情報処理学会論文誌 Vol.57 No.2 562–572 (Feb. 2016)

クラウドにおける自律的な応答性向上に向けた広域スケーリング

藪崎 仁史[1,a)] 中越 洋[1] 村山 耕一[1] 加藤 崇利[1]

受付日 2015年5月14日, 採録日 2015年11月5日

概要：広域に分散する端末のクラウド利用において，アプリケーションの構成要素の配置を最適化することで，応答時間を削減する繰返し配置方式 WATS（Wide area tentative scaling）を提案する．定式化された応答時間が最小になる配置構成を計算するアプローチでは，アプリケーションごとに異なる構成要素間の通信時間をすべて分析して定式化する必要があり，その分析量が膨大になる．WATS においては，構成要素間の通信時間を用いずに，配置構成の変更と応答時間を実測するという試行を繰り返すことにより，応答時間を段階的に削減する．その際，配置構成の変更にともなう計算資源コストの増加を抑制するため，ベイズ推定を活用して，次の試行において評価すべき配置構成の推定，および，停止判定を行う．クラウドを模擬したシミュレーション評価の結果，WATS を適用すると，40〜130%の計算資源コストの一時的な増加を許容することにより，応答時間が平均で最適解の 1〜5%程度大きな値に収束する見込みを得た．

キーワード：クラウドコンピューティング，応答時間，配置構成，ベイズ推定

Wide Area Tentative Scaling for Quick Response in Cloud Computing

Hitoshi Yabusaki[1,a)] Hiroshi Nakagoe[1] Koichi Murayama[1] Takatoshi Kato[1]

Received: May 14, 2015, Accepted: November 5, 2015

Abstract: Wide area tentative scaling (WATS) is proposed which improves the response time in a phased manner by repetitively replicating a part of an application and related data at other DCs and selecting a better organization. WATS approach is to repetitively change the organization to reduce the response time. The drawback of this approach is that it consumes more computing resources due to repetitive replication. We therefore, applied Bayesian inference to search for a better organization with fewer trials. Evaluation results showed that WATS successfully reduced the response time in a phased manner.

Keywords: cloud computing, response time, application organization, Bayesian inference

1. はじめに

グローバル化にともない，クラウドに配備されたアプリケーションを利用する端末が地理的に分散する．アプリケーションが格納されるデータセンタから遠隔にある端末においては，データ処理要求を送信してから処理結果を受信するまでの応答時間が数百ミリから数秒になる．この値は，ユーザビリティを損なわずに利用できる応答時間の限界 [1], [2] を超える．

エンドユーザが閲覧のみ行う静的な Web コンテンツにおいては，端末近傍のサーバにキャッシュすることにより，応答時間が削減される．一方，エンドユーザが閲覧と書き込みを行うアプリケーションにおいては，分散した端末の近傍に分散配置させても，必ずしも応答時間が削減されない．これは，広域に分散したデータの同期遅延や，Web フロントエンドサーバ，バックエンドサーバ，および，ストレージ等のアプリケーションの構成要素間の通信遅延が生じるためである．アプリケーションサービスプロバイダ

[1] 株式会社日立製作所
Hitachi Ltd., Yokohama, Kanagawa 244–0817, Japan
[a)] hitoshi.yabusaki.vw@hitachi.com

が，アプリケーションの応答時間を削減するためには，アプリケーションの構成要素の配置場所を最適化する必要がある．

アプリケーションの構成要素の配置場所を最適化する従来研究を以下に示す．文献 [3] においては，クラウドコンピューティングにおいて CPU，メモリ，および，ストレージの容量を制約として課し，応答時間が最小になるように最適化問題を解く．文献 [4] においては，文献 [3] と同様の問題をデータセンタ内において解く．文献 [5] と [6] においては，エンドエンドの応答時間を定式化し，最適化問題を解く．また，文献 [7], [8], [9] においては，CPU，メモリ，帯域，および，エネルギー等の複数制約条件下において最適化問題を解く．

これらの従来研究では，アプリケーションの構成要素間の通信時間等から応答時間を定式化し，整数計画法等で最適化問題を解く．しかし，実運用に適用するためには，アプリケーションごとに異なる構成要素間の通信時間をすべて分析して定式化する必要があり，それには以下の理由から，膨大な分析をともなう．まず，アプリケーションのどの構成要素からどの構成要素にデータ参照や更新等の処理要求が送信されるかといった相互参照関係を把握する必要がある．また，データの同期書き込みや非同期等といった一貫性の強さを把握する必要がある．また，相互参照関係や要求される一貫性の強さが応答時間に及ぼす影響を定量化する必要がある．さらに，パブリッククラウドにおいては性能がばらつく [10]．そのため，応答時間を正確に定式化することが難しい．

本研究の狙いは，クラウドを利用するアプリケーションサービスプロバイダが，配置計画や分析に要する運用管理の負荷を増やさずに，アプリケーションの構成要素を最適なデータセンタに配置し，応答時間を削減することである．本論文では，アプリケーションとデータの配置最適化方式について述べる．提案方式の新規性は以下の 2 点である．1 つ目は，応答時間を計測しながら配置構成を繰り返し変更することで，段階的に応答時間を削減する点である．2 つ目は，各配置構成において計測される応答時間を含む実行時間の履歴に基づいて，配置先の決定と繰返し配置の停止判定をベイズ推定により判定する点である．本判定により，繰返し配置による計算資源コストの増加を抑えつつ，応答時間を削減可能にする．クラウドを模擬したシミュレーション評価の結果，繰返し配置方式 WATS（Wide area tentative scaling）を適用すると，40〜130%の計算資源コストの一時的な増加を許容することにより，応答時間が平均で最適解の 1〜5%程度大きな値に収束する見込みを得た．以下，広域分散クラウドについて説明し，繰返し配置方式 WATS，および，評価結果を示す．

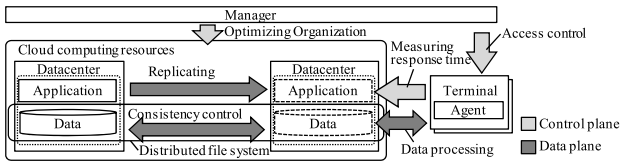

図 1 広域分散クラウド
Fig. 1 Wide area distributed cloud-computing system.

