令和 7 年度

修士学位論文

クラウドコンピューティングにおける複数区間のワークロード予測に基づく仮想マシン配置手法

A Virtual Machine Placement Method Based on Multi-Interval Workload Prediction in Cloud Computing

合田 和樹

指導教員　横山 和俊

2026 年 2 月 27 日

高知工科大学大学院 工学研究科 基盤工学専攻

情報学コース

要 旨

クラウドコンピューティングにおける複数区間のワークロード予測に基づく仮想マシン配置手法

合田 和樹

近年，様々な分野でクラウドコンピューティングが活用されている．クラウドコンピューティングの普及に伴い，事業者には SLA（Service Level Agreement）に基づくサービス品質の維持が求められている．一方で，品質維持を目的としたリソースの過剰な増強は，運用コストの増大を招くという課題がある．このトレードオフを解消するため，リソースの仮想化や，ライブマイグレーションなどの技術が存在し，効率的にリソースを物理サーバに配置することが重要である．

仮想マシン配置手法として，機械学習によるワークロード予測結果を利用した手法が提案されている．しかし，これらの多くは短期または長期のいずれか単一の予測期間に基づくものであり，直近の負荷スパイクや中長期的なトレンドを同時に考慮することが困難である．

そこで，本研究では異なる予測期間を持つ複数区間のワークロード予測を統合的に利用した VM 配置手法を提案し，有効性を評価する．

キーワード　クラウドコンピューティング，仮想化，ワークロード予測

Abstract

A Virtual Machine Placement Method Based on Multi-Interval Workload Prediction in Cloud Computing

In recent years, information technology has advanced across various fields, and cloud computing has been widely adopted. With the proliferation of cloud computing, service providers are required to maintain service quality in accordance with Service Level Agreements (SLAs). However, excessive resource provisioning to ensure service quality leads to increased operational costs, which remains a critical challenge. To address this trade-off, technologies such as resource virtualization and live migration have been introduced, and efficient allocation of resources to physical servers has become increasingly important.

As virtual machine (VM) placement strategies, several approaches utilizing workload prediction based on machine learning have been proposed. However, most existing methods rely on either short-term or long-term prediction periods alone, making it difficult to simultaneously account for sudden workload spikes and mid- to long-term workload trends.

Therefore, in this study, we propose a VM placement method that integratively utilizes workload predictions over multiple time horizons with different prediction intervals, and evaluates its effectiveness.

key words Cloud Computing, Virtualization, Workload Prediction

目次

図目次

表目次

第 1 章

はじめに

近年，様々な分野でクラウドコンピューティングが盛んに活用されている．総務省が発表した令和 7 年版情報通信白書によると，世界のパブリッククラウドサービスの売上高は 2024 年に 7,733 億ドルに達すると見込まれており，今後も AI 需要の加速などが重なり，今後もクラウドサービス市場は拡大していくと見込まれている [1]．この需要に対してクラウドサービス事業者は安定したサービスを提供するために，Service Level Agrement(以降 SLA と略す) として稼働率を保証する場合が多い．実際に Microsoft 社が提供する Microsoft Azure では要件によって異なりはするが基本的に 99.9%の稼働率が保証されている [2].

SLA を遵守するためにクラウド事業者は多くの計算資源を保持するが，過剰な資源確保は消費電力の増加やコスト増大を招く一方，過少な資源確保はサービスの中断や SLA 違反のリスクを高める．

そこで，クラウド事業者は，ユーザのサービスを仮想化することによって，物理ホストに複数台の仮想マシンを搭載することで最適化を行っている．また，物理マシンが過負荷になると見込まれる場合には，VM を別のホストへ移動させるライブマイグレーション技術などが用いられる．

Farahnakian[4] らは k-NN での予測を導入することで SLA 違反とマイグレーション回数の削減に成功したが，そこで用いられている予測モデルは主に「短期的な負荷変動」に焦点を当てたものである．しかし，実際のクラウド環境での負荷は「長期的なトレンド」と突発的な「短期的な変動」があり，単一区間の予想期間による予測だけでは，リソースの予測の最適化と，短期的なスパイクへの即応を同時に達せすることは困難である．

そこで本研究では，物理ホストの容量に達する状態を SLA 違反と定義した上で，短期的

な予測と長期的な予測を組み合わせることにより，SLA 違反の低減を目指す VM 配置手法を提案する．

本稿の構成を以下に示す．2 章で関連技術を述べ，3 章で関連研究を示す．4 章では本研究が作成したデータセット・ワークロード・モデルについて説明し，5 章では本研究が想定する環境モデル，6 章では提案手法を述べる．7 章では提案手法の有用性を確かめるための評価方法と評価項目，結果と評価を示し，8 章でまとめと今後の課題を述べる．

第 2 章

関連技術

本章では，本研究の基盤となる技術について述べる．具体的には，計算資源の基盤となるクラウドコンピューティング，それを支える仮想化技術，および時系列データの学習に用いる LSTM について解説する．

2.1　クラウドコンピューティング

クラウドコンピューティングとは，ネットワーク，サーバー，ストレージ，アプリケーション，サービスなどの構成可能な計算リソースを共有し，利便性が高く，オンデマンドなネットワークアクセスを可能にするモデルである．NIST(米国国立標準技術研究所) の定義では，クラウドコンピューティングは以下の 5 つの本質的な特徴を持つとされる [3].

- オンデマンド・セルフサービス
- 幅広いネットワークアクセス
- リソースの共有
- スピーディな拡張性
- 測定可能なサービス

クラウドが浸透する前は，利用者が自前でハードウェアを保有・管理するオンプレミス型が主流であったが，クラウドコンピューティングの普及により，利用者は物理的なインフラストラクチャを意識することなく，必要な時に必要な分だけ計算資源を利用することが可能となった．また，クラウドコンピューティングには種類があり，プラットフォームそのもの

を貸し出す IaaS(Infrastructure as a Service)，OS やミドルウェア以外のものを貸し出す PaaS(Platform as a Service)，メールなどの具体的なサービスを提供する (Softwere sa a Service) などがある．

2.2 仮想化

仮想化（Virtualization）は，物理的なハードウェアリソースを論理的に分割・統合し，OS やアプリケーションに対して仮想的なリソースとして提供する技術である．クラウドコンピューティングを支えるコア技術であり，1 台の物理サーバー上で複数の仮想マシン（Virtual Machine: VM）を稼働させることを可能にする．

また，近年では OS レベルの仮想化である「コンテナ技術」やそれを管理するオーケストレーション技術も普及しているが，いずれの技術も，ハードウェアリソースの利用効率を最大化し，環境の隔離を実現する点で共通している．

2.3 LSTM (Long Short-Term Memory)

LSTM（Long Short-Term Memory）は，RNN（Recurrent Neural Network）を改良したものであり，時系列データや文章などのデータの学習に適したニューラルネットワークモデルである．

従来の単純な RNN では，誤差逆伝播法において勾配が指数関数的に減衰または爆発してしまう「勾配消失問題（Vanishing Gradient Problem）」や「勾配爆発問題」が発生しやすく，時系列といった長期的な依存関係（Long-term dependencies）を学習することが困難であった．そこで LSTM は，セル状態（Cell State）と呼ばれる情報の通り道と，ゲート（Gate）と呼ばれる情報の取捨選択を行う機構を導入することで，この問題を解決している．LSTM ブロックは主に，忘却ゲート（Forget Gate），入力ゲート（Input Gate），出力ゲート（Output Gate）の 3 つのゲートで構成される．

このように，LSTM はゲート機構を通じて長期的な情報を保持することが可能であり，音

2.3 LSTM (Long Short-Term Memory)

声認識，自然言語処理，異常検知など多岐にわたる分野で応用されている．本研究においても，時系列データの変動パターンを捉えるために LSTM を採用する．

第 3 章

関連研究

3.1 予測を用いた仮想マシン管理手法に関する研究

負荷の変動を事前に予測し，仮想マシンのリソースを割り当てる手法は様々なものが検討されている．

文献 [4] では，現在のリソース利用率のみに基づく管理は不必要なマイグレーションや SLA 違反を招くと指摘し，k 近傍回帰（k-NNR）UP-BFD アルゴリズムを提案した．

文献 [5] では，統計的な ARIMA モデルを用いて将来の要求負荷を予測し，動的なプロビジョニングを行う手法を開発した．彼らの手法はウェブサーバーの要求履歴に基づき，サービス品質の影響を最小限に抑えつつリソース効率を高めることに成功している．

文献 [6] では，予測分析と機械学習を組み合わせた動的な VM 配置手法を提案した．文献では 3 重指数平滑法でリソース需要を予測し，SVM（サポートベクターマシン）を用いて最適な配置を決定することで，静的な配置手法と比較してコスト削減と応答時間の改善を実現している．

しかしながら，これらの手法の多くは予測期間が単一の時間尺度に限定されている．実際のクラウド環境では短期的なスパイクと長期的なトレンドが混在しており，単一の予測期間では十分な対応が困難な場合がある．そこで本研究では，短期的な予測 (直前 60 区間から 12 区間の予測) と長期的な予測 (直前 200 区間から 60 区間) という複数区間の予測情報を組み合わせることで，SLA 違反をより効果的に低減する VM 配置・再配置手法を提案する．

3.2　負荷予測モデルの精度向上に関する研究

より複雑な負荷特性を捉えるため，ディープラーニングを用いたモデル精度の向上も検討されている．

文献 [7] では，特徴抽出に優れた CNN と時系列情報のモデル化に長けた LSTM を組み合わせたハイブリッド CNN-LSTM モデルを提案した．このモデルは，単一の CNN や LSTM と比較して，周期的，成長的，あるいは予測不可能な負荷パターンに対して高い予測精度を示すことが実証されている．

文献 [8] では，モデル自体の構造だけでなく，予測プロセスの効率化に焦点を当てた MSFS（Multi-time Series Forecasting System）を提案している．この研究では，多数の VM が存在する環境において，類似した利用パターンを持つ VM をグループ化し，グループごとに専門の予測モデルを構築することで，スケーラビリティと精度の両立を図っている．

このように予測モデル自体の高度化については多くの研究が存在するが，本研究の主目的は予測精度の極限的な向上ではなく，予測情報をいかに VM の配置・再配置戦略に活用するかという点にある．そのため，本研究では予測モデルとして標準的な LSTM を用い，実用的な予測能力を確保しつつ，提案する多角的な配置アルゴリズムの有効性検証に焦点を当てる．

第 4 章

データセット・ワークロード・モデル作成

本研究では，クラウド環境における VM の配置およびライブマイグレーション戦略を評価するため，実運用環境に近い CPU 負荷特性を持つ合成データセットおよび動的ワークロードを生成する．本研究では，VM の配置およびライブマイグレーション戦略を評価するため，実運用環境に近い CPU 負荷特性を持つ合成データセットおよび動的ワークロードを生成した．

4.1 データセット生成

4.1.1 概要

本研究では，VM の役割ごとに異なる負荷特性を再現するため，合成的に生成された CPU 使用率の時系列データセットを構築した．生成される各時系列データは，5 分間隔でサンプリングされた 1 週間分（2016 点）の CPU 使用率を表す．

実運用環境における VM の CPU 使用率には，単純な周期変動だけでなく，ノイズ，スパイク，バースト，トレンド変化など多様な非定常性が含まれる．そこで本研究では，確率的にそれらのノイズを付与することで，予測が容易すぎず，かつ完全にランダムでもない負荷波形を生成した．

表 4.1 にデータセット生成の基本パラメータを示す．

表 4.1　データセット生成パラメータ

パラメータ	値
サンプリング間隔	5 分
1 日あたりのデータ点数	288 点
時系列長（1VM）	2016 点（7 日分）
学習用波形数	2800（単一 2500 ＋混合 300）
評価用波形数	3500（単一 3000 ＋混合 500）

4.1.2　VM プロファイル

本研究では，クラウド環境における典型的な VM の役割として，Web Server，Batch Server，Database，Idle の 4 種類のプロファイルを定義した．また，予測が難しい波形を生成するためにいくつかのプロファイルを混合させたプロファイルも定義した．各プロファイルの負荷特性を表 4.2 に示す．

表 4.2　VM プロファイル別の負荷特性

プロファイル	負荷範囲	特徴
Web Server	低〜高（2–90）	日次周期，週末減少，サブサイクル
Batch Server	低〜高（0.5–70）	定期的なバースト，不規則なスパイク
Database	中（5–50）	安定したベースライン，小規模な周期変動
Idle	極低（0–10）	ほぼ一定の低負荷
Mixed	可変	複数プロファイルの重み付き合成

Web Server は，ユーザアクセスに応じた日次の周期的変動を持ち，週末には負荷が減少するなど特性を再現した．主周期として 288 点（1 日）または 144 点（半日）を確率的に選択し，さらに 1〜8 時間程度周期も追加した．

