WordPressにおける転送データの削減による移行にかかる実行時間の短縮

手塚 雄星[1]　西村 克己[1]　串田 高幸[1]

概要：東京工科大学のクラウド・分散システム研究室では, WordPress サイトを運用し, 研究室に所属する学生がブログを執筆している. このサイトは毎日フルバックアップを作成し, バックアップデータを NAS に保存している. アクセス数の増加に対応し, Web サイトに継続的にアクセスできるように, 管理者は Web サイトのデータを移行する. 課題は, Web サイトのデータを移行元の Ubuntu サーバー（以後, 移行元サーバーと記述する）から移行先の Ubuntu サーバー（以後, 移行先サーバーと記述する）へ移行する際に時間がかかることだ. 提案は，WordPress の記事を執筆した際に生成されるリサイズされた画像のうち，記事に使用されている画像のみを移行の実行対象とし, 移行するファイルの転送量を減らすことで，移行時の時間を短縮を行った. 評価は, 提案適用前と提案適用後における, 移行時のファイルの転送を 10 回行い, 実行時間を比較した. 実験環境では, 移行元サーバーから NAS を経由して移行先サーバーへ Web サイトのデータを転送した. 結果として, 提案適用前において, 移行元サーバーから NAS へ移行した際にかかる時間の平均は約 92.6 秒であるのに対して, 提案適用後は約 4.4 秒となり, 約 95.1%削減できた. また, 提案適用前において, NAS から移行先サーバーへ移行した際にかかる時間の平均は約 157.1 秒かであるのに対して, 提案適用後は約 9.5 秒となり, 約 93.9%削減できた. 移行する時間を短縮することによって, ダウンタイムが短くなるため, Web サイトに掲示されている項目を閲覧したい人にとっては, 閲覧するまでに待機する時間が短くなる.

1. はじめに

背景

インターネット上には, Web サイトが運営されている. Web サイトを構築するためのソフトウェアの一つに, WordPress がある [1]. WordPress は Contents Management System (以後, CMS と記述する) の一種であり, ユーザーが簡単に Web サイトを管理・運営できるオープンソースのソフトウェアである [2]. インターネット上の Web サイトにはユーザーがアクセスしており, 必要な情報を取得している. Web サイトへアクセスする際に, アクセス数が増加しても, 継続的にコンテンツを閲覧できるように, 管理者は Web サイトが閲覧できることを維持しなければならない [3]. アクセス数が増加するとページの読み込みが遅くなり, ユーザーが閲覧を諦めてしまい, 結果として悪影響を与える [4]. そのため, 現在の Web サービスでは, アクセス数の増加に対して, 迅速かつ柔軟に対応することが求められている [5]. Web サイトへのアクセス数の増加への対応として, データを移行する方法がある.

東京工科大学コンピュータサイエンス学部のクラウド・分散システム研究室では, 仮想マシン (以後, VM と記述する) 上に WordPress を構築して, 研究室に所属する学生が執筆するブログやテクニカルレポートを公開するサイト（以後, CDSL サイトと記述する）を運用している[*1]. Web サイトは, 毎日 0 時に圧縮してフルバックアップを作成している. バックアップは作成後, Network Attached Storage(以後, NAS と記述する) に 1 ヶ月間保存している.

インターネット上へ実際に公開している CDSL サイトを移行の対象とすると, サービスを運営する上で悪影響を及ぼしてしまうため, バックアップされたデータを利用して, 移行元サーバーに CDSL サイトを構築する.

CDSL サイトのデータを移行する際には，WordPress の記事の内容やアクセスデータの内容とデータベースの Dump ファイルの 2 つが必要である. 移行元サーバーは, Web サーバーとして Nginx やウェブサーバーと組み合わせて使用される PHP-FPM, データベースとして MariaDB をセットアップする. 2 つのデータを tar コマンドで圧縮して, 移行元サーバーから NAS の場合と NAS から移行先サーバーの場合の 2 つの間を rsync コマンドで移行する [6].

