情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

高集約サーバ統合環境における Web アプリケーションの性能に関する考察

越智俊介[†]　山口実靖[††]

ネットワーク技術の発展に伴い，ネットワークを通じてサービスを提供するクラウドコンピューティングに注目が集まっており，多種多様なサービスがネットワークを通じて行われるようになっている．クラウドコンピューティングでは仮想化技術が利用されており一台の物理計算機に複数の仮想マシンを動作させそれぞれの仮想マシンに個々の Web アプリケーションを動作させることが可能となっている．

近年では web アプリケーションの開発が増え，開発自体が複雑になっていたり，開発期間が以前と比べ短くなり Web アプリケーションの開発が難しくなってきている．この問題に対して，高級言語型 web アプリケーションフレームワークが登場し，生産性の高い開発と短い納期に適した開発環境が得られるようになった．

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とクラウドコンピューティングに着目し，高集約サーバ統合環境における仮想計算機の性能向上手法について考察を行った．具体的には，一台の物理計算機上に 100 台以上の多数の仮想計算機を稼働させる高集約環境を想定し，この環境における仮想計算機の性能について考察を行い，性能向上手法を提案した．性能評価の結果，提案手法により性能が向上することが確認された．

A Study on Performance of Highly Consolidated Virtualized Environment

Shunsuke Ochi[†]　Saneyasu Yamaguchi[††]

A large number of server computers are running in data centers or other companies. Their huge power consumption is one of the most important issue in recent computer systems. Server virtualization, or cloud computing with it, is a promising solutions for this problem. Most of these servers have only very low loads. Virtualization technology enables to consolidate these sever computers into one physical computer. In this paper, we explore a highly consolidated virtualized environment, in which more than 140 server computers are consolidated into one physical computer. This environment enables an application to have a dedicated OS. We present performance evaluation of the highly consolidated environment, and propose a performance improving method.

1. はじめに

近年高級言語型 Web アプリケーションフレームワークの登場により Web アプリケーションの開発が比較的容易になっている．しかし，複数の Web アプリケーションを単一の OS 上にて運用する場合，他の Web アプリケーションとの使用資源(ポート番号など)の衝突，使用ソフトウェアのバージョンの衝突などの問題が発生する．この問題は，各 Web アプリケーションに固有の OS を与えることにより回避することができる．

本研究では，各 Web アプリケーションに固有の VM と OS を与える環境を想定し，多数の VM を単一の物理計算機上で動作させた場合の性能について考察を行う．

2. 仮想化環境と Web アプリケーション

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) [(2)] がある．

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，開発の困難さが低いと考えられる．

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[(3) (4)]．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，この環境の性能向上を目指す．

3. 基本性能調査

本章では高い独立性，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．

[†] 工学院大学大学院 工学研究科 電気・電子工学専攻
Electrical Engineering and Electronics, Kogakuin University Graduate School
[††] 工学院大学 工学部 情報通信工学科
Department of Information and Communications Engineering, Kogakuin University

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

3.1 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 1〜144 台の VM を稼動させ，各 VM 上に Rails を用いて作成した 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リードオンリーのものである．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，コンテキストスイッチ数を測定した．リクエスト対象はランダムに選択した．

測定を行った環境を図 1 に示す．測定は計算機 1 と計算機 2 を用いて 2 回行った．計算機 1 の仕様を表 1 に，計算機 1 を用いた実験における VM の仕様を表 2 に示す．同様に，計算機 2 の仕様を表 3 に，計算機 2 を用いた実験における VM の仕様を表 4 に示す．仮想計算機としては KVM [4] を使用した．

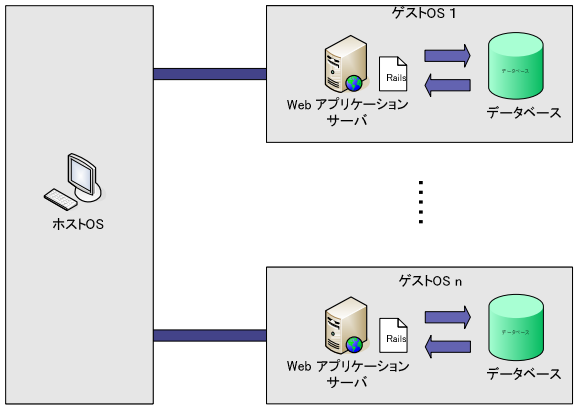

図 1　測定環境

表 1　計算機 1 の仕様

CPU	Athlon64 3500+(2.2GHz)1 コア
Memory	12GB
HDD	2TB
OS	Fedora12(2.6.31)
仮想化ソフト	KVM+QEMU