2. 広域分散クラウド

これまで，我々は，複数のデータセンタを活用し，端末近傍のデータセンタにアプリケーションとデータを分散配置して端末近傍でデータ処理する広域分散クラウドの開発を進めてきた [11], [12]．

提案方式 WATS が動作する広域分散クラウドについて述べる．**図 1** に広域分散クラウドの概要を示す．広域分散クラウドにおいては，地理的に分散する複数のデータセンタにアプリケーションやデータが配備され，端末近傍のデータセンタでデータ処理される．WATS による広域分散クラウドの特徴は 2 つある．

1 つ目の特徴は，アプリケーションを管理するマネージャと端末に格納されるエージェントとが連携動作する点である．エージェントは，端末が近傍のデータセンタをマネージャから取得すること，および，応答時間を計測してマネージャに通知することを目的に端末に格納される．以下に動作内容を示す．エージェントはデータセンタに格納されたマネージャに DNS クエリを送り，近傍のデータセンタに格納されたアプリケーションの URL を把握する．エージェントは把握した URL にリクエストを送信してからクエリを受信するまでの応答時間を計測し，計測結果をマネージャに通知する．エージェントにおける応答時間の計測においては，計測専用の模擬的な通信は行われず，ユーザが Web 閲覧や書き込みを行った際の応答時間を計測する．これは，計測専用の模擬的な通信を行う場合，計測のためのダミーの処理の追加等，アプリケーションの動作を変更する必要が生じるためである．

マネージャは，アプリケーションの構成要素の配置構成を変更する．なお，本論文においては，アプリケーションの構成要素を，Web フロントエンドサーバ，バックエンドサーバ，および，ストレージと見なす．

2 つ目の特徴はデータの一貫性を制御する分散ファイルシステムである．広域分散クラウドにおいては，マネージャが複数のデータセンタに分散された最新のデータの格納場所を管理する．一貫性の制御方法について述べる．トランザクション処理等のために強い一貫性を保つ方法として，データ同期とメタデータ同期の 2 種類がある．データ同期においては，分散ファイルシステムがデータ更新の要求を受け取ると，データを同期した後に応答を返す．メタデータ同期においては，分散ファイルシステムがデータを

同期せず，メタデータ（最新のデータの格納場所）を同期した後に応答を返す．前者はデータ同期の時間だけ，後者はメタデータ同期の時間だけ応答が遅くなる．一方，緩い一貫性を許容するデータが更新される際には，複製されたデータ間の一貫性を保証する必要がないため，分散ファイルシステムは応答を返した後に，複製されたすべてのデータを更新する．

3. 繰返し配置方式 WATS

従来方式における問題は，1 章に記載したとおり，各アプリケーションの応答時間を定式化するために膨大な分析をともなう点である．我々は計算資源の単価が下落傾向にあることに着眼し，一時的な計算資源の消費量の増加を許容するアプローチをとる．具体的には，アプリケーションとデータの配置（試行）を繰り返し変更して最適な配置構成を自律的に探索する．ただし，配置変更を無制限に続けると，計算資源コストが膨大になる．そこで，試行における評価結果に基づいて，次の試行における配置先の決定と繰返し配置の停止判定を行う．これにより，繰返し配置によるコスト増加を抑制する．

3.1 想定するアプリケーション

本論文においては，クラウド等で提供される PaaS 基盤上で 1 つ以上の疎結合な構成要素からなるアプリケーションを対象とする．疎結合な構成要素とは，異なる VM やサーバ上で動作可能なデータ処理機能である．具体的には，Web フロントエンドサーバ，バックエンドサーバ，および，ストレージ等を構成要素と想定する．実運用においては，構成要素よりも細かい粒度，たとえば，Web フロントエンドサーバの機能の一部やストレージのデータの一部のみを移動することも考えられるが，本論文では想定外とする．

3.2 動作概要

提案方式 WATS を適用した際のアプリケーションとデータの構成変更を例示し，動作概要を述べる．**図 2** は，Web フロントエンドが複数のデータセンタに分散配置される際の概念図である．前提条件として，計算資源コストの上限から，定常的に設置可能なフロントエンド数の上限を 2 とする．

1 回目の試行に関して説明する．各試行においては，構成要素を実際にデータセンタに複製し，応答時間を実測して評価する．

まず，最適な配置先を推定して同配置先にフロントエンドと関連するデータをあるデータセンタ（図では DC 2）に複製する．推定方法は 3.4 節において後述する．複製後にとりうる配置構成（暫定的な配置構成）は 3 種類である．それらは，複製元（DC1）のみ利用する元の構成，複製元（DC1）を停止して複製先（DC2）のみ利用する構成，および，複製先（DC1）と複製元（DC2）の両方を利用する構成である．次に上記 3 種類の構成において応答時間を評価する．まず，各エージェントをランダムでいずれかの配置構成に割り当てる．各エージェントは割り当てられた配置構成の最寄りの Web フロントエンドと通信する．また，その際の応答時間を計測し，マネージャに通知する．マネージャは応答時間の平均値が最小である配置構成を採用する．

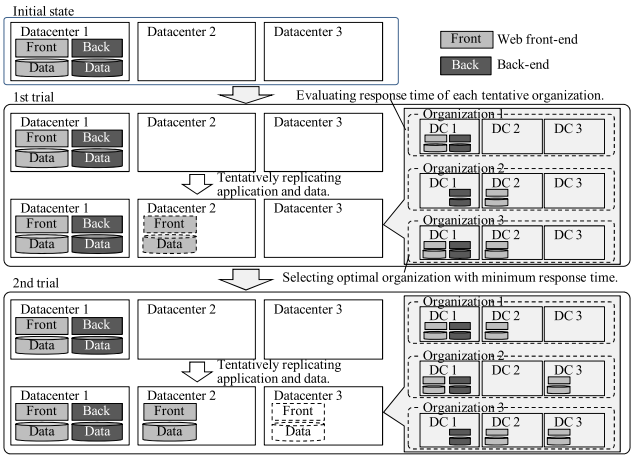

図 2 適切な配置構成の探索例図
Fig. 2 Example of searching a better organization.

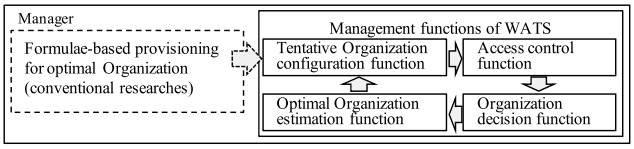

図 3 WATS のシステム構成
Fig. 3 Organization of WATS.