[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://ja.tak-cslab.org (閲覧日:2024-12-20)

WordPress サイトのデータを移行する手順を 7 つに分けて説明する．1 つ目に移行元のサーバーから WordPress のデータをバックアップする．バックアップする対象は WordPress のデータベースと WordPress のファイルの 2 つである．2 つ目に移行先サーバーの準備をする．移行先サーバーに WordPress をインストールして，プラグインやテーマをインストールする．また，データベースを作成して，適切な設定を行う．3 つ目にデータを転送する．バックアップしたデータを，rsync コマンドを利用して移行先サーバーに移行する．4 つ目に移行後サーバーでデータベースの接続設定を変更する．新しいデータベースを作成する．MySQL を利用して移行元サーバーでバックアップした DUMP ファイルをインポートする．5 つ目に移行後サーバーで Web サイトの URL を変更する．6 つ目に全てのデータが正しく移行されたことを確認するために，移行先サーバーの Web サイトをテストする．ページの表示や機能が正常に動作するかを確認する．7 つ目にテストが完了して，問題がないことを確認してから移行先サーバーの Web サイトを公開する．

課題

課題は Web サイトのデータを移行元サーバーから移行先サーバーへ移行する際に，全てのデータを移行すると時間がかかることである．移行元サーバーから移行先サーバーへ Web サイトのデータを移行する流れを図 1に示す．

図 1 移行元サーバーから移行先サーバーへ Web サイトのデータを移行する流れ

図 1の移行元サーバーには，Web サイトを運営するために保管しているデータがある．移行元サーバーから移行先サーバーへ移行する際に，全ての Web サイトのデータを移行すると時間がかかる．

各章の概要

第 2 章では，関連研究について説明する．第 3 章では，課題について解決するための提案方式について説明する．第 4 章では，提案した手法の実装について説明する．第 5 章では，評価実験として実験内容と実験結果と分析について説明する．第 6 章では，提案方式についての議論を説明する．最後に，第 7 章にて結論を説明する．

2. 関連研究

VM のメモリページをコピーして移行先サーバーの物理サーバーに送信する方法が提案されている研究がある [7]．VM を移行することは行っているが，Web サイトのアプリケーションである WordPress のデータの移行は行ってない点に改善の余地がある．

インタークラウド移行と動的リソース管理に基づいて，異なるクラウド環境間で VM とコンテナを効率的に移行する方法が提案されている研究がある [8]．VM が一時的に停止するため，サービスが利用できなくなる問題があるが，移行時間を削減し，サービスの停止を最小限に抑える方式を提案していない点に改善の余地がある．

VM マイグレーションによって生成されるトラフィックの合計を計算するためのコスト見積もりモデル「Remedy」が提案されている研究がある [9]．Remedy はネットワークリソースのインテリジェントな割り当てを提案しているが，移行コストのデータを削減する手法を取っていない点に改善の余地がある．

VM のライブマイグレーションを最適化するために，ソフトウェア定義ネットワークを活用して，複数の VM 間でのパフォーマンス干渉と移行コストを最小化する手法 (以後，ITEM と記述する）が提案されている研究がある [10]．ITEM 戦略は，移行コストとネットワーク帯域幅の最適化に優れているが，未使用データを除外することで，全体のデータ量を削減することには触れていない点に改善の余地がある．

3. 提案

提案方式

提案手法は，WordPress の記事を執筆した際に生成されるリサイズされた画像のうち，記事に使用されている画像のみを移行の実行対象とし，移行するファイルの転送量を減らすことで，移行時の時間を短縮を行うことである．

WordPress で使用しているコンテンツのディレクトリ構造を図 2で示す．図 2にあるように，移行元サーバーに wp-admin や wp-content, wp-includes というディレクトリが存在する．wp-content ディレクトリの中には，plugins や uploads, languages というディレクトリが存在する．uploads というディレクトリの中には，画像が保管されている．保管されている画像はデータベースに紐づいている．データを移行する前に，移行元サーバーでデータベースを参照し，Web サイトで未使用画像を識別する．未使用と識別された画像をアーカイブし，ファイルの転送量を削減することで，移行時間を短縮する．

WordPress を利用して構築した Web サイトにおいて，データベースを参照し，未使用画像をアーカイブする流れを図 3で示す．移行元サーバーに紐づいているデータベース

