ファイルのアクセスと更新回数にもとづく重要度の決定によるリストア時間の短縮とストレージ容量の確保

遠藤 空[1]　西嶌 知良[1]　串田 高幸[1]

概要：人的ミスによりデータを損失してしまう企業がある．これにより，ダウンタイムが発生してしまう場合がある．課題は圧縮データにより，復旧させたいデータのリストア時間が遅れ，許容ダウンタイム内にデータ復旧が終了しないことである．本稿の提案手法ではファイルのアクセス回数と更新回数を取得し，これらを用いてフォルダに重要度をつけている．この重要度が低いものから圧縮していき，重要度が高いものは圧縮せずにバックアップを行う．これにより，災害発生時に重要度が高いデータのリストア時間を短縮する．本稿の提案を用いて，実験データの重要度を計測し，アクセス・更新頻度が高くなるにつれて，重要度も高くなるという結果が得られた．アクセス・更新回数がともに高頻度の重要度は 42，両方の回数が低頻度のファイルの重要度は 1 となった．この時のアクセス・更新回数の高頻度とは 15〜20 の回数，低頻度とは 0〜3 回の回数を意味する．実験に用いたファイルの種類は IIJ のファイルサーバの約半数を占めていた，PDF，pptx 形式の PowerPoint ファイルを取り扱った．これにより災害発生時，ユーザは行っているプロジェクトデータを非圧縮により，リストア時間を短縮することが出来る．

1. はじめに

背景

中小企業の約 76 %が深刻なデータ損失を経験している[*1]．データ損失がビジネスに与える影響は，業務が止まってしまうことの他に，復旧までに時間がかかることや，信用問題に関わることである[*2]．このような事態に備えるために，データのバックアップは必要不可欠である．

バックアップとは障害が発生した際にデータを復旧できるように，データやシステムを複製して保存しておくことである [1]．バックアップデータの増大により，それを解決するための圧縮技術が近年開発されている [2]．バックアップには「フルバックアップ」,「部分バックアップ」があり，一般的に使用されているバックアップ手法は，フルバックアップと部分バックアップを組み合わせる手法である [3]．また，バックアップデータの中で，大部分を占めているフルバックアップは圧縮されることがある[*3]．

また，近年のビッグデータの時代の到来により，データストレージのバックアップの重要性が高まっている [4]．

企業において上記のようにバックアップを取っていても，システム障害により，ダウンタイムが発生してしまう場合がある．この障害は大きく分けると主に「自然災害」,「ハードウェア障害」,「ヒューマンエラー」,「ソフトウェア障害」,「コンピュータウイルス」がある [5]．例えば，ファイルの書き換えミスによって起こるファイル損失が挙げられる．一例として，FAA による航空機の運航停止の事例がある[*4]．これはメンテナンス中に作業員があるファイルを別のファイルに置き換えたことが原因で発生した．上記の事例のようにシステムが停止すると、重大な経済的損失が発生したり、会社が廃業に追い込まれたりする可能性があるため，企業は数時間以上続くシステムダウンを許容することができない [6]．しかし，中堅・中小企業では約 50 % の企業しか 12 時間という許容ダウンタイム内にデータ復旧がすんでいない[*1]．

課題

図 1 をもとに，本稿における課題を説明する．本稿における課題はシステム障害が発生した際のバックアップサーバからリストアする際にかかる時間が長くなり，許容ダウンタイム内にデータ復旧が終わらないことである．この時，バックアップデータを圧縮するとリストア時に展開する必

[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://insights-jp.arcserve.com/data-loss-global-research#3
[*2] https://www.fleekdrive.com/blog/archive/security10/
[*3] https://x.gd/tQWXS
[*4] https://www.arnnet.com.au/article/704679/database-error-cyberattack-led-air-travel-stoppage-faa-says/
[*1] https://insights-jp.arcserve.com/data-loss-global-research#3

