はグループ管理テーブルというテーブルで管理する。

上記の演算を用いると，ファイルはほぼランダムに各ファイルグループに分散される。従って，各ファイルグループに属するファイルの数はほぼ同じとなる。また，ファイルグループの数を要素 NAS の 10 倍～1000 倍程度とし，各要素 NAS に対し容量に比例した数のファイルグループが割り当たるようにグループ管理テーブルを設定する。これにより，容量が同じなら，各要素 NAS にほぼ同数のファイルが格納される。以後，各要素 NAS の容量は同じであると仮定する。

(3) クライアント要求の処理

クライアントが X-NAS に対してファイル f の生成要求(NFS では CREATE プロシージャ)を発行すると，Xnfsd は管理ディスクの仮想ファイルツリーにファイル f のダミーファイル f を生成する。次に，生成したダミーファイル f の inode 番号にモジュロ演算を適用してファイルグループ番号を求め，グループ管理テーブルを参照してファイル f の格納先要素 NAS を求める。そしてその要素 NAS に NFS の CREATE プロシージャを発行し，ファイル f の実体 **f** を生成する。最後に，Xnfsd はクライアントにダミーファイル f のファイルハンドルを返す。

クライアントが X-NAS に対してファイル f の読み出し要求(NFS では READ プロシージャ)を発行すると，Xnfsd は管理ディスクの仮想ファイルツリーのダミーファイル f にアクセスして inode 番号を求め，その inode 番号にモジュロ演算を適用してファイルグループ番号を求め，グループ管理テーブルを参照してファイル f の格納先要素 NAS を求める。そして求めた要素 NAS に対して NFS の READ プロシージャを発行し，ファイル f の実体 **f** の読み出しを行い，得られたデータをクライアントに返す。書き込みについても同様である。

2.2.3. 容量拡張・縮退方式

容量拡張・縮退処理は親 NAS の Manager が行う。以下では，容量拡張と容量縮退に分けて説明する。

(1) 容量拡張

容量拡張とは，X-NAS に新規子 NAS を追加し，X-NAS の容量を大きくすることである。容量拡張処理は以下の 3 つのステップからなる。

(step1) 追加する新規子 NAS をシステムに登録
(step2) 追加子 NAS にディレクトリ作成
(step3) 既存要素 NAS の幾つかのファイルグループを追加子 NAS へ移動しグループ管理テーブルを変更

step2 まで実行することにより，追加子 NAS を要素 NAS として使用する準備が整う。ただし，子 NAS を追加しただけでは，追加子 NAS にファイルグループが割り当てられておらず，ファイルが書き込まれることはない。そこで， step3 を実行し各要素 NAS に同数のファイルグループを割り当て直す。

ファイルグループの移動は Manager の move モジュールによって行われる。move モジュールは，管理ディスクの仮想ファイルツリーを探索して移動するファイルグループに属するファイルを特定し，それらのファイル全てを移動元要素 NAS

から移動先要素 NAS に移動する。move モジュールがファイルグループを移動している最中も，Xnfsd はクライアントからのファイルアクセス要求を処理し続ける。ただし，クライアントが移動中のファイルグループに属するファイルにアクセスしようとした場合，Xnfsd はクライアントのファイルアクセス要求を破棄する。するとクライアントは NFS の規約に従いファイルアクセス要求を再送する。従って，その要求はそのファイルグループの移動が終了した後，実行されることになる。ファイルアクセス処理の停止よりも同期メカニズムの簡単化を優先しこの手法を採用した。ファイルグループ数を 10000～100000 とし，ファイルアクセスが移動中のファイルグループにぶつかる確率を下げることで，ファイルアクセス要求をほぼ遅延させることなく，容量拡張が可能となる。

(2) 容量縮退

容量縮退とは，格納されているデータは消さずに，X-NAS の子 NAS をシステムから切り離して，X-NAS の容量を小さくすることである。容量縮退は一部の子 NAS を X-NAS から切り離して別の用途に使用したい場合や，子 NAS を別の子 NAS とリプレースしたい場合に用いる。容量縮退が実行できるのは，縮退前に X-NAS に格納されているデータのサイズが縮退後の小さくなった容量より小さいことが前提である。容量縮退処理は以下の二つのステップからなる。

