Vol.2011-IOT-12 No.30
2011/3/1

クラウドコンピューティングを用いた大規模コンテンツ配信基盤の構築と運用

岡本慶大[†1]　寺田直美[†1]　赤藤倫久[†2]
岡本裕子[†3]　関谷勇司[†4]　河合栄治[†5]
藤川和利[†1]　砂原秀樹[†1,†6]

インターネットにおける大規模コンテンツ配信では，突発的なアクセス増加によりリソースの不足が起こると，大幅に配信品質が低下してしまう．この問題に対して，クラウドから計算機資源を一時借用し，品質低下を防ぐという解決法が考えられる．本論文では，全国高等学校野球選手権大会のインターネット中継において実施した，クラウドを利用した大規模コンテンツ配信実験について報告する．

Construction and Operation of Large Scale Web Contents Distribution Platform using Cloud Computing

Yoshihiro Okamoto ,[†1]
Naomi Terada and Tomohisa Akafuji ,[†1,†2]
Yuko Okamoto ,[†3] Yuji Sekiya ,[†4] Eiji Kawai ,[†5]
Kazutoshi Fujikawa [†1] and Hideki Sunahara[†6]

On the Internet, flash crowd (unexpected increase of web access) debase content quality due to insufficient resources. We think cloud computing, borrows computer resources through the internet, may solve this matter.

In this paper, we report an experimentation of large-scale content distribution using cloud computing in Japanese High School Baseball Championship (Summer Koshien) 2010.

1. はじめに

ブロードバンドインターネットの普及により，様々なコンテンツがインターネットを通じて配信されている．また，インターネットへの接続ユーザ数も日々増加しており，さらには配信されるコンテンツ高画質化の要求も高まっている．全国高校野球選手権大会 (以下，甲子園) のインターネット中継もその１つであり，コンテンツを配信するサーバやネットワークへの負担は年々増加している．

配信用に構築されたサーバやネットワークの処理能力を超えてリクエストが集中する場合があり，このような場合に，一部のユーザへコンテンツが届かない場合や，または，全体への配信品質が低下してしまう場合があり，最悪の場合では，スラッシングの発生によってコンテンツ配信自体が停止してしまう可能性も考えられる．しかしながら，リクエストの集中に合わせて配信システムを構築すると，平均的な負荷率が下がってしまい，さらに構築に掛かるコストも増加する．このような一時的な IT リソースの要求に応えられる概念として，クラウドコンピューティングが挙げられる．

本論文では，甲子園インターネット中継の一部としてクラウドコンピューティングを実験的に利用し，大規模 Web コンテンツ配信基盤としてのクラウドコンピューティングの活用について検証した．

†1 奈良先端科学技術大学院大学
Nara Institute of Science and Technology
†2 朝日放送株式会社
Asahi Broadcasting Corporation
†3 NTT スマートコネクト株式会社
NTT Smartconnect Corporation
†4 東京大学
University of Tokyo
†5 情報通信研究機構
National Institute of Information and Communications Technology
†6 慶應義塾大学
Keio University

2. 甲子園インターネット中継

2.1 プロジェクト概要

甲子園のインターネット中継は，朝日放送，奈良先端科学技術大学院大学 (以下，NAIST)，NTT スマートコネクトの共同プロジェクトとして 1998 年から行われており，試合経過や過去の試合のハイライトなどを Web でリアルタイムに提供している．

