ファイル更新回数とアクセス頻度にもとづくディレクトリ圧縮の優先度付けによるストレージ使用量の削減

橋本 健[1]　井田 尚樹[1]　串田 高幸[1]

概要：課題は, ファイルサーバに格納されている使用頻度の低いディレクトリのバックアップにより, バックアップサーバのストレージをが圧迫されることである. 本稿の提案では, ファイルサーバに格納されているディレクトリの使用頻度を数値として監視し, それをもとに使用頻度の昇順に, ストレージ使用量がストレージ総容量の 70%に収まるまでディレクトリを圧縮する. 評価方法として, 使用頻度を数値の比較, ディレクトリの使用頻度を昇順に並べた際の全ディレクトリの 50%に当たる個数分のバックアップディレクトリを圧縮する前後のサイズを比較する. 実験で使用するディレクトリの個数, ディレクトリに格納されているファイルの内容は東京工科大学コンピュータサイエンス学部コンピュータサイエンス学科先進情報専攻 (以下, 本大学と呼ぶ.) の学生が本大学の授業で使用しているものを参考にした. 本大学はセメスター制であるため, 2021 年から 2023 年までの前期, 後期分の計 6 個のディレクトリを使用した. また, ファイルは本大学の授業で使用していたパワーポイントファイルを使用した. 使用頻度の数値を比較する実験の結果として平均使用頻度は, アクセス, 更新回数共に低頻度なディレクトリでは 10.0[回/日], アクセスは高頻度だが更新が低頻度なディレクトリでは 20.0[回/日], アクセス, 更新共に高頻度なディレクトリでは 30.0[回/日] となった. ディレクトリの使用頻度を昇順に並べた際の全ディレクトリの 50%に当たる個数分, バックアップディレクトリを圧縮する前後のサイズを比較する実験の結果として, ディレクトリのサイズは圧縮前のディレクトリのサイズは 307.8MB, 圧縮後のディレクトリのサイズは 284.5MB と約 7.6%減少した.

1. はじめに

背景

バックアップとは, 削除または破損したデータを回復することのみを目的として作成および保持される運用データのコピーである [1]. データを損失から保護することは多くの分野, 特にビジネスにおいて重要である [2]. データの損失の原因は, ユーザーの誤操作, コンピュータの故障, ウイルスの感染である [3].

情報社会の進展によりデータは爆発的に増加している [4]. International Data Corporation(IDC) の報告によると, デジタルデータの量は 2006 年から 2010 年にかけて 50%の割合で増加し, 5 年後には 10 倍のストレージスペースが占有されることになる [5]. そこで, ストレージスペースを確保する方法の 1 つにデータ圧縮がある. この方法は元のシステムよりも少ない容量内で必要なバックアップを維持するのに役立つ [6]. データ圧縮はファイル圧縮, 画像圧縮, 動画圧縮, 音声圧縮の 4 つの種類に分類される[*1].

ファイルサーバはローカル通信ネットワークを介して他の多数のコンピュータに接続されているコンピュータに提供されるユーティリティである [7]. これにより, OS のファイル共有機能を用いたネットワーク上でのファイル交換ができる[*2]. しかし, ファイルサーバに格納されている全ファイルの内, その多くは使用頻度の低いファイルが占めている. 日本電気株式会社が企業, 団体 50 社に実施した調査によると, ファイルサーバに格納された全ファイルのうち, 1 年以上更新がないファイルは 76%にも及ぶという数値が出ている[*3]. ファイルサーバのデータはバックアップサーバに定期的にバックアップされる. そのため, バックアップサーバ内にもファイルサーバ同様に使用頻度の低いファイルが格納されている.

バックアップサーバは利用不能な時間を短縮し, 可用性を高めるための効果的なソリューションである. 主にハードウェア障害や災害が発生した場合にサービスを復旧するために使用される [8]. しかし, 企業によって毎日生成されるデータにより, バックアップシステムの規模が異常に増