要があるためリストア時間が長くなる．しかし，全データを圧縮せずにバックアップしてしまうと，ストレージの空き容量が確保されずに容量不足になってしまう．

そのため，重要なデータは圧縮せずにバックアップし，重要でないデータは圧縮してバックアップしておく必要がある．

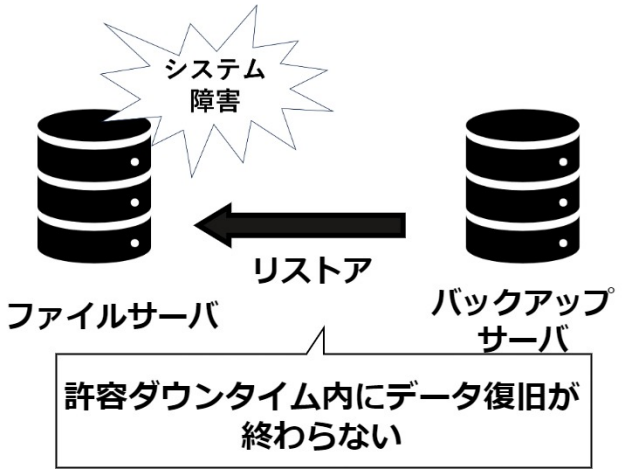

図 1　課題となるケース

基礎実験

データをリストアする上で，圧縮データが非圧縮データよりもリストア時間が長いことを確認する．基礎実験に使用したフォルダ内に含まれるファイルの種類とファイルの個数について表 1 に示す．表 1 のファイルの種類はファイルの拡張子を表している．

表 1　ファイルの種類

ファイルの種類	個数	ファイルの種類	個数
java	197	css	15
class	182	csv	11
html	108	json	9
jpg	94	md	2
docx	80	zip	2
png	71	pkl	1
txt	68	exe	1
pdf	40	asta	1
pptx	27	lnk	1
ipynb	22	xml	1
xlsx	18		

データ転送は rsync コマンドを使用し，圧縮データの圧縮形式は zip コマンドを使用して zip 形式にし，時間の計測には time コマンドを使用した．これを 10 回行った結果の圧縮データと非圧縮データの平均リストア時間を図 2 に示す．非圧縮のデータの平均時間は約 17.95 秒，圧縮データの平均時間は約 26.89 秒であった．

また，この時の対象フォルダの圧縮率は約 39.73 %であった．この基礎実験から圧縮データが非圧縮データよりもリストア時間が長いという結果が得られた．

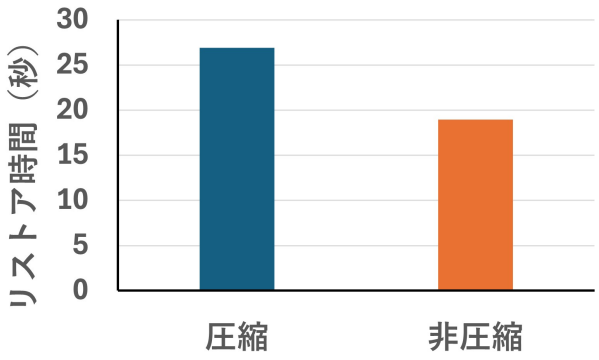

図 2　基礎実験の結果

各章の概要

第 2 章では関連研究について述べる．第 3 章では本稿における課題を解決する提案手法についての説明とユースケースについての説明をする．第 4 章では, 提案方式を用いて作成したソフトウェアの実装について述べる．第 5 章では評価実験を行った際の実験環境と実験結果について述べる．第 6 章では，本稿の提案方式についての議論を述べる．第 7 章では，本稿のまとめを述べる．

2. 関連研究

アクセス頻度の低いデータの予測とファイルの優先順位付けを行い，ホット/コールドデータ分類による新しい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 [7] ．アクセスログの記録、ストレージ使用状況の監視、システム入力データセットとなるメタデータの生成を継続的に行い，そこから機械学習を用いてファイルのアクセスパターンを予測し，優先度付けを行っている．本稿では，災害発生時のリストアを想定しているため，アクセスだけでなくファイルの更新も考慮しなければならない．そのため，この提案では本稿の課題を解決できない．

アクセス頻度にもとづいて，離れた場所からデータを取得またはアクセスするためのデータのバックアップを提案している研究がある [8]．データのバックアップ先をアクセス頻度にもとづいて選択している．本稿では，企業のサーバを想定してアクセス回数を用いて，バックアップデータの重要度を決定しているため，データのバックアップ先を決定する必要はないことから，この提案では本稿の課題を解決できない．

Oracle Database In-Memory Option を提案している研究がある [9]．アクセス頻度の低いデータをメモリに格納することは無意味であることから，アクセス頻度の低いデータを IM 列ストアに最適化しないように，メモリ内圧縮をしている．IMCU 内で，各列が圧縮ユニットとして格納