表 2　計算機 1 における VM の仕様

割当 CPU	1 コア
Memory	512MB
HDD	8GB
OS	Fedora12(2.6.31)

表 3　計算機 2 の仕様

CPU	Phenom II X4 920(2.8GHz)4 コア
Memory	12GB
HDD	4TB
OS	Fedora13(2.6.34.7)
仮想化ソフト	KVM+QEMU

表 4　計算機 1 における VM の仕様

割当 CPU	1 コア
Memory	256MB
HDD	15GB
OS	CentOS5.5(2.6.18)

3.2 測定結果

3.2.1 計算機 1

計算機 1 を用いた測定の結果を図 2，図 3 に示す．図 2 より，VM 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とが分かる．これにより VM にアクセス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．次に図 3 のコンテキストスイッチの結果より，稼働 VM の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ことが分かる．VM 数 32 では一秒あたり 30000 回以上のコンテキストスイッチが発生しており，これが大きな負荷になっていると考えられる．

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

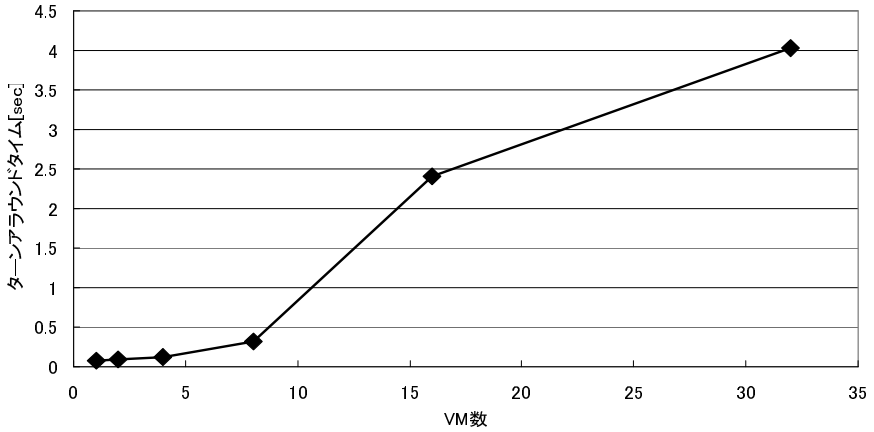

図 2　ターンアラウンドタイム

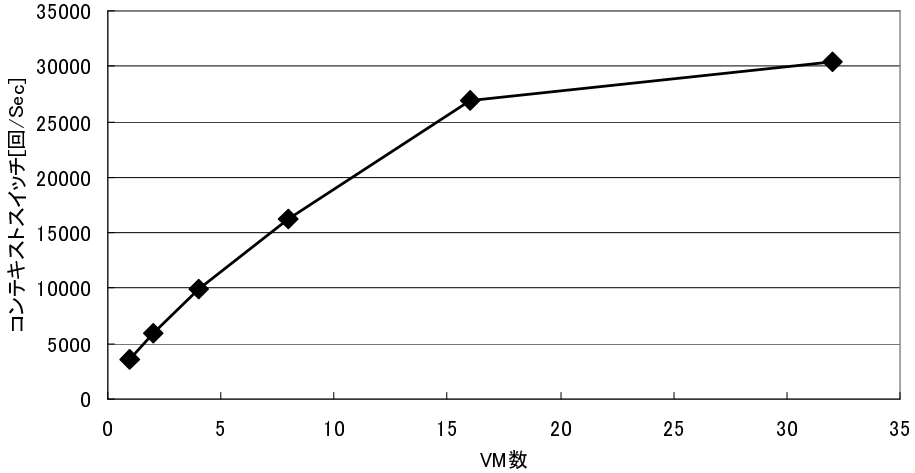

図 3　コンテキストスイッチ数

3.2.2 計算機 2 の結果

計算機 2 を用いた測定の結果を図 4，図 5 に示す．図 4 より，計算機 1 の実験と同様に VM 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おり，VM にアクセス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．また図 5 のコンテキストスイッチの結果より，稼働 VM の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ことが分かる．VM 数 144 では一秒あたり約 200000 回のコンテキストスイッチが発生しており，これが非常に大きな負荷になっていると考えられる．