[1] 東京工科大学コンピュータサイエンス学部
〒 192–0982 東京都八王子市片倉町 1404-1

[*1] https://info.securesamba.com/media/13138/

[*2] https://www.ntt.com/business/lp/file-server.html

[*3] https://jpn.nec.com/co-creation/showroom/images/nec_nias_20210312.pdf

大している [9]．これにより，データ保護を効率的な実行が困難になっている．

課題

課題は，ファイルサーバに格納されている使用頻度の低いディレクトリのバックアップディレクトリにより，バックアップサーバのストレージが圧迫されることである．課題の概要を図 1 に示す．使用頻度の低いディレクトリはアクセス，更新が低頻度のディレクトリを指している．その他のディレクトリはアクセス，更新が高頻度または，低頻度と高頻度の中間に位置する頻度のディレクトリを指している．ファイルサーバに格納されているディレクトリをバックアップサーバにバックアップした際，使用頻度の低いディレクトリを含む全ディレクトリがバックアップされる．そのため，使用頻度の低いディレクトリのバックアップディレクトリがバックアップストレージを余計に占有してしまう．これにより，バックアップサーバのストレージの空き容量が枯渇し，更なるバックアップができない．

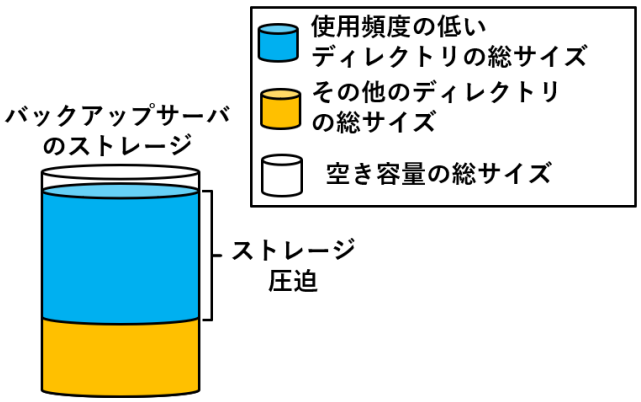

図 1　課題の概要

各章の概要

第 2 章では，関連研究について述べる．第 3 章では，課題を解決するための提案方法述について述べる．第 4 章では，提案方式を実現させるための実装するソフトウェアの実装について述べる．第 5 章では，提案の評価の実験環境，実験結果について述べる．第 6 章では，提案方式についての議論を述べる．第 7 章では，本稿のまとめについて述べる．

2.　関連研究

多次元データに基づく新しいホット，コールドデータ識別メカニズムを提案している研究がある [10]．ホットおよびコールドデータ識別メカニズムでは，データアクセス時間，データアクセス頻度，データ間の依存関係にもとづき，ホットデータとコールドデータを定量化することでホットデータとコールドデータを識別している．提案手法では，ニュートンの冷却の法則時間の経過によるデータの冷却，周囲のデータへの熱伝導によるホットデータの冷却を考慮することで，これまでに高頻度でアクセスされてきたデータの冷却問題を効果的に解決している．しかし，データの更新頻度を提案に使用しておらず，アクセスと更新を識別できないため，本稿の手法に適さない．

リクエストサイズに基づく予測スキームと共に，受信書き込みトランザクションのデータ圧縮率にもとづくホット，コールド識別の提案をしている研究がある [11]．ホストシステムから転送された書き込み要求データをバッファメモリに格納し，データの先頭部分を圧縮している．それにより，バッファに対するヒット率が増加し，ホット，コールドの識別性能が増加している．しかし，実験で使用されたデータについて明記されていない．本稿の実験はデータの内容に依存するため，この関連研究は使用できない．

ホット，コールドデータ分類とデータ移行によるデータ温度予測タスクのための自動クラウドストレージ階層化システムを提案している研究がある [12]．この研究では，データを予測，分類し，頻繁に使用されるファイルを自動的，動的，正確にホットストレージに割り当て，使用頻度の低いデータはコールドストレージメディア (テープや光学ドライブ) に移動している．しかし，2 ヶ月のデータセットでモデル構築をしており，2 ヶ月の間のデータでしか他の月のデータを判断できないため，本稿の手法に適さない．

3.　提案

提案方式

本稿の提案では，ファイルサーバに格納されているディレクトリの使用頻度を監視し，それをもとにバックアップサーバに格納されているディレクトリをストレージ使用量がストレージ総容量の 70%に収まるまで圧縮する．提案の流れを図 2 に示す．