Batch Server は，定期実行されるバッチ処理を想定し，一定周期でバースト的な負荷上昇が発生する波形を生成した．バーストの形状として，矩形波，山形，スパイクの 3 種類を

用意し，確率的に選択することで多様性を持たせた．

Database は，比較的安定した負荷を持つ VM を想定し，一定のベースライン負荷に小規模な周期変動を加えた波形を生成した．

Idle は，待機状態の VM を想定し，極めて低い負荷が継続する波形を生成した．

Mixed は，複数の基本波形を重み付き線形結合することで生成した．

それぞれのプロファイルにて生成された波形の代表例をそれぞれ図 4.1，図 4.2，図 4.3，図 4.4，図 4.5 に示す．

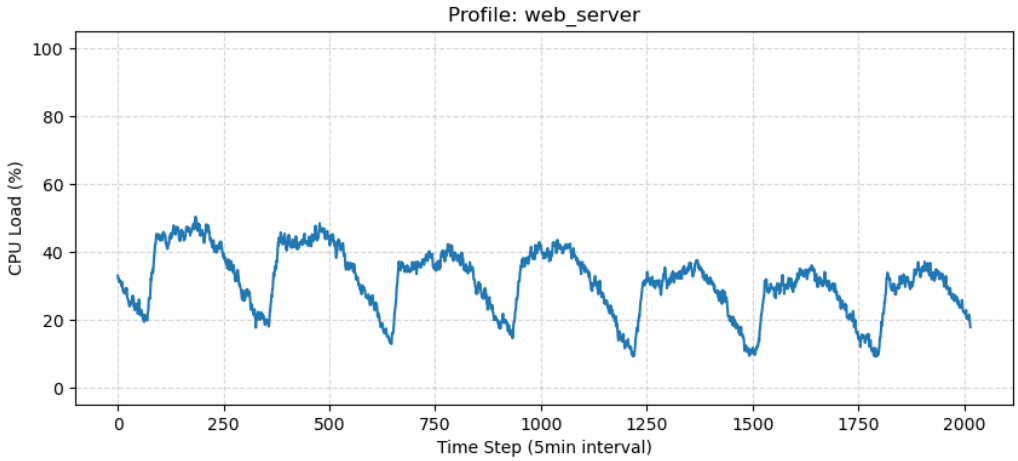

図 4.1　Web プロファイル

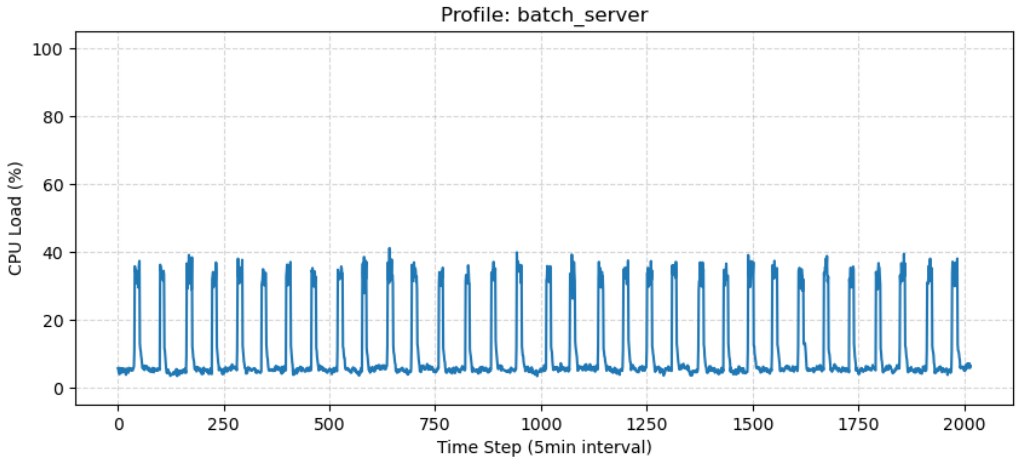

図 4.2　Batch プロファイル

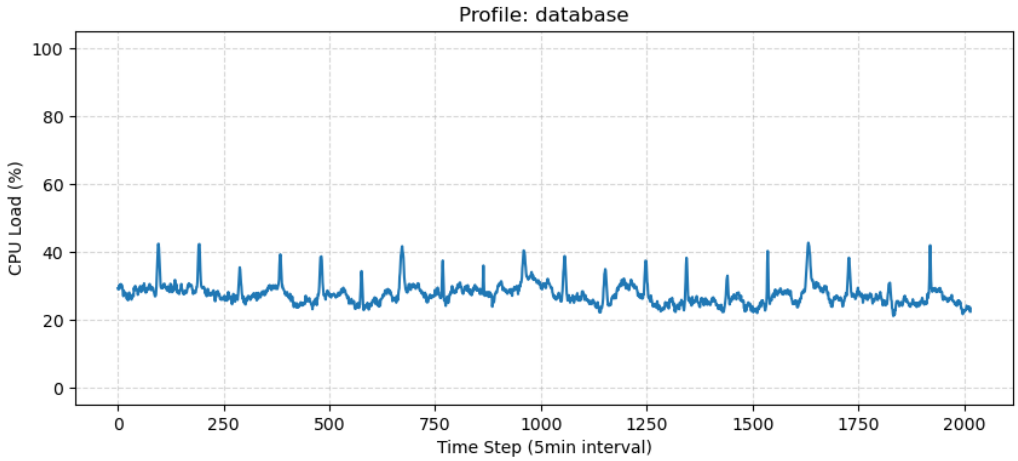

図 4.3　Database プロファイル

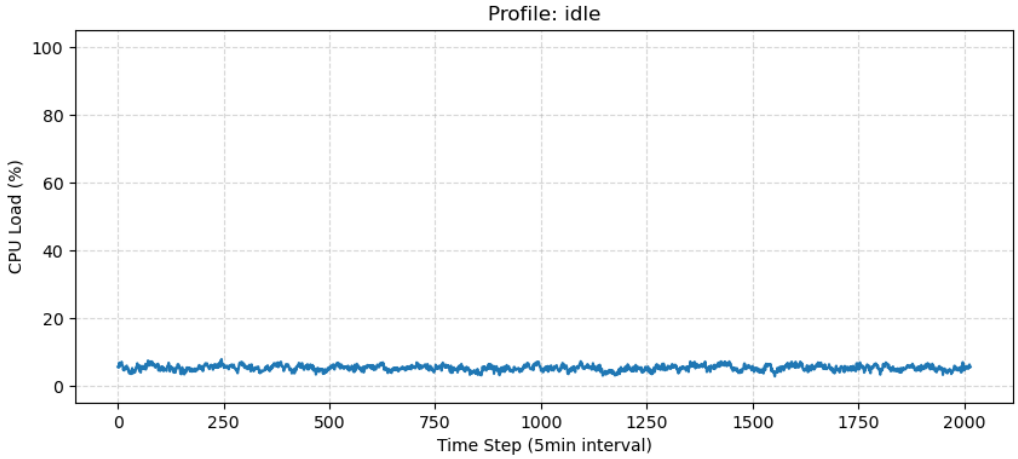

図 4.4　Idle プロファイル

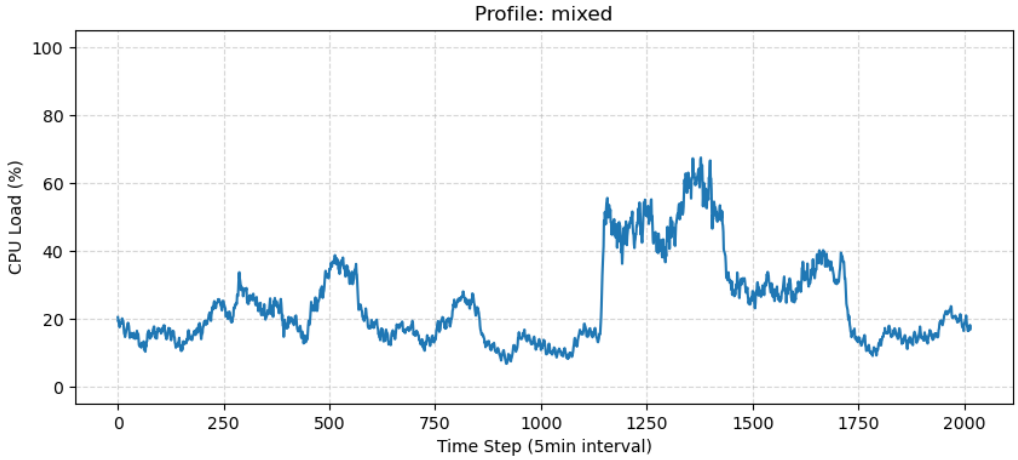

図 4.5　Mixed プロファイル

4.2　ワークロード生成

4.2.1　概要

静的な負荷波形のみならず，VM の到着および終了を含む動的なワークロードを生成した．表 4.3 にワークロード生成の基本パラメータを示す.

表 4.3　ワークロード生成パラメータ

パラメータ	値
シミュレーション期間	7 日間
時間刻み	5 分
ホスト数	500 台
ホスト CPU 容量	110%

4.2.2　VM 寿命モデル

クラウド環境では，短命な VM が多数を占める一方で，長期間稼働する VM が一定数存在するロングテール分布を持つことが知られている [9].

そこで本研究では，ランダムな寿命ではなく，短期稼働 VM と長期稼働 VM を確率的に混在させ，短期的な予測と長期的な予測が可能な寿命を設定する．また本研究において，長期的な分布だけでなく短期的な分布も考慮するのは，比較的長期的な VM しか存在しない環境では，過負荷などのイベントが発生した場合でも，同じ VM により起因することや，評価に偏りが生じる可能性があると考えた．そこで，シミュレーション期間内に多様な状況を再現できるようにした．具体的には，各 VM 到着時に確率 r で長期，確率 $(1-r)$ で短期を選択し，対応する対数正規分布から寿命をサンプリングする．表 4.4 にプロファイル別の寿命パラメータを示す.

混合寿命の期待値 $E[W]$ は以下のように計算される：

$$E[W] = (1-r) \cdot \mu_{\text{short}} + r \cdot \mu_{\text{long}} \tag{4.1}$$

表 4.4　VM 寿命モデルのパラメータ

プロファイル	長期確率 r	短期平均（時間）	長期平均（時間）
Web サーバ	40%	6	48
Batch サーバ	40%	6	48
Database サーバ	40%	6	60
Idle サーバ	10%	2	12
Mixed サーバ	40%	6	24

ここで，μ_{short} および μ_{long} はそれぞれ短期・長期の平均寿命である．

4.2.3　VM 到着モデル

それぞれ特徴のあるワークロードを作成するにあたり，データセットにより負荷強度が異なる．具体的には，Batch のような低負荷から高負荷になる VM が多く存在する環境と Web のような比較的高負荷で周期的な VM が多く存在する環境で，同一の VM 到着率を用いてしまうと，極端に過密・過疎なワークロードが生成される．そこで，異なるワークロード間でも比較的公平な比較を行えるように，ワークロード強度を考慮して VM の到着率を決定する．

具体的な手順として，まず目標稼働 VM 数 L を以下のように決定する：

$$L = \frac{N_{\text{host}} \cdot C_{\text{host}} \cdot u_{\text{target}}}{\bar{c}} \tag{4.2}$$

ここで，N_{host} はホスト数，C_{host} はホストの CPU 容量，u_{target} は目標利用率，$\bar{c}$ は平均 VM 負荷である．

目標利用率 u_{target} は，プロファイル構成のバースト特性を考慮して以下のように決定する：

$$u_{\text{target}} = \frac{s}{\sum_p w_p \cdot b_p} \tag{4.3}$$

ここで，s は安全係数，w_p はプロファイル p の構成比，b_p はプロファイル p のバースト率（P95/Mean）である．

次に，Little の法則に基づき到着率 λ を導出する：

$$\lambda = \frac{L}{E[W]} \tag{4.4}$$

VM の到着はポアソン過程に従い，各時間刻みにおいて期待値 $\lambda \cdot \Delta t$ のポアソン分布から到着数をサンプリングする．この際，Idle ワークロードのみ VM の到着数が現実的に離れすぎた個数や，誤った分布になることから，到着率を一定割合にすることで調整を行った．

したがって，本手法では各 VM のプロファイルの激しさによって，目標稼働 VM 数が異なり，適切な負荷状態が維持できると考えた．

4.2.4　評価ワークロード

提案手法の汎用性を評価するため，プロファイル構成比が異なる 5 種類のワークロードを用意した．

表 4.5 に各ワークロードの構成比を示す．

表 4.5　評価ワークロードにおけるワークロード構成比

ワークロード	Web	Batch	DB	Idle	Mixed
Standard	0.2	0.2	0.2	0.2	0.2
Batch	0.1	0.6	0.1	0.1	0.1
Web	0.6	0.1	0.1	0.1	0.1
DB	0.1	0.1	0.6	0.1	0.1
Idle	0.1	0.1	0.1	0.6	0.1