し，ユーザによって圧縮レベルを選択している．そのため，ユーザが圧縮レベルを指定する操作があり，バックアップウィンドウ中に行う本稿の提案では適用できないことから，本稿の課題を解決できない．また，Oracle 社のデータベースのパフォーマンス向上を目的としており，Oracle Database の一部であるため，本稿の課題を解決できない．

3. 提案

提案方式

本稿における提案方式ではユーザがファイルサーバにファイルをアップロードしたときからのファイルのアクセス回数・更新回数をもとにフォルダごとの重要度を決定する．その後，この重要度をもとに対象のフォルダを圧縮するかしないかを決める．本稿では人的ミスによってデータ損失が発生した場合を想定している．そのため，先週からの定期的な更新があるということは現在進行中のプロジェクトデータである可能性が高いと考え，更新回数を用いる．また，ファイルサーバは同じネットワークに繋いでいる他の人とファイルを共有するためのサーバのことである[*5]．このことから更新回数取得のためにファイルサーバを監視する本稿では，更新はしないが共有のためにアクセスするファイルも考慮し，アクセス回数を用いる．

また，上記の理由から本稿ではアクセス回数と更新回数は平等として扱う．

本稿の提案を用いて重要度が高いデータはフルバックアップデータも非圧縮にする．これにより，災害発生時に重要度の高いデータをより早くリストアできるようにする．

ここからは提案方式を重要度の決定方法とバックアップサーバでの圧縮・非圧縮をどのように設定しているかの 2 つに分けて説明する．

重要度の決定方法

本稿における提案方式の重要度を決定する方法の流れを図 3 に示す．

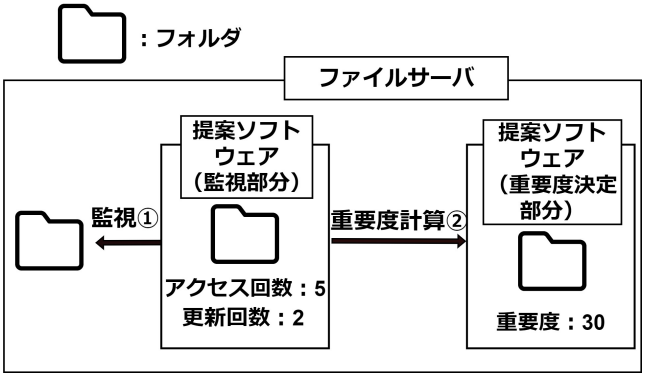

図 3 重要度の決定方法

[*5] https://x.gd/to06i

重要度は図 3 の提案ソフトウェア（監視部分）と提案ソフトウェア（重要度決定部分）を用いて決定する．

まず，提案ソフトウェア（監視部分）では重要度を決定するために必要な各ファイルのアクセス回数と更新回数を記録するためのフォルダ監視を行う．次に，提案ソフトウェア（重要度決定部分）で先ほど記録されたアクセス回数と更新回数からフォルダごとの重要度を決定する．フォルダは監視プログラムを配置しているディレクトリ内のフォルダである．式（1）に重要度を決定する時に使用する式を示す．

$$I = \frac{n_a}{tn_a} + \frac{n_u}{tn_u} \tag{1}$$

変数 I は重要度を表し，n_a はフォルダのアクセス回数，tn_a は監視ディレクトリ内にある全ファイルのアクセス回数の平均，n_u はフォルダの更新回数，tn_u は監視ディレクトリ内にある全ファイルの更新回数の平均を表す．また，平均を取る期間は 1 週間とし，更新回数・アクセス回数を扱っているため，フルバックアップを取る期間と同様にした．