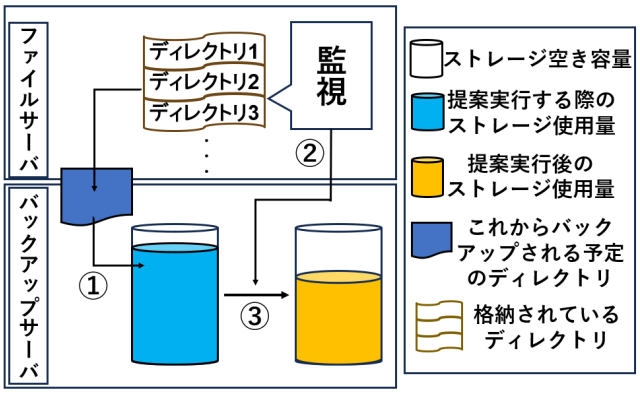

図 2　提案の流れ

提案の流れを以下に示す．

① これからバックアップサーバに転送される予定の，ファイルサーバに格納されているディレクトリのバック

アップディレクトリのサイズと既にバックアップサーバに格納されているバックアップディレクトリのサイズの合計がストレージ総容量の 70%を超過しているか確認する．

② ファイルサーバに格納されているディレクトリを監視することで, 各ディレクトリのアクセス回数, 更新回数, ディレクトリの作成されてからの経過日数を取得する．そして, これらの要素をもとに使用頻度の数値を算出する．計算式 (1) に使用頻度の数値を算出する際に使用する式を示す．

$$F = \frac{A + U}{C} \tag{1}$$

F はディレクトリの使用頻度, A はディレクトリへのアクセス回数, U はディレクトリの更新回数, C はディレクトリ作成日からの経過日数を示す．計算式 (1) ではまず, ディレクトリの作成日から今日までの A と C を加算し, ディレクトリの使用回数を求める．そして, 使用回数を C で除算することにより, ディレクトリの 1 日当たりの使用回数の平均を求めている．この平均をディレクトリの使用頻度とする．

③ ①で合計したディレクトリのサイズがバックアップサーバのストレージ総容量の 70%に収まるまで, 使用頻度の低い順に圧縮し続ける[*4]．この理由により, 圧縮するディレクトリの個数は変動する．

ユースケース・シナリオ

ユースケースシナリオはインターネットサービスを提供する企業のバックアップサーバ管理者を想定する[*5]．ユースケースシナリオを図 3 に示す．

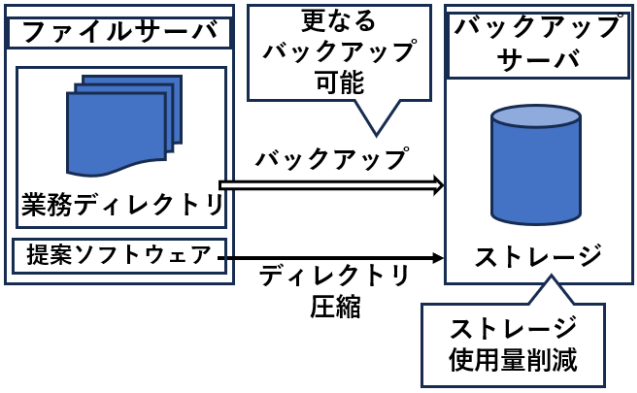

図 3 ユースケースシナリオ

ユーザは業務時間に, ファイルサーバに格納されているディレクトリにアクセスや更新をする．ファイルサーバのディレクトリは毎日, 業務終了時間から業務開始時間までにバックアップサーバへバックアップされる．そのため, 使用頻度の低いディレクトリもバックアップされ, バックアップサーバのストレージに使用頻度の低いディレクトリのバックアップディレクトリが格納されてしまう．これにより, ストレージ使用量が余計に増加し, ストレージが圧迫されてしまう．そこで, 本稿の提案により, ファイルサーバに格納されている, 使用頻度の低いディレクトリを確認できる．そして, 確認した使用頻度の低いディレクトリのバックアップディレクトリを圧縮することでバックアップストレージ使用量を削減できる．これにより, 提案前よりバックアップできる回数を増加できる．