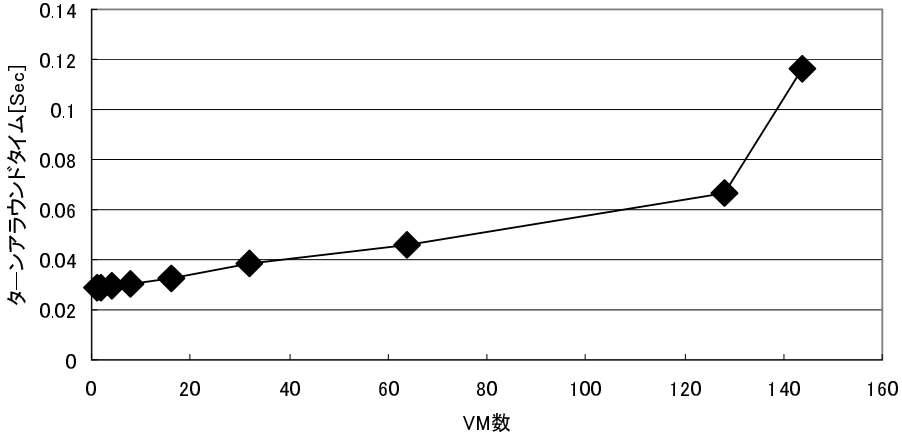

図 4　ターンアラウンドタイム

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

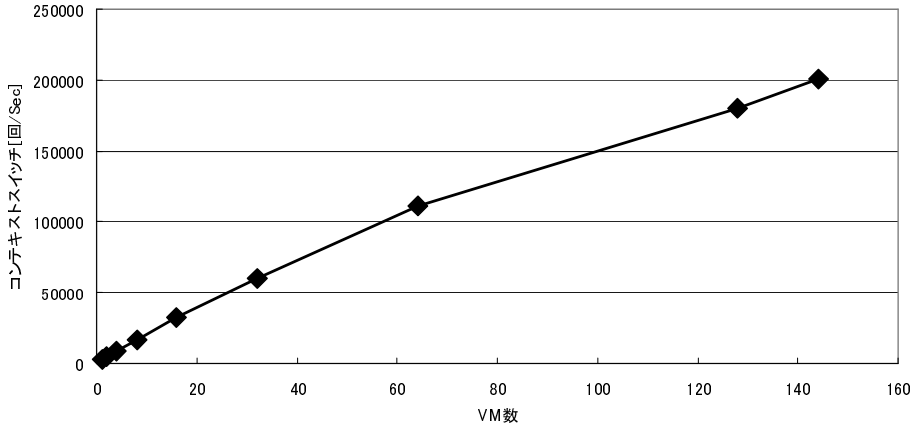

図 5　コンテキストスイッチ数

4. 性能向上手法

前章の測定結果よりアクセス負荷がなくても VM を複数立ち上げるだけで VM がホスト OS に対して高い負荷をかけていることが分かった．この章ではホスト OS に対する負荷を低減するためにゲスト OS 軽量化を行う．軽量化手法としては（1）ホスト OS 上でゲスト OS のプロセス優先度を変更する手法と（2）ゲスト OS のカーネルタイマー周波数を変更する手法を用いる．

4.1 ホスト OS に対するゲスト OS のプロセス優先度

各プロセスには優先度(nice 値)が存在しており，CPU がそれぞれのプロセスを順番に処理する際に，当該プロセスの 重要度（優先度）が高いものを優先的に処理する．通常のプロセスの優先度は nice0 であり，優先度は最高(nice-20)から最低(nice19)の範囲で変更することができる．

本稿では，VM プロセスの優先度を nice19 に変更して VM 負荷を軽減することを試みる．

4.2 タイマー割り込み

本節では，ゲスト OS における Linux カーネルのタイマー割り込みの頻度を減らし VM の負荷を軽減する手法を提案する．カーネルタイマーの初期値は 250Hz である．

5. 性能評価

前章の手法を適用した環境の性能を調査した．VM プロセス優先度は nice0 から nice19 に変更し，カーネルタイマー周波数は 250Hz から 100Hz，17Hz に変更した．なおカーネルタイマー周波数 16Hz 以下ではカーネルを構築できなかった．それ以外の測定条件は第 3 章と同じである．以下に，性能評価結果を記す．