2.2 甲子園システムの概要

甲子園インターネット中継システムは，3 種類のサーバから構成される．

koshien 甲子園の Web コンテンツ全体を配信

sentryc ライブ配信ページのパラパラアニメを配信

sentryd NTSC の入力から静止画の切出しと圧縮を行い sentryc へ提供

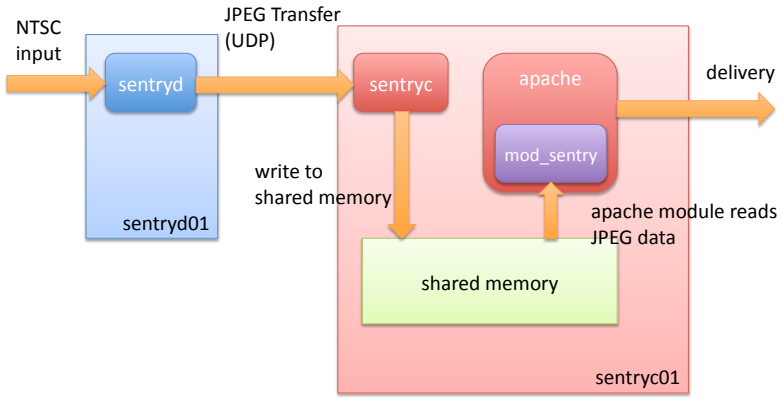

図 1 sentry サーバプログラムの動作

sentryd と sentryc の動作モデルを図 1 に示す．sentryd は入力された映像をパラパラアニメの要領で 1 秒毎に JPEG 画像データとして切出し及び圧縮し，sentryc サーバに UDP 伝送する．JPEG 画像データを受け取った sentryc は共有メモリへ JPEG データを書き出す．同一サーバ上で動作している apache には mod_sentry と呼ばれるモジュールを組み込んであり，特定の URL にアクセスするとこの共有メモリの内容が配信されるようになっている．これは，FTP や NFS の等のファイル転送ではディスクアクセスが配信のボトルネックとなるためである．

koshien の配信する Web ページには画像表示用の JavaScript が配置してあり，この JavaScript が一定間隔で sentryc から画像を取得することで，パラパラアニメのように動作し，ストリーミングが閲覧出来ないといった環境下でも，試合進行や球場の雰囲気を伝えることができる．

sentryd から sentryc へのデータ転送は，プライベートネットワークを利用しており，sentryd はインターネットへは接続されていない．

2.3 昨年度までの課題

昨年度までの構成では，アクセス集中時に sentryc 系のサーバ能力不足や対外線の逼迫により，一部のユーザへ画像が配信されないといった問題があった．これに対し，sentryd 側で JPEG 画像の圧縮率を上げる，パラパラアニメの更新間隔を伸ばすなどの対策を行ったが，その分ユーザへの配信品質が低下してしまうという問題もあった．

3. WIDE クラウド

WIDE プロジェクトでは，WIDE クラウドワーキンググループを立ち上げ，WIDE クラウドの運用を通して，研究活動や産学連携に適したサービスを提供できる柔軟な VM リソース管理手法の研究および構築を行っている．WIDE クラウドは，複数の大学に Hypervisor を分散配置し，その間を高速・広帯域なネットワークで接続している．また，それらを統合的に管理できる管理ツール ”WIDE クラウドコントローラ” の開発を行い，IaaS (Infrastructure as a Service) の形式で実験的に VM を実行できるサービスを提供している．[1)] 図 2 に WIDE クラウドコントローラのスクリーンショットを示す．

4. クラウドを利用したコンテンツ配信基盤の構築

今年度の甲子園システムでは，sentryc 系サーバの一部を WIDE クラウド上の VM とすることで，外部の計算機リソースやネットワーク帯域を利用するという運用を行った．

今年度の甲子園システムのネットワーク構成を図 3 に示す．対外線はサイバー関西プロジェクトにより koshien シリーズおよび sentryc シリーズそれぞれに 1Gbps 提供された．WIDE クラウド上の VM は，奈良 (NAIST) と根津 (東大) に配置し，1VM 毎に 1 グローバル IPv4/v6 アドレスを割り当てた．

従来プライベートネットワークで運用していた sentryd は，JGN2plus 経由でインターネットへ接続することで，WIDE クラウド上の sentryc へも画像転送を可能とした．

