分散 Web システムにおけるオートスケールアルゴリズムの改良と評価

14G481　堀内 晨彦（最所研究室）

クラウド環境において負荷量に応じて動的に仮想キャッシュサーバ数を増減させることで，応答性を確保しつつ運用コストを低減する分散 Web システムのための負荷量の移動平均や増加率を用いたオートスケールのアルゴリズムの改良や，負荷の予測に基づいて予め仮想キャッシュサーバを起動する機能の実装と評価について述べる．

1　はじめに

近年，クラウドの発展にともなって負荷分散に用いるキャッシュサーバを容易に構築できるようになったが，その台数を負荷量に応じて動的に増減できなければ，応答の向上が不十分であったり無駄な運用コストが発生したりする．そのため，クラウド上で作成した仮想マシンを振分先に追加したり，過負荷の際に自動で仮想マシンを追加したりする LBaaS（Load Balancer as a Service）[1] が提供されている．しかし，複数のクラウドを横断的に利用できなかったり，負荷量とするパラメタが Web サービスに適していなかったり，キャッシュサーバの利用を想定していなかったりする．このような問題を解決するため，本研究ではオープンソースのソフトウェアロードバランサと Web サーバソフトウェアを用いた分散 Web システムを開発している．

2　分散 Web システムについて

本システムは図 1 に示すように，以下の処理を行う拡張ロードバランサを開発しソフトウェアロードバランサと協調動作させる．監視及びキャッシュサーバの増減は拡張ロードバランサで，リクエストの制御はソフトウェアロードバランサで行う．

A. オリジンサーバと仮想キャッシュサーバ群の負荷監視
B. 負荷量に応じた仮想キャッシュサーバの起動・停止
C. 仮想キャッシュサーバの起動・停止に合わせたソフトウェアロードバランサの振分先の更新

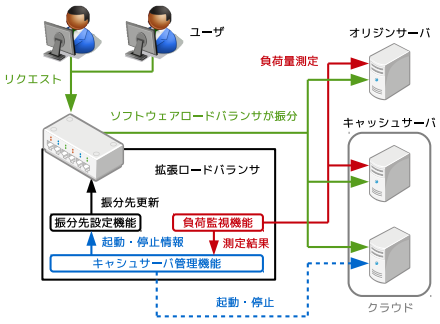

図 1:　分散 Web システムの概要

3　負荷量の移動平均による改良

卒業研究 [2] では，同時リクエスト数が変化していないにも関わらずスケールアウトとスケールインが繰り返される問題が発生した．その原因が負荷量の測定値のバラツキにあると考え，負荷量の移動平均を用いて改良することにした．

負荷量の測定結果をリングバッファに保存し，その末尾から指定個数（平均区間）の平均を求める機能を実装した．スケールアウトではより早く負荷分散を行うために平均区間を短く，スケールインでは上記の問題の発生を防ぐため平均区間を長く設定した．詳細は省くが，スケールアウトで 4 区間・スケールインで 8 区間の移動平均で問題は発生しなくなったが，スケールアウトが遅れるという副作用も確認できた．

4　負荷量の増加率による改良

キャッシュサーバ管理機能を有効にし実際に仮想キャッシュサーバの起動・停止を行ったところ，負荷量が閾値を超えてから新規の仮想キャッシュサーバが起動するまでの間の過負荷により応答時間が悪化する問題が発生した．この過負荷を低減するため，負荷量の増加率に基づいてスケールアウト時に起動する仮想キャッシュサーバの台数を決定することにした．9 秒前の負荷量に対する現在の負荷量の割合を増加率とし，切り上げて整数化した値を起動する仮想キャッシュサーバの台数とする．

この機能をキャッシュサーバ管理機能に実装し，実験を行った結果を図 2 に示す．負荷量が急激に増加しているときに 2 台同時に起動しており，その結果負荷量が 0.65 と大きく減少している．なお，1 台ずつ起動したときは 0.5 であった．この結果は，仮想キャッシュサーバを複数台同時に起動することで，振り分け開始時の過負荷がより解消されたことを示している．350 秒以降で仮想キャッシュサーバの停止が行われていないのはハイパーバイザでエラーが発生したためであり，その対応が必要である．

5　負荷量の予測による改良

負荷の予測に基づいて予め仮想キャッシュサーバを起動することで，過負荷になる前にスケールアウトすることにした．現在の負荷量の増加がそのままの割合で続いたと仮定し，仮想キャッシュサーバを起動してから振り分けを開始するまでにかかる時間 S 秒後の負荷量を予測し，そのとき必要な台数を求め，起動を開始する．起動する台数 M は $AVGOR_n$，$AVGOR_{n-1}$，Th_{high}，N を現在の負荷量，9 秒前の負荷量，スケールアウト時の閾値，稼動台数として以下の式で求める．式の右辺の第 1 項の分母は S 秒後の負荷量を予測した値であり，それを Th_{high} で割ることで必要台数を求め，そこから稼働中の仮想キャッシュサーバの台数を引くことで M が求まる．

$$M = \frac{\frac{AVGOR_n - AVGOR_{n-9}}{10} \times S + AVGOR_n}{Th_{high}} - N - 1$$

負荷予測機能をキャッシュサーバ管理機能に実装し実験を行った．詳細は省くが，スケールインによる一時的な負荷量の微増にも予測機能が反応しスケールアウトした．そこで，稼動台数に応じた閾値の決定機能と，予測機能の有効化・無効化の切替機能を実装することにした．稼動台数が少ない場合は同時リクエスト数の増減による影響が大きいため，閾値を高く設定し予測機能を有効にすることで早期にスケールアウトを図る．反対に稼動台数が多い場合は同時リクエスト数の増減による影響が小さいため，負荷予測機能を無効にし適切にスケールインが行われるようにする．