重要度の決定後，この値が高い順にフォルダを並び替える．その後，バックアップサーバへバックアップするデータのサイズを送信する．

圧縮・非圧縮の基準

本稿の提案方式における圧縮・非圧縮の流れを図 4 に示す．

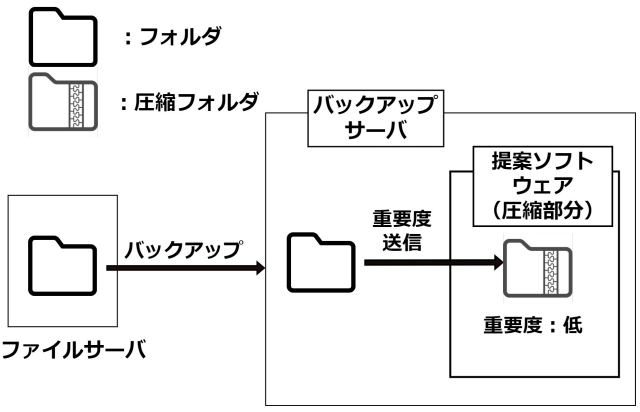

図 4 圧縮・非圧縮の流れ

図 4 の提案ソフトウェア（圧縮部分）を用いて，求めた重要度からフォルダの圧縮・非圧縮を決める．まず，ファイルサーバから送られてきたバックアップ分のデータサイズがバックアップサーバの空き容量内に収まるかを確認する．この時，vSAN ストレージを提供している VMware 社の推奨値は「70 %」である[*6]．これはサーバ 1 台が壊れてもデータロストは起こらず、リビルド処理の過程でも問題

[*6] https://x.gd/Q6EEx

は起らないためである．そのため，本稿における提案でも上記を使用して，バックアップデータ分のデータサイズが，バックアップサーバの 70 %の容量を超える場合，重要度の低い順からバックアップデータ毎にデータサイズを確認し，70 %を超えていれば圧縮するようにする．この大まかな流れを図 5 に示す．

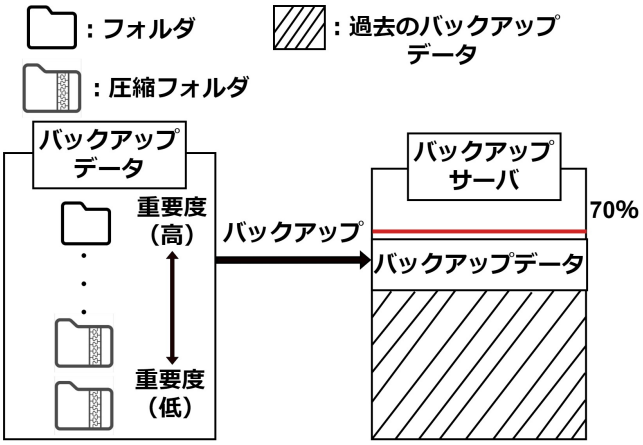

図 5　70 %を超えないように圧縮する

式（1）により，導き出されたフォルダの中で重要度の値が低いものから順に圧縮する．

これにより重要度が高いフォルダ，つまり使用頻度の高いフォルダや現在のプロジェクトデータはシステム障害発生時にリストア時間を短縮できるようにしている．

ユースケース・シナリオ

本稿ではファイルの書き換えミスによって，企業データを損失してしまった時に，データを早急に復旧したい中小の情報処理系企業をユースケースとして想定する．

バックアップのスケジュールに関して，一般的な中小企業では週末にフルバックアップを取っており，それ以外の日は増分バックアップを取っていることから，本ユースケースのバックアップ・スケジュールも上記と同様とする[*7]．

増分バックアップとは一般的に，最初にフルバックアップを実行し、その後ファイルの変更内容内容を毎回保存することである [10].

図 6 に本稿の提案を使用した後のユースケースを示す．図 6 のバックアップ管理者は災害発生時に災害復旧計画の実行をし，人的ミスによりデータ損失したデータをバックアップサーバからファイルサーバへリストアの動作を行う．この図に示すように災害発生時に重要度の高いデータが非圧縮なためにリストア時間を短縮することでダウンタイムの短縮をすることが出来る．