Vol.2011-IOT-12 No.30
2011/3/1

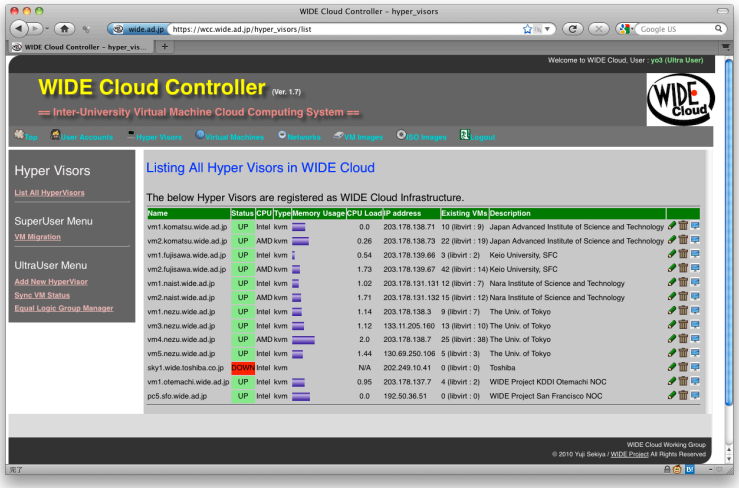

図 2 WIDE クラウドコントローラ

実計算機である sentry.asahi.co.jp と，WIDE クラウド上の sentryc への振り分けは，koshien が配信する JavaScript の参照先サーバを書き換えることで実現した．

(1) クライアントが koshien から，パラパラアニメの JavaScript を含む Web コンテンツを取得する
(2) JavaScript を実行すると，画像を取得するサーバ名と更新間隔が記述された設定ファイルを koshien から取得する
(3) 取得した設定ファイルを元に，更新間隔毎に指定されたサーバから画像を取得する
(4) 設定ファイルは 1 分毎に再度取得される

koshien シリーズは本年度は 9 台で運用したので全体の 1/9 ずつのトラフィックを振り分けられる．さらに，koshien シリーズの負荷に余裕がある場合は前段のロードバランサのウェイトを変更することでより細かいウェイト調整が可能である．

負荷分散手法としては DNS ラウンドロビンを用いる方法も，過去の甲子園実験で検討されている[2] が，クライアントによって反映時間がまちまちであり，トラブル発生時にクラウドから実計算機へ切り戻すのが困難となる．

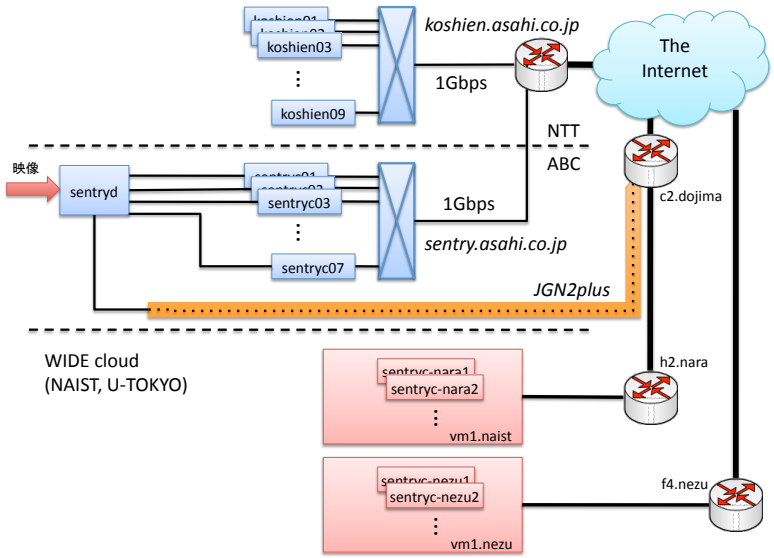

図 3 2010 年度甲子園システムネットワーク概要図

4.1 VM ベンチマーク

実際の運用前に，WIDE クラウド上の VM での Apache の性能評価を行った．Hypervisor は，CPU による性能差を調査するため，Intel Xeon と AMD Opteron のサーバを利用した．Hypervisor の構成を表 1 に示す．

表 1 Hypervisor の構成

	vm1.naist	vm2.naist
CPU	Intel Xeon L5520 (2.26GHz) x2	AMD Opteron 2387 (2.8GHz) x2
Memory	36GB DDR3	64GB DDR2
OS	Ubuntu 10.04 amd64	
Hypervisor	KVM 84	