各ワークロードについて，5 種類の乱数シードを用いてワークロードを生成し，結果の統計的安定性を確保した．

図 4.6 に各ワークロードにおける稼働 VM 数の時間推移を示す．シミュレーション開始時は VM が存在しない状態から始まり，VM の到着と終了を繰り返しながら 3〜4 日目以降に定常状態へ収束している．

Idle ワークロードが最も多くの VM を抱えているのは，Idle VM の平均負荷が極めて低

いことに加えて寿命が短いことから，同じシステム負荷を達成するにはより多くの VM が必要となるためである．一方，Web，Batch，DB ワークロードは平均負荷が高い VM が多いため，相対的に少ない VM 数で同等のシステム負荷に達する．

本研究のワークロード生成手法では，プロファイル構成が異なりホストに搭載されている VM 種別の割合が異なる場合でも，適切な負荷強度が維持されるよう到着率を調整した．

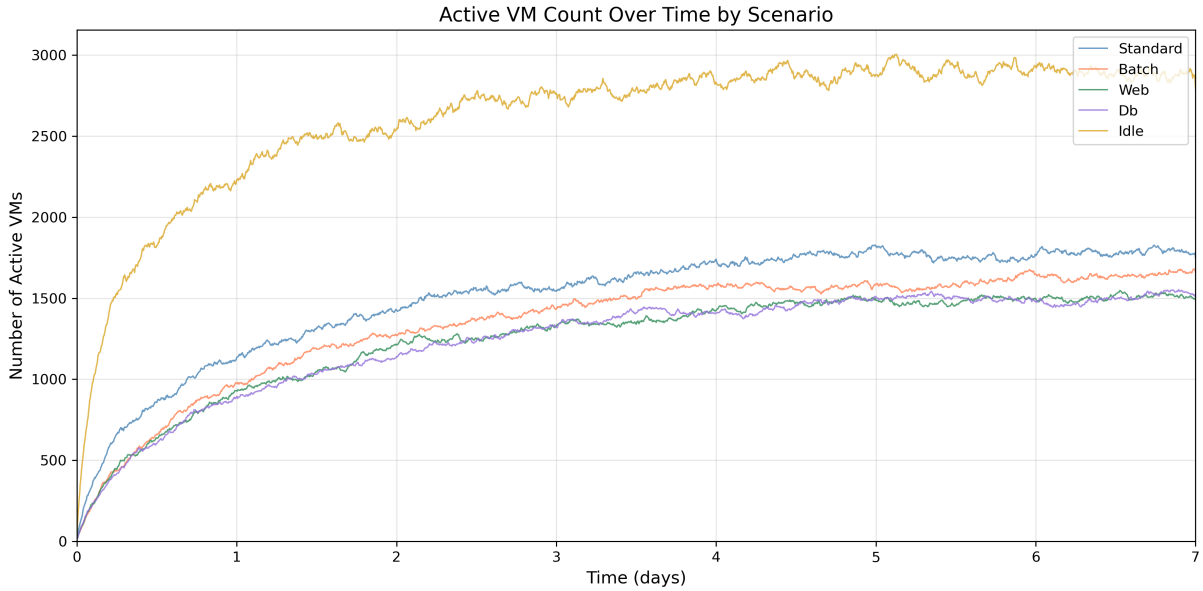

図 4.6　各ワークロードにおける稼働 VM 数の時間推移

4.3　予測モデル生成

本研究では，動的 VM 配置手法への入力となる将来負荷を予測するため，LSTM モデルを採用した．VM 配置問題においては，予測精度だけでなく，システム全体の計算リソースを圧迫しない軽量なモデル構成が求められる．そのため，一般的に用いられる 2 層の LSTM ネットワークを予測モデルとして採用した．用いたパラメータを表 4.6，ネットワーク構造を表 4.7 に記述する．

表 4.6　学習パラメータおよびデータセット設定

項目	設定値
入力系列長 (T_{past})	60 区間 または 200 区間
出力系列長 (T_{future})	12 区間 または 60 区間
バッチサイズ	256
学習率	0.001（Adam）
損失関数	平均二乗誤差（MSE）
ドロップアウト率	0.2
L2 正則化係数	1×10^{-4}
最大エポック数	100（Early Stopping 使用）

表 4.7　予測モデルのネットワーク構造 (60 区間から 12 区間の場合)

層の種類	ユニット数	活性化関数	出力形状
入力層	–	–	(None, 60, 1)
LSTM 層（第 1 層）	256	tanh	(None, 60, 256)
ドロップアウト層	–	–	(None, 60, 256)
LSTM 層（第 2 層）	256	tanh	(None, 256)
ドロップアウト層	–	–	(None, 256)
全結合層（Dense）	64	ReLU	(None, 64)
ドロップアウト層	–	–	(None, 64)
出力層（Dense）	12	線形（Linear）	(None, 12)

第 5 章

システムモデル

本研究で想定するクラウド環境のモデルについて説明する.

5.1　シミュレーションシステム

本実験では，クラウド環境の物理ホスト (PM) と VM の挙動を模擬シミュレーションを行う．シミュレーションは区間単位 t で進行し，1 区間は実時間の M 分（本実験では 5 分）に相当する.

5.1.1　システムモデル定義

システムを構成する要素を示す.

- 物理ホスト集合 H: データセンター内のホスト群．各ホスト h_j は CPU 容量 C_{host}（110）を持つ.
- 仮想マシン集合 V: 時間経過に伴い到着・終了する VM．各 VM v_i は実際のワークロード・トレースに基づいた CPU 要求量 $u_i(t)$ を持つ.
- 予測モデル: 各 VM に対し，直近の負荷履歴から将来の負荷を予測する LSTM モデル（短期予測 P_{short} および長期予測 P_{long}）を行う.

次にシステムの構成要素がそれぞれ保有する情報を以下の表に示す. ここで各 ID はホスト，VM が一意に決まるユニークな識別子である.

表 5.1　ホストが保有する情報

情報	説明
ホスト ID	ホストを一意に識別する ID
CPU 容量	ホストが提供可能な CPU 容量
配置 VM 集合	当該ホスト上で稼働する VM の集合
現在 CPU 負荷	配置 VM の実測 CPU 使用率の合計
短期予測負荷	短期予測に基づく将来 CPU 負荷
長期予測負荷	長期予測に基づく将来 CPU 負荷

表 5.2　VM が保有する情報

情報	説明
VM ID	VM を一意に識別する ID
VM タイプ	ワークロード特性を表す分類
割当ホスト	配置先ホスト
寿命	VM が稼働する残り区間数
CPU 使用履歴	過去の CPU 使用率の時系列データ
短期予測負荷	短期予測による将来 CPU 使用率
長期予測負荷	長期予測による将来 CPU 使用率
最終マイグレーション時刻	直近でマイグレーションが発生した時刻

5.1.2　シミュレーション手順

各区間 t における処理順序は以下の通りである．

1. VM の到着と初期配置: 時刻 t に発生する新規 VM 群 V_{new} を読み込む．各 $v \in V_{new}$ に対し，指定された配置戦略に基づき，ホスト h_j を決定し割り当てる．
2. 負荷計測と状態更新: 稼働中の全 VM v_i について，現在の CPU 使用率 $u_i(t)$ をワーク

ロードデータから取得し，履歴バッファ D_i に記録する．

$$D_i^{(t)} = D_i^{(t-1)} \cup \{u_i(t)\}$$

3. SLA 違反の検出: 各ホスト h_j について，現在の実負荷 $L_j(t)$ を算出する．

$$L_j(t) = \sum_{v_i \in h_j} u_i(t)$$

もし $L_j(t) > C_{host}$ である場合，SLA 違反（Overload）として記録する．この判定はマイグレーション実行前に行われ，現時点での配置の適否を評価する．

4. 将来負荷の予測: 履歴 D_i の長さが所定の入力長を満たす VM に対し，学習済み LSTM モデルを用いて将来の負荷を予測する．
   - 短期予測: K_{short} 区間（例: 12 区間=60 分）
   - 長期予測: K_{long} 区間（例: 60 区間=300 分）

   これにより得られた予測値列に基づき，各 VM の変動性（Volatility）や信頼性を評価し，ホストごとの将来負荷マップを更新する．
5. 動的再配置: 予測された将来負荷および現在のホスト状態に基づき，過負荷の解消を行う．この際に，移動した VM の個数分だけマイグレーション回数としてカウントする．
6. 時間進行と VM 離脱: シミュレーション時刻を $t \leftarrow t+1$ に進める．各 VM の残りライフタイムを減算し，ライフタイムが 0 になった，またはワークロードデータが終了した VM をシステムから削除し，リソースを解放する．

第 6 章

提案手法

本研究では，複数のワークロード予測を組み合わせた VM 配置手法を提案する．本手法の構築にあたり，堀口ら [10] によって提案された，エッジコンピューティング環境におけるタスクの貸し借り手法を参考にし，これをクラウドデータセンターにおける VM 配置問題へと最適化した．具体的には，将来の負荷予測に基づき，SLA 違反のリスクが高いホストから余裕のあるホストへと事前に VM を再配置することで，リソースの安定供給を目指す．

本研究では，提案する再配置手法を **MP-SLP**（Mixed Prediction Short-term and Long-term Placement）と呼ぶ．また，MP-SLP の有効性を検証するためのアブレーションスタディとして，移動先選定に短期予測のみを用いる比較手法を **MP-SOP**（Mixed Prediction Short-term Only Placement）と呼ぶ．

6.1 制約条件

VM を移動した後，12 区間（シミュレーション上で 1 時間）の間，当該 VM の再移動を禁止する．

本制約は，短時間に同一 VM が繰り返し移動されるピンポン現象を抑制することを目的としている．負荷予測に基づく再配置では，予測誤差や突発的な負荷変動により，一度移動した VM が直後に再び移動対象となる場合がある．このような頻繁な再移動は，マイグレーション回数の増加やシステム全体の不安定化を招き，結果として運用コストや SLA 違反の増加につながる可能性がある．

そこで，本研究では同定期再配置において，VM が新たなホストに配置された後，予測上

は SLA 違反を起こさないと判定されていることから，一定時間（12 区間）その配置を維持することで，移動効果を観測するための猶予期間を設ける．この猶予期間により，一時的な負荷変動に起因する過剰な再配置を抑制し，より安定したリソース管理を実現する．

6.2　予測結果の合成

各 VM は履歴に基づき負荷予測を行い，履歴不足時は直近値を用いる．履歴が十分な場合は短期・長期予測を行う．本手法の特徴として，複数区間の予測が存在する重複部分では，各区間の大きい値を選択する．本研究での予測ウィンドウとして以下の二つの期間の予測を行う．

- 短期予測: 直近 60 区間から 12 区間の予測
- 長期予測: 直近 200 区間から 60 区間の予測

具体例

予測結果の活用について具体例を示す．ここでは，説明を簡潔にするため，短期予測期間を $t_s = 3$，長期予測期間を $t_l = 6$ とする．

	$t+1$	$t+2$	$t+3$	$t+4$	$t+5$	$t+6$
短期予測	10	20	10	–	–	–
長期予測	20	10	30	20	50	55

このとき，各時刻で最大値を取ることで，合成波形は

$$(20,\ 20,\ 30,\ 20,\ 50,\ 55) \tag{6.1}$$

となる．

6.3　配置アルゴリズム

本研究の配置アルゴリズムは，サーバ間の資源の貸し借り関係に基づいて負荷集中を解消する関連研究 [10] を参考にして設計している．具体的には，負荷予測結果から各ホストの余剰資源と不足資源を算出し，不足量が大きいホストから順に処理を行うという基本的な枠組みを踏襲している．また，予測誤差を考慮するために安全係数（マージン）を導入し，将来的な過負荷を事前に回避する点も既存研究と共通している．

一方で，本研究では配置対象が連続的な資源量ではなく，VM という単位である点が関連研究と大きく異なる．関連研究では，不足資源量が最大のサーバと余剰資源量が最大のサーバが見つかり次第，そのサーバ間で資源の貸し借りを即座に行う．これは，資源量が連続的に調整可能であり，貸し借りによる影響が予測上ほぼ一意に定まることを前提としている．

これに対し，VM 配置では，同一の不足量を解消できる場合であっても，どのホストに VM を配置するかによって，将来の負荷時系列が大きく異なる．そのため，本研究では，不足が大きいホスト・VM から処理するという順序は維持しつつ，余剰が見つかり次第即座に配置を決定するのではなく，移動先となり得るすべての候補ホストを評価した上で，ホストを選択する方式を採用した．