5.1 計算機 1

計算機 1 の測定結果を図 6～図 9 に示す．図 7 において，VM の優先度が最低(nice19)の場合は，稼働 VM 数の増加に伴う Web アプリケーションの応答時間の増加が大幅に抑制されている結果となっている．また，VM プロセス優先度の低減に加えてタイマー割り込み周波数を 100Hz，17Hz に変更した場合，優先度を下げたのみの場合よりもさらに応答時間性能が改善している．

次に図 8 のコンテキストスイッチ数について考察する．図より，VM の数の増加に伴いコンテキストスイッチ数が大幅に増加していることが分かる．この負荷がターンアラウンドタイムの悪化の大きな原因の一個になっていると考えられる．また， VM プロセスの優先度を下げ，カーネルタイマー周波数を減少させることによりコンテキストスイッチ数の増加が抑制できていることが分かる．図 9 より各手法の Load Average には大きな差は無いことが分かる．

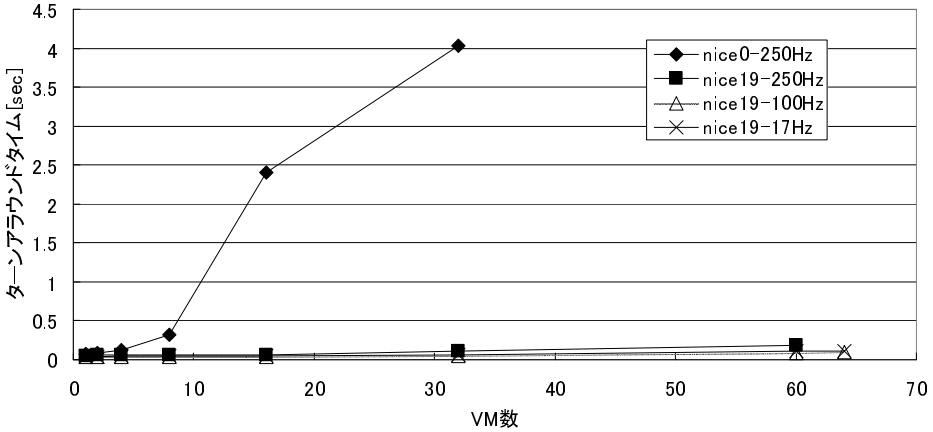

図 6　ターンアラウンドタイム

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

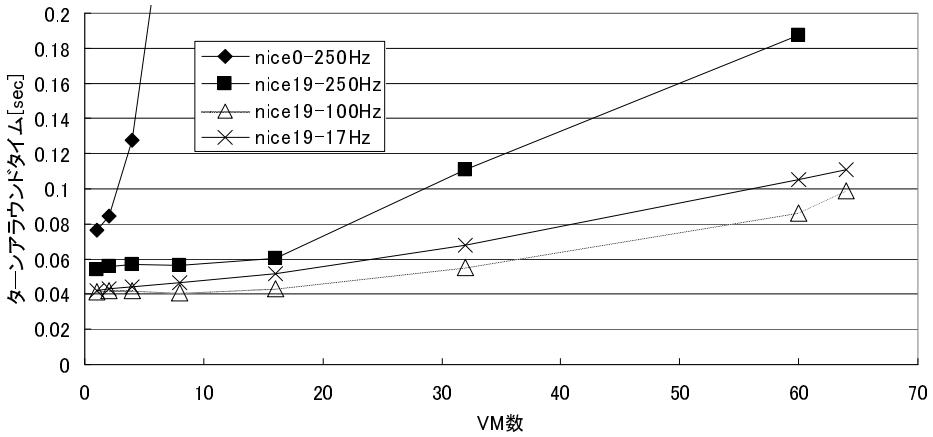

図 7　ターンアラウンドタイム(図 6 の拡大図)