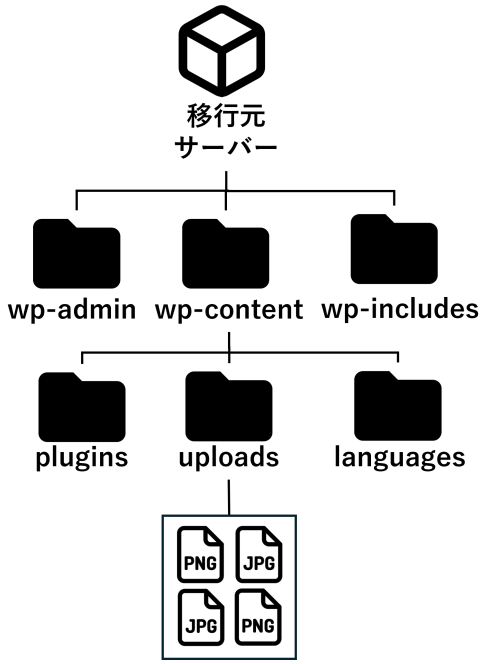

図 2 WordPress のディレクトリ構造

は複数のカラムに分かれている．カラムである post_content には, Web サイトで使用している画像や未使用画像が記載されている．データベースの post_content には, 使用している画像と未使用画像のどちらも記載されているため, 未使用画像を移行時のファイル転送の対象から除外する．

記事の本文や画像が参照されているデータベースの wp_posts のテーブルの一部を表 1に示す．

表 1 wp_posts のテーブルの一部

SELECT_ID	Post Title	Post Content
117	CDSL の立ち上げを振り返る	http://ja.takcslab.org/wp-content/uploads/2020/01/IMG_1109-1024x768.jpg
244	Qiita 週間 Organization ランキングで 3 位	http://ja.takcslab.org/wp-content/uploads/2020/01/B8AEE39-1024x986.jpeg

post_content の img タグの部分を見ると，Web サイトで使用している画像を認識できる． SELECT_ID が 117 の post_content を見ると，JPG 形式で IMG_1109-1024x768.jpg という画像を使用している．よって, 表 1を参照すると, Web サイトで使用している画像である IMG_1109-1024x768.jpg をアーカイブとして別のファイルに保存しておくことで, Web サイトのデータを移行する時間を短縮することができる．

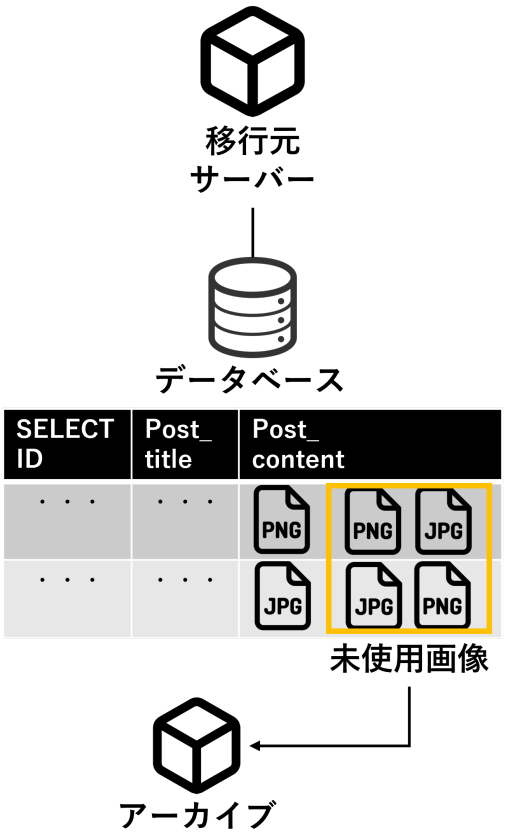

図 3 Web サイトでの未使用画像をアーカイブする図

ユースケース・シナリオ

サーバーに Ubuntu Server をインストールし, WordPress で動作するブログサイトを移行する場面を想定する．ユースケース・シナリオを図 4に示す．