以上より，本研究の配置アルゴリズムは，関連研究における「不足資源量が大きいホストから順に処理する」という基本思想を維持しつつ，VM 配置問題および複数時系列予測を扱うために，配置判断部分を拡張している．

6.3.1　初期配置手法

初期配置では，新規に到着した VM を対象として，短期の CPU 負荷予測に基づき配置先ホストを決定する．本手法の目的は，単に現在の負荷が低いホストを選択するのではなく，将来の負荷変動を考慮し，配置後も余裕を維持できるホストを優先的に選択することにある．

具体的には，各ホスト h_i が有する CPU 容量を Cap_i，短期予測により得られる将来時

刻 $t+k$ における負荷予測値を $L_i(k)$ とする．このとき，時刻 $t+k$ における予測上の空き CPU 容量 $C_i(k)$ を次式で定義する．

$$C_i(k) = Cap_i - L_i(k) \tag{6.2}$$

配置判断においては，近い将来の負荷変動ほど重要であると考え，時間減衰を表す重み関数 $w(k)$ を導入する．本研究では，時間減衰係数として初期値 0.9 を用い，重み関数を次式で定義する．

$$w(k) = 0.9^{k-1} \tag{6.3}$$

この重み付けにより，直近の予測負荷が配置判断に強く反映され，時間が離れるにつれてその影響は緩やかに減衰する．以上を踏まえ，ホスト h_i に対する空き容量スコア S_i を次式で定義する．

$$S_i = \sum_{k=1}^{K} w(k)\, C_i(k) \tag{6.4}$$

新規 VM は，この空き容量スコア S_i が最大となるホストに配置される．これにより，直近の瞬間的な余裕だけでなく，将来の負荷上昇を見越した初期配置が可能になると考える．

具体例

初期配置アルゴリズムの例を示す．ここでは，CPU 容量がすべて $Cap_i = 100$ の 2 台のホスト h_1, h_2 を考える．短期予測（$t = 3$）により得られた将来負荷が表 6.1 の通りであるとする．

表 6.1　短期予測による将来負荷の例

ホスト	$L_i(1)$	$L_i(2)$	$L_i(3)$
h_1	60	70	80
h_2	65	65	65

このとき，予測上の空き CPU 容量は h_1 では $[40, 30, 20]$，h_2 では $[35, 35, 35]$ となる．時間減衰係数 $w(k) = [1.0, 0.9, 0.81]$ を用いて空き容量スコアを計算すると，

$$S_1 = 1.0 \times 40 + 0.9 \times 30 + 0.81 \times 20 = 83.2 \tag{6.5}$$

$$S_2 = 1.0 \times 35 + 0.9 \times 35 + 0.81 \times 35 = 94.85 \tag{6.6}$$

となる．この結果，$S_2 > S_1$ であるため，新規 VM はホスト h_2 に配置される．

6.3.2　再配置手法

本手法では，ホストおよび VM の短期および長期の CPU 負荷予測という複数の時系列情報を統合的に用い，過負荷状態にあるホストから余裕のあるホストへ VM の再配置を行う．本研究の主題は，単一時点や単一予測に依存するのではなく，異なる時間スケールの予測時系列を組み合わせて配置判断を行う点にある．

パラメータ設定

本手法で使用する主要なパラメータを表 6.2 に示す．

表 6.2　再配置手法のパラメータ

パラメータ	値	説明
t_s	12	短期予測期間（ステップ数）
t_l	60	長期予測期間（ステップ数）
α	1.1	安全係数（予測誤差を考慮したマージン）
T_{cool}	12	クールダウン期間（ステップ数）
$N_{\max}$	$5 \times \lvert\mathcal{H}\rvert$	最大反復回数（$\lvert\mathcal{H}\rvert$ はホスト数）

過負荷ホストの判定

各ホスト h_i について，短期予測により得られるピーク負荷と現在負荷の最大値を L_i^{peak} とし，安全係数 α を用いて余剰容量 R_i を次式で定義する．

$$L_i^{\text{peak}} = \max\left(\max_{t\in[1,t_s]} L_i^{\text{short}}(t),\ L_i^{\text{current}}\right) \tag{6.7}$$

$$R_i = Cap_i - \alpha \cdot L_i^{\text{peak}} \tag{6.8}$$

$R_i < 0$ となるホストを過負荷ホスト集合 $\mathcal{B}$（borrowers）として分類し，$R_i \geq 0$ となるホストを移動先候補集合 $\mathcal{L}$（lenders）として分類する．$\mathcal{B}$ 内のホストは不足量（$-R_i$）の降順にソートされ，最も不足量が大きいホストから優先的に処理される．

移動 VM の選定

過負荷ホスト h_s 上の VM の中から，以下の条件を満たす VM を候補として抽出する．

- 当該再配置周期内で既に移動または移動失敗として記録されていない
- 直近のマイグレーションからクールダウン期間 T_{cool} が経過している

候補 VM の中から，短期予測区間におけるピーク負荷が最大となる VM を移動対象として選定する．これにより，直近の過負荷要因となる VM を優先的に移動させる．

移動先ホストの時系列評価

移動先ホストの候補は $\mathcal{L}$ に属するホスト（ただし移動元ホストを除く）とし，当該 VM を配置した場合の短期および長期の負荷時系列を評価する．評価結果に基づき，各候補ホストを以下の 4 段階に分類する．

- **Score 0**：短期予測時系列において，VM 追加後の合計負荷に安全係数を乗じた値が容

量を超過するため，配置不可.

- **Score 1**：短期予測では容量制約を満たすが，長期予測時系列において容量超過が予測される.
- **Score 2**：短期・長期の両予測時系列において容量制約を満たし，かつ長期予測データが十分に存在する.（ホストに搭載されている VM の過半数が長期予測を持っている場合）
- **Score 3**：短期・長期ともに容量制約は満たすが，長期予測に用いる時系列データが一部不足しており，長期的な安全性を完全には保証できない.

移動先ホストの選択方針

本手法では，短期および長期の時系列の両方において安全と判断できる Score 2 を最優先で選択する．Score 2 のホストが複数存在する場合は，短期予測における余裕容量（$Cap_d - \max_t(L_d^{\mathrm{short}}(t) + m_t) \cdot \alpha$）が最大のホストを選択する．余裕容量が同一の場合は，長期予測における余裕容量，さらに現時点での余剰容量 R_d の順で比較する.

Score 2 が存在しない場合には，Score 1 および Score 3 のホストを統合し，短期予測における余裕容量が最大のホストを選択する．余裕容量が同一の場合は Score 3 を Score 1 より優先する．これは，Score 3 は長期予測データが部分的に欠落しているものの観測可能な範囲では容量超過が発生していない状態を表す一方，Score 1 は長期予測時系列において明示的な容量超過が予測されるためである．本手法では，時系列情報が不完全であっても安全側の挙動を示す配置を，将来的な過負荷が明確に予測される配置よりも優先することで，過度に悲観的な再配置判断を回避する.

再配置のアルゴリズム

本手法の再配置処理は，単一の過負荷判定に基づく一括的な移動ではなく，VM 単位の移動を反復的に行いながら，負荷状態を逐次再評価する単純な貪欲法（Greedy approach）を

採用する．以下に，1 シミュレーションステップ（1 回のスケジュール周期）における再配置手順を示す．

1. **初期化**：当該ステップにおける処理済み VM 集合 $\mathcal{M} = \emptyset$，解決不能ホスト集合 $\mathcal{U} = \emptyset$ とする．
2. **ホストの分類とソート**：クラスタ内の全ホストについて短期予測と現在負荷に基づく余剰容量 R を算出し，$\mathcal{U}$ に含まれない過負荷ホストを不足量の降順にソートして $\mathcal{B}$ に，非負となるホストを余剰容量の降順にソートして $\mathcal{L}$ に分類する．$\mathcal{B}$ または $\mathcal{L}$ が空の場合，処理を終了する．
3. **移動元ホストの選択**：$\mathcal{B}$ の先頭（最も不足量が大きい）ホスト h_s を選択する．
4. **移動 VM の決定**：h_s 上の VM のうち，$\mathcal{M}$ に含まれず，かつクールダウン期間 T_{cool} を経過している VM の中から，統合予測時系列に基づく短期予測区間でのピーク負荷が最大の VM v_m を選択する．該当する VM が存在しない場合，h_s を解決不能ホストとして $\mathcal{U}$ に追加し，手順 2 へ戻る．
5. **移動先ホストの探索**：v_m を $\mathcal{L}$ 内の各ホスト h_d に仮想的に配置し，前述のスコアリング手法に基づいて評価する．最も条件の良い（Score 2 優先，次点で余裕容量最大）ホスト h_d を選択する．
6. **配置の実行**：
   - 適切な h_d が見つかった場合（配置成功）：v_m を h_s から h_d へマイグレーションし，移動回数を 1 増加させる．
   - 適切な h_d が見つからなかった場合（配置失敗）：マイグレーションを見送る．
7. **状態の更新**：配置の成否にかかわらず，評価を終えた v_m を処理済み VM 集合 $\mathcal{M}$ に追加する．一度評価された VM は，当該再配置周期（一連の反復処理全体）において再試行（再評価）の対象としない．これにより，無限ループを防止する．
8. 手順 2 から再度繰り返す．

具体例

再配置手法の具体例を示す．ここでは，説明を簡潔にするため，短期予測期間を $t_s = 3$, 長期予測期間を $t_l = 6$ とし，CPU 容量がすべて $Cap = 100$，安全係数 $\alpha = 1.1$ の 3 台のホスト h_1, h_2, h_3 を考える．各ホストには表 6.3 に示す VM が配置されているものとする．

表 6.3　各ホストに配置されている VM

ホスト	VM 数	配置 VM
h_1	3	v_1, v_2, v_3
h_2	2	v_4, v_5
h_3	2	v_6, v_7

■過負荷ホストの判定　各ホストの短期予測負荷を表 6.4 に示す.

表 6.4　各ホストの短期予測負荷

ホスト	$L_i(1)$	$L_i(2)$	$L_i(3)$	L_i^{peak}
h_1	85	95	90	95
h_2	50	55	60	60
h_3	45	50	55	55

各ホストの余剰容量 $R_i = Cap - \alpha \cdot L_i^{\text{peak}}$ を計算すると，

$$R_1 = 100 - 1.1 \times 95 = -4.5 \tag{6.9}$$

$$R_2 = 100 - 1.1 \times 60 = 34.0 \tag{6.10}$$

$$R_3 = 100 - 1.1 \times 55 = 39.5 \tag{6.11}$$

となる．$R_1 < 0$ であるため，ホスト h_1 は過負荷状態と判定され $\mathcal{B} = \{h_1\}$ となる．ホスト h_2, h_3 は $R_i > 0$ であり，$\mathcal{L} = \{h_3, h_2\}$（余剰容量の降順）となり，$h_3$ から解決を行う．

■移動 VM の選定　過負荷ホスト h_1 上の各 VM について，短期予測と長期予測を統合した負荷時系列を評価する．表 6.5 に各 VM の予測負荷を示す．なお，短期予測と長期予測の両方が存在する期間（$t = 1$–3）においては，各時刻で両者の最大値を採用することで，より保守的な負荷見積もりを行う．

表 6.5　ホスト h_1 上の各 VM の予測負荷

VM	予測種別	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	短期ピーク
v_1	短期予測	25	30	28	–	–	–	30
	長期予測	22	25	20	18	15	12	
	統合後	25	30	28	18	15	12	
v_2	短期予測	30	35	32	–	–	–	**38**
	長期予測	28	33	**38**	25	22	20	
	統合後	30	35	**38**	25	22	20	
v_3	短期予測のみ	30	30	30	–	–	–	30

v_2 においては，$t = 3$ で長期予測（38）が短期予測（32）を上回るため，統合後の値として 38 が採用される．この結果，統合後の短期ピーク負荷が最大となる v_2（ピーク値 38）を移動候補として選定する．

■移動先ホストの評価　VM v_2 を各候補ホストに配置した場合の負荷時系列を評価する．表 6.6 に，各候補ホストの現在の負荷予測と，v_2 追加後の合計負荷を示す．

各ホストの評価結果は以下の通りである．

- **ホスト** h_2：短期予測において $t = 3$ で合計負荷が $107.8 > 100$ となり，容量超過が発生する．したがって，**Score 0**（配置不可）と判定される．
- **ホスト** h_3：短期予測において $t = 3$ で合計負荷が $102.3 > 100$ となり，容量超過が発生する．したがって，**Score 0**（配置不可）と判定される．

表 6.6　移動先候補ホストの評価

ホスト		短期（$t = 1$–3）			長期（$t = 4$–6）		
h_2	ホスト負荷	50	55	60	58	55	50
	v_2 統合後	30	35	38	25	22	20
	合計 $\times\alpha$	88.0	99.0	**107.8**	91.3	84.7	77.0
h_3	ホスト負荷	45	50	55	–	–	–
	v_2 統合後	30	35	38	25	22	20
	合計 $\times\alpha$	82.5	93.5	102.3	–	–	–

■代替 VM の選定　両候補ホストがいずれも Score 0 となったため，v_2 の移動は失敗となる．v_2 を移動失敗として $\mathcal{M}$ に記録し，次にピーク負荷が大きい v_1 または v_3（ともにピーク値 30）を対象として再度移動先を探索する．このように，特定の VM の移動が困難な場合でも，同一ホスト上の他の VM を順次試行することで，可能な限り過負荷状態の解消を図る．

このような反復手順により，短期予測と長期予測の統合に基づく配置判断が行われ，単一の予測に依存する場合よりも将来の負荷変動を適切に捉えた再配置が可能となる．

本手法では，ホストおよび VM の短期および長期の CPU 負荷予測という複数の時系列情報を統合的に用い，過負荷状態にあるホストから余裕のあるホストへ VM の再配置を行う．本研究の主題は，単一時点や単一予測に依存するのではなく，異なる時間スケールの予測時系列を組み合わせて配置判断を行う点にある．

第 7 章

評価と考察

7.1　評価環境

表 7.1 に評価に用いた計算機を示す.