(step1) 削除する子 NAS に格納された全てのファイルグループを他の要素 NAS に移動

(step2) 削除する子 NAS を X-NAS から切り離す

step1 は Manager の move モジュールによって行う。容量拡張の場合と同様に，容量縮退中もクライアントからのファイルアクセスはほぼ止まらない。

2.3. 課　題

2.2.2で述べたように，Xnfsd は各要素 NAS にほぼ同数のファイルを配置することができる。しかし，ファイルサイズにはばらつきがあるため，各要素 NAS に同数のファイルを配置しても，各要素 NAS のディスク使用量に偏りが生じる可能性がある。偏りが生じると，特定の要素 NAS が一杯になり，他の要素 NAS のディスクに空きがあるにもかかわらず，ファイルの生成や追加書き込みができなくなる。これをディスク残量問題と呼ぶ。X-NAS の課題は，システムが自律的にディスク使用量の偏りを是正し，ディスク残量問題を予防する機能を実現することにある。

3. ファイルサイズ分布とディスク残量問題

3.1. ファイルサイズ分布の調査

ファイルサイズはファイルごとに異なる。Word 等の文書系ファイルは 10KB～2MB 程度，画像等のマルチメディアファイルは 100KB～20MB 程度，プログラム等のアーカイブファイルは 1MB～500MB 程度のサイズである。ブロードバンドの普及により，ファイルサイズは年々大きくなっている。

著者が日常使用しているファイルサーバや NAS に格納されている 5 つのファイル群についてサイズ分布を調査した。

表 1: ファイルサイズ分布の調査結果

#	ディスク使用量[MB]	ファイル数 ～1MB	～10MB	～100MB	～1GB
1	615	19,491	50	5	0
2	22,962	81,156	1,157	396	15
3	7,288	44,679	348	68	16
4	10,623	49,367	839	156	9
5	23,335	87,348	1,660	302	30

ファイル群 1: UNIX サーバ A における 1 ユーザのホームディレクトリ
ファイル群 2: UNIX サーバ B における 24 ユーザのホームディレクトリ
ファイル群 3: UNIX サーバ C における 20 ユーザのホームディレクトリ
ファイル群 4: Windows クライアント向け NAS A における共有フォルダ
ファイル群 5: Windows クライアント向け NAS B における共有フォルダ

結果を表 1 に示す。この表はファイルサイズを 0～1MB，1～10MB，10～100MB，100MB～1GB に分け，それぞれの範囲に入るファイルの数を調査したものである。サイズが 1GB を超えるファイルは存在しなかった。ファイル群 1～3 は UNIX プログラムの開発や実験に使用されている。一方，ファイル群 4 と 5 は Windows 系各種文書の共有やバックアップに使用されている。用途は異なるがいずれも傾向は同じとなった。10MB 未満の小さいファイルがファイル数の大半を占めるが，10MB 以上の大きいファイルも存在する。10MB 未満のファイルの大半は，UNIX 系(#1～3)ではプログラムソースやログ等のテキストファイルであり，Windows 系(#4,5)では Word 等の文章系ファイルであった。10MB 以上のファイルは UNIX 系か Windows 系かによらずほとんどすべてアーカイブファイルであった。表 1 の五つのデータを合計し，ファイル数とディスク使用量の累積率をグラフ化したものが図 2 である。この図から，全体の高々0.3%を占めるにすぎない 10MB 以上の大きなファイルが，ディスク使用量の 67%を占めることがわかる。従って，ディスク使用量は 10MB 以上の大きいサイズのファイルの数で大体決まってしまい，10MB 未満の小さなファイルの数にはあまり影響を受けない。