また，VM の GuestOS として これまでの甲子園での運用実績のある CentOS 5.5 と，Hypervisor と同じ OS である Ubuntu 10.04 をインストールし，OS の違いによる性能差を比較した．また，仮想 NIC について，準仮想化ドライバである virtio と，Intel Pro 1000

のエミュレーションする e1000 を比較した．仮想 CPU は 1，割り当てたメモリは 4GB，利用した Apache のバージョンは 2.2.16 で，Multi Processing Module は event MPM を利用し，mod_sentry を組み込んである．

ベンチマークは httperf を利用し，mod_sentry によって画像が提供される URL を対象とした．sentryd の提供するコンテンツサイズは 25KB とした．

表 2 Apache ベンチマーク結果

構成	レスポンスレート (s)
Intel+CentOS+virtio	2153
AMD+CentOS+virtio	1107
Intel+CentOS+e1000	1188
Intel+Ubuntu+virtio	1886

表 2 にベンチマーク結果を示す．最も性能が高かったのは，Intel Xeon のサーバ上で CentOS と virtio を利用した組み合わせであった．AMD のパフォーマンスが伸びないのは，kvm_amd カーネルモジュールの実装の問題である可能性が高い．virtio が e1000 より速いのは，準仮想化によるエミュレーションのオーバーヘッド削減の効果によるものである．

Linux カーネルのバージョンが古いにもかかわらず CentOS (Linux 2.6.18) が Ubuntu (Linux 2.6.32) より速い理由は，タイマ割り込み間隔などのカーネルの設定に起因すると考えられる．

5. 運用報告

甲子園システムの運用は，開幕の 8 月 7 日から行われた．WIDE クラウド側では，NAIST(奈良) と東大 (根津) にそれぞれ 4 台ずつの VM を準備した．決勝戦でのトラフィックは甲子園システム全体で 650Mbps を記録した．また，最大トラフィックは準決勝で 800Mbps を記録している．

5.1 実戦投入

WIDE クラウド上の sentryc の投入は，大会 2 日目 第 3 試合から開始され，koshien のうち 1 台の JavaScript の参照先を NAIST の sentryc へ変更することでトラフィックが分散されることを確認した．VM の TCP セッション数の変化を図 4 に示す．14:30 付近からセッション数が増加しているのがわかる．

随時，クラウドの sentryc へ切り替えたが特に問題は起こらず，大会 8 日目は準備した

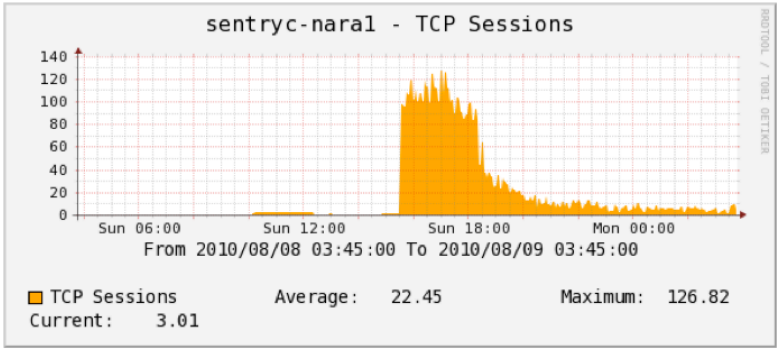

図 4 sentryc 切替時の TCP セッション数の変化

8VM を全て投入した．また，大会 6 日目には NAIST の全学ネットワークメンテナンスがあったため，クラウドから実計算機への切り戻しも行った．この日の奈良の Hypervisor におけるネットワークトラフィックを図 5 に示す．4VM の合計となっており，最大 80Mbps，東大側と合わせて約 150Mbps，約 8000TCP セッションとなった．