7.1.1　ハードウェア環境

本実験で用いたハードウェア環境を示す.

表 7.1　ハードウェア環境

プロセッサ	メモリ	GPU
Intel Core i7-14700K	32.0 GB	NVIDIA RTX 4080 Super（16 GB VRAM）

7.1.2　ソフトウェア環境

本実験で用いたソフトウェア環境を示す.

表 7.2　ソフトウェア環境

OS	Python
Ubuntu 22.04.05 LTS	Python 3.10.18

7.1.3　機械学習・数値計算ライブラリ

本実験で用いた主な機械学習・数値計算ライブラリは下記のとおりである.

表 7.3　機械学習・数値計算ライブラリ

ライブラリ	Version
TensorFlow	2.20.0
scikit-learn	1.7.2
NumPy	2.1.3
SciPy	1.15.3

7.1.4　GPU/CUDA 環境

GPU 計算には CUDA 12.6 および cuDNN 9.11.0 を使用した．TensorFlow は GPU 版を用い，nvidia-cuda-runtime-cu12 環境下で実行した．

7.2　評価方法

7.2.1　評価指標

(1) SLA 違反回数

本研究では，クラウドサービスの品質維持において，継続時間だけでなく「過負荷が発生した事実そのもの（瞬間的なスパイク）」を極小化することが重要であると考える．短時間であっても CPU リソースが枯渇すれば，その瞬間に処理遅延やパケットロスが発生し，ユーザー体験を損なう可能性があるためである．そこで本研究では，SLA 違反を「過負荷状態（CPU 使用率が閾値を超過した状態）が発生したタイム区間の総数」として定義し，これを評価指標とする．

本研究における SLA 違反数 $SLAV_{count}$ を以下の式で定義する．

$$SLAV_{count} = \sum_{t=1}^{T} \sum_{i=1}^{M} V_{i,t} \tag{7.1}$$

ここで，T は全シミュレーション区間数，M はホスト数である．$V_{i,t}$ は，時刻 t における

ホスト i の過負荷状態を示すバイナリ変数であり，以下のように定義される．

$$V_{i,t} = \begin{cases} 1 & (\text{if } U_{CPU}(i,t) > 1.0) \\ 0 & (\text{otherwise}) \end{cases} \tag{7.2}$$

ここで，$U_{CPU}(i,t)$ は時刻 t におけるホスト i の CPU 使用率を表す．本定義により，過負荷が頻発する状況をより敏感に検知し，その発生回数を目的関数として評価することが可能となる．

(2) マイグレーション回数

マイグレーション回数は，シミュレーション期間中に実行されたライブマイグレーションの総数である．マイグレーションは SLA 違反を回避するための手段であるが，実行時にはネットワーク帯域の消費や VM 性能の一時的な低下を伴う．したがって，SLA 違反を抑制しつつマイグレーション回数も抑えることが望ましい．

7.3 比較手法

MP-SLP の有効性を検証するため，以下の 2 つの手法との比較を行う．表 7.5 に，各手法の違いを示す．

表 7.4 各手法の比較

	MP-SLP（提案手法 2）	MP-SOP（比較手法 2）	比較手法 1
VM 選定	Mix（短期・長期の最大値）	Mix（短期・長期の最大値）	短期のみ
移動先選定	短期＋長期（Score 評価）	短期のみ	短期のみ

7.3.1 比較手法：単一のワークロード予測を用いた手法

比較手法は，VM 選定および移動先選定の両方において短期予測のみを使用する．本手法は，堀口ら [10] の手法を VM 配置問題に適用したものであり，複数の時間スケールの予測

表 7.5 各手法の比較

	比較手法 1	MP-SOP（比較手法 2）	MP-SLP（提案手法）
VM 選定	短期のみ	Mix（短期・長期の最大値）	Mix（短期・長期の最大値）
移動先選定	短期のみ	短期のみ	短期＋長期（Score 評価）

を組み合わせることの効果を検証するための基準手法として位置づけられる．

移動 VM の選定

過負荷ホスト上の VM の中から，短期予測のピーク負荷が最大となる VM を移動候補として選定する．MP-SLP および MP-SOP とは異なり，長期予測に伴う時系列統合は行わない．

具体的には，各 VM v_j の短期予測負荷を $P_j^{(s)}(k)$ とするとき，短期ピーク負荷 P_j^{peak} を次式で定義する．

$$P_j^{\mathrm{peak}} = \max_{k \in \{1,\ldots,K_s\}} P_j^{(s)}(k) \tag{7.3}$$

この P_j^{peak} が最大となる VM を移動候補として選定する．

移動先ホストの選定

移動先ホストの選定は MP-SOP と同様に，短期予測のみを用いて評価を行い，短期的な空き容量が最大となるホストを選択する．

MP-SLP との違い

表 7.6 に，各手法における VM 選定の違いを具体例で示す．ここでは，短期予測期間を $K_s = 3$，長期予測期間を $K_l = 6$ とする．

この例では，短期予測のみを用いる比較手法では v_1 と v_2 の短期ピークがともに 30 であ

表 7.6 VM 選定における各手法の違い

VM	短期予測			長期予測			MP-SLP/MP-SOP	比較手法
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	（Mix）	（短期のみ）
v_1	25	30	28	20	40	45	30	30
v_2	30	25	20	35	50	55	30	30

り，どちらを選定しても同等と判断される．一方，MP-SLP および MP-SOP では，長期予測を考慮した統合後のピークも同じ 30 であるが，v_2 は長期的に負荷が上昇する傾向（$k=6$ で 55）を示しており，このような情報が MP-SLP の移動先選定において活用される．

この比較により，VM 選定および移動先選定の両方において複数の時間スケールの予測を活用することが，配置判断の質の向上にどの程度寄与するかを検証する．

7.3.2 MP-SOP（MP-SOP）：移動先選定に短期予測のみを用いた手法

MP-SOP（Mixed Prediction Aware Short-term Only Placement）は，移動対象 VM の選定には MP-SLP と同様に短期予測と長期予測の最大値（Mix）を用いるが，移動先ホストの選定には短期予測のみを使用する．本手法は，長期予測を移動先選定に用いることの効果を検証するためのアブレーションスタディとして位置づけられる．

移動先ホストの選定

移動先ホストの候補は，MP-SLP と同様に余剰容量 $R_i > 0$ を満たすホストとする．各候補ホストについて，当該 VM を配置した場合の短期予測負荷のみを評価し，短期予測期間内で容量超過が発生しないホストの中から，短期的な空き容量（headroom）が最大となるホストを選択する．

具体的には，候補ホスト h_i の短期予測負荷を $L_i^{(s)}(k)$，移動対象 VM の統合後予測負荷を $P_v(k)$ とするとき，配置後の合計負荷 $L_i^{(\mathrm{after})}(k)$ を次式で計算する．

$$L_i^{(\mathrm{after})}(k) = \alpha \cdot \left(L_i^{(s)}(k) + P_v(k) \right) \tag{7.4}$$

すべての $k \in \{1, \ldots, K_s\}$ において $L_i^{(\mathrm{after})}(k) \leq Cap_i$ を満たすホストを配置可能と判定し，その中から短期的な最小空き容量が最大となるホストを選択する．

$$h^* = \arg\max_{h_i} \min_{k} \left(Cap_i - L_i^{(\mathrm{after})}(k) \right) \tag{7.5}$$

MP-SLP との違い

MP-SLP では，移動先ホストの選定において長期予測も評価し，Score 0–3 の 4 段階で分類を行う．これに対し，MP-SOP では長期予測を評価せず，短期予測のみで配置可否を判断する．そのため，短期的には安全であっても，長期的に容量超過が発生するホストへの配置が許容される．

この比較により，移動先選定において長期予測を考慮することが，将来の再マイグレーション抑制や SLA 違反率の低減にどの程度寄与するかを検証する．

7.4　結果と考察

7.4.1　SLA 違反回数

それぞれのワークロードでの SLA 違反回数について考察する．

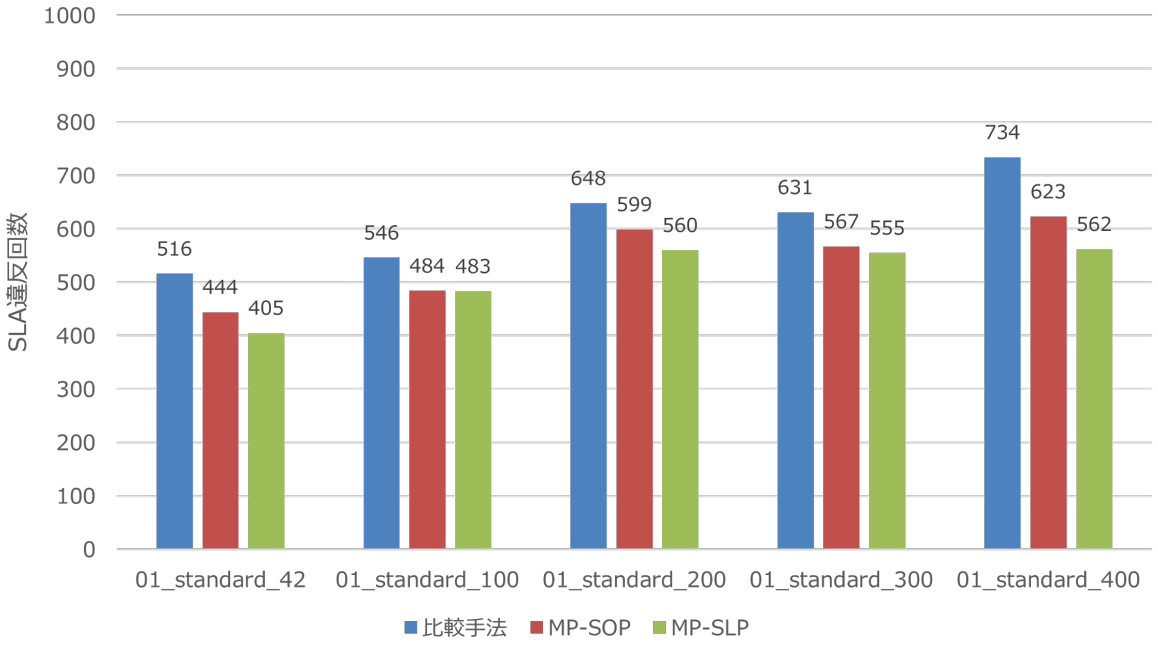

図 7.1　SLA 違反回数 (Standard)