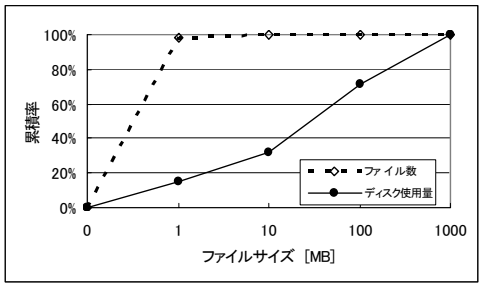

図 2: ファイル数とディスク使用量の累積率

3.2. ディスク偏りの見積もりとディスク残量問題

3.1節の調査結果の合計と同じファイルサイズ分布を持つファイル群が 100GB あり，これを親 NAS と 3 台の子 NAS から構成される X-NAS に格納すると仮定する。サイズが 10MB 未満であるファイルは 33GB 分，10MB 以上のファイルは 67GB 分ある。サイズが 10MB 未満のファイルは大量にあるため，X-NAS のファイル配置方式によって各要素 NAS に均等に分配される。このため，各要素 NAS に 33GB/4=8.3GB

ずつ配置されると見なせる。一方，10MB 以上のファイルは数が少ないため，必ずしも各要素 NAS に均等に配置できず偏りが生じる。一つの要素 NAS に 10MB 以上のサイズのファイルがすべて配置されてしまった場合を考えると，その要素 NAS のディスク使用量は 67GB+8.3GB= 75.3GB となる。一方，それ以外の要素 NAS のディスク使用量は 8.3GB である。使用量の最大と最小の差は67GBもある。このケースは極端であるが，偏りの最大値はこのように大きくなる。

X-NAS の各要素 NAS のディスク容量が 40GB であると仮定すると，X-NAS 全体の容量は 160GB であるから，100GB 分のファイルを格納できるはずである。しかし，大きなサイズのファイルが偏って格納される要素 NAS には最低 75.3GB のディスク容量が必要であるのに，実際には 40GB の容量しかない。従って，一つの要素 NAS が一杯となり，100GB のファイル群のうちの 53%すなわち 53GB しか X-NAS に書き込むことができず，深刻なディスク残量問題が発生する。

4. X-NAS の自律リバランス機能

これまで各要素 NAS の容量が同じであると仮定してきた。以後各要素 NAS の容量が異なる場合も考慮に入れる。各要素 NAS の容量が異なる場合，ディスク残量問題を予防するには，各要素 NAS の**ディスク残量**の偏りを是正する必要がある。偏りを是正するため，ユーザに手作業でファイルグループを移動させる方法が考えられるが，専門の管理者でないと実施は困難である。新規ファイルを書き込む際，ディスク残量の最も多い要素 NAS にそのファイルを配置すれば，偏りを抑えることができるように思える。しかし，この方法でもすでに X-NAS に格納済みの従来ファイルに追加書き込みが発生してファイルサイズが大きくなると偏りが生じる。

そこで，ディスク残量の少ない要素 NAS から多い要素 NAS に幾つかのファイルグループを移動する平準化処理を自律的に実施し，ディスク残量の偏りを是正する。この機能を**自律リバランス機能**と呼ぶ。ファイルグループの移動には，前述の Manager の move モジュールを利用する。X-NAS システムは，クライアントからのファイルアクセス処理と並列にバックグラウンドで平準化処理を行う。自律リバランス機能は図 1 の Manager の一機能として実現する。

4.1. 自律リバランス機能の要件

自律リバランス機能の要件を以下のように定める。

(a) 自律リバランス機能は，move モジュールを利用してファイルグループを移動する。移動中のファイルに対しクライアントからアクセス要求があると，それを破棄し再送させるため，ファイルアクセス性能を低下させる恐れがある。そこで，ファイルアクセス処理に対する影響を最小化する。

(b) 平準化処理は，ディスク残量の偏りを目的関数とした一種のフィードバック処理である。しかし，ファイルサイズは一様でないため，平準化処理を行っても偏りを是正できない場合があり，このとき処理が停止しない可能性がある。そこで，平準化処理の確実な停止を実現する。