採用された配置構成をもとに 1 回目の試行と同じ要領で 2 回目の試行を行う．その際，1 回目の計測結果を活用して，最適な配置先を推定する．

上記試行を繰り返すことで，応答時間を段階的に削減する．なお，複製先のデータセンタの選択方法と繰返し配置の停止判定は 3.4 節において述べる．

3.3 システム構成

WATS のシステム構成について**図 3** を用いて説明する．WATS は，最適構成の推定機能，暫定的構成の設定機能，名前解決機能，および，構成決定機能で構成される．

暫定的構成の設定機能は，最適構成の推定機能が指定する複製先にアプリケーションを配置する．名前解決機能は，エージェントからの名前解決の要求に対し，アプリケーションが配置されたサーバのうち，端末の最寄りのアクセス先を通知する．構成決定機能は，複数エージェントから受信した応答時間から配置構成ごとに平均値を求め，暫定的な配置構成の中から応答時間の平均値が最小である配置構成を選択する．

最適構成の推定機能は，実行履歴（各配置構成において計測される応答時間）から次の試行におけるアプリケーションとデータの配置先を推定し，繰返し配置の停止判定を行う．これにより，繰返し配置に要する計算資源コストを削減する．

暫定的構成の設定機能はアプリケーションの配置ツールである Chef [13] を用いた．アクセス制御機能には，既存の DNS と同様の名前解決の仕組みを適用した．応答時間の収集と管理する仕組みには，既存のログ収集ツールである Fluentd [14] を用いた．以下，最適構成の推定機能の詳細を示す．

3.4 最適構成の推定機能

最適構成の推定機能は，計算資源コストを削減するため，試行における計測結果に基づいて，最適な（応答時間が最小になる）配置構成を推定する．

（1）前提条件

前提条件を以下に示す．

- 各端末の位置は未知とする．
- 端末は固定端末とし，試行中における端末の移動は無視できるものとする．
- アプリケーションの構成要素間の通信の有無やデータ一貫性の強さは未知とする．すなわち，構成要素間の通信時間は未知とする．
- 過去の試行において端末が計測した，暫定的な配置構成の応答時間を蓄積し参照可能とする．

（2）最適な配置構成の推定方法

最適な配置構成を推定するため，応答時間が最小である確率が最も高い配置構成を計算する．確率を計算するうえで，試行によって得られる計測結果を利用するため，計測結果を条件とする条件付き確率を計算する．以下に計算方法の詳細を述べる．

条件付き確率の計算に用いる記号の定義を**表 1** に示す．t 回の試行後に，測定結果として特徴変数 $x_{i_1}, x_{i_2}, \ldots, x_{i_t}$（$x_i$ は $\bar{x}_i$ であってもよい）が得られたと仮定する．このとき，データセンタ k が最適な配置先であるという依存クラス変数事象 $x_{i_1}, x_{i_2}, \ldots, x_{i_t}$ が起こった場合に事象 y_k が起こる条件付き確率 $P(y_k|x_{i_1}, x_{i_2}, \ldots, x_{i_t})$ はベイズの定理より，式 (1) で計算される．

$$P(y_k|x_{i_1}, x_{i_2}, \ldots, x_{i_t}) = \frac{P(y_k) \times P(x_{i_1}, x_{i_2}, \ldots, x_{i_t}|y_k)}{P(x_{i_1}, x_{i_2}, \ldots, x_{i_t})}$$
$$= \frac{P(y_k) \times P(x_{i_1}|y_k) \times P(x_{i_2}|y_k, x_{i_1}) \times \cdots \times P(x_{i_t}|y_k, x_{i_1}, \ldots, x_{i_{t-1}})}{P(x_{i_1}, x_{i_2}, \ldots, x_{i_t})} \quad (1)$$

右項の条件付き確率の値は，過去の実行履歴等による統計情報として得られるが，試行回数の増加にともない，条件付き確率の値が指数的に増加する（次元の呪い）．次元の呪いの問題を回避する方法として，単純ベイズ分類器が有効である [15]．単純ベイズ分類器は，特徴変数の独立性を仮定することにより，次元の呪いから生じる問題を緩和する．独立性の仮定は一般に成り立たないことが多いが，単純ベイズ分類器の目的が，特徴クラス変数の正確な条件付き確率の算出ではなく，最も確からしい特徴クラス変数の算出である場合に有効である．単純ベイズ分類器は適用例としてスパムフィルタがある．

表 1 条件付き確率の計算に用いる記号
Table 1 Symbols used for formulating conditional probability.

記号	説明
$DC_{opt,t}$	t 回目の試行における配置先データセンタ
N	過去の試行において未配置先のデータセンタの集合
$P(x)$	事象 x が起きる確率
$P(y\|x)$	事象 x が起こった場合に，事象 y が起こる条件付き確率
r_i	データセンタ i に配置されるアプリケーションを利用する端末の平均応答時間
x_{i_t}	“t 回目の試行における配置先のデータセンタ i_t の応答時間 r_i が閾値αよりも小さい”という事象
$\bar{x}_{i_t}$	“t 回目の試行における配置先のデータセンタ i_t の応答時間 r_i が閾値αよりも大きい”という事象
y_k	“データセンタ k が応答時間を最小とする配置先である”という事象
ε	繰り返し配置の停止判定の閾値
α	応答時間が小さいか否かを判断する為の閾値．α は過去に計測された応答時間の平均値とした．

単純ベイズ分類器を適用することにより式 (1) は式 (2) のように単純化される．

$$P(y_k|x_{i_1}, x_{i_2}, \ldots, x_{i_t}) = P(y_k) \prod_{j=1}^{t} P(x_{i_j}|y_k) / P(x_{i_1}, x_{i_2}, \ldots, x_{i_t}) \quad (2)$$

実行履歴による統計情報として得られる右項の各値の算出方法に関して説明する．$P(y_k)$ は，収束時にデータセンタ k が最適な配置構成に含まれたケースの回数をすべてのケースの回数で割った割合として求められる．$P(x_{i_t}|y_k)$ は，収束時にデータセンタ k が最適な配置構成に含まれたすべてのケースにおいて，データセンタ i_t に配置して計測した応答時間 r_i が閾値 α よりも小さくなった回数を同ケースの回数で割った値として求められる．$P(x_{i_1}, x_{i_2}, \ldots, x_{i_t})$ は $x_{i_1}, x_{i_2}, \ldots, x_{i_t}$ がすべて成り立った回数をいずれか 1 つでも成り立たなかった回数で割って求められる．なお，ベイズ推定においては，分母の値は重要ではなく，右辺の値の合計値が 1 になるように正規化されてもよい．

式 (2) で算出される条件付き確率の最大値であり，かつ，配置先として選択されていないデータセンタ k を $t+1$ 回目の試行における配置先のデータセンタ $DC_{opt,t+1}$ とす

る．$DC_{opt,t+1}$ は式 (3) で表される．

$$DC_{opt,t+1} \approx \mathrm{argmax}_k \left\{ \frac{P(y_k) \times \prod_{j=1}^{t} P(x_{i_j}|y_k)}{P(x_{i_1}, x_{i_2}, \ldots, x_{i_t})} \right\} \quad (3)$$