従来の方法である移行元サーバーにある全てのデータを移行先サーバーへ移行することは, 時間がかかる．提案方式をもちいることで, WordPress によってリサイズされた画像の中にある未使用画像をアーカイブとして別のファイルに保管して, 使用されている画像を転送することで，移行時のファイルの転送量を削減した．これにより, 移行元サーバーから移行先サーバーへの移行時間を短縮することができる．

移行する時間を短縮することによって, ダウンタイムが短くなる．そのため, Web サイトに掲示されている項目を閲覧したい人にとっては, 閲覧するまでに待機する時間が短くなる．よって, Web サイトを閲覧できない人を最小限に抑えることができる．

4. 実装

移行元サーバーに対して, 記事に使用している画像の取り出しと, 記事の未使用画像をアーカイブし, 移行を実施するソフトウェアを開発言語である Python を使い，2 つ作成した．

1 つ目のソフトウェアは, WordPress の記事の中身が記

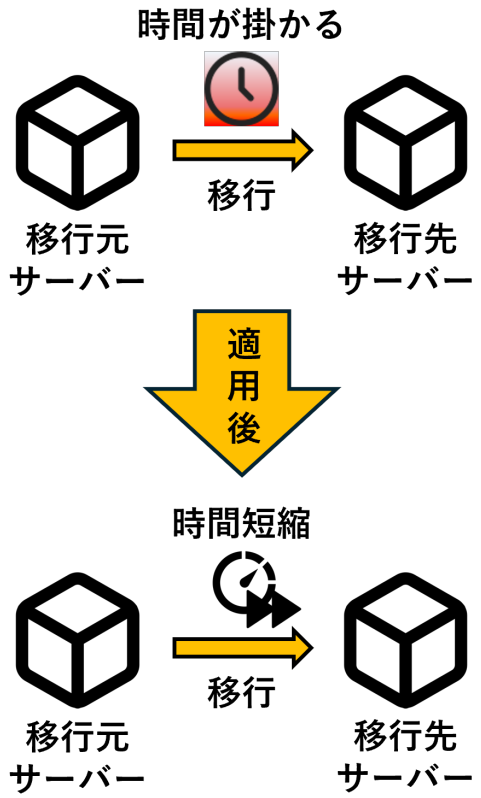

図 4 ユースケース・シナリオ

録されているデータベースにアクセスし, 記事データから画像のファイル名を取り出す File-Search である.

2 つ目のソフトウェアは, 1 つ目で記事データから画像のファイル名を取り出した情報を使い, 未使用画像をアーカイブとして圧縮を行う Remove-Png である. それぞれのソフトウェアを分けて説明する.

File-Search

WordPress は, 記事を執筆したデータの html がデータベース内の wp_posts というテーブルに保存する. wp_posts の post_content 内にある img タグを取り出す. その後, img タグを取り出した URL と該当の PostID を取り出して, csv ファイルに書き出す.

CSV ファイルに書き出した出力の一部を表 2に示す. 表

表 2 CSV ファイルの出力結果の一部

Post ID	Image URL
177	2020/01/IMG_1109-1024x768.jpg
177	2020/01/IMG_1110-1024x768.jpg
177	2020/01/IMG_1108-768x1024.jpg
177	2020/01/IMG_1118-1024x768.jpg
177	2020/01/IMG_1121-1024x768.jpg

2では, Post ID の 177 では複数の画像が記事に使われている. 記事に使われている画像のタイトルを CSV ファイルに記録しておくことで, 記録されていない画像をアーカイブとして移行の際の転送対象から外すことができる.

Remove-Png

1 つ目のソフトウェアである File-Search にて, 記事の本文データが記録されているデータベースにアクセスし, 使用している画像のファイル名を取り出すことができたことを前提とする. 処理の流れを図 5に示す. Remove-PNG