4.2. 自律リバランスアルゴリズム

定期的に各要素 NAS のディスク残量を監視し，偏りが生じた場合には残量の最も少ない NAS から，最も多い NAS にファイルグループを移動し偏りを是正することを，自律リバランス機能の骨格とする。これをアルゴリズム化すると図 3 のようになる。これを基本アルゴリズムと呼ぶ。

```
残量監視処理
01:while(1) {
02:  要素 NAS1～NASn のディスク残量 A1～An を計測;
03:  if ( max(A1,…,An) - min(A1,…,An) > Td )
04:    if ( ! 平準化処理起動中 )
05:      平準化処理起動;
06:  sleep(t);
07:}

平準化処理
08:while( max(A1,…,An) - min(A1,…,An) > Td ) {
09:  ディスク残量が最小の要素 NAS を移動元 NAS Ns とする
10:  ディスク残量が最大の要素 NAS を移動先 NAS Nd とする
11:  Ns に格納されているファイルグループを何らかの方法で
       一つ選択し Nd に移動
12:  要素 NAS1～NASn のディスク残量 A1～An を計測;
13:}
```

図 3: 基本アルゴリズム

基本アルゴリズムは残量監視処理と平準化処理から構成される。残量監視処理は，各要素 NAS のディスク残量を一定間隔 t 毎に計測する(02 行目)。計測したディスク残量の最大値と最小値の差分が差分閾値 Td より大きくなった場合に(03 行目)ディスク残量の偏りが発生したと考え，平準化処理が起動されていなければ(04 行目)，平準化処理を起動する (05 行目)。平準化処理は，ディスク残量の最も少ない要素 NAS に格納されているファイルグループの一つを，何らかの方法で選択し，それをディスク残量の最も多い要素 NAS に移動し(09～11 行目)，再び各要素 NAS のディスク残量を計測し(12 行目)，偏りが是正されていれば平準化処理を終了し，そうでなければ処理を続ける(08 行目)。

この基本アルゴリズムをベースとしながら，4.1節の二つの要件を満足するアルゴリズムを検討した。検討にあたっては，アルゴリズムとしての最適性や完璧性を追求するのではなく，実用性を重視した。

(1) 移動ファイルグループの選択方法

まず平準化処理の 11 行目の移動するファイルグループの選択方法について考える。最適な方法は，移動によって残量の偏りを是正するのに最も適したサイズのフィルグループを選択することである。これを実現するには，ファイルグループのサイズを求める必要がある。X-NAS の構造上，ファイルグループのサイズを求めるには仮想ファイルツリーを辿って全てのファイルを探索する必要がありコストがかかる。そこで，次善の策として，移動元 NAS に格納されているファイルグループの中からランダムに一つを選択するものとする。

(2) 要件(a)への対応

各要素 NAS のディスク残量の偏りを是正する目的は，ディスク残量問題を予防することにある。従って，各要素 NAS のディスクに多くの空きがあり，ディスク残量問題が起こりそう

もない状況のときは，平準化処理を行う必要はない。そこで，ディスク残量に関する開始閾値 Ts を設け，各要素 NAS のディスク残量がすべて Ts 以上の場合は，ディスク残量に偏りがあっても平準化処理を行わないようにする。

(3) 要件(b)への対応

3.1節で説明したようにファイルグループの中にはサイズが非常に大きなものもある。(1)で説明したように，ファイルグループはランダムに選択される。たまたま大きなファイルグループを選択しそれを移動すると，移動元 NAS のディスク残量は大幅に増えるが，移動先 NAS のディスク残量が大幅に減り，移動元 NAS と移動先 NAS の立場が逆転しただけで，ディスク残量の偏り具合は改善されない。ファイルグループの選択によっては，このような処理が連続的に発生し，大きなサイズのファイルグループが要素 NAS 間を行ったり来たりして振動し，平準化処理は終了しない可能性がある。これを防ぐには，大きなファイルグループの移動を禁止すればよい。しかし，(1)で説明したようにファイルグループのサイズを求めるにはコストがかかる。そこで，移動自身を禁止するのではなく，ファイルグループの移動後に移動元 NAS と移動先 NAS の立場が逆転したら平準化処理を強制的に停止するものとする。このように平準化処理を強制停止すると，ディスク残量の偏りは是正されない。しかし，偏りがあれば，その後の残量監視処理において再び平準化処理が起動されるので，いずれ偏りは是正されることになる。