繰返し配置の停止判定について述べる．繰返し配置の目的が応答時間の削減であることから，応答時間が削減される見込みがない場合に停止されるべきである．応答時間が削減される見込みがないとは，試行においてアプリケーションの暫定的な配置先として選択されていない各データセンタに配置することで応答時間が削減される確率の和が十分に小さいことである．すなわち，$t+1$ 回目の試行において，式 (4) が成り立つ場合に繰返し配置を停止する．

$$\sum_{k \in N} P(y_k|x_{i_1}, x_{i_2}, \ldots, x_{i_t}) = \sum_{k \in N} \frac{P(y_k) \times \prod_{j=1}^{t} P(x_{i_j}|y_k)}{P(x_{i_1}, x_{i_2}, \ldots, x_{i_t})} < \varepsilon \quad (4)$$

繰返し配置の手順をまとめる．まず，データセンタ $DC_{opt,t+1}$ を計算する．次に，アプリケーションとデータを $DC_{opt,t+1}$ に配置する．新しい配置構成と元の配置構成の応答時間を比較し，応答時間が小さい配置構成のみ残す．複数の配置構成の応答時間が同一の場合，複製数が少ない配置構成を採用する．これらの手順を式 (4) が成り立つまで繰り返す．なお，1 回目の試行における配置先データセンタ $DC_{opt,t}$ は最適化問題を解く従来方式によって計算されてもよい．

表 2 応答時間のモデルに用いる記号
Table 2 Symbols used for modeling response time.

記号	説明
$\rho_{k,i}$	端末 k がデータセンタ i に配置されたアプリケーションを利用する際の応答時間
$\rho_{k,i}'$	端末 k とデータセンタ i 間の通信時間
$\rho_{i,j}''$	データセンタ i とデータセンタ j 間の通信時間
ρ_i'''	データセンタ i におけるアプリケーションの処理時間
δ	データ転送量
$b_{i,k}$	データセンタ（または端末）k とデータセンタ i との間の TCP スループット
$d_{k,i}$	データセンタ（または端末）k とデータセンタ i との間のラウンドトリップタイム
I	アプリケーションが配置されるデータセンタの集合
K	全ての端末 k の集合
s	通信のオーバーヘッド（セッション確立等）
x	CPU 利用率

4. システム評価

4.1 評価環境

応答時間と計算資源コストに関して評価した．機械学習を行う提案方式 WATS においては，端末位置等の初期条件によって結果が変わるため，有意な統計結果を得るために十分な回数の評価が必要である．そのため，大手クラウドサービスを模擬したシミュレーション評価を行った．

評価環境を**表 3** に示す．データセンタの位置は，大手クラウドサービスのデータセンタの位置を考慮しつつ，すべての大陸に分散するように選定された．端末の位置は，すべての大陸に分散する 70 拠点から 10 拠点をランダムで選択された．データセンタ間，および，データセンタと端末間の通信遅延と通信帯域は計測結果 [16] を適用した．

IT 機器の仕様，および，演算処理に関連するパラメータを**表 4** に，IT 機器の単価を**表 5** に示す．ばらつきの大きいクラウドにおける CPU の性能の変動係数は $\sigma = 13.79\%$ [17] である．本評価では，CPU 利用率を $\pm 3\sigma$（信頼度 99.7%）とした．IT 機器の単価においては，大手クラウドサービス

表 3 評価環境
Table 3 Environment of system evaluation.

項目	説明
データセンタ位置	東アジア：中国，日本，マレーシア 南アジア：インド，パキスタン 西アジア：ヨルダン 欧州：イタリア，スイス，ドイツ アフリカ：アルジェリア，ブルキナファソ 北米：カナダ，米国 南米：ブラジル，ボリビア
端末位置候補	70 拠点から 10 拠点をランダムで選択
通信遅延（RTT）	DC 間、及び，端末-DC 間の通信遅延は共に，[16]による計測結果を適用
通信帯域	DC 間、及び，端末-DC 間のスループットは共に，[16]による計測結果を適用

表 4 IT 機器仕様
Table 4 Specification of IT equipments.

項目		設定値
CPU		2.4 GHz
メモリ		8 GB
コア数		1
インスタンス数		1
演算処理時間係数	a	7.94
	b	-0.2
	c	45.17
CPU の変動係数 σ		13.79%
CPU 使用率のばらつき		±3 σ（信頼度 99.7%）

表 5 IT 機器の単価
Table 5 Prices of IT equipments.

項目		単価
インスタンス	オンデマンド	$0.4 / h
	リザーブド	$20.25 / month
ストレージ	データ保存	$0.1 / GB month
	I/O 処理	$0.3 / 1 million access
ネットワーク	入力	$0.1 / GB
	出力	$0.17 / GB

表 6 アプリケーションの特徴 [18]
Table 6 Characteristics of Applications.

項目	Montage	Broadband	Epigenome
入力データ通信量	4291 MB	4109 MB	1843 MB
出力データ通信量	7970 MB	159 MB	299 MB
Logs データ通信量	40 MB	5.5 MB	3.3 MB
I/O 処理回数	7176 回	7176 回	7176 回
ストレージサイズ	5 GB	5 GB	2 GB
通信オーバーヘッド	1.5 回	1.5 回	1.5 回
端末送受信量	1 MB	1 MB	1 MB

の価格帯を採用した．

アプリケーションの特徴を**表 6** に示す．実環境においてモデル化したアプリケーションのコストと応答時間を実測された通信量やデータ量 [18] と揃えるため，上記クラウドサービス上で評価されたアプリケーションである Montage [19]，Broadband [20]，Epigenome [21] の 3 種類とした．各アプリケーションの特徴を述べると，Montage は処理遅延の 95%が I/O であり，通信量が多いアプリケーションである．Broadband は処理の 75%が 1 GB 以上であり，メモリ消費量が多い．Epigenome は処理時間の 99%が演算処理であり，CPU 負荷が大きい．データの一貫性制御の種類は，運用ポリシに依存する．本論文では，強い一貫性，弱い一貫性，メタデータ同期の中からランダムで選択した．

応答時間は，端末の地理的な分布，アプリケーション種別，一貫性制御の種類に依存する．そのため，応答時間の平均値を算出する際には，1,000 万ケース行った．

なお，提案方式の評価の前に，本評価環境において改善可能な上限を示すため，最適な（応答時間が最小となる）配置構成における応答時間と計算資源コストの平均値を算出した．応答時間が最小になる配置構成の算出方法は，すべての配置構成の組合せにおいて応答時間を計算し，応答時間が最小となる配置構成を選択することによって算出された．応答時間が最小になる配置構成，および，応答時間，計算資源コストは初期条件（端末の地理的な分布とアプリケーション種別）に依存するため，初期条件を変更して算出された 1,000 万ケースの平均値をプロットした．

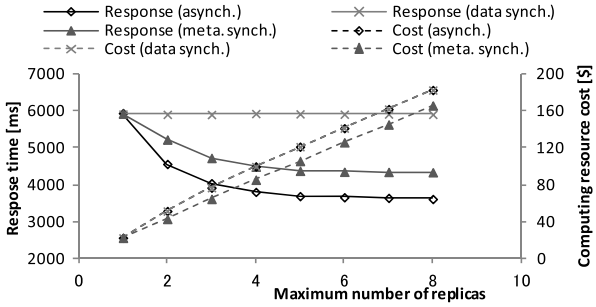

図 4 最適な配置構成における応答時間と計算資源コスト
Fig. 4 Response time and computing-resource cost in optimal organization.