[*7] https://www.climb.co.jp/blog_vmware/bcp-7372

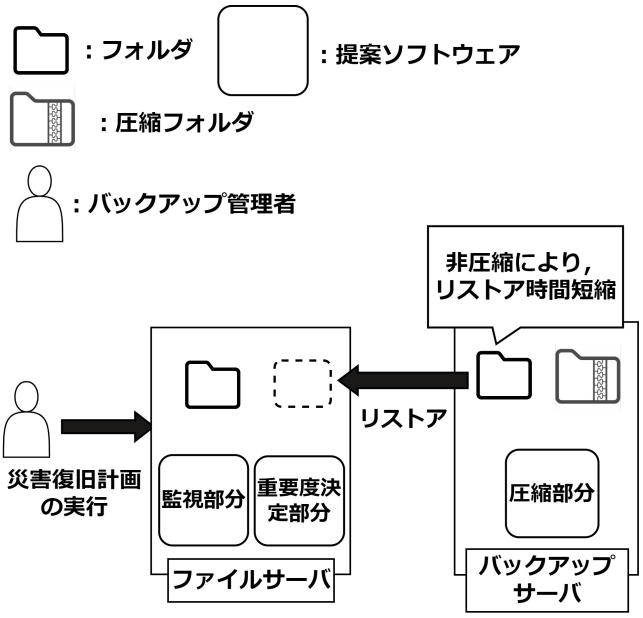

図 6　提案を使用した後のユースケース

4. 実装

提案ソフトウェアは Python で 3 つのプログラムを作成した．

図 7 に実装するソフトウェアの流れを示す．monitoring.py は，フォルダ監視を行い，ファイルのアクセス回数と更新回数をフォルダごとに記録する機能を持つ．importance.py は，記録したアクセス回数と更新回数をもとに重要度を決定し，この情報とバックアップデータの情報をバックアップサーバへ送信する機能を持つ．また，増分分のバックアップデータがバックアップできない場合，重要度の低い順から圧縮するようにする．この時，バックアップサーバ対象ディレクトリがバックアップサーバ側で圧縮されていれば，その対象ディレクトリの情報を取得し，増分バックアップ分を圧縮して送信する．バックアップサーバ側で圧縮されていなければそのまま送信する．compression.py は，バックアップを行うデータを重要度が低い順に並び変えてその情報と，バックアップサーバの容量をファイルサーバへ送る機能を持つ．このプログラムは増分バックアップデータにより，バックアップサーバの容量の 70 %を超える場合に実行する．以下にそれぞれのプログラムの詳しい説明を行う．

monitoring.py

monitoring.py ではファイルサーバ内を watchdog を用いて，フォルダ監視を行い，ファイルに対する更新回数を取得し，csv ファイルへ記述する．アクセス回数は，gettaime 関数を用いて対象のファイルの最終アクセス日時との比較を行い，csv ファイルへ記述する．

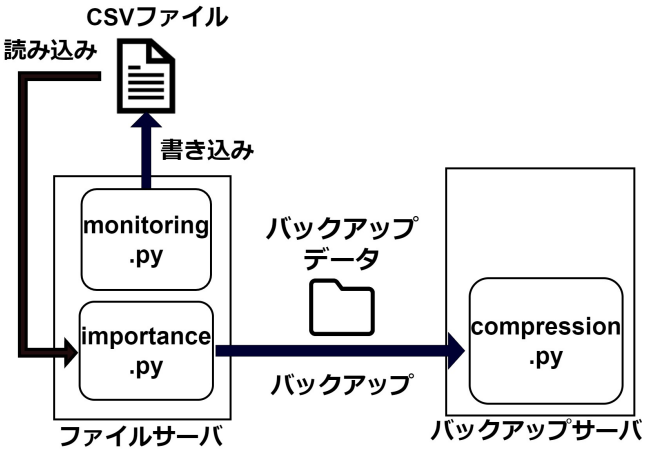

図 7 実装するソフトウェアの流れ

importance.py

importance.py では csv ファイルを読み込み，式（1）からフォルダに対する重要度を決定し，バックアップデータのデータサイズと共にバックアップサーバへ socket を使用して送信する．その後，compression.py から socket を用いて，バックアップサーバの容量，重要度が低い順に並び替えたバックアップデータの情報を受信するようにする．受信したデータを用いて，バックアップデータがバックアップサーバの 70 %を超えるかの確認をバックアップデータ毎に行う．この時，対象のバックアップデータの非圧縮分のデータサイズを引いた後に，圧縮したデータサイズを加算して，データサイズの比較を行う．

compression.py

compression.py ではファイルサーバから送られてきたデータを用いて，バックアップデータのサイズとバックアップサーバの容量のサイズを取得する．また，ファイルサーバから送られてきたデータを重要度が低い順に並び替えて，上記と合わせて importance.py へデータを送信する．

5. 評価実験

作成した提案ソフトウェアによって，自身が定義した重要なデータが正しく分類され，非圧縮でのバックアップになるかの評価を行う．

実験環境

図 8 に実験環境を示す．本稿のユースケースと同業にあたる IIJ では Windows のファイルサーバを使用している[*8]．そのため，本稿における実験環境ではファイルサーバは Windows のローカル PC を使用し，バックアップサーバには VM（仮想マシン）を用いる．