これらの機能をキャッシュサーバ管理機能に実装し，実験を行った結果を図 3 に示す．負荷量のグラフの推移が，同時リクエスト数のグラフの推移に追従していることが分かる．稼動台数が少ない場合に早期にスケールアウトするようにしたため，全クライアントがリクエストしている状態での平均稼働率は約 0.7 で安定している．また，このときのクライアントへの応答時間を図 4 に示す．応答時間推移が，稼動台数の推移に追従していることが分かる．100 秒〜200 秒付近で応答時間が 5〜6 秒に達しているが，その後はスケールアウトによって改善され約 1.5 秒と安定して推移している．これらの結果は，負荷予測機能とアルゴリズム切替機能による改良で改善されたことを示している．

6　システムの可視化アプリケーション

拡張ロードバランサは CSV 形式のログを出力するが，負荷量の推移が分かり難い，仮想キャッシュサーバの起動・停止が確認できないという問題がある．そこで，分散 Web システムの動作をリアルタイムで可視化する Web アプリケーションを開発することにした．

Web アプリケーションは拡張ロードバランサとのインタフェースとなるバックエンドと，グラフ表示などを行うフロントエンドから成る．それぞれの通信には WebSocket[3] を使用し，バックエンドが負荷量と仮想マシンの稼動状態を受信すると，フロントエンドがその内容を表示する．図 5 に動作例を示す．画面の左側が各 Web サーバの負荷量のグラフ，右側が各仮想マシンの稼動状態を示す．図より，オリジンサーバと仮想キャッシュサーバ 1〜3 台目にリクエストが振り分けられ，4 台目が起動中であることが分かる．この Web アプリケーションを用いることで，オートスケールアルゴズムのデバッグをより簡単に行うことが可能になった．

7　おわりに

分散 Web システムにおける仮想キャッシュサーバの起動・停止の繰り返し，起動時の過負荷による応答時間の悪化という問題を解決するため，負荷量の移動平均や増加率を用いたオートスケールアルゴリズムの改良，負荷の予測に基づいて仮想キャッシュサーバを起動する機能の実装を行った．実験により，これらの機能によって同時リクエスト数に追従してオートスケールが行えたことを確認した．更に，クライアントへの応答時間を概ねユーザの許容範囲に改善できたことも確認した．

今後の課題として，ヘテロなクラウド環境への対応や，キャッシュサーバ管理機能の改良，実用的な環境での評価実験などがある．

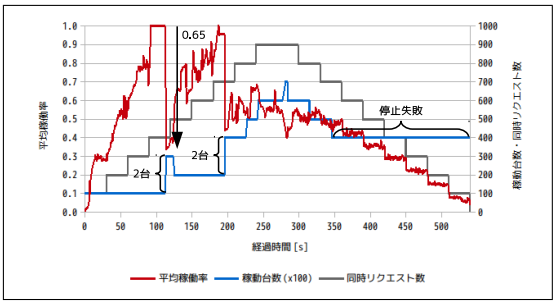

図 2:　複数台起動機能の実験結果

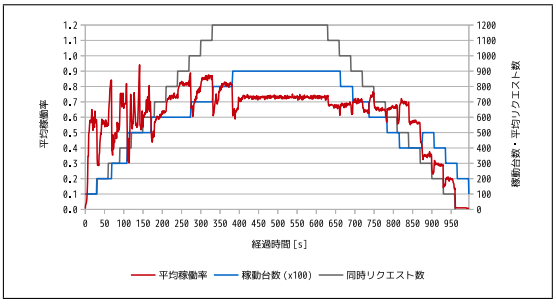

図 3:　負荷予測機能の実験結果

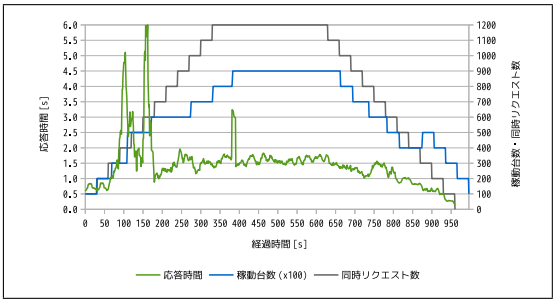

図 4:　負荷予測機能の応答時間

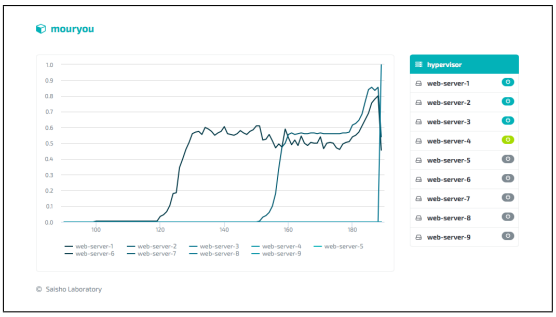

図 5:　可視化アプリケーションの動作例

参考文献

[1] Rahman M., Iqbal S. and Gao J., “Load Balancer as a Service in Cloud Computing,” in *Proc. IEEE 8th Int. Symp. on Service Oriented System Engineering*, Apr. 2014, pp. 204-211.

[2] 堀内晨彦, “クラウドに適した Web システムの負荷監視機能の改善と評価”, 香川大学 卒業論文, 2014

[3] RFC 6455 - The WebSocket Protocol, `https://tools.ietf.org/html/rfc6455`