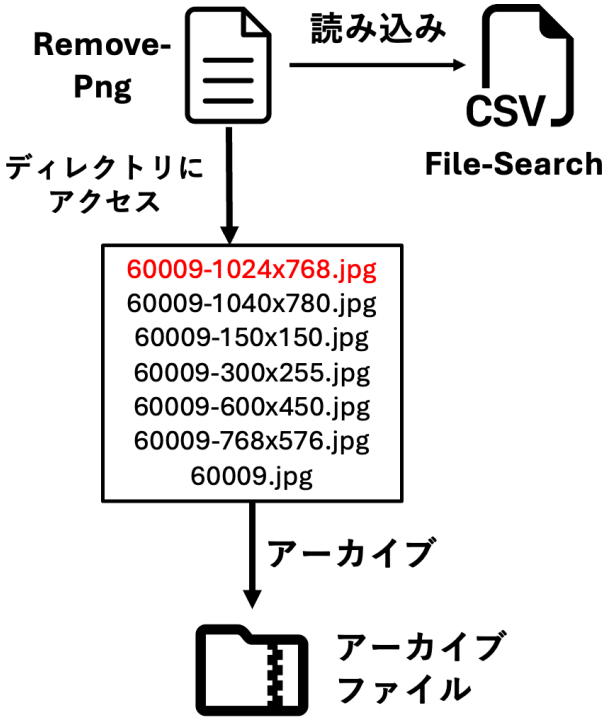

図 5 アーカイブするソフトウェアの処理の流れ

が, File-Search にて出力された CSV ファイルを読み込む. CSV ファイルを元にして, Remove-PNG が移行元サーバーの/var/www/html/wp-content/uploads 内のディレクトリのうち, CSV ファイルに記録されていない画像ファイルを取り出してアーカイブファイルとして保管をする.

図 5では, 60009-1024x768.jpg のファイルが WordPress の記事に使用されているデータであると記録して, それ以外のファイルはアーカイブする画像として Remove-Png で操作を行った.

5. 評価実験

提案適用前と提案適用後における, 移行元サーバーから移行先サーバーへ移行を行う際のファイルの転送を行う時間を比較して, 評価した.

基礎実験

移行元サーバーから NAS の間と NAS から移行先サーバーの間において, Web サイトを運営するために必要な全

てのファイルを転送したときの実行時間を計測した．移行元サーバーから移行先サーバーへデータを転送できず，NAS を経由してデータを転送し，移行を行うことを想定し，実験を行った．移行元サーバーにある全てのファイルを NAS に移行した際の実行時間を図 6に示す．図 6は，移

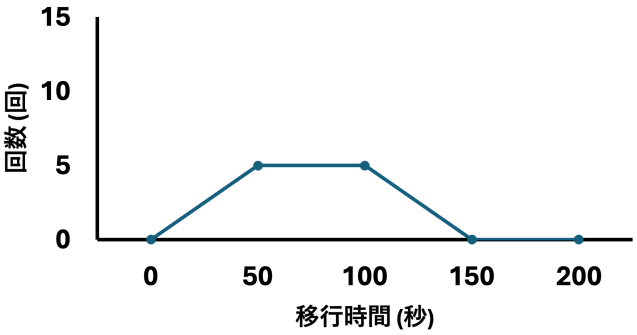

図 6 移行元サーバーにある全てのファイルを NAS に移行した際の実行時間

行元サーバーにある全てのファイルを NAS へ移行する時間を 10 回計測した．移行元サーバーから NAS に移行したときの時間の最小値は約 75.0 秒であり，最大値は約 122.7 秒であった．NAS から移行先サーバーに移行したときの時間の平均は約 92.6 秒であった．

NAS にある全てのファイルを移行先サーバーへ移行した際の実行時間を図 7に示す．図 7は，NAS にある全ての

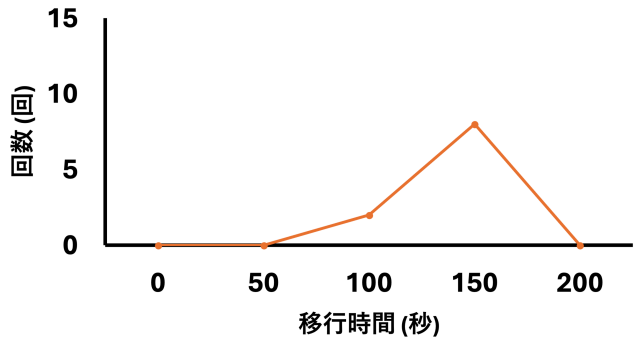

図 7 NAS にある全てのファイルを移行先サーバーへ移行した際の実行時間

ファイルを移行先サーバーへ移行する時間を 10 回計測した．NAS から移行先サーバーに移行したときの時間の最小値は約 144.1 秒であり，最大値は約 183.5 秒であった．NAS から移行先サーバーに移行したときの時間の平均は約 157.1 秒であった．