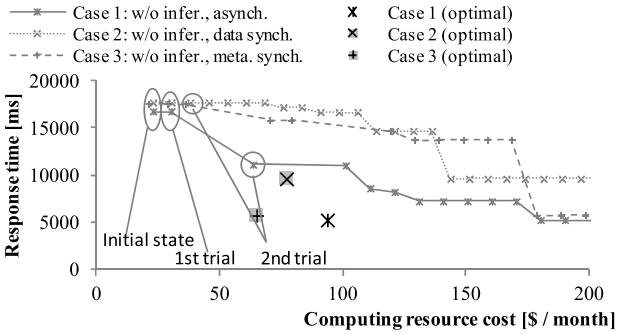

図 5 各ケースにおける繰返し配置による応答時間と計算資源コストの変化（ケース 1〜3）
Fig. 5 Response time and computing resource cost in each case with repetitive replication (Case1〜3).

4.2 評価結果

（1）応答時間と計算資源コストの最良値

本評価環境において改善可能な上限を示す．**図 4** においては，最適な（応答時間が最小となる）配置構成における応答時間と計算資源コストの平均値である．

図 4 から以下の点が分かる．

- 緩い一貫性とメタデータ同期においては，複製数の増加にともない，応答時間が減少した．また，その減少幅が徐々に縮小し，複製数が 5 以上では応答時間はほぼ一定となった．
- 強い一貫性においては，複製数の上限の変化に対し，応答時間は一定であった．
- 計算資源コストにおいては，増加の幅が一貫性制御の種類によって異なるが，いずれの場合でも複製数の増加にともなって増加した．

以上より，コストの増加を抑えつつ，応答時間を向上させるうえで，複製数は要求されるデータの一貫性によって変更されるべきであることが分かる．

（2）応答時間の向上のばらつき

各ケースにおける応答時間と計算資源コストの評価結果を**図 5**，および，**図 6** に示す．図 5 においては，ベイズ推定を未適用としたケース 1〜3 を示し，図 6 においてはベイズ推定を適用したケース 4〜6 を示す．ベイズ推定を未

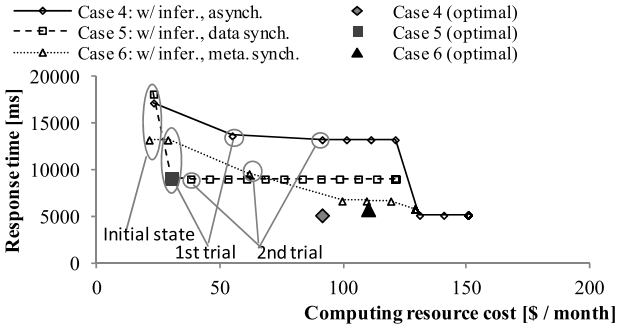

図 6 各ケースにおける繰返し配置による応答時間と計算資源コストの変化（ケース 4〜6）

Fig. 6 Response time and computing resource cost in each case with repetitive replication (Case4〜6).

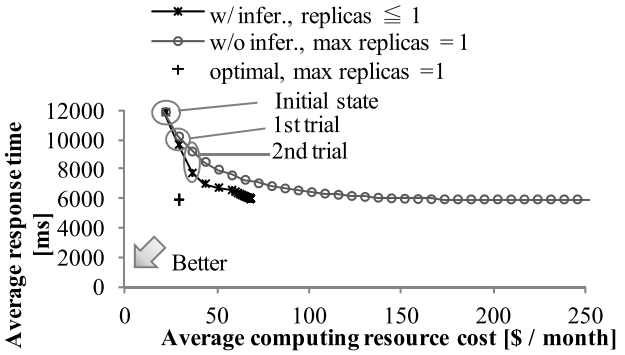

図 7 各ケースにおける繰返し配置による応答時間と計算資源コストの変化（複製数の上限：1）

Fig. 7 Response time and computing resource cost in each case with repetitive replication (Max replicas = 1).

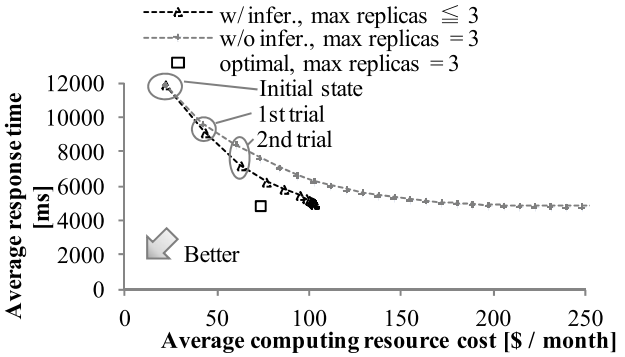

図 8 各ケースにおける繰返し配置による応答時間と計算資源コストの変化（複製数の上限：3）

Fig. 8 Response time and computing resource cost in each case with repetitive replication (Max replicas = 3).

適用の場合，配置されていない，すなわち，応答時間が評価されていないデータセンタの中から等しい確率で配置先を選択した．端末最寄りのデータセンタを優先的に選ぶ手法も考えられるが，端末最寄りのデータセンタに分散配置することによって応答時間が必ずしも最小にならないためである．なお，すべてのケースにおいて複製数の上限を 3 とした．

試行回数が増えるにつれて，応答時間が減少し，計算資源コストが増加した．いずれの試行においても，応答時間の増加が見られないが，これは試行前後の配置構成を比較して応答時間が短い構成を採用しているためである．強い一貫性が要求されるケースにおいては，分散配置することで応答時間が悪化するため，複製数が 2 以上にならなかった．図 5 と図 6 から以下の点が分かる．

- いずれのケースにおいても，試行回数が増加するにつれて，応答時間は単調に減少しなかった．また，応答時間が収束するまでに要する試行回数はケースごとに異なった．
- 収束時の応答時間は，各ケースでばらばらであった．条件を確認したところ，これは端末位置の分布と要求される一貫性の強さに起因していた．
- 最適構成の推定機能が停止を判定したときの試行回数はケースによって様々であった．また，停止と判定した後，試行を繰り返した場合においても，大部分のケースにおいて応答時間はそれ以上減少しなかった．