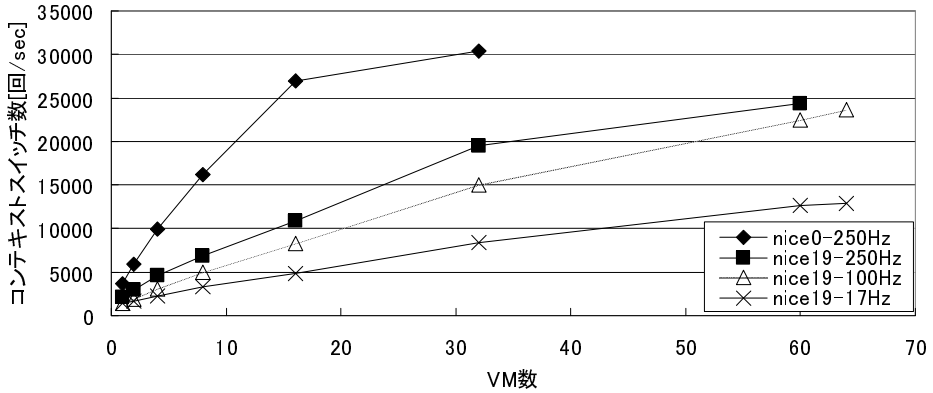

図 8　コンテキストスイッチ数

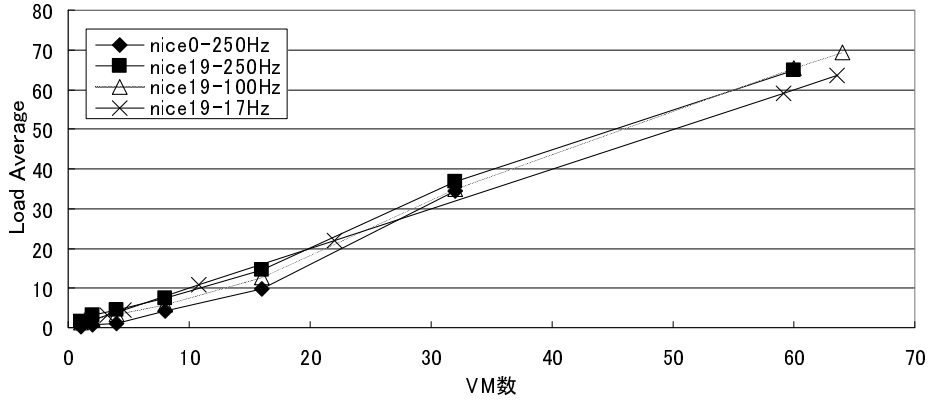

図 9　Load Average

5.2 計算機２

計算機 2 の測定結果を図 10～図 12 に示す．図 10 において，VM の優先度が最低(nice19)の場合は，稼働 VM 数の増加に伴う Web アプリケーション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．またタイマー割り込みの値を変更することにより応答性能のさらなる改善がなされていることが分かる．次に図 11 のコンテキストスイッチ数を比較すると，VM プロセスの優先度に関わらず稼働 VM 数が増加するに従いコンテキストスイッチ数が同程度に増加することが分かる．これに対して，タイマー割り込みの値を変更すると，コンテキストスイッチ数の増加が大幅に抑えられる結果になった．図 12 より，提案手法により Load Average が必ずしも減少していないことが分かる．すなわち，VM プロセス優先度 19，カーネルタイマー周波数 17Hz の環境（nice19-17Hz）の Load Average は初期設定（nice0-250Hz）のものより低くなっているが，それ以外の環境においては Load Average はむしろ高くなっていることが分かる．

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

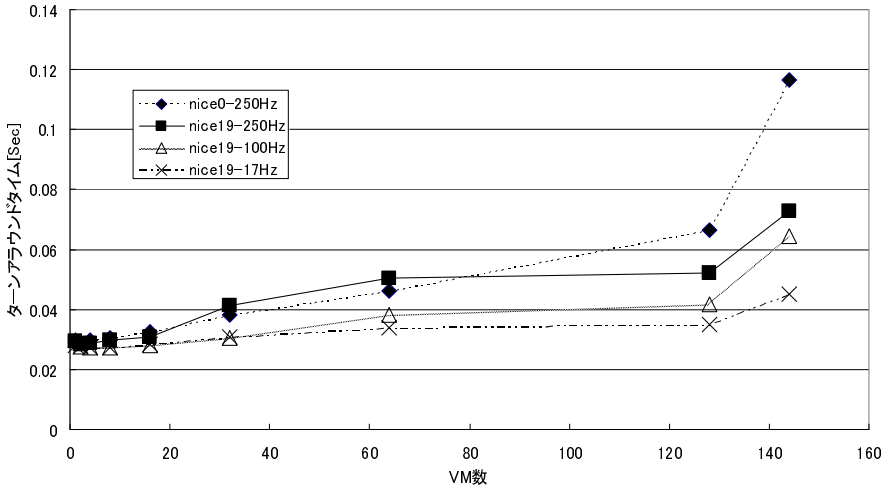

図 10　ターンアラウンドタイム

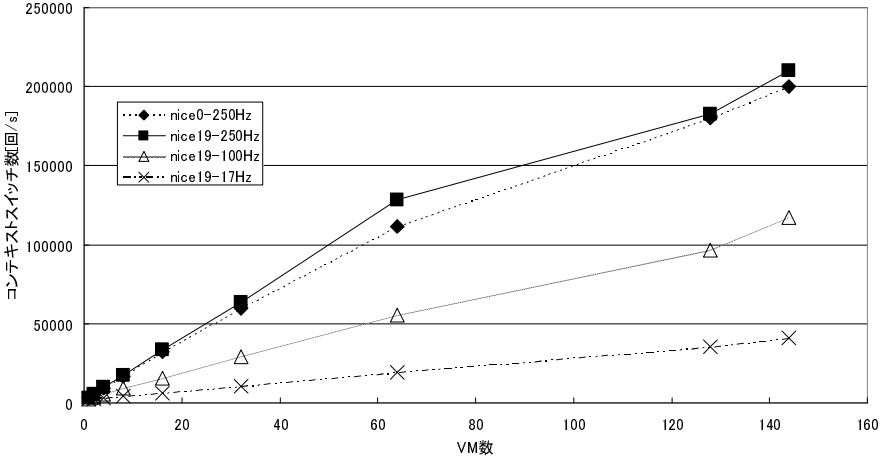

図 11　コンテキストスイッチ数

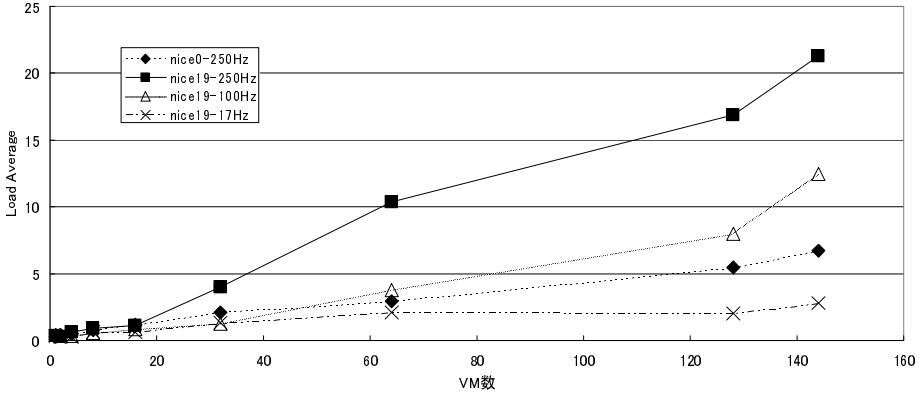

図 12　Load Average

6. 考察

本章では，VM プロセスの増加に伴いターンアラウンドタイムが増加する原因について詳細に考察する．

Web アプリケーションの動作は，複数の処理により構成されている．Web アプリケーションのターンアラウンドタイムを以下の 3 要素に分離し，それぞれの時間の変化を考察する．

(1) **Web アプリケーションの前処理(C-W-D)**：Web アプリケーションサーバがクライアントからリクエストを受け取ってから，RDBMS へリクエストを送出するまでの時間
(2) **RDBMS 処理(W-D-W)**：RDBMS が Web アプリケーションサーバからクエリを受け取ってから，応答を返すまでの時間
(3) **Web アプリケーションの後処理(D-W-C)**：Web アプリケーションが RDBMS からデータを受け取ってから，それを HTML に変換しクライアント(Web ブラウザ)に送出するまでの時間

各時間は，クライアント-Web アプリケーション間通信のパケット，Web アプリケーション-RDBMS 間の通信のパケットを観察することにより測定することができる．

前章と同一の Web アプリケーションを作成し，そのレスポンスタイム，各構成要素の時間を測定した．ただし，パケット観察を容易にするために RDBMS として PostgreSQL を用いた．測定方法は以下の通りである．1 台の物理計算機上に 1〜144 台の VM を稼動させ，各 VM 上に Rails を用いて作成した Web アプリケーションを 1