図 4に上記(1)～(3)を導入した X-NAS の自律リバランスアルゴリズムを示す。

```
残量監視処理
01:while(1) {
02:  要素 NAS1～NASn のディスク残量 A1～An を計測;
03:  if ( (max(A1,…,An) - min(A1,…,An) > Td)
           && (min(A1,…,An) < Ts) )
04:    if ( ! 平準化処理起動中 )
05:      平準化処理起動;
06:  sleep(t);
07:}

平準化処理
08:while( max(A1,…,An) - min(A1,…,An) > Td ) {
09:  ディスク残量が最小の要素 NAS を移動元 NAS Ns とする
10:  ディスク残量が最大の要素 NAS を移動先 NAS Nd とする
11:  Ns に格納されているファイルグループをランダムに一つ
       選択し Nd に移動
12:  要素 NAS1～NASn のディスク残量 A1～An を計測;
13:  if ( Ns のディスク残量 > Nd のディスク残量 )
14:    exit;
15:}
```

図 4: 自律リバランスアルゴリズム

(4) パラメータに関する考察

開始閾値 Ts は，ディスク残量問題が発生する可能性のあるディスク残量を見極めるための閾値である。上述のように平準化処理の起動インターバルは，最短で残量監視処理の監視インターバル t [s]である。このインターバルの間は平準化処理は起動しない。そこで，インターバル t の間にディスク残量の最も少ない要素 NAS にファイル書き込みが集中しても，その要素 NAS のディスクが溢れてしまわないようにする必要がある。X-NAS の書き込みスループットを T [B/s]とすると，最大 tT[B]のファイルが書き込まれる可能性がある。そこで，開始閾値 Ts はこの値以上に設定する必要がある。マージンを考慮し Ts は 2tT～3tT 程度に設定する。

差分閾値 Td は，ディスク残量の偏りをどの程度許容するかによる。Td を小さくすると，ディスク残量の偏りの是正効果は高くなるが，平準化処理が頻繁に起動されクライアントからのファイルアクセス処理への影響が大きくなる。一方，Td を大きくとると，平準化処理はあまり起動されないのでクライアント処理への影響は小さくなるが，ディスク残量の偏り是正の効果は低くなる。これらを考慮して Td を決定する必要がある。なお，Td をファイルグループ一個のサイズより小さく設定しても効果はないため，Td の最小値はファイルグループのサイズとなる。Td は是正の効果に関係あるため，X-NAS の全容量に対して，どの程度書き込みできれば良いかという視点も考慮する必要がある。たとえば，UNIX のファイルシステムは全容量の 95%以上のユーザ書き込みを禁止している。これと同様に，容量の 5%程度は使用できなくても良いとすると，容量 Ci である要素 NASi が N 台ある X-NAS では，Td = $0.05\Sigma Ci\ /(N-1)$ となる。

5. X-NAS プロトタイプシステムによる評価

5.1. 実験環境と実験概要

4章で述べた自律リバランス機能を備えた X-NAS プロトタイプシステムを実現した。親 NAS，子 NAS ともハードウェアには PC を使用し，OS は RedHat7.2 Linux を使用した。親 NAS には Xnfsd と Manager を搭載した。子 NAS には新しいソフトウェアは導入していない。このプロトタイプシステムは，自律リバランス機能に関連する各種パラメータの値を起動時オプションによって変更できるようにしてある。このプロトタイプシステムを用いて自律リバランス機能の評価実験を行った。実験に使用した X-NAS プロトタイプシステムの構成と設定を表 2 に示す。

表 2: X-NAS プロトタイプシステムの構成と設定

X-NAS の構成	親 NAS + 子 NAS×3
データディスクの容量	親 NAS，子 NAS 各 1GB (ユーザ使用可能容量 960MB)
全体容量	4GB(ユーザ使用可能容量 3.8GB)
ファイルグループ数	500
残量監視間隔 (t)	30s