1 つ目と 2 つ目の点は，試行の停止を判定する際に特定の応答時間や試行回数で判定することが不適切であるため，試行の停止判定が容易でないことが分かる．一方，3 つ目の点は，最適構成の推定機能が適切に停止判定していることを示唆している．

（3）ベイズ推定の応答性向上への効果

最適構成の推定機能の有効性を定量的に評価する．応答時間と計算資源コストの平均値を**図 7**，および，**図 8** に示す．図 7 においては複製数の上限を 1 とし，図 8 においては複製数の上限を 3 とした．図 7 と図 8 から以下の点が分かる．

- 応答時間を収束するのに要する計算資源コストは，最適構成の推定機能を適用することによって，推定機能を適用しない場合に比べて，複製数の上限が 1，3 の場合共に約 70%削減された．また，最適構成の推定機能を適用した場合の計算資源コストは，最適解に比べて，複製数の上限が 1 の場合に 130%，複製数の上限が 3 の場合に 40%増加した．
- 一方，収束時の応答時間は，同推定機能の適用によって増加した．その増分は，最適解と比べ，複製数の上限が 1 の場合に 1%であり，上限が 3 の場合に 5%であった．なお，応答時間の増加の原因は，応答時間が完全に収束する前に停止判定してしまうケースがあったためである．
- 応答時間がほぼ収束するまでに要する試行回数が，同推定機能の適用によって減少した．この理由は，同推定機能の適用により，最初の数回の試行における応答時間の減少幅が，適用しない場合に比べて大きかったためである．
- 応答時間がほぼ収束している近傍において，1 回の試

行あたりの計算資源コストの増加幅が，同推定機能の適用によって減少した．この理由は，応答時間削減の可能性が十分に小さいケースにおいて，配置変更が停止されたためである．

以上の点より，最適構成の推定機能が，配置変更の回数，および，1回の配置変更に要する計算資源コストの平均値の両方を削減し，その結果計算資源コストが削減されたことが分かる．また，同推定機能の停止判定よって収束時の応答時間が1〜5%増加するものの，計算資源コストは70%抑制された．また，最適解と比べると，収束時の応答時間が1〜5%だけ大きく，計算資源コストは40〜130%だけ大きい値となった．

（4）評価結果の制限と考察

本評価における制限事項と考察を述べる．

(a) 配置変更時のサービスの一時的な停止

インメモリに格納されるデータの同期を必要とするアプリケーションの構成要素を移動する際，サービスの一時的な停止が必要になるが，本評価においては，サービスの一時的な停止は考慮されていない．

一方，インメモリに格納されるデータの同期が不要であるアプリケーションの場合，以下の手順により，サービスを一時的に停止する必要はない．その手順とは，まず，構成要素を複製して起動させ，複製元と複製先の構成要素をアクティブ・アクティブ構成として動作させる．次に，端末のアクセス先を複製元から複製先に変更する．すべての端末のアクセス先が変更した後，複製元の構成要素を停止する．

(b) 配置変更時のセッションの切断

アプリケーションの構成要素の配置変更後に，クラウド上のアプリケーションとhttpセッションを張る端末が接続先を変更すると同セッションは切断される．そのため，端末の接続先の変更はセッションを再接続するタイミング等に行う必要がある．また，構成要素の移動時には，移動元の構成要素はすべての端末とのセッションが終了した際に行う必要がある．本評価においては，このようなセッションの再接続が行われることを前提としている．しかし，永続的にセッションを維持する端末が存在し，かつ，同セッションを切断することが許されないアプリケーションにおいては，移動元の構成要素を停止できない．このような場合，構成変更できないため，応答時間が評価結果のとおりに削減されない．

(c) 配置変更時の負荷急増にともなう応答時間の劣化

本評価においては，クラウドのリソースが十分に大きく，アプリケーションの構成変更による応答時間への影響を無視できることを前提とした．しかし，クラウドのリソースが構成変更されるアプリケーションによって消費されるリソース量を無視できるほど大きくない場合，アプリケーションの構成変更にともなうアクセス数増加により，処理負荷や通信量が増加し，応答時間が劣化する可能性がある．

(d) 従来方式との比較

本評価においては，最適解や推定アルゴリズムを適用しない場合の応答時間と計算資源コストは比較されているが，1章に示す従来方式との比較が含まれない．パブリッククラウドにおける性能のばらつきにより，従来方式が最適解とどの程度乖離が生じるかを示すためには，従来方式の評価が有効である．今後の課題として，従来方式と最適解との評価があげられる．

5. おわりに

一時的な計算資源の消費量の増加を許容することにより，暫定的な配置変更と実測を繰り返し，最適な配置構成を自律的に探索するWATS（Wide area tentative scaling）を提案した．WATSにおいては，試行を繰り返すことにより，応答時間を段階的に削減する一方，計算資源コストが増加する．そこで，計算資源コストの増加を抑制するため，ベイズ推定を活用して，次の試行において評価すべき配置構成を推定し，停止判定を行う最適構成推定機能を提案した．シミュレーション評価の結果，WATSを適用すると，40〜130%の計算資源コストの一時的な増加を許容することにより，応答時間が平均で最適解の1〜5%程度大きな値に収束する見込みを得た．

参考文献

[1] Miller, R.B.: Response time in man - computer conversational transactions, *Proc. AFIPS Fall Joint Computer Conference*, Vol.33, pp.267–277 (Dec. 1968).

[2] Nielsen, J.: *Usability Engineering*, Morgan Kaufmann, Morgan Kaufmann Publishers Inc., San Francisco (1994).

[3] Yusoh, Z.I.M. and Tang, M.: A penalty-based genetic algorithm for the composite SaaS placement problem in the cloud, *IEEE Congress on Evolutionary Computation*, Barcelona, Spain, pp.1–8 (July 2010).

[4] Zhu, X., Santos, C.A., Beyer, D., Ward, J. and Singhal, S.: Automated application component placement in data centers using mathematical programming, *Int. J. Network Management*, Vol.18, No.6 (2008).

[5] Chang, F., Viswanathan, R. and Wood, T.L.: Placement in clouds for application-level latency requirements, *Proc. IEEE Intl. Conf. on CLOUD*, Hawaii, USA, pp.327–335 (June 2012).

[6] Agarwal, S., Dunagan, J., Jain, N., Saroiu, S. and Wolman, A.: Volley: Automated data placement for geo-distributed cloud services, *7th USENIX Symp. on Networked Systems Design and Impl* (*NSDI*), Berkeley, CA, USA, p.2 (Jan. 2010).