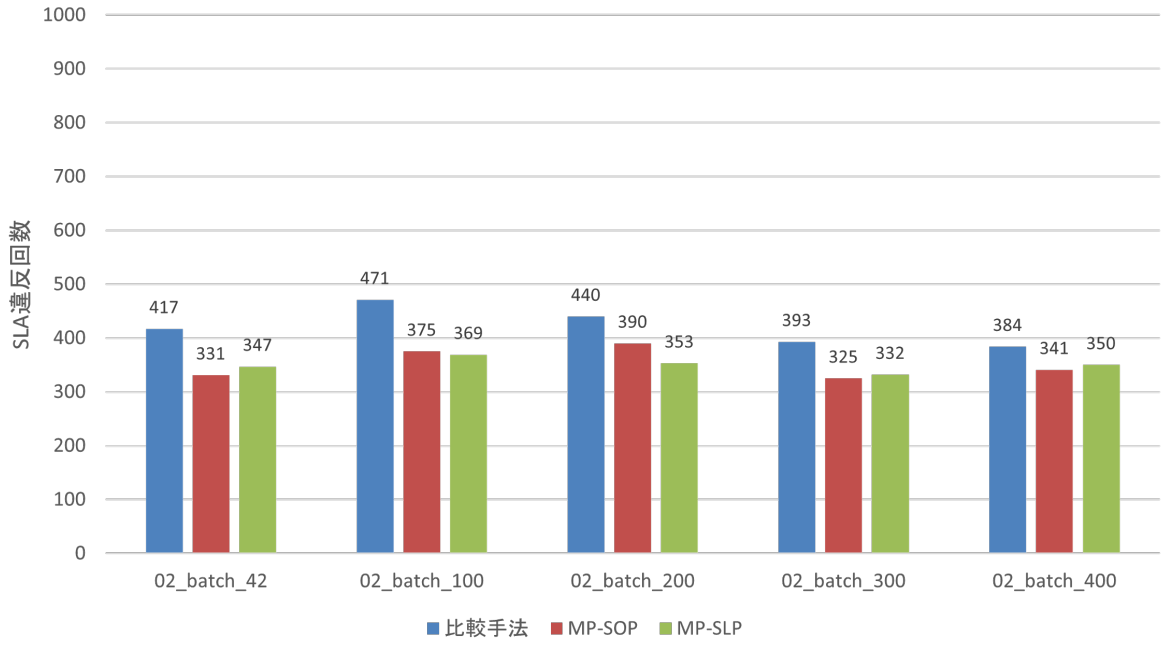

図 7.2　SLA 違反回数 (Batch)

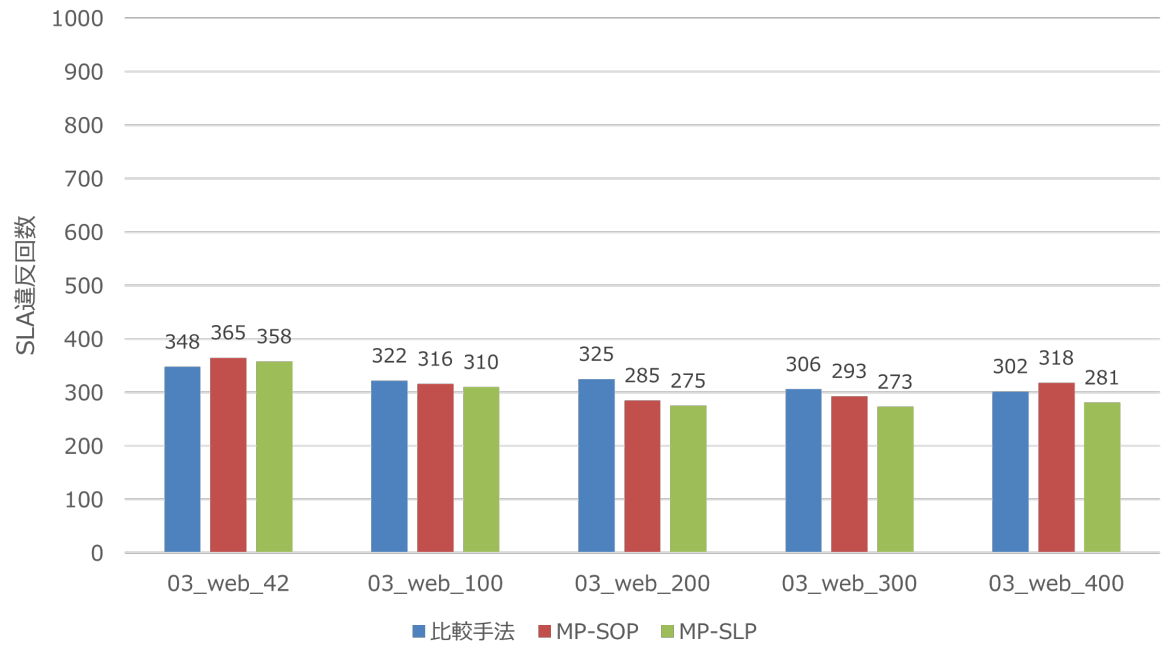

図 7.3　SLA 違反回数 (Web)

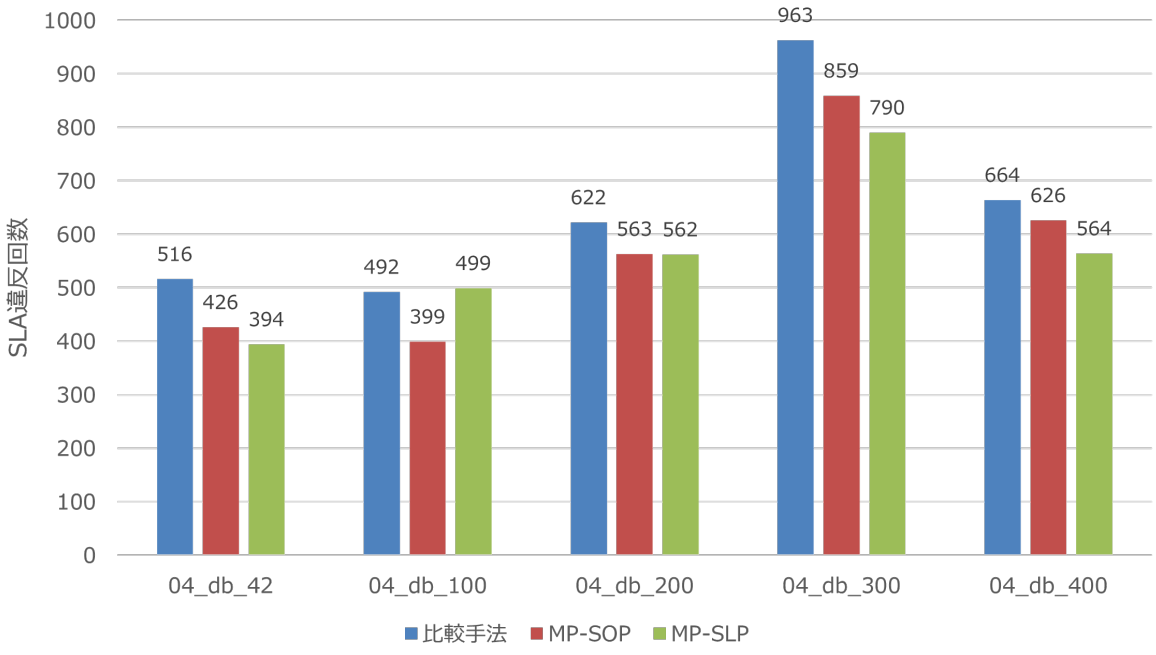

図 7.4　SLA 違反回数 (Database)

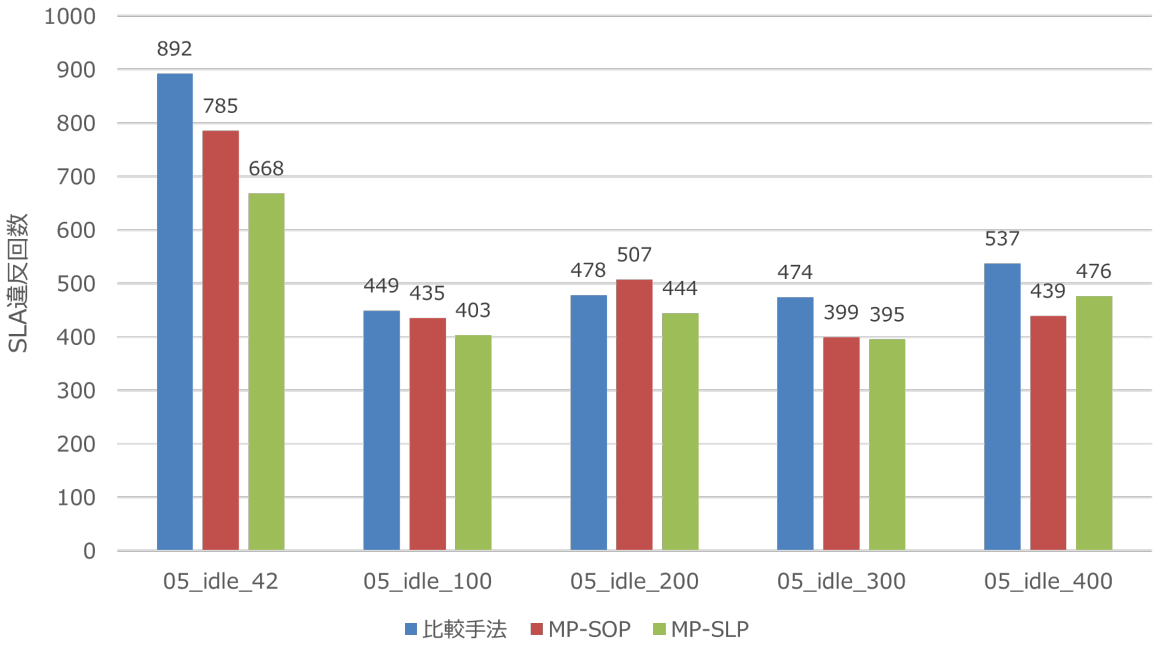

図 7.5　SLA 違反回数 (Idle)

Standard ワークロード

Standard ワークロードでは，MP-SOP が平均 543.4 回，MP-SLP が 513.0 回，比較手法が 615.0 回の SLA 違反を記録した．MP-SOP は比較手法に対して約 11.6%，MP-SLP は約 16.6%の削減を達成した．このワークロードでは MP-SLP が全 5 ケースで MP-SOP を上回ることを確認できた．Standard ワークロードは全プロファイルが均等に混在するため，多様な負荷パターンが共存する環境において，長期的な視点を活用した配置判断が効果を発揮したと考えられる．

Batch ワークロード

Batch ワークロードでは，MP-SOP が平均 352.4 回，MP-SLP が 350.2 回，比較手法が 421.0 回の SLA 違反を記録した．MP-SOP は比較手法に対して約 16.3%，MP-SLP は約 16.8%の削減を達成した．Batch VM は定期的なバーストパターンを持つため，長期予測によってバースト発生タイミングを事前に捉えることができ，将来のバーストに備えた配置判断が有効に機能したと考えられる．ただし，MP-SOP と MP-SLP の差は 0.5 ポイントと比較的小さく，5 ケース中 3 ケースでは MP-SOP が優位であった．これは，バーストの発生タイミングや規模が予測と異なる場合に，長期予測に基づく判断が必ずしも最適とならないケースが存在するためであると考える．

Web ワークロード

Web ワークロードでは，MP-SOP が平均 315.4 回，MP-SLP が 299.4 回，比較手法が 320.6 回の SLA 違反を記録した．MP-SOP は比較手法に対して約 1.6%，MP-SLP は約 6.6%の削減を達成した．しかし，他のワークロードに比べて改善幅が小さく，一部ケースでは比較手法に比べて MP-SOP および MP-SLP が SLA 違反回数が増加している現象が確認できた．

この結果について，以下の解釈が考えられる：

1. Web VM の日次周期パターンは短期予測で十分に捕捉可能であり，Mix 予測（短期・長期の最大値）による負荷見積もりが過大評価となった可能性
2. MP-SLP の長期的視点を活用した配置判定により安全と判定され配置を行われるが，制約条件等により移動ができなくなり連続的な違反が発生した可能性

Database ワークロード

Database ワークロードでは，MP-SOP が平均 574.6 回，MP-SLP が 561.8 回，比較手法が 651.4 回の SLA 違反を記録した．MP-SOP は比較手法に対して約 11.8%，MP-SLP は約 13.8%の削減を達成した．このワークロードでは MP-SLP が全 5 ケースで MP-SOP を上回り，長期予測の有効性が示された．

Database VM は比較的安定した負荷特性を持つが，周期的なバックアップ処理やクエリ集中による負荷変動が発生する．MP-SLP は長期予測によりこれらの変動を事前に考慮することで，より適切な配置判断が可能となったと考えられる．

Idle ワークロード

Idle ワークロードでは，MP-SOP が平均 513.0 回，MP-SLP が 477.2 回，比較手法が 566.0 回の SLA 違反を記録した．MP-SOP は比較手法に対して 9.4%，MP-SLP は 15.7%の削減を達成した．

Idle VM は低負荷であるため多くの VM がホストに配置される．また，Idle VM はスポット的に消失することから，他の VM の影響を受けやすい環境にある．このような高密度配置環境では，将来の負荷変動を考慮した配置判断が重要となり，MP-SLP の長期予測に基づく視点が効果を発揮したと考えられる．5 ケース中 4 ケースで MP-SLP が優位であり，高密度環境における長期予測の有効性が示された．