評価実験には表 1 のファイル群 1 を用いた。このファイル群の総ディスク使用量は 615MB である。このファイル群 1 とまったく同じファイルをクライアントのホームディレクトリに持つ 6 人のユーザ(A～F)がいると仮定する。バックアップ作業を想定し，各ユーザのホームディレクトリを A～F の順に X-NAS にコピーしていく。初期状態において X-NAS は空であるものする。A～F 全体の総ディスク使用量は 3690MB であるから，要素 NAS 間のディスク残量の偏りがなければ，すべてのユーザのホームディレクトリを書き込める。

上記のコピー処理を順次実施し，各ユーザのコピーが終了する度に各要素 NAS のディスク使用量を計測し，自律リバランス機能に関する各種情報を取得した。

5.2. パラメータの決定

4.2節(4)の通り，開始閾値は 2tT～3tT とする。残量監視間隔 t は 30s，プロトタイプシステムの書き込みスループットTは約 5MB/s であるから，Ts を 300MB に設定する。

一方，プロトタイプシステムのファイルグループ数は 500 であるから，全容量を使い切ったときのファイルグループの平均サイズは約 8MB となる。従って Td は 8MB 以上でなければならない。また，本評価実験では，用意した 3690MB のデータがすべて格納できるようにしたい。ユーザ権限で使用可能な容量 3891MB に対して 3690MB は 94.8%にあたる。そこで，4.2節(4)の考え方を応用すると Td = 66MB と求まる。8～66MB の間の値として Td=30MB に設定する。

5.3. 実験結果

実験は自律リバランス機能なしの場合と自律リバランス機能あり場合について行った。自律リバランス機能ありの場合は5.2節で決定したように開始閾値 Ts=300MB，差分閾値 Td=30MB とし，比較のため，ディスクが空の時から平準化処理を実行することを意味する Ts=1000MB の場合も実験した。各ユーザのホームディレクトリをプロトタイプシステムにコピーするたびに，各要素 NAS のディスク使用量を計測した。結果を図 5～図 7 に示す。各図の横軸はユーザのホームディレクトリのコピー回数を示す。1 ならユーザ A のホームディレクトリのコピーをし終えた状態を，2 なら A に加えてユーザ B のホームディレクトリのコピーをし終えた状態を示す。縦軸は各要素 NAS のディスク使用量を示す。

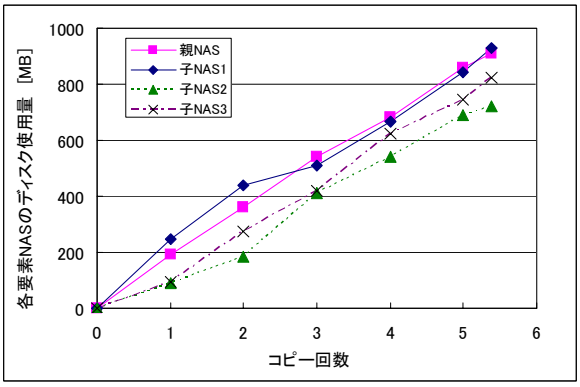

図 5: 自律リバランス機能なし

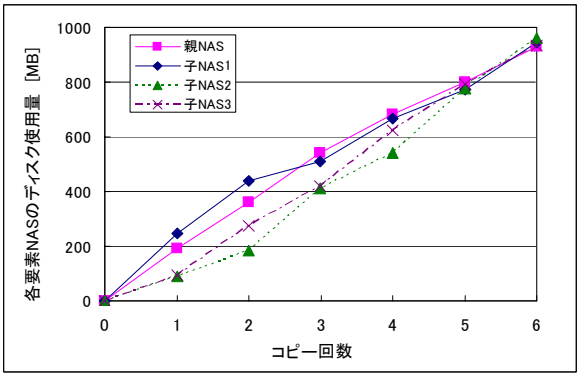

図 6: 自律リバランス機能あり(Ts=300MB)