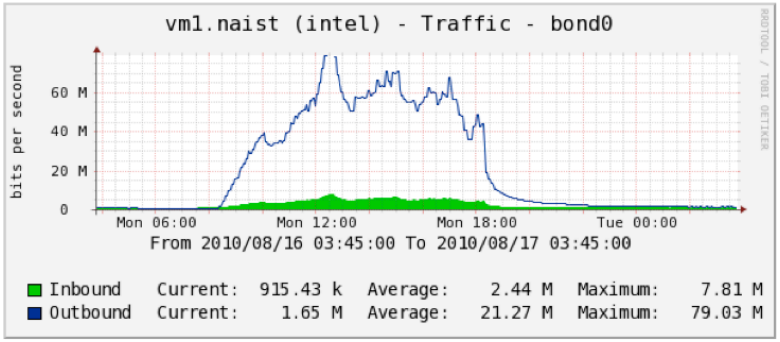

図 5 4VM 合計ネットワークトラフィック

5.2 リバースプロキシ導入

koshien と sentryc の紐付けにおけるさらに細かいウェイト調整や，AMD の Hypervisor の活用，グローバル IPv6 アドレスのみを持つ VM の追加などを目的に，リバースプロキシを導入した．リバースプロキシがボトルネックにならないよう，実計算機を利用した．表

図 6 mod_proxy_balancer の構成ファイル (例)

```
<Proxy balancer://cluster>
# Intel
    BalancerMember http://sentryc1.naist.wide.ad.jp/ loadfactor=10
    BalancerMember http://sentryc2.naist.wide.ad.jp/ loadfactor=10
# AMD
    BalancerMember http://sentryc3.naist.wide.ad.jp/ loadfactor=2
    BalancerMember http://sentryc4.naist.wide.ad.jp/ loadfactor=2
</Proxy>
```

3 にリバースプロキシのサーバ構成を示す.

表 3 リバースプロキシのサーバ構成

CPU	Xeon X5570 (2.8GHz) x2
Memory	24GB DDR3
Network	10 Gigabit
OS	CentOS 5.5
Rev.Proxy	Apache 2.2.16 (event MPM) mod_proxy_balancer

mod_proxy_balancer のセッティングを図 6 に示す. ここに示しているのは, guest1 および guest2 を Intel Xeon の Hypervisor 上, guest3 および guest4 を AMD Opteron の Hypervisor 上に配置した状態で, 負荷を均等にかけたい場合の設定である.

大会 14 日目第 1 試合において, トラブルに見舞われたものの, リバースプロキシのみで 166Mbps, 甲子園システム全体で約 800Mbps を記録した.

5.3 運用中のトラブル

全 VM 投入後, 大会 11 日目のピーク時間帯に VM や Hypervisor の応答が著しく悪化する問題が発生した. 問題発生時の Hypervisor の CPU 使用率のグラフを図 7 に示す. 30 分程の間, 監視マシンから SNMP データが取得できていない様子がわかる. CPU 使用率やメモリ使用量, ネットワークトラフィックはまだ頭打ちになっておらず, 各 VM から発行されるネットワーク割り込みの処理によって Hypervisor の処理が逼迫し, パケットに応答できなくなった可能性が考えられる.

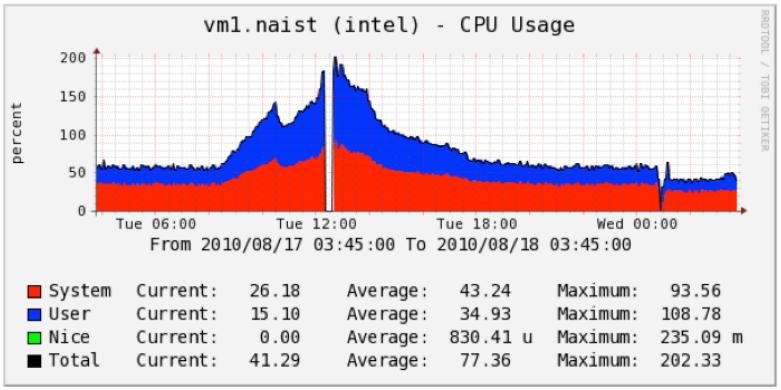

図 7 VM 応答悪化時の Hypervisor の CPU 使用率

さらにリバースプロキシ導入後にも応答が悪化し, 画像が途中で切れるなどの問題が発生した. こちらは, 奈良に設置したリバースプロキシのバックエンドに根津の VM を配置したのが原因で, バックエンドを奈良の VM のみにしたところ解消した. 問題発生時のトラフィック及び TCP セッション数を図 8 および図 9 に示す.