SLA 違反に関する考察

全体として，MP-SOP は比較手法に対して SLA 違反回数を 8.4%，MP-SLP は 13.1%削減することに成功した．25 ケース中，MP-SOP は 22 ケース（88%），MP-SLP は 23 ケース（92%）で比較手法を上回った．

MP-SOP と MP-SLP を比較すると，25 ケース中 20 ケース（80%）で MP-SLP が優位であった．これらの結果から，長期予測を移動先選定に併用することでさらなる改善が期待できることが示された．

一方で，MP-SOP が MP-SLP を上回るケースも 5 ケース（20%）存在した．これは長期予測の不確実性に起因すると考えられる．長期的な視点を重視することにより，短期的なマージン（余裕）が減少し，予測誤差や突発的な負荷変動に対する耐性が低下する場合がある．また，長期予測が不十分な場合，判断精度が低下する可能性がある．

7.4.2　マイグレーション回数

それぞれのワークロードでのマイグレーション回数を考察する．

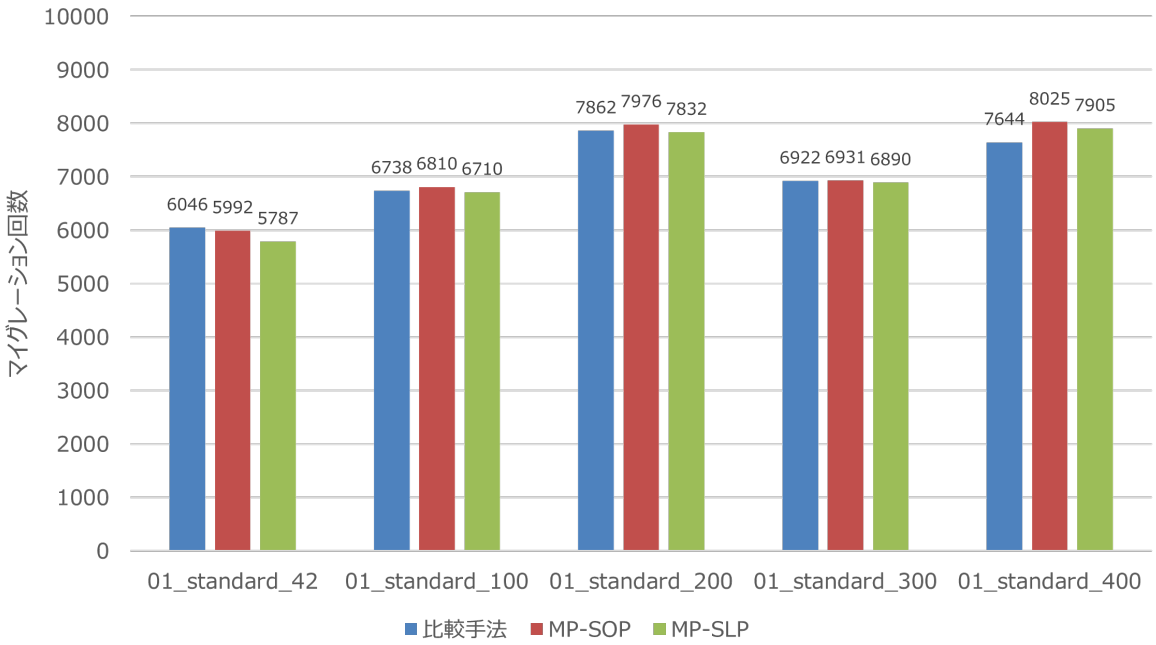

図 7.6　マイグレーション回数 (Standard)

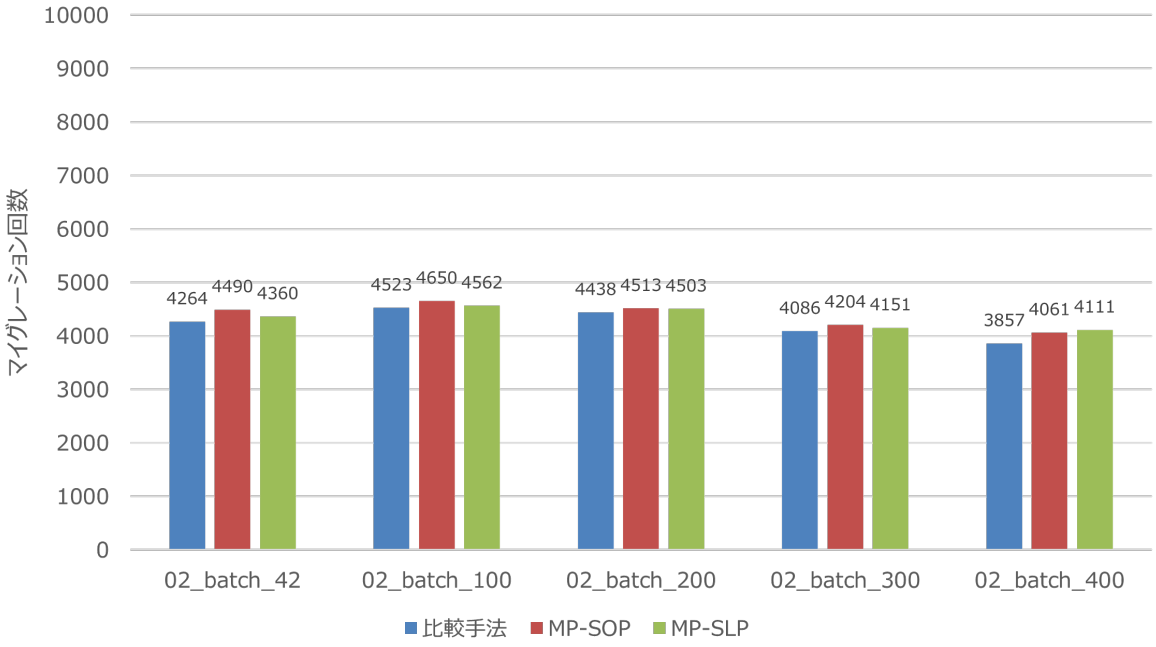

図 7.7　マイグレーション回数 (Batch)

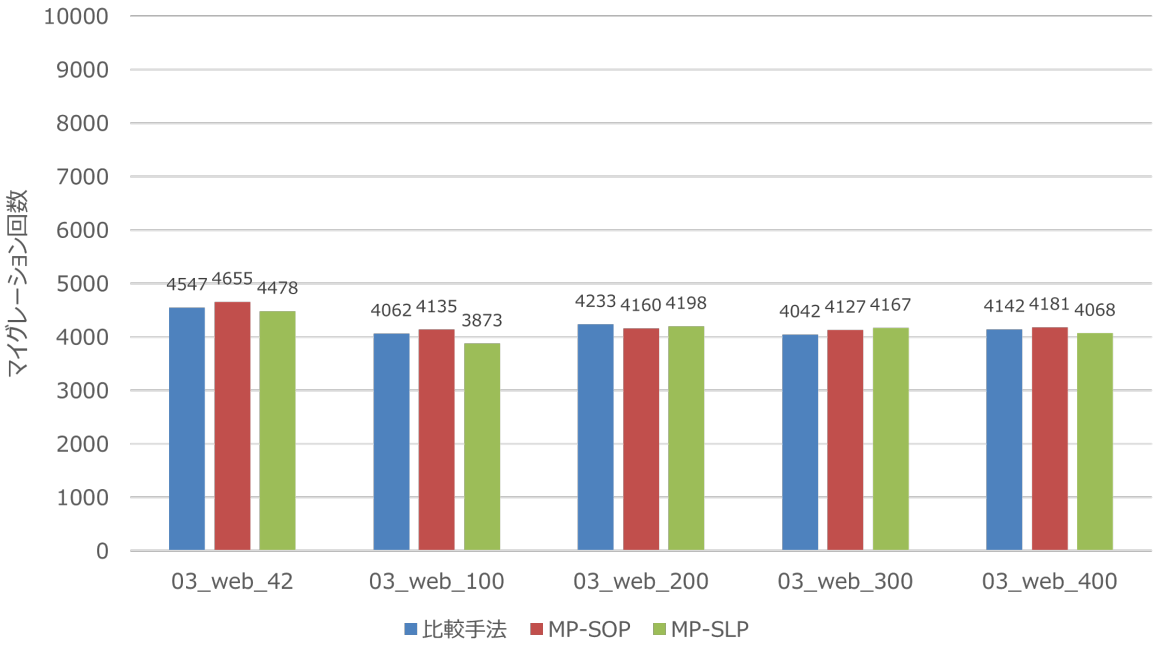

図 7.8　マイグレーション回数 (Web)

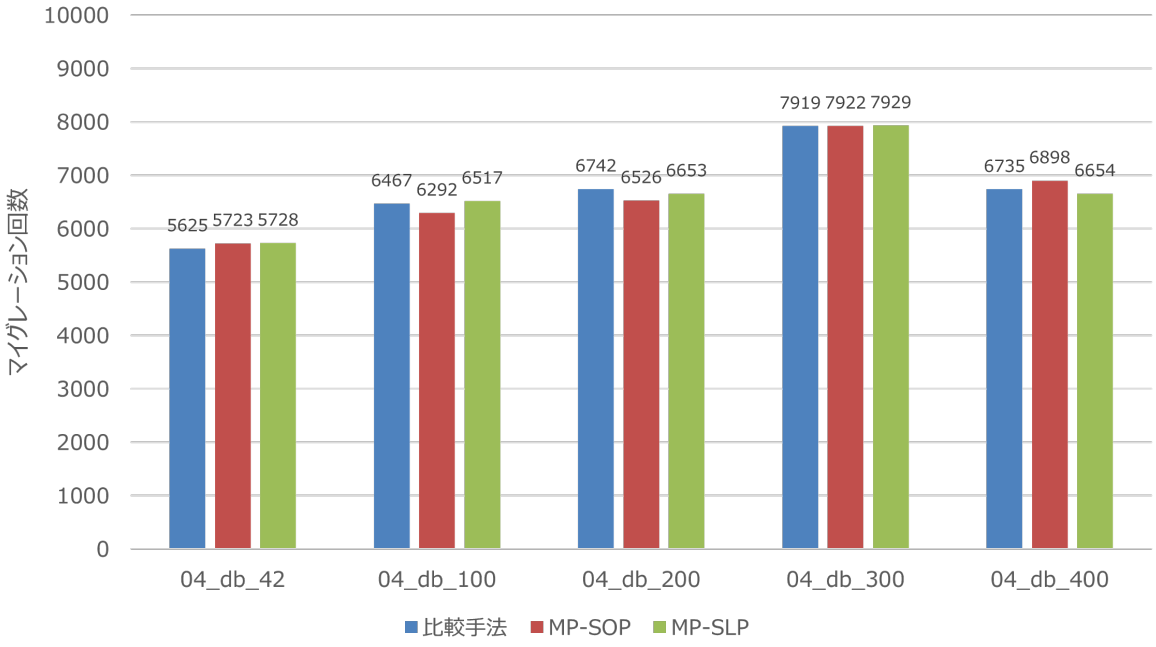

図 7.9　マイグレーション回数 (Database)

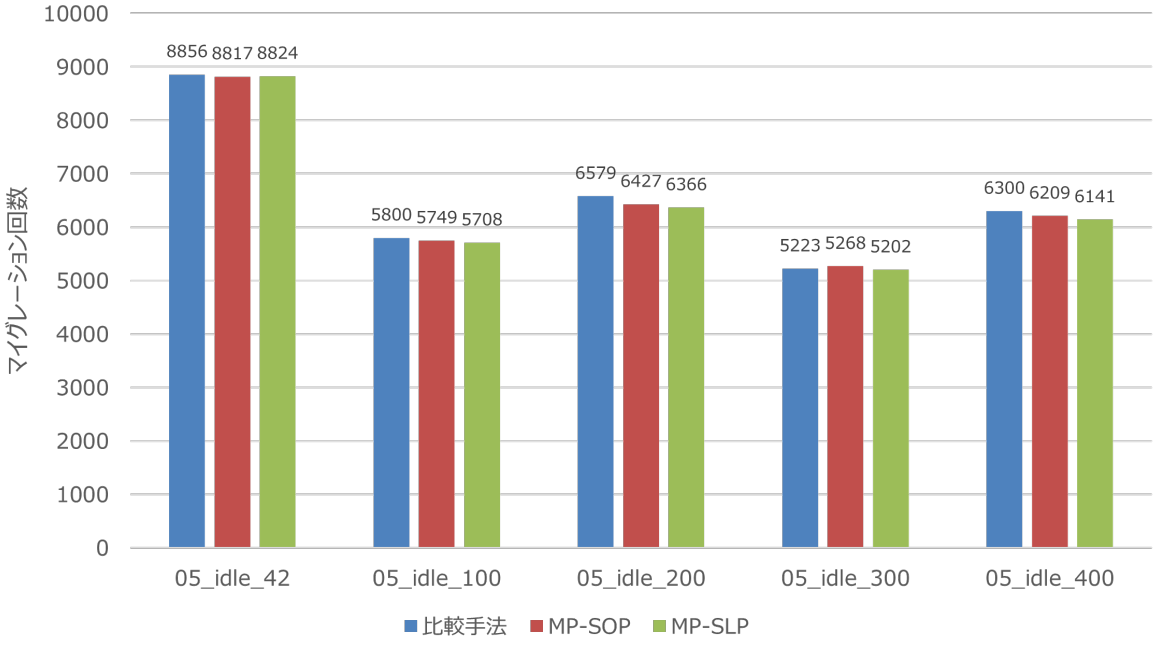

図 7.10　マイグレーション回数 (Idle)