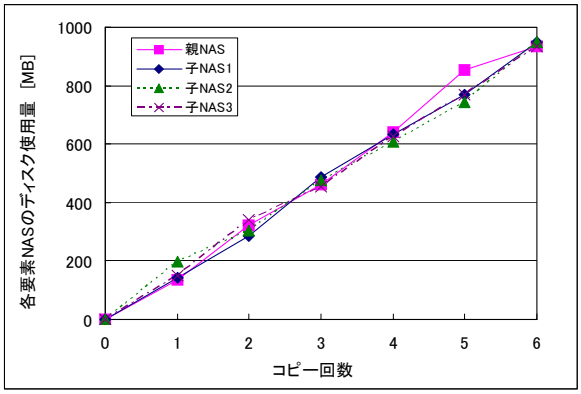

図 7: 自律リバランス機能あり(Ts=1000MB)

図 5 の自律リバランス機能なしでは，1～5 回のすべてにおいて，各要素 NAS のディスク残量に偏りがあることが分かる。6 回目のコピーの途中で子 NAS1 のディスクが一杯になりコピー処理は最後まで完了しなかった。これがディスク残量問題である。実験終了時の総ディスク使用量は 3383MB であり，これは使用可能容量の約 87%にしかすぎない。

図 6 の自律リバランス機能あり(Ts=300MB)では，4 回目のコピーまでは自律リバランス機能なしと同じであり，ディスク残量に偏りがある。この理由は各要素 NAS のディスク残量が 300MB より多くあるため，平準化処理が起動されないからである。一方，5 回目と 6 回目は要素 NAS のディスク残量が 300MB 以下となるため，平準化処理が起動され，偏りが是正されている。6 回目のコピーもエラーになることなく完了し，ディスク残量問題は発生しなかった。

図 7 の自律リバランス機能あり(Ts=1000MB)の場合は，1 回目から偏りが是正されている。5 回目のコピーの際に 80MB 程度の偏りが発生しているが，これは平準化処理の際に，たまたま大きいサイズのファイルグループを移動してしまったために発生したものである。移動したファイルグループのサイズは 52MB であった。このファイルグループの移動直後に移動元と移動先のディスク残量の逆転現象が発生したが，振動することなく平準化処理は停止した。6 回目のコピーも途中でエラーになることなく完了し，ディスク残量問題は発生しなかった。

実験中に平準化処理の起動回数とファイルの総移動量を計測した。結果を表 3 に示す。表中の 1～6 はコピー回数を示す。Ts=1000 では，1～6 のすべてにおいて平準化処理が実行される。総ファイル移動量は 763MB にもなり，これだけのファイルの移動によって，クライアントからのファイルアクセス処理にも大きな影響を及ぼしているものと考えられる。

表 3: 平準化処理の起動回数とファイルの総移動量

実験条件	平準化処理起動回数						総移動量 [MB]
	1	2	3	4	5	6	
Ts=300	0	0	0	0	33	15	203
Ts=1000	108	79	32	23	25	19	763

一方，Ts=300 の場合は，1～4 回は平準化処理は実行されず，従ってクライアントからのファイルアクセス処理に影響を及ぼさない。なお，図 7 の Ts=1000 の場合の 1 回目と 2 回目のコピーにおける平準化処理の起動回数が他より多いが，その理由は以下の通りである。コピー回数が少ないときは多いときよりファイルグループのサイズが小さい。一方，偏りを解消するために移動しなければならないファイルグループ群のサイズは回数によらずほぼ一定である。従って，サイズが小さい時はより多くのファイルグループを移動する必要がある。以下に結果をまとめる。

- 一つのファイル群のみを用いた実験ではあるが，仮想一元化 NAS において各要素 NAS のディスク残量に偏りが生じ，ディスク残量問題が発生することを実証した。
- 自律リバランス機能によってディスク残量の偏りを是正すれば，ディスク残量問題を予防できることを検証した。
- 自律リバランスアルゴリズムに開始閾値 Ts を導入することで，不要な平準化処理を抑止し，クライアントからのファイルアクセス処理に与える影響を低く抑えられることを検証した。これにより，自律リバランス機能は要件(a)を満足しているといえる。
- 平準化処理はたまたまサイズの大きなファイルグループを移動してしまった場合でも振動せず，確実に停止することを確認した。自律リバランス機能は要件(b)を満足しているといえる。