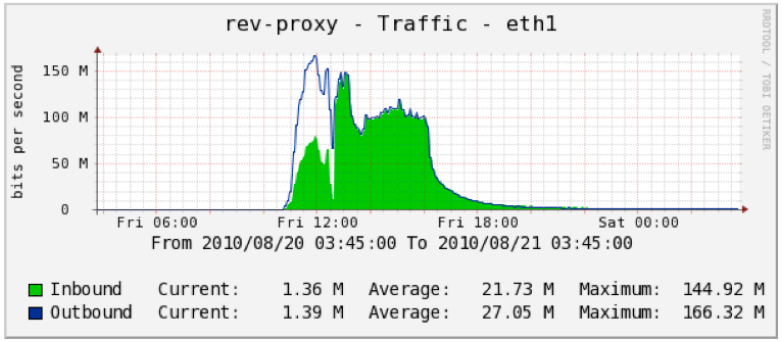

図 8 リバースプロキシ応答悪化時のトラフィックの変化

また, Ubuntu 10.04 の net-snmp の実装に問題があり, 80Mbps 以上のトラフィックを SNMP で取得できないという問題や, Internet Explorer の JavaScript エンジンの実装では, 今回の JavaScript における sentryc の切替を無視してしまうという問題があった.

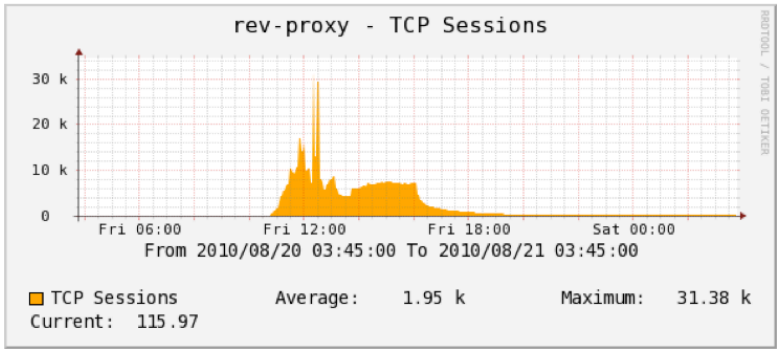

図 9 リバースプロキシ応答悪化時の TCP セッション数の変化

6. 実験における考察と課題

今回の実験は，期間限定の大規模配信基盤としてクラウドを利用するという試みであったが，おおむね良好な結果を得ることができたと考えている．特に，昨年度は数度以上行う必要があったパラパラアニメの更新間隔や JPEG 圧縮率の調整が，今年度はリバースプロキシのトラブルが発生した 1 試合のみで済んだため，クラウドを利用することでクライアントへの配信品質を向上できたと結論づけられる．

運用面では，ベンチマークほど実戦では性能が出ない点，同一ワークロード多重化による性能劣化，リバースプロキシの構成法など，検証不足な点があった．

今後，クライアントのソースアドレスや AS によって配信元の VM を変更する，つまりクラウドを CDN として利用する方法についても検討を行いたい．

7. おわりに

今後も，インターネットにおける大規模コンテンツ配信はさならる高品質化が求められるようになり，配信サーバ・ネットワークへの負荷が問題になると考えられる．本研究では，甲子園インターネット中継の配信基盤としてクラウドコンピューティングを利用し，大規模 Web コンテンツ配信基盤の一部としてのクラウドコンピューティングの活用について検証した．

昨年度まではトーナメントが進むにつれ，クライアント数の増加に伴うリソース不足が度々発生していたが，今年度はトラブル発生した一度だけで済み，Web コンテンツの配信品質の向上にクラウドが活用可能であることを示した．

今後は，経路情報などを利用した VM の動的分散配置なども検討していきたい．

謝　辞

本実験を行う機会を与えてくださったサイバー関西プロジェクトおよび WIDE プロジェクトの皆様に感謝する．

参 考 文 献

1) WIDE Cloud Controller. