多VM環境における高級言語型Webアプリケーションフレームワークの性能解析

越智　俊介†　　山口　実靖‡

†工学院大学大学院　工学研究科　電気・電子工学専攻

‡工学院大学　工学部　情報通信工学科

1. はじめに

近年，SNSやブログなどの普及によりWebアプリケーションの需要が増加し，多数のWebアプリケーションが作成されるようになった．

本稿では，高級言語型Webアプリケーションフレームワークの使用による高速な開発と，各Webアプリケーションに高い独立性を提供することの実現を目指し，そのための基礎調査として多VM環境における高級言語型Webアプリケーションフレームワークの性能について考察する．

2. 開発の高速性 と 高い独立性 を有する Web アプリケーション環境

Webアプリケーションの普及により，短い納期での高速な開発が求められるようになってきている．高速なWebアプリケーション開発を実現するための環境の一つとしてRoR (Ruby on Rails)[1] があり，これを用いることにより極めて高速にWebアプリケーションを開発することができる．

また，多数のWebアプリケーションを運用する手法として，Webアプリケーション群が単一のOSを共有する手法と，各アプリケーションが独占的にOSを有する手法が考えられる．前者の方が効率的に計算資源を使用できるが，後者の方が他のアプリケーションとの運用上の衝突を回避できるなどの高い独立性が得られ，少ない工数での開発が期待できる．

本研究では少ない工数での開発を目的として，(1) 各Webアプリケーションに独占的にOSを有する環境を与える，(2) 高級言語型Webアプリケーション開発フレームワークRoRを用いる，の2点を前提とし，これを支援する環境の構築を目指す．具体的には，各Webアプリケーションに VM(仮想計算機)を独占的に与え，それらの負荷が高くないとの前提のもとに多数のVMを単一の物理計算機で動作させる環境を想定し，これの性能向上を目指す．

Performance Evaluation of Web Application Framework using High-Level Programming Language in Virtualized Environment

Shunsuke OCHI †, Saneyasu YAMAGUCHI ‡

† Graduate School of Electrical and Electronics Engineering, Kogakuin University

‡ Department of Information and Communications Engineering, Kogakuin University

3. Ruby on Rails

Ruby on Railsは2004年に登場した高級言語Rubyを用いるWebアプリケーションフレームワークである．特徴としては，MVC (Model-View-Controller) アーキテクチャをサポートしていること，RDBMSを用いるCRUD処理のコードを自動生成できることなどがあげられる．

ただし，高級言語やRDBMSを用いるため要求される計算資源は少なくなく，得られる性能は高くない．

4. 仮想化技術を用いるサーバ統合

データセンターをはじめとする多くの企業で多数のサーバコンピュータが稼動しており，それらの消費電力や管理負荷が情報システムの大きな問題の1個となっている．また，それらのサーバの多くは利用率が低いことが多く，サーバ資源の効率的な活用が課題となっている．この問題に対する解決策として，仮想化技術によるサーバ統合がある．仮想化技術を用いて複数のサーバOSやサーバソフトウェアを単一の物理計算機上で動作させることにより，消費電力の削減や，サーバ資源の効率的な運用が可能となる．

本研究では，各Webアプリケーションに高い独立性を提供するために各Webアプリケーションに独占的に仮想計算機環境を与える．

5. 性能評価

本研究で想定した高い独立性，高い生産性を有する環境を少数の物理計算機で実現したときの性能を調査するため，独占的にVMを保持するRailsアプリケーションを単一の物理計算機上に多数起動させ，その応答性能を測定した．

5.1 測定方法

測定方法は以下の通りである．1 台の物理計算機上に 1～20 台の VM を稼動させ，各 VM 上に Rails にて作成した Web アプリケーションを 1 つ動作させた．Rails Web アプリケーションは，RDBMS からデータを読み込みその内容を返す静的なものと，RDBMS に対して読み書きを行う動的なものの 2 種類を用いた．RDBMS には MySQL を使用した．上記環境にて，ホスト計算機から Rails Web アプリケーションに対して繰り返し HTTP 要求を送信し，Web アプリケーションのターンアラウンドタイムを測定した．リクエスト対象はランダムに選択した．また，ホスト OS における VM プロセスのスケジューリング優先度(nice 値)についても，初期値の 0 と，最低の 19 に変更して測定した．

測定に使用した計算機の仕様は表 1 の通り，VM の仕様は表 2 に通りである．

表 1．ホスト計算機の測定環境

CPU	Athlon64 3500+(2.2Ghz)
Memory	12GB
HDD	2TB
OS	Fedora12(2.6.31)
仮想化ソフト	KVM+QEMU

表 2．ゲスト計算機の測定環境

CPU	Athlon64 3500+(2.2GHz)
Memory	512MB
HDD	8GB
OS	Fedora12(2.6.31)

5.2 測定結果

各条件におけるターンアラウンドタイムとホスト OS におけるコンテキストスイッチ数を図 1,2 に示す．

図 1 において，VM の優先度が初期値(nice0)の状態では VM の数が増えるに従い Web アプリケーションの応答時間が大幅に増加している．これより，VM の優先度が初期値の場合は VM に負荷がかかっていなくても稼働 VM の数を増加させるだけで VM 性能が大きく低下することが分かる．これに対して VM の優先度が最低(nice19)の場合は，稼働 VM 数の増加に伴う Web アプリケーショの応答時間の増加が大幅に抑制されていることが分かる．

次にコンテキストスイッチ数を比較すると，VM の優先度が初期値の場合も最低の場合も稼働 VM 数が増加するとコンテキストスイッチ数が増加する傾向にあるが，VM の優先度が最低の場合はその増加が大幅に抑えられており，コンテキストスイッチの増加がアプリケーション応答時間増加の大きな理由の 1 個になっていると考えられる．

図 1 と図 2 を比較すると，稼働 VM 数の増加に伴いアプリケーション応答時間が増える点，

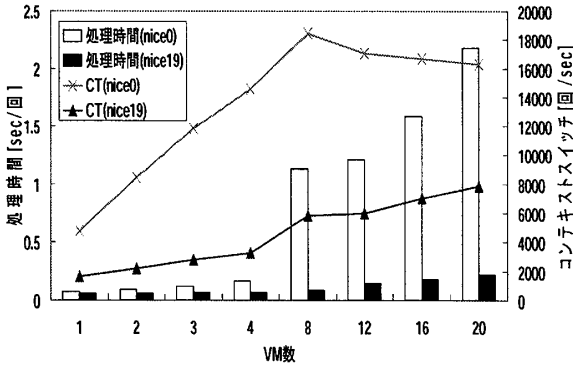

図 1　測定結果 (静的 Web アプリケーション)

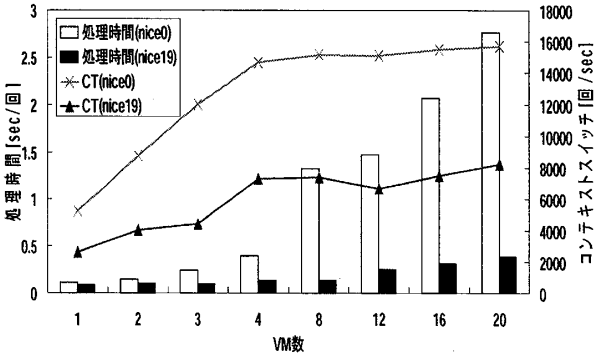

図 2　測定結果 (動的 Web アプリケーション)

コンテキストスイッチ数が増加する傾向にある点は共通であるが，RDBMS への書き込みを伴う図 2 の方がターンラウンドタイムが 16%～135%長くなっている．また，図 2 の方が少ない VM 数(3 あるいは 4)で応答時間が上昇している．

6. おわりに

本稿では高い生産性と高い独立性を有する Web アプリケーション環境として VM を占有的に有する環境で Rails を動作させる状況を想定し，単一の物理計算機上で多数の Web アプリケーションを起動させたときの性能の調査を行った．調査の結果，多くの VM を起動することにより性能が大きく劣化してしまうが，各 VM プロセスの優先度を下げて実行することにより劣化を抑制させることが可能であることが確認された．

今後は，上記調査結果を考慮した性能向上手法について検討していく予定である．

参考文献

[1]Ruby on Rails
http://rubyonrails.org/