6. 関連研究

複数のストレージを仮想一元化するアプローチとして分散ファイルシステムである Zebra，DiFFS，Slice などがある。Zebra[2]は，ファイルをストライピングしてストレージサーバ群に分散配置する。DiFFS[3]はファイル全体を分割せずに特定のストレージサーバに格納する。Slice[4]は小さいファイルはファイル単位でストレージサーバに格納し，大きいファイルはブロック単位でストライピングする。一方，ネットワークルーティング技術によって複数ストレージを仮想化する NAS Switch[5][6]もある。NAS Switch はストレージ装置の前に特別なスイッチを置き，このスイッチが特定のディレクトリ以下のすべてのファイル群を特定のストレージ装置に分散配置する。Zebra や Slice などのストライピングアプローチは，各要素 NAS 間のディスク使用量の偏りがなく，従ってディスク残量問題は発生しない。しかし，容量拡張等の構成変更が難しい。一方，ファイル単位で格納する DiFFS や NAS Switch は，構成変更は容易だが，ディスク残量問題が発生する。DiFFS や NAS Switch はファイル移動の枠組みについては検討しているものの，システムが自律制御によってディスク残量問題を未然に予防する機能は実現していない。

複数のストレージから構成される自律ディスク[7]では，各ストレージが自律分散的にデータの再配置を行いデータの偏りを是正する研究が行われている[8]。自律ディスクにおける自律リバランス機能は各ディスクが分散処理するが，X-NAS は親 NAS が集中処理する点で異なる。

7. おわりに

筆者らは，安価で容量スケーラブルな NAS を提供するため，複数のエントリ NAS から構成され単一ファイルシステムビューを提供する仮想一元化 NAS システム X-NAS を提案している。仮想一元化 NAS システムにはファイルのサイズのばらつきに起因して，ディスク残量問題が発生する。本稿では，この問題を解決するために，自律リバランス機能を提案した。また，プロトタイプシステムによる評価実験を行い，提案機能がディスク残量問題の発生を予防できることを検証した。

自律リバランス機能の今後の課題は，ディスク残量問題を予防しつつ，ファイルアクセス処理の負荷均衡化を実現することである。また，X-NAS システム全体としての課題は性能と信頼性の向上である。

謝辞

X-NAS の着想時から X-NAS について議論して頂いているスタンフォード大学情報科学科の J. D. Ullman 先生に謝意を表します。

参考文献

[1] Y. Yasuda, S. Kawamoto, A. Ebata, J. Okitsu, T. Higuchi, “Concept and Evaluation of X-NAS: a Highly Scalable NAS System”, to appear in 20th IEEE Symposium on MASS Storage Systems, 2003.

[2] John H.Hartman, John K.Ousterhout, “The Zebra Striped Network File system”, ACM Transactions on Computer Systems, Vol.13, No.3, pp274-310, 1995.

[3] C. Karamanolis, L. Liu, M. Mahalingam, D. Muntz, Z. Zhang, “An Architecture for Scalable and Manageable File Services”, HP Laboratories Palo Alto, HPL-2001-173, 2001.

[4] D. C. Anderson, J. S. Chase, A. M. Vahdat, “Interposed Request Routing for Scalable Network Storage”, ACM Transactions on Computer systems, Vol.20, No.1, 2002.

[5] 山川 聡，石川 潤，菊地芳秀，“NAS スイッチ:NFS サーバの仮想化統合技術の開発”，信学技報 CPSY2002-36, pp13-18, 2002.

[6] 桂島 航，石川 潤，“ファイルサーバ仮想化アプライアンス NAS スイッチの提案”，情処研報，2002-DSM-26-6, 2002.

[7] H.Yokota, “Autonomous disks for advanced database applications”, Proc. International Symposium on Database Applications in Non-Traditional Environment (DANTE’99), pp.441-448, 1999.

[8] H. Yokota, Y. Kanemasa, J. Miyazaki, “Fat-Btree: An Update-Conscious Parallel Directory Structure”, Proc. 15th Int’l Conf. on Data Engineering, pp.448-457, 1999.