4. 実装

実装の流れを図 4 に示す．

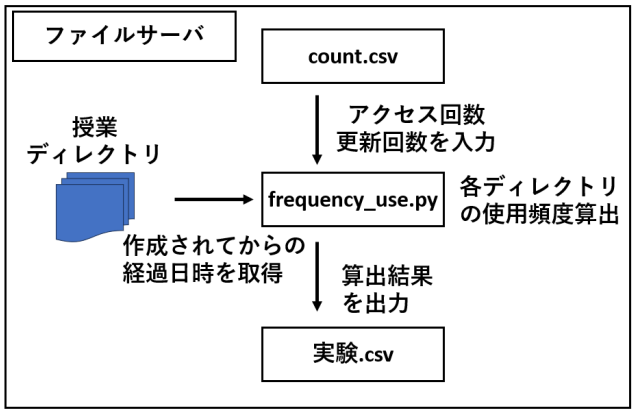

図 4 実装の流れ

frequency_use.py

本稿では提案手法をもとにしたソフトウェアを作成した．Frequency_use.py では, ファイルサーバに格納されている各ディレクトリのディレクトリ名, アクセス回数, 更新回数が記入された count.csv ファイルを読み込む．ディレクトリが作成されてからの経過日数を取得する．そして, ディレクトリ名を除いたこれらの 3 つの要素から圧縮の優先度付けのために, ディレクトリの使用頻度を計算式 (1) を使用し数値を算出する．プログラム終了時, 各ディレクトリ名と算出した結果を実験.csv ファイルに書き込む．

5. 評価実験

本稿では 2 種の評価実験を行った．ファイルサーバに格納されているディレクトリの使用頻度を比較する評価実験を実験 1 とする．ファイルサーバに格納されているディレクトリの使用頻度を昇順に並べた際の全ディレクトリの 50%に当たる個数分, バックアップディレクトリを圧縮する前後のサイズを比較する実験を実験 2 とする．本稿では, 実装でバックアップサーバに転送される予定のバックアッ

[*4] https://blog.idcf.jp/entry/vSan_storage
[*5] https://ent.iij.ad.jp/articles/1579/

プディレクトリのサイズを取得できない. そのため, バックアップサーバに転送される予定のディレクトリのサイズと既にバックアップサーバに格納されている全ディレクトリのサイズを合計したサイズをストレージの総容量の 70%に抑えるまで圧縮する際, 圧縮するディレクトリの個数を把握することができない. そこで実験 2 では, ファイルサーバに格納されているディレクトリの使用頻度を昇順に並べた際の全ディレクトリの 50%に当たる個数分バックアップディレクトリを圧縮する.

実験環境

実験環境を図 5 に示す.

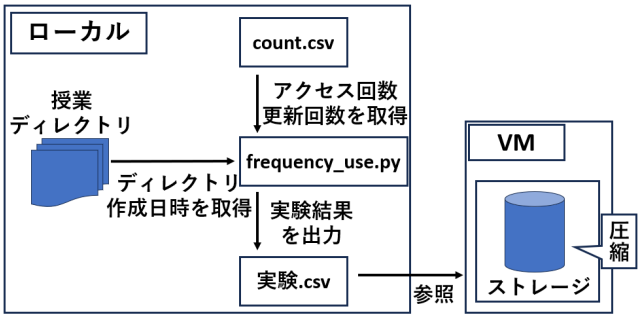

図 5 実験環境

実験環境には, ファイルサーバに windows のローカル環境, バックアップサーバに VM(仮想マシン) を用いる. 以下に VM の構成要素を示す.

- **VM**

OS Ubuntu 20.04
vCPU 2 コア
RAM 8GB
HDD 40GB

本稿の実験では事前準備として, 本大学の学生の環境に則したディレクトリ, ディレクトリに格納するファイルを用意する.

ディレクトリとして, 本大学はセメスター制であるため, 2021 年から 2023 年までの前期, 後期分計 6 個のディレクトリ (2021 前期, 2021 後期, 2022 前期, 2022 後期, 2023 前期, 2023 後期) を使用する.

本稿のユースケースとして参考にした企業, 株式会社インターネットイニシアティブで利用されているファイルサーバに格納されているファイルの内, ファイルの合計サイズが最も大きいファイル形式はパワーポイント形式であった[*6]. そのため, 本稿の実験で使用するディレクトリにパワーポイントファイルを格納した. パワーポイントファイルには本大学のこれまでの授業で使用したパワーポイントファイルを使用する. 利用する各パワーポイントファイルの配置場所は本大学の学生の環境に則している. 本稿の実験環境で使用する各ディレクトリにはそれぞれ, 2021 前期ディレクトリには 2 個, 2021 後期ディレクトリには 1 個, 2022 前期ディレクトリには 36 個, 2022 後期ディレクトリには 10 個, 2023 前期ディレクトリには 4 個, 2023 後期ディレクトリには 22 個, のパワーポイントファイルが格納されている.