このことから，移行元サーバーの全てのファイルを移行先サーバーへ移行する平均時間の合計が約 249.7 秒ということが分かる．

実験環境

実験環境には，提案ソフトウェアを導入し実行するサーバー 1 台，WordPress をインストールした移行元サーバー 1 台，WordPress をインストールした移行先サーバー 1 台の計 3 台を用意した．以下にサーバーの構成要素を，図 8に実験環境を示す．

- サーバー構成情報 (提案ソフトウェア実行用)
  OS: Ubuntu Server 24.04.1 LTS
  vCPU: 2[Core]
  RAM: 2[GB]
  SSD: 25[GB]
- サーバー構成情報 (移行元サーバー，移行先サーバー)
  OS: Ubuntu Server 24.04.1 LTS
  vCPU: 2[Core]
  RAM: 2[GB]
  SSD: 32[GB]

提案ソフトウェア実行用のサーバーでは，Python で実装した提案ソフトウェアを導入し，動作させた．サーバーは，(vCPU:2[Core]，RAM:2[GB]，SSD:25[GB]) の性能で作成した．移行元サーバー，移行先サーバーの WordPress 動作のそれぞれのサーバーでは，2 台とも同じ性能の (vCPU:2[Core]，RAM:2[GB]，SSD:32[GB]) で作成した．

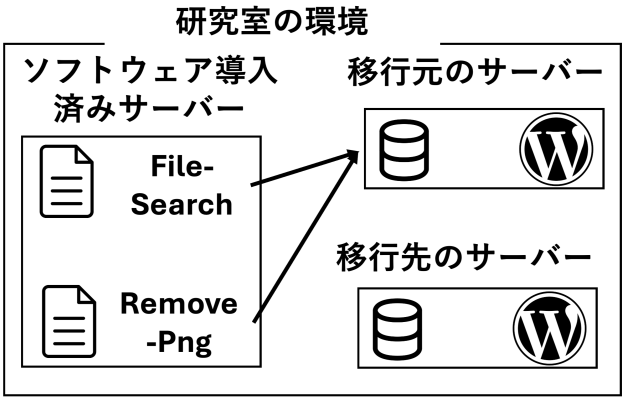

図 8 評価実験の構成

実験結果と分析

実験では，移行元サーバーから NAS を経由して，移行先サーバーへ Web サイトのデータを転送した．提案方式を適用し，移行元サーバーから NAS へファイルを転送する時間と NAS から移行先サーバーへファイルを転送する時間の 2 つを計測した．

提案方式を適用し，移行元サーバーから NAS へ移行した際の時間を図 9に示す．図 9は，提案方式を適用して，移行元サーバーから NAS に移行する時間を 10 回計測した結果を示している．移行元サーバーから NAS に移行したときの時間の最小値は約 4.3 秒であり，最大値は約 4.6 秒で