Vol.2011-DPS-146 No.50
Vol.2011-CSEC-52 No.50
2011/3/11

つ動作させる．ホスト計算機から Rails Web アプリケーションに対してラウンドロビン方式で 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，各構成要素の時間を測定対した．アクセス対象の VM の数は 1，10，40，144 と変化させた．アクセス対象 VM 数が 10 の場合は，10 台の VM にのみアクセス要求が送信され，残りの 134 台は稼働しているのみでアクセス要求が来ないことになる．アクセス VM 数に関わらずアクセスは多数回行い，アクセス VM 数が 10 の場合はこの 10 台に対して繰り返しアクセス要求が送信される．

調査結果を図 13，図 14 に示す．

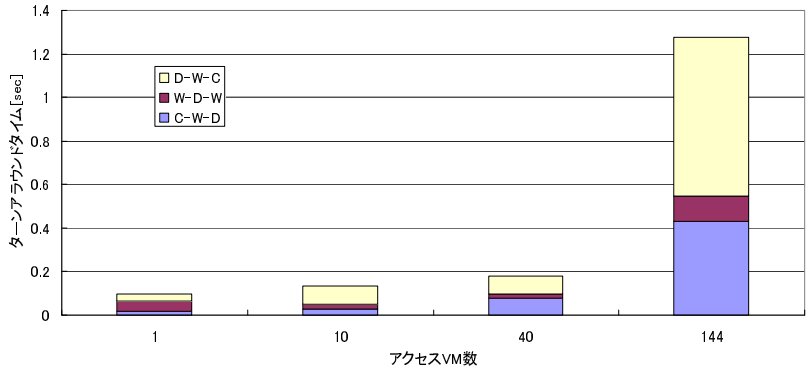

図 13　VM 毎の TA 解析（1 アクセス目から 50 回アクセス目まで）

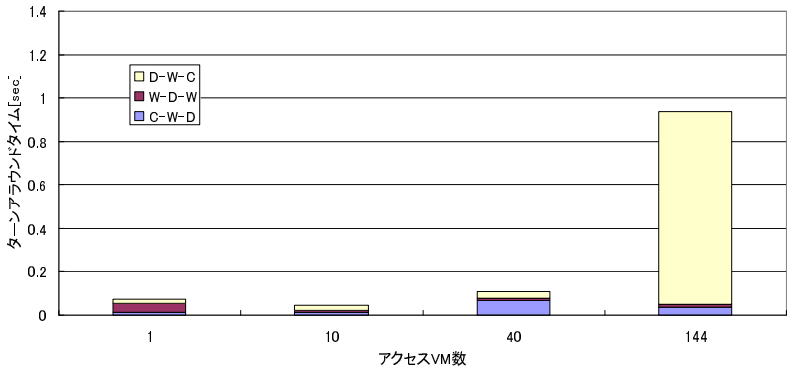

図 14　VM 毎の TA 解析（1440 アクセス目から 1490 アクセス目まで）

図より稼働 VM 数の増加に伴うターンアラウンドタイムの増加の主たる要因は，「(3)Web アプリケーションの後処理(D-W-C)」の増加であることが分かった．本処理は Ruby 言語で実装されており，RDBMS から受信したデータからのビュー(HTML)の作成などが含まれている．

7. おわりに

本稿では，各 Web アプリケーションに VM を独占的に与え，各 Web アプリケーション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，この環境における Web アプリケーションの性能について考察した．性能向上手法としてホスト OS における VM プロセスの優先度を下げる手法と，カーネルタイマー周波数を下げる手法を提案した．性能評価の結果，提案手法により性能劣化を大幅に抑制できることが確認された．また，Web アプリケーションの性能解析を行い，VM 数の増加に伴うターンアラウンドタイム増加の主たる原因について考察した．

今後は，ボトルネック箇所の性能改善手法の検討，Load Average が増加する理由の調査を行っていく予定である．

謝辞　本研究は科研費（22700039）の助成を受けたものである．

参考文献

1)　KVM
http://www.linux-kvm.org/
2)　Ruby on Rails
http://rubyonrails.org/
3)　越智俊介，山口実靖：高集約サーバ統合環境における仮想計算機の性能に関する考察，情報処理学会 FIT2010，
4)　越智俊介，山口実靖：仮想計算機による多数の低負荷サーバの集約に関する一考察，Internet Conference2010