実験で使用する計 6 個のディレクトリを, アクセス, 更新共に低頻度なディレクトリをケース 1, アクセスは高頻度だが更新は低頻度なディレクトリをケース 2, アクセス, 更新共に高頻度なディレクトリをケース 3 とする. ディレクトリは更新される際, 同時にアクセスされる. つまり, アクセスが低頻度だが更新は高頻度のケースは実在しないため, このケースは除外する. 2021 前期, 2021 後期のディレクトリをケース 1, 2022 前期, 2022 後期のディレクトリをケース 2, 2023 前期, 2023 後期のディレクトリをケース 3 とする.

また, ディレクトリの 1 日当たりのアクセス, 更新回数が記入された count.csv ファイル, 実行結果を出力する実験.csv ファイルをそれぞれ用意する.

本稿の実装ではローカル環境に格納されているディレクトリのアクセス, 更新を監視できない. そのため, リアルタイムのアクセス, 更新回数を取得できない. そこで, 本稿では各ディレクトリの 1 日当たりのアクセス, 更新回数を定数で定義する. ディレクトリのアクセス, 更新が低頻度, 高頻度の場合の 1 日当たりのアクセス回数, 更新回数をそれぞれ, アクセスが低頻度の場合のアクセス回数を 5〜10[回/日], アクセスが高頻度の場合のアクセス回数を 15〜20[回/日], 更新が低頻度の場合の更新回数を 0〜5[回/日], 更新が高頻度の場合の更新回数を 10〜15[回/日] と定義する. 今回 count.csv に記入する, 各ディレクトリの 1 日当たりのアクセス, 更新回数をそれぞれ以下のようにした.

- ケース 1(アクセス, 更新が共に低頻度なディレクトリ)
  - 2021 前期のディレクトリの 1 日当たりのアクセス回数を 5[回/日], 更新回数を 0[回/日] とする.
  - 2021 後期のディレクトリの 1 日当たりのアクセス回数を 10[回/日], 更新回数を 5[回/日] とする.
- ケース 2(アクセスは高頻度だが, 更新が低頻度なディレクトリ)
  - 2022 前期のディレクトリの 1 日当たりのアクセス回数を 15[回/日], 更新回数を 0[回/日] とする.
  - 2022 後期のディレクトリの 1 日当たりのアクセス回数を 20[回/日], 更新回数を 5[回/日] とする.
- ケース 3(アクセス, 更新が共に高頻度なディレクトリ)
  - 2023 前期のディレクトリの 1 日当たりのアクセス回

[*6] https://ent.iij.ad.jp/articles/1579/

数を 15[回/日], 更新回数を 10[回/日] とする.
– 2023 後期のディレクトリの 1 日当たりのアクセス回数を 20[回/日], 更新回数を 15[回/日] とする.

frequency_use.py の実行終了時の各ディレクトリ名と使用頻度を数値化した実験結果を実験.csv に出力する.

実験結果と分析

図 6 に実験 1 の各ケースの使用頻度の平均を示す.

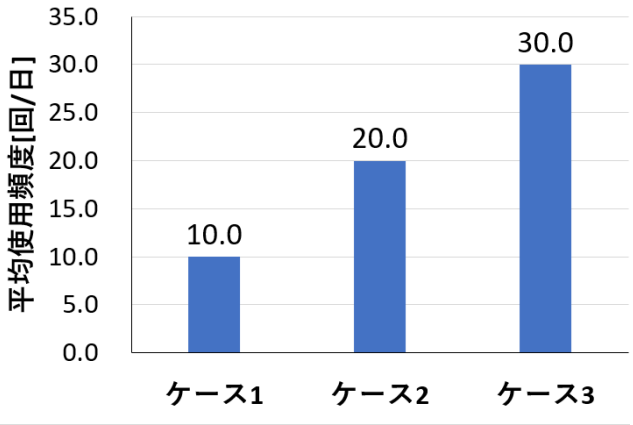

図 6 各ケースの使用頻度の平均

縦軸は各ケースの使用頻度の平均を表している. ケース 1 の使用頻度の平均は 10.0[回/日], ケース 2 の使用頻度の平均は 20.0[回/日], ケース 3 の使用頻度の平均は 30.0[回/日] となった. これは, アクセス, 更新回数が増加することにより, 計算式 (1) の分子の重みが増加したからである.