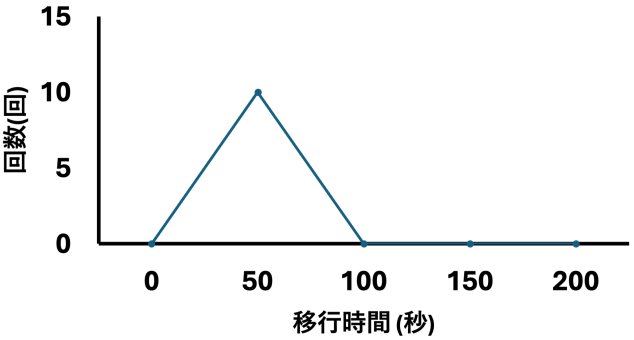

図 9 提案方式を適用後に移行元サーバーから NAS へ移行した際の実行時間

あった．移行元サーバーから NAS に移行したときの時間の平均は約 4.4 秒であった．

提案方式を適用後に NAS から移行先サーバーへ移行した際の時間を図 10に示す．図 10は，提案方式を適用して，

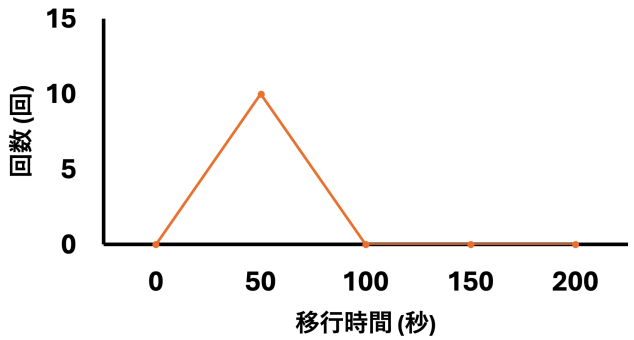

図 10 提案方式を適用後に NAS から移行先サーバーへ移行した際の実行時間

NAS から移行先サーバーへ移行する時間を 10 回計測した．NAS から移行先サーバーに移行したときの時間の最小値は約 9.3 秒であり，最大値は約 9.8 秒であった．NAS から移行先サーバーに移行したときの時間の平均は約 9.5 秒であった．

移行元サーバーから NAS へ移行した際の実験結果としては，移行時間の平均が約 92.6 秒から約 4.4 秒まで減少し，約 95.1%削減できた．NAS から移行先サーバーへ移行した際の実験結果としては，移行時間の平均は約 157.1 秒から約 9.5 秒まで減少し，約 93.9%削減できた．

このことから，移行元サーバーから移行先サーバーへ移行する際の平均時間が約 249.7 秒から約 13.9 秒まで減少し，約 94.4%削減できたことが分かる．

6. 議論

提案方式では，WordPress の記事に使用している画像のみを取り出し，移行する際のデータと定め， 記事に使用していると判定しなかった画像はアーカイブを行った．移行するファイルの転送量を減らすことによって，移行時間を短縮することを試みた．移行する Web サイトを動かしている WordPress の設定によって，リサイズされた画像が自動的に生成されていた．Web サイトに使用していないのにも関わらず，リサイズされた画像が生成されているため，画像を生成した根拠が明確ではない．移行時に参照されていないリサイズされた画像が移行後に使用されると，画像が表示されない問題が発生する．そのため，移行元サーバーから，移行先サーバーに全てのファイルを送る必要がある．

この問題に対して，移行先サーバーにファイル転送し，移行先サーバーで WordPress が起動したのちに，提案によってアーカイブされた画像を送ることで移行元サーバーから移行先サーバーに全てのファイルを送ることができる．

提案方式では，WordPress の記事に使用されていなかった画像を取り出し，移行するファイルの転送量を減らすことで，移行時間を短縮することを試みた．WordPress は，記事をユーザーが自由に執筆することができる．記事の執筆時，同一の画像を 2 回アップロードすると，それぞれ別のファイル名で保存される．例えば，IMG-A.png を 2 回アップロードすると，IMG-A-1.png という別のファイル名として保存される．これらの画像は，同一の画像であるが，別のファイル名で保存されるため，余分な画像が残る．そのため，全く同一の画像データがアップロードされた時に，画像の縦横比と画像が同じサイズであった場合ははじめにアップロードされた画像を残すことで，重複画像を削減できる．重複画像を削減することによって，ファイルの転送量が減り，移行時間の短縮に繋がる．

提案方式では，WordPress の記事に使用されていた画像を取り出し，移行するファイルの転送量を減らすことで，移行時間を短縮することを試みた．WordPress は記事に使用する画像をリサイズして，容量を削減する機能がある．リサイズする前の画像を残し，移行先サーバーにて画像のリサイズを行うことで，移行元サーバーと移行先サーバーと同じ状態にすることができる．画像のリサイズの作業の時間が，移行元サーバーから移行先サーバーに全てのデータを転送する際の時間と比較して早くなれば，有効である．