実験に使用したフォルダは不定期による低頻度なアクセス・更新回数の 2023projectA，定期的による高頻度なアクセス回数・不定期による低頻度な更新回数の 2023projectB，定期的による高頻度なアクセス・更新回数の 2023projectD の 3 つのフォルダを用意した．この時の高頻度とは 15〜20 の回数，低頻度とは 0〜3 回の回数とし，定期的とは 1 週間毎日を意味する．また，2023projectC は不定期による低頻度なアクセス回数・定期的な高頻度の更新回数を持つが，更新をするためにはアクセスが必須のため，物理的に作成できないことからこのフォルダは本稿の実験から取り除いた．また，本実験で使用したアクセス回数・更新回数の平均は 1 週間分である 5 日間の期間を設定し，実験を行った．

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

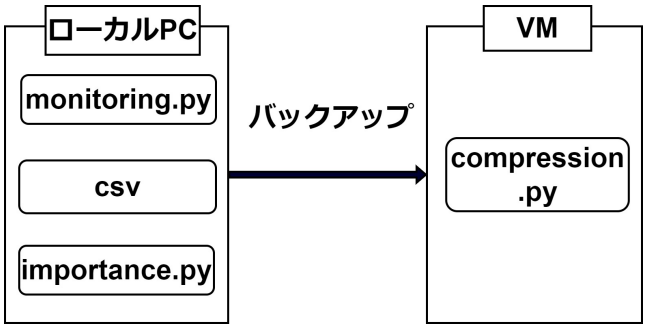

図 8 実験環境

実験結果と分析

本稿の提案ソフトウェアを用いて，実験を行った結果を図 9 に示す．実験環境で定義した定期的による高頻度なアクセス・更新回数の 2023projectD のフォルダの重要度が 1 番高くなっていることが分かる．これにより，アクセス頻度が高く，更新頻度が高いデータはバックアップ時に優先的に非圧縮でバックアップされるようにする．基礎実験により，非圧縮のデータが圧縮データよりもリストア時間が短いことが証明されていることから，重要度の高い 2023projectD のフォルダは災害発生時に，圧縮されている場合よりもリストア時間を早めることができる．

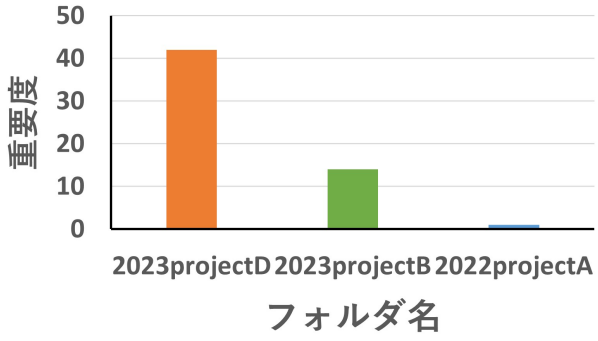

図 9 評価実験の結果

6. 議論

本稿ではアクセス回数と更新回数にもとづいて重要度の

決定を行い，重要度が高いデータを非圧縮にし，重要度が低いデータを圧縮するようにした．本稿ではこれをバックアップデータの容量とバックアップサーバの容量を足した値が，バックアップサーバの 70 %を超えるかを確認するようにしていた．しかし，圧縮した場合に非圧縮データよりもデータサイズが大きくなるデータが存在する場合，そのデータを圧縮にした方がバックアップサーバの容量を圧迫してしまうため，本来圧縮しなくてもよいデータも圧縮することに繋がってしまう．

また，本稿の手法では各バックアップデータに対して容量の比較を行っていた．しかし，この場合バックアップ時間が長引いてしまい，効率が悪くなってしまう．そのため，今回の手法のように重要度が低いデータから順に圧縮するだけでなく，各バックアップデータに対する圧縮率の取得を行い，データサイズの考慮が必要である．これにより，圧縮データと非圧縮データを分けることが出来，バックアップサーバへ送信するための操作を削減することが出来る．