図 7 に, ディレクトリの使用頻度を昇順に並べた際の全ディレクトリの 50%に当たる個数分のバックアップディレクトリを圧縮する実験 2 の, 圧縮前後のディレクトリのサイズを示す. 縦軸はディレクトリのサイズを表している.

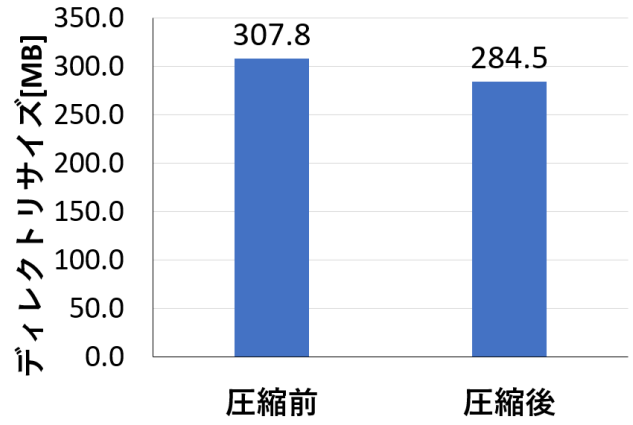

図 7 圧縮前後のディレクトリのサイズ

圧縮前のディレクトリのサイズは 307.8MB であったが圧縮後のディレクトリのサイズは 284.5MB と約 7.6%減少している. ディレクトリ圧縮でサイズの縮小幅が狭かった要因に, 実験で利用したパワーポイントファイルのスライドに多数の画像が含まれていたことが挙げられる.

6. 議論

本稿の評価実験では各ディレクトリの 1 日当たりのアクセス, 更新回数を定数で定義した. しかしアクセス, 更新回数を定数で定義した場合, ファイルサーバを活用するユーザの行動による本来のアクセス, 更新回数や使用頻度の変動に対応できない. そこで, ファイルサーバに格納されているディレクトリのアクセス, 更新をリアルタイムに監視することで, 現実に則した本来の使用頻度を求められる. ディレクトリの監視の実装方法として, Watchdog モジュールの導入が挙げられる. Watchdog モジュールにより, 特定フォルダ内のファイル・フォルダの状態を監視し, 状態に合わせた処理を設定できる.

本稿の実験ではディレクトリに格納するファイルとして本大学の学生が授業で使用していたパワーポイントファイルを利用した. しかし, 本稿の実験 2 の結果から, ディレクトリの圧縮率は格納されているファイル形式, またはファイル内容に依存することが確認できた. そこで, 本稿の提案では圧縮とは異なるアプローチをとる必要がある. 異なるアプローチとして, 新たなストレージへのアーカイブ, 既存のバックアップサーバのストレージ容量の拡張が挙げられる.

計算式 (1) では, ディレクトリが作成されてから現在までのアクセス回数, 更新回数を加算した後, ディレクトリが作成されてからの経過日数で除算することで, 1 日当たりのディレクトリの使用回数の平均を算出している. この場合, 定期的に使用されるディレクトリ, 過去には頻繁に使用されていたが直近の使用頻度は低いディレクトリ, 過去では使用頻度が低かったが直近の使用頻度が高いディレクトリ, の 3 つのケースに計算式 (1) を使用して数値を算出した際, 算出した 3 つの数値が同値や使用頻度が逆転する場合がある. この場合, 重要視されるべきである直近の使用頻度の高いディレクトリ場合が優先的に圧縮されてしまう. そこで, 使用回数が直近のものほど重要度を増加させることで, より正確な使用頻度を求められる. また, ディレクトリのアクセス回数と更新回数の重みが等しいため, アクセス回数と更新回数の重要度が等しいユースケースにしか本稿の提案を導入できない. そこで, ユースケースによりアクセス回数及び更新回数の重みを調整することにより, ユースケースにとって重要度の高いディレクトリをより正確に求められる.