[7] Anke, J., Wolf, B., Hackenbroich, G. and Kabitzsch, K.: A planning method for component placement in smart item environments using heuristic search, *DAIS*, ser. LNCS, Vol.4531, Springer (2007).

[8] Urgaonkar, B., Rosenberg, A.L. and Shenoy, P.J.: Application placement on a cluster of servers, *Int. J. Found. Comput. Sci.*, Vol.18, No.5 (2007).

[9] Li, B., Li, J., Huai, J., Wo, T., Li, Q. and Zhong,

L.: Enacloud: An energy-saving application live placement approach for cloud computing environments, *IEEE Intl. Conf. on Cloud Computing*, Bangalore, India (Sep. 2009).
[10] 首藤裕一，波戸邦夫：パブリッククラウドの時間的性能変動及び性能分布，信学技報，Vol.113, No.206, IN2013-60, pp.13–18 (2013).
[11] Yabusaki, H., Nakagoe, H., Murayama, K. and Kato, T.: Wide Area Tentative Scaling (WATS) for Quick Response in Distributed Cloud Computing, *Proc. IEEE INFOCOM'14 Conference Workshop*, CrossCloud, pp.31–36 (Apr. 2014).
[12] 藪崎仁史，中越洋一，村山耕一，加藤崇利：広域分散クラウドにおいて応答性を段階的に向上するアプリケーションとデータの繰り返し配置方式，信学総大，D-23-2, No.206, IN2013–60, pp.13–18 (Mar. 2014).
[13] Michelotti, L. and Ostiguy, J.-F.: CHEF: An Interactive Program for Accelerator Optics Calculations, *IEEE Proc. PAC*, pp.988–990 (May 2005).
[14] Fluentd: Open Source Log Management, available from ⟨http://fluentd.org/⟩.
[15] Zhang, H.: The Optimality of Naive Bayes, *Proc. 17th Int. Florida Artificial Intelligence Research Society Conference FLAIRS*, Florida, USA (May 2004).
[16] The PingER Project, available from ⟨www-iepm.slac.stanford.edu/pinger/⟩ (accessed 2014-02-21).
[17] 沖津 潤，平島陽子，朝 康博，加藤 猛，齊藤達也：環境配慮型データセンタ向け空調連係 IT 負荷配置最適化方式，信学会論文誌 D，Vol.J95-D, No.4, pp.919–927 (2012).
[18] Juve, G. et al.: Scientific workflow applications on Amazon ec2, *Proc. Workshop on Cloud-based Services and Applications in conjunction with e-Science*, pp.59–66 (2009).
[19] Katz, D.S. et al.: A comparison of two methods for building astronomical image mosaics on a grid, *International Conference on Parallel Processing* (*ICPP'05*), pp.85–94 (2005).
[20] Southern California Earthquake Center: Community Modeling Environment (CME), available from ⟨http://www.scec.org/cme⟩.
[21] USC Epigenome Center, available from ⟨http://epigenome.usc.edu⟩.
[22] Hashemian, R. *et al.*: Improving the scalability of a multi-core web server, *Proc. ACM ICPE*, Prague, Czech Republic, pp.161–172 (April 2013).
[23] Hazelhurst, S.: Scientific computing using virtual high-performance computing: A case study using the Amazon elastic computing cloud, *Proc. 2008 Annu. Research Conf. SAICSIT*, pp.94–103 (Oct. 2008).

付　　録

A.1 評価環境に適用されたモデル

4 章のシステム評価に用いたシミュレーション環境において，適用した応答時間と計算資源コストのモデルを示す．なお，後述のモデル化は，シミュレーション環境を構築するために行ったが，実際のクラウド環境では，モデル化に用いるすべての値を分析することは困難である．そのため，評価を行う際，提案方式はこれらのモデルの情報は未知として配置構成を推定した．

表 A·1　計算資源コストのモデルに用いる記号
Table A·1　Symbols used for modeling computing resource cost.

記号	説明
Aio_i	データセンタ i における I/O トランザクション数
$An_{k,i}$	端末 k とデータセンタ i 間のデータ転送量
As_i	データセンタ i に保存されるデータ量
C	計算資源コスト
$Cc_{i,j}$	データセンタ i とデータセンタ j 間の通信コスト
$Cc'_{k,i}$	端末 k とデータセンタ i 間の通信コスト
Ci_i	データセンタ i におけるインスタンスコスト
Cr_t	t 回目の試行に要するコスト
Cs_i	データセンタ i におけるストレージコスト
C_{total}	計算資源コストと繰り返し配置に要するコストを併せた総コスト
N_i	データセンタ i におけるリザーブドインスタンス数
N'_i	データセンタ i におけるオンデマンドインスタンス数
Pi_i	データセンタ i におけるリザーブドインスタンス価格
Pi'_i	データセンタ i におけるオンデマンドインスタンス価格
Pn_i	データセンタ i における通信の価格
Ps_i	データセンタ i におけるデータ保存の価格
Ps'_i	データセンタ i における I/O トランザクションの価格
T_i	データセンタ i においてオンデマンドインスタンスを利用する平均時間

A.1.1 応答時間のモデル

応答時間は，端末とデータセンタ間の通信時間，データセンタ間の通信時間，および，データセンタにおける処理時間の和とする．

（1）端末とデータセンタ間の通信時間

端末 k とデータセンタ i 間の通信時間 $\rho'_{k,i}$ は，セッション確立等に要するオーバヘッドとデータ送受信時間の和で計算される．通信時間 $\rho'_{k,i}$ を式 (A.1) に示す．

$$\rho'_{k,i} = d_{i,k} \times s + \frac{\delta}{b_{i,k}} \tag{A.1}$$

（2）データセンタ間の通信時間

データセンタ間の通信時間は，データの一貫性制御に依存する．本論文では，強い一貫性制御，緩い一貫性制御，および，メタデータ同期の種類を考慮する．メタデータ同期とは，データを同期せずにローカルに保存し，メタデータのみ同期する．メタデータとは，最新データにアクセスする際に必要となる，データの保存場所とバージョン情報である．

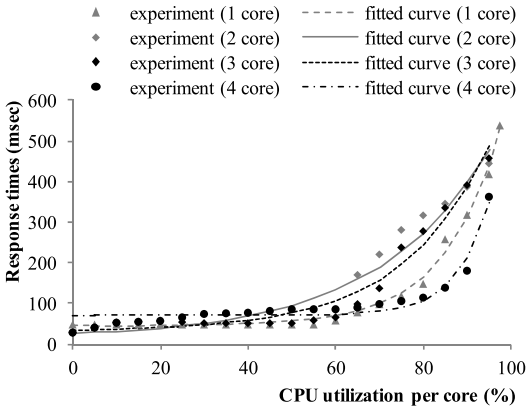

図 **A·1** 演算処理遅延のモデル
Fig. A·1 Model of processing delay.