7. おわりに

東京工科大学のクラウド・分散システム研究室では，WordPress サイトを運用し，研究室に所属する学生がブログを執筆している．このサイトは毎日フルバックアップを作成し，バックアップデータを NAS に保存している．アクセス数の増加に対応し，Web サイトに継続的にアクセスできるように，管理者は Web サイトのデータを移行する．課題は，Web サイトのデータを移行元サーバーから移行先サーバーへ移行する際に時間がかかることである．提案は，WordPress の記事を執筆した際に生成されるリサイズされた画像のうち，記事に使用されている画像のみを移行の実行対象とし，移行するファイルの転送量を減らすことで，移

行時の時間を短縮を行った．評価は, 提案適用前と提案適用後における, 移行時のファイルの転送を 10 回行い, 実行時間を比較した. 実験環境では, 移行元サーバーから NAS を経由して移行先サーバーへ Web サイトのデータを転送した. 結果として, 提案適用前において, 移行元サーバーから NAS へ移行した際にかかる時間の平均は約 92.6 秒であるのに対して, 提案適用後は約 4.4 秒となり, 約 95.1%削減できた．また, 提案適用前において, NAS から移行先サーバーへ移行した際にかかる時間の平均は約 157.1 秒かであるのに対して, 提案適用後は約 9.5 秒となり, 約 93.9%削減できた．移行する時間を短縮することによって, ダウンタイムが短くなるため, Web サイトに掲示されている項目を閲覧したい人にとっては, 閲覧するまでに待機する時間が短くなる.

参考文献

[1] Cabot, J.: WordPress: A Content Management System to Democratize Publishing, *IEEE Software*, Vol. 35, No. 3, pp. 89–92 (online), DOI: 10.1109/MS.2018.2141016 (2018).

[2] Patel, S. K., Rathod, V. and Parikh, S.: Joomla, Drupal and WordPress - a statistical comparison of open source CMS, *3rd International Conference on Trendz in Information Sciences & Computing (TISC2011)*, pp. 182–187 (online), DOI: 10.1109/TISC.2011.6169111 (2011).

[3] Mallikarjuna, B. and Kumar, M.: Analysis of the Performance, Scalability, Availability, and Security of Cloud Computing in Different Cloud Environments, *2022 International Conference on Futuristic Technologies (INCOFT)*, pp. 1–7 (online), DOI: 10.1109/INCOFT55651.2022.10094446 (2022).

[4] Grassi, G., Teixeira, R., Barakat, C. and Crovella, M.: Leveraging Website Popularity Differences to Identify Performance Anomalies, *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10 (online), DOI: 10.1109/INFOCOM42981.2021.9488832 (2021).

[5] Grassi, G., Teixeira, R., Barakat, C. and Crovella, M.: Leveraging Website Popularity Differences to Identify Performance Anomalies, *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10 (online), DOI: 10.1109/INFOCOM42981.2021.9488832 (2021).

[6] Handa, C., Wadhawan, S., Manocha, A. and Shalu: Enhancing Virtual Machine Migration in Cloud Environments, *2023 1st DMIHER International Conference on Artificial Intelligence in Education and Industry 4.0 (IDICAIEI)*, Vol. 1, pp. 1–6 (online), DOI: 10.1109/IDICAIEI58380.2023.10406495 (2023).

[7] Hu, B., Lei, Z., Lei, Y., Xu, D. and Li, J.: A Time-Series Based Precopy Approach for Live Migration of Virtual Machines, *2011 IEEE 17th International Conference on Parallel and Distributed Systems*, pp. 947–952 (online), DOI: 10.1109/ICPADS.2011.19 (2011).

[8] Grassi, G., Teixeira, R., Barakat, C. and Crovella, M.: Leveraging Website Popularity Differences to Identify Performance Anomalies, *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10 (online), DOI: 10.1109/INFOCOM42981.2021.9488832 (2021).

[9] Maziku, H. and Shetty, S.: Towards a network aware VM migration: Evaluating the cost of VM migration in cloud data centers, *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, pp. 114–119 (online), DOI: 10.1109/CloudNet.2014.6968978 (2014).

[10] Qin, Y., Zhang, L., Xu, F. and Luo, D.: Interference and Topology-Aware VM Live Migrations in Software-Defined Networks, *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 1068–1075 (online), DOI: 10.1109/HPCC/SmartCity/DSS.2019.00152 (2019).