本稿ではファイルのアクセス回数と更新回数から重要度を設定した．しかし，アクセス回数と更新回数が少なくても重要なファイルが存在する．また，本稿の重要度ではアクセス回数や更新回数が 1 回でも多ければ，重要度が高くなる．そのため，100 回のアクセスと 101 回のアクセスでは大きな差はないが，1 回のアクセスの差で圧縮・非圧縮が決まりかねない．そのため，アクセス回数と更新回数以外の要素も用いる必要がある．

本稿ではバックアップサーバにあるフルバックアップデータも非圧縮にすることを提案したが、現段階ではこの機能の実装は行っていない．そのため，本稿における提案を実現するには，今後この機能の実装を行う必要がある．

評価実験では本稿の提案が上手く適用されたかの確認するための実験を行った．しかし，本稿の評価実験ではファイルの内容から事前に重要であるファイルの識別を行うことが出来ていなかった．そのため，事前に実験対象のユーザを用意し，そのユーザにとって重要なデータを順番に決定し，提案を用いた結果との比較を行う必要がある．

7. おわりに

課題は災害発生時に，復旧したいデータが圧縮されていることにより，許容ダウンタイム内にデータ復旧が終わらないことである．提案方式はファイルのアクセスと更新回数にもとづく重要度の決定によるリストア時間の短縮とストレージ容量の確保である．評価実験では作成した提案ソフトウェアによって，自身が定義した重要なデータが正しく分類され，非圧縮でのバックアップになるかの実験を行った．結果はアクセス回数・更新回数が高くなるにつれて重要度が高くなり，正しく分類された．

謝辞

本テクニカルレポートを執筆にあたりご指導いただきました東京工科大学コンピュータサイエンス学部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] 原仁志，ハラヒトシ：コンピュータの障害対策とバックアップ，鈴鹿大学 鈴鹿大学短期大学部紀要 人文科学・社会科学編， No. 2, pp. 69–81 (2019).

[2] Zhang, J. and Li, H.: Research and Implementation of a Data Backup and Recovery System for Important Business Areas, *2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Vol. 2, pp. 432–437 (online), DOI: 10.1109/IHMSC.2017.209 (2017).

[3] Xia, R., Yin, X., Alonso Lopez, J., Machida, F. and Trivedi, K. S.: Performance and Availability Modeling of ITSystems with Data Backup and Restore, *IEEE Transactions on Dependable and Secure Computing*, Vol. 11, No. 4, pp. 375–389 (online), DOI: 10.1109/TDSC.2013.50 (2014).

[4] Zhao, Y. and Lu, N.: Research and Implementation of Data Storage Backup, *2018 IEEE International Conference on Energy Internet (ICEI)*, pp. 181–184 (online), DOI: 10.1109/ICEI.2018.00040 (2018).

[5] Gaonkar, S., Keeton, K., Merchant, A. and Sanders, W. H.: Designing Dependable Storage Solutions for Shared Application Environments, *IEEE Transactions on Dependable and Secure Computing*, Vol. 7, No. 4, pp. 366–380 (online), DOI: 10.1109/TDSC.2008.38 (2010).

[6] Mendonça, J., Lima, R., Queiroz, E., Andrade, E. and Kim, D. S.: Evaluation of a backup-as-a-service environment for disaster recovery, *2019 IEEE Symposium on Computers and Communications (ISCC)*, IEEE, pp. 1–6 (2019).

[7] 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).

[8] Janpet, J. and Wen, Y.-F.: Reliable and Available Data Replication Planning for Cloud Storage, *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, pp. 772–779 (online), DOI: 10.1109/AINA.2013.125 (2013).

[9] Lahiri, T., Chavan, S., Colgan, M., Das, D., Ganesh, A., Gleeson, M., Hase, S., Holloway, A., Kamp, J., Lee, T.-H., Loaiza, J., Macnaughton, N., Marwah, V., Mukherjee, N., Mullick, A., Muthulingam, S., Raja, V., Roth, M., Soylemez, E. and Zait, M.: Oracle Database In-Memory: A dual format in-memory database, *2015 IEEE 31st International Conference on Data Engineering*, pp. 1253–1258 (online), DOI: 10.1109/ICDE.2015.7113373 (2015).

[10] Chao, Y., Wen, X., Guohui, W., Xinge, Q., Sai, L. and Lele, Z.: Incremental local data backup system based on bacula, *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pp. 429–432 (online), DOI: 10.1109/IICSPI.2018.8690434 (2018).