7. おわりに

課題はファイルサーバに格納されている使用頻度の低いディレクトリのバックアップディレクトリにより, バック

アップサーバのストレージをが圧迫されることである. 提案方式はファイル更新回数とアクセス頻度にもとづくディレクトリ圧縮の優先度付けによるストレージ使用量の削減である. 評価実験では, ファイルサーバに格納されているディレクトリの使用頻度を比較する実験, ファイルサーバに格納されているディレクトリの使用頻度を昇順に並べた際の全ディレクトリの 50%に当たる個数分のバックアップディレクトリを圧縮する前後のサイズを比較する実験を行った. 使用頻度の数値を比較する実験の結果として, ディレクトリのアクセス, 更新共に低頻度なケース 1 の使用頻度は 10.0[回/日], ディレクトリのアクセスは高頻度だが更新が低頻度なケース 2 の使用頻度は 20.0[回/日], ディレクトリのアクセス, 更新共に高頻度なケース 3 の使用頻度は 30.0[回/日] となった. 全ディレクトリの 50%に当たる個数分のバックアップディレクトリを圧縮する前後のサイズを比較する実験では, 圧縮前のディレクトリのサイズが 307.8MB, 圧縮前のディレクトリのサイズが 284.5MB と約 7.6%減少した.

謝辞 本テクニカルレポートの執筆にあたりご協力いただいた東京工科大学コンピュータサイエンス学部コンピュータサイエンス学科先進情報専攻の三上智徳さんに御礼申し上げます.

参考文献

[1] Akbar, R., Husain, M. S. and Suaib, M.: Comparative study of various backup and monitoring techniques, *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, pp. 1530–1537 (online), DOI: 10.1109/ICGCIoT.2015.7380710 (2015).

[2] Qin, Y., Hoffmann, B., Wang, Y. and Lilja, D. J.: Exploring A Forecasting Structure for the Capacity Usage in Backup Storage Systems, *2019 IEEE 10th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, pp. 0126–0134 (online), DOI: 10.1109/UEMCON47517.2019.8992955 (2019).

[3] Ishida, H., Uchiya, T., Takumi, I. and Kinoshita, T.: Development of Distributed Backup System among Individuals by Introducing Deduplication, *2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*, pp. 327–331 (online), DOI: 10.1109/CISIS.2016.78 (2016).

[4] Sun, G.-Z., Dong, Y., Chen, D.-W. and Wei, J.: Data Backup and Recovery Based on Data De-Duplication, *2010 International Conference on Artificial Intelligence and Computational Intelligence*, Vol. 2, pp. 379–382 (online), DOI: 10.1109/AICI.2010.200 (2010).

[5] Du, J., Yu, H. and Zheng, W.: MassStore: A low bandwidth, high De-duplication efficiency network backup system, *2012 International Conference on Systems and Informatics (ICSAI2012)*, pp. 886–890 (online), DOI: 10.1109/ICSAI.2012.6223150 (2012).

[6] Jose, J. and V, M. K. K.: Solid compression strategy for BTRFS snapshot, *2015 National Conference on Recent Advances in Electronics Computer Engineering (RAECE)*, pp. 160–164 (online), DOI: 10.1109/RAECE.2015.7510246 (2015).

[7] Birrell, A. and Needham, R.: A Universal File Server, *IEEE Transactions on Software Engineering*, Vol. SE-6, No. 5, pp. 450–453 (online), DOI: 10.1109/TSE.1980.230493 (1980).

[8] Hu, B., Sudo, Y., Hato, K., Murata, Y. and Murayama, J.: Cost reduction evaluation of sharing backup servers in inter-cloud, *2013 19th Asia-Pacific Conference on Communications (APCC)*, pp. 256–261 (online), DOI: 10.1109/APCC.2013.6765952 (2013).

[9] Qin, Y., Hoffmann, B. and Lilja, D. J.: HyperProtect: Enhancing the Performance of a Dynamic Backup System Using Intelligent Scheduling, *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, pp. 1–8 (online), DOI: 10.1109/PCCC.2018.8711182 (2018).

[10] Song, Y., Fan, S., Xu, J. and Liao, J.: A Novel Hot-cold Data Identification Mechanism Based on Multidimensional Data, *2022 5th International Conference on Data Science and Information Technology (DSIT)*, pp. 1–5 (online), DOI: 10.1109/DSIT55514.2022.9943890 (2022).

[11] Kim, K., Jung, S. and Song, Y. H.: Compression ratio based hot/cold data identification for flash memory, *2011 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 33–34 (online), DOI: 10.1109/ICCE.2011.5722616 (2011).

[12] Hsu, Y.-F., Irie, R., Murata, S. and Matsuoka, M.: A Novel Automated Cloud Storage Tiering System through Hot-Cold Data Classification, *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 492–499 (online), DOI: 10.1109/CLOUD.2018.00069 (2018).