Standard ワークロード

Standard ワークロードでは，MP-SOP が平均 7146.8 回，MP-SLP が 7024.8 回，比較手法が 7042.4 回のマイグレーションを実行した．MP-SLP は比較手法に対して約 0.3%のマイグレーション数削減，MP-SOP は比較手法に対して約 1.5%の増加となった．

Batch ワークロード

Batch ワークロードでは，MP-SOP が平均 4383.6 回，MP-SLP が 4337.4 回，比較手法が 4233.6 回のマイグレーションを実行した．両提案手法とも比較手法よりマイグレーション数が増加しており，MP-SOP は約 3.5%，MP-SLP は約 2.5%の増加となった．

Batch VM はバースト特性を持つため，マイグレーション判断が難しく，全体としてマイグレーション数が増加しやすい．しかし，長期予測により将来のバースト発生を事前に考慮することで，過度なマイグレーションの増加は抑制されていると考えられる．

Web ワークロード

Web ワークロードでは，MP-SOP が平均 4251.6 回，MP-SLP が 4156.8 回，比較手法が 4205.2 回のマイグレーションを実行した．MP-SOP は比較手法に対して約 1.1%の増加となった一方，MP-SLP は約 1.2%の削減を達成した．

Web VM は日次周期の規則的なパターンを持ち，予測精度が高いため，全手法においてマイグレーション回数の差は比較的小さかったと考えられる．その中でも，MP-SLP は長期的な安定性を考慮した移動先選定により，わずかながらマイグレーション数の抑制に成功したと考えられる．

Database ワークロード

Database ワークロードでは，MP-SOP が平均 6672.2 回，MP-SLP が 6696.2 回，比較手法が 6697.6 回のマイグレーションを実行した．MP-SOP は比較手法に対して約 0.4%の削減，MP-SLP は約 0.0%とほぼ同等のマイグレーション回数であった．

Database VM は比較的安定した負荷特性を持つため，マイグレーション数自体に大きな差は生じにくい．一方で，SLA 違反数では MP-SLP が改善を示していることから，マイグレーション回数ではなくマイグレーション先の選定が向上していると考えられる．

Idle ワークロード

Idle ワークロードでは，MP-SOP が平均 6494.0 回，MP-SLP が 6448.2 回，比較手法が 6551.6 回のマイグレーションを実行した．MP-SOP は約 0.9%，MP-SLP は約 1.6%のマイグレーション数削減を達成した．

Idle VM は低負荷であるため高密度に配置され，他の VM の影響を受けやすい環境にある．このような環境では，将来の負荷を考慮した VM 負荷の見積もりが，不要なマイグレーションの抑制に寄与したと考えられる．また，MP-SOP と MP-SLP の差が小さいことから，Idle VM においては長期予測の追加効果が限定的であることが示唆される．

マイグレーション数に関する考察

全体として，マイグレーション数の平均は，比較手法が 5746.0 回，MP-SOP が 5789.6 回，MP-SLP が 5732.6 回となった．MP-SOP は比較手法に対して 0.8%の増加，MP-SLP は 0.2%の削減であり，全体として大きな差は見られなかった．

これは，提案手法がマイグレーション数の削減を直接の目的とせず，予測修正や長期的視点での移動先の多様性を確保し，移動後の安定性を重視した移動先選定を行っているためであると考えられる．

一方，MP-SOP と MP-SLP を比較すると，25 ケース中 18 ケースで MP-SLP の方がマイグレーション回数を抑制できていることが確認された．これは，MP-SLP が長期予測に基づいて将来も安定したホストを選択することで，マイグレーション後の再移動が抑制されたためと考えられる．

しかし，SLA 違反数との関係を見ると，マイグレーション回数が減少している一方で，短期的な負荷変動への対応が遅れ，SLA 違反が増加するケースも一部確認された．このことから，長期的な視点はマイグレーション数抑制に有効であるものの，短期予測とのバランスが重要であることが示唆される．

7.4.3　SLA 違反回数とマイグレーション回数より

以上の結果より，長期的視点を含むことで，SLA 違反回数を減少させる可能性が示唆された．これは，長期的視点による配置先の選択肢が増えたことや，予測修正により正しい情報から正しい配置場所へマイグレーションができたためであると考えられる．一方で，MP-SOP と MP-SLP の効果がどのワークロードでも限定的な場面になり得ることも示唆された．これは，予測修正が大きい値を取る戦略と選択肢が増えた結果，予測誤差を吸収できる箇所に配置ができなかったことや，予測の誤りによる不適切な配置なども考えられる．

第 8 章

おわりに

8.1 まとめ

近年，様々な分野でクラウドコンピューティングの需要が高まっている．その需要に伴い，クラウド事業者には SLA（Service Level Agreement）に基づくサービス品質の維持が求められている．サービス品質を維持するためにはリソースの増強などが考えられるが，運用コストの増大を招くという課題がある．このトレードオフを解消するため，リソースの仮想化や，ライブマイグレーションなどの技術が存在し，効率的にリソースを物理サーバに配置することが重要である．

仮想マシン配置手法として，機械学習によるワークロード予測結果を利用した手法が提案されている．しかし，これらの多くは短期または長期のいずれか単一区間の予測期間に基づくものであり，直近の負荷スパイクや中長期的なトレンドを同時に考慮することが困難である．

そこで，本研究では異なる予測期間を持つ複数区間のワークロード予測を統合的に利用した VM 配置手法を提案し，SLA 違反とマイグレーション数で比較し，複数区間のワークロード予測を活用することが有効であることが示唆された．一方で，予測結果の単純な合成や，スコアリングでは限界もあり，比較手法に劣る場合も確認された．

8.2　今後の課題

本研究では，制御された合成ワークロードを用いることで，長期予測に基づく動的配置手法が，特定の構造的な負荷変動（周期的バースト等）に対して有効に機能する条件が示唆された．一方で，本研究の設計上の前提および評価条件に起因し，実環境への適用に向けての課題が残る．

8.2.1　実際のワークロードへの適応

本研究で用いた合成ワークロードは，予測モデルが学習可能な範囲での設計である．そのため，本研究での特徴のあるワークロードのようなある特定のシステムやサービスが偏在する場合には有効であると考える．しかし，実際のクラウドデータセンターにおけるワークロードには，スポット的な VM が多いことや，予測困難なノイズ成分や，時間とともに統計的性質が変化する非定常性が含まれることが知られている．このような環境では，長期予測精度が低下し，期待されるリスク回避効果が得られない可能性がある．今後は，より現実的な負荷特性を持つ実運用トレースを用いた検証を通じて，本手法のロバスト性を評価する必要がある．

8.2.2　予測誤りが配置判断に与える影響

本手法は予測結果に強く依存すると考えられる．実際のワークロードへの適応でも述べたが，特に負荷変動が激しい場合や，ある VM がホストを占有した場合などは本手法は有効ではない．将来の負荷変動を過大に予測した場合などには不要なリソース確保が発生する恐れがあり，逆に過小に予測した場合には制約条件等により連続した SLA 違反に直結する可能性がある．また，本手法は予測結果が最大の結果を採用している．これは SLA 違反を防止するためには有効ではある一方，VM 移動時により多くのリソースを要求することで配置先が見つからないなどの恐れがあり，SLA 違反増加の恐れがある．現状の手法では，予測の不確実性を明示的に考慮した配置判断は行っておらず，予測誤りに対して保守的に振る舞

う制御機構の導入が課題として残されている．

8.2.3　配置・再配置手法の高度化

本研究では，短期予測と長期予測を統合したリスク評価に基づく再配置手法を提案したが，実環境の多様な要件を満たすための発展的課題も残されている．

具体的には，再試行戦略，VM 選択，スコアリング等の過程の高度化である．本手法では計算量（決定遅延）の抑制を優先し，一度移動に失敗した VM を当該再配置周期内では再試行の対象から除外する貪欲的な設計や，過負荷を解消できない低負荷な VM の移動も選択しうる戦略を採用している．しかし，他 VM の移動成功によってクラスタ全体の空き容量分布が変化した場合，再評価を行えば収容可能となるケースも存在する．したがって，スケジューラのリアルタイム性と再配置成功率（過負荷解消率）のトレードオフを考慮し，状況に応じて探索空間を動的に拡張する適応的な再試行戦略の導入は今後の重要な課題である．また，本設計ではスコアリングを行っているが，スコア 2 以外の効果が薄いことも考えられるため，よりバランスを考慮した配置先の選定や，移動をしないなどの戦略的な手法が考えられる．

以上の課題から，ホスト資源に一定の余裕が存在する環境において，特定の VM の挙動が支配的となる環境条件下での有効性を示すものであると考えられる．今後は，これらの課題解決に取り組むとともに，ワークロード特性（特定の VM の挙動が支配的となる環境など）に応じて予測情報の利用方法を適応的に調整し，より現実的かつ動的なクラウド環境における提案手法の有効性を検証していく．

謝辞

本研究を進めるにあたり，様々な角度からアドバイスや適切な指導をしていただいた指導教員の横山和俊教授に心から感謝いたします．また，副査を引き受けていただき，質問等を通して研究をより良くしていただいた敷田幹文教授，松崎公紀教授に心より感謝します．研究室の皆様には最後まで支えられつつ，研究を最後まで続けることができました．ありがとうございました．

参考文献

[1] 総務省, ”令和 7 年度版情報白書”, <https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r07/html/nd218200.html>, (参照 2025 年 12 月 28 日).

[2] Microsoft, ”Service Level Agreements (SLA) for Online Services”, <https://www.microsoft.com/licensing/docs/view/Service-Level-Agreements-SLA-for-Online-Services?lang=18>, (参照 2025 年 12 月 28 日).

[3] National Institute of Standards and Technology U.S. Department of Commerce, ”The NIST Definition of Cloud Computing”, <https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf>, (参照 2025 年 12 月 28 日).

[4] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, and H. Tenhunen, ”Utilization Prediction Aware VM Consolidation Approach for Green Cloud Computing,” in Proc. IEEE 8th Int. Conf. Cloud Comput. (CLOUD), 2015, pp. 381–388.

[5] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, ”Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications ’ QoS,” IEEE Transactions on Cloud Computing, vol. 3, no. 4, pp. 449–458, 2015.

[6] U. C. De, R. Satapathy, and S. S. Patra, ”Optimizing Resource Allocation using Proactive Predictive Analytics and ML-Driven Dynamic VM Placement,” 2023 4th IEEE Global Conference for Advancement in Technology (GCAT), Bangalore, India, 2023, pp. 15, doi: 10.1109/GCAT59970.2023.10353234.

[7] H. L. Leka, Z. Fengli, A. T. Kenea, A. T. Tegene, P. Atandoh, and N. W. Hundera, “A Hybrid CNN-LSTM Model for Virtual Machine Workload Forecasting in Cloud Data Center,” in 2021 18th International Computer Conference on Wavelet Active

Media Technology and Information Processing (ICCWAMTIP). IEEE, Conference Proceedings, pp. 474–478.

[8] M. Smendowski and P. Nawrocki, ”Optimizing multi-time series forecasting for enhanced cloud resource utilization based on machine learning,” Knowledge-Based Systems, vol. 304, 112489, 2024.

[9] Barbalho, Hugo and Kovaleski, Patricia and Li, Beibin and Marshall, Luke and Molinaro, Marco and Pan, Abhisek and Cortez, Eli and Leao, Matheus and Patwari, Harsh and Tang, Zuzu and Rozales Gonçalves, Larissa and Dion, David and Moscibroda, Thomas and Menache, Ishai, ”Virtual Machine Allocation with Lifetime Predictions，”Proceedings of Machine Learning and Systems (MLSys 2023), Vol. 5, pp.232–253，2023.

[10] 堀口 幹展，高橋 竜一，深澤 良彰，”エッジコンピューティングにおける予測を用いた負荷分散手法”，情報処理学会研究報告，Vo.2023-SE-214，No.29，pp. 1-8，2023.