強い一貫性制御の場合，データセンタ i とデータセンタ j との間の通信時間 $\rho''_{i,j}$ は，端末とデータセンタ間の通信時間と同様に，式 (A.2) で計算させる．

$$\rho''_{i,j} = d_{i,j} \times s' + \frac{\delta}{b_{i,j}} \tag{A.2}$$

弱い一貫性制御の場合，端末への応答後に，他のデータセンタにデータを送受信する．データセンタ間の通信時間は，端末の応答時間に影響しないため，式 (A.3) で計算される．

$$\rho''_{i,j} = 0 \tag{A.3}$$

次に，メタデータ同期について説明する．グローバルに分散するデータセンタをつなぐ広域網においては，RTT が数十～数百ミリ秒である．TCP スループットは，一部の地域を除くと 100 kbps 以上である．一方，メタデータのデータ量 δ が一般的に数十バイトと小さい．式 (A.2) において，$d_{i,j} \times s' \gg \delta / b_{i,j}$ であるため，通信時間 $\rho''_{i,j}$ は式 (A.4) で計算される．

$$\rho''_{i,j} \approx d_{i,j} \times s' \tag{A.4}$$

(3) データセンタにおける処理時間

処理時間は CPU 使用率の増加にともない指数的に増加する [22] ため，本論文では式 (A.5) に示すとおりにモデル化した．

$$\rho'''_i = \exp(a \times x + b) + c \tag{A.5}$$

ここで，x は CPU 使用率であり，a，b，c は比例定数（以下，演算処理時間係数と呼ぶ）である．文献 [22] における評価結果に基づいて，演算処理時間係数 a，b，c を最小二乗法で算出した．比較した結果を**図 A·1** に，比例定数の値と評価結果との相対誤差の大きさを**表 A·2** に示す．評価結果と数式は，1 コアにおいて 0.08，2 コア以上において 0.22～0.3 の相対誤差の範囲で一致した．

以上より，端末 k がデータセンタ i にデータ処理要求する際，その応答時間 $\rho_{k,i}$ は，式 (A.6) で計算させる．

$$\rho_{k,i} = \rho'_{k,i} + \rho''_{i,j} + \rho'''_i \tag{A.6}$$

表 **A·2** 処理時間のモデルにおける係数と誤差
Table A·2 Constants and errors in processing time model.

	a	b	c	relative error
1 core	7.94	-1.56	45.17	0.08
2 core	3.89	2.44	14.96	0.3
3 core	5.12	1.27	30.85	0.22
4 core	14.09	-7.74	71.69	0.23

A.1.2 計算資源コストのモデル

クラウド利用にともなう計算資源のコストは，一般に，インスタンスコスト，ストレージコスト，および，通信コストに分類される．通信コストは，さらに端末とデータセンタ間の通信コストとデータセンタ間の通信コストに分類される．大手クラウドサービスの料金体系 [23] にならい，各コストをモデル化した．

(1) インスタンスコスト

インスタンスには，一定期間で契約するリザーブドインスタンスと時間帯で契約するオンデマンドインスタンスがある．データセンタ i におけるインスタンスコストは，これらの和として，式 (A.7) により計算される．

$$Ci_i = N_i * Pi'_i + N'_i \times T_i \times Pi'_t \tag{A.7}$$

なお，オンデマンドインスタンスは，試行におけるアプリケーションの暫定的な配置に利用される．

(2) ストレージコスト

ストレージコストは，データ格納にともなうコストと I/O 処理にともなうコストの和であり，式 (A.8) により計算される．

$$Cs_i = As_i \times Ps_i + Aio_i \times Ps'_i \tag{A.8}$$

(3) 通信コスト

端末とデータセンタ間の通信コストには，端末のアクセス網のコスト，広域網のコスト，データセンタ入口のコストに分類される．アクセス網のコストと広域網のコストは，近年では固定課金が多い．固定課金の場合，他の用途との間で負担割合は多様である．また，コストを負担するのは，サービスプロバイダではなく，エンドユーザである．そのため，本論文では，通信コストは，データセンタ入口のコストのみ考慮する．端末とデータセンタ間通信コストは，式 (A.9) により計算される．

$$Cc'_{k,i} = An_{k,i} \times Pn_i \tag{A.9}$$

同様に，データセンタ間の通信コストは，式 (A.10) により計算される．

$$Cc_{i,j} = An'_{i,j} \times (Pn_i + Pn_j) \tag{A.10}$$

以上より，試験的な配置にともなうコストを除く，計算資源の総コストは式 (A.11) で計算される．

$$C = \sum_{i \in I}(Ci_i + Cs_i) + \sum_{i \in I}\sum_{j \in I, j \neq i} Cn_{i,j} + \sum_{k \in K}\sum_{i \in I} Cn'_{k,i} \quad \text{(A.11)}$$

（4）繰返し配置にともなうコスト

試験的な配置にともなうコストは，配置にともなう通信コスト，配置先で一時的に利用するストレージコストとインスタンスコストの和である．ストレージコストは（1）と同様であり，インスタンスコストは（2）のオンデマンドインスタンスに相当する．配置にともなう通信コストは（3）のデータセンタ間の通信コストにおいて，$An'_{i,j}$ を配置するデータ量として計算される．試行を t 回繰り返す際の繰返し配置にともなうコストは式 (A.12) により計算される．

$$C_{total} = C + \sum_{l=1}^{t} Cr_l \quad \text{(A.12)}$$

よって，計算資源のコストは式 (A.13) で計算される．

$$C = \sum_{i \in I}(Cs_i + Cs'_i) + \left(\sum_{i \in I}\sum_{j \in I, i \neq j} Cc_{i,j} + \sum_{i \in I}\sum_{k \in K} Cc'_{i,k}\right) \quad \text{(A.13)}$$

藪崎 仁史

2008 年日立製作所入社．日立製作所情報通信イノベーションセンタ所属．現在，クラウドの運用管理技術の研究，および，広域網の制御技術の研究に従事．

中越 洋

2005 年日立製作所入社．日立製作所情報通信イノベーションセンタ所属．現在，クラウドの運用管理技術の研究に従事．

村山 耕一

2002 年日立製作所入社．日立製作所情報通信イノベーションセンタ所属．現在，クラウドの運用管理技術の研究に従事．

加藤 崇利

1995 年日立製作所入社．日立製作所情報・通信システム社経営戦略統括本部所属．現在，情報・通信システムの事業戦略と技術戦略の企画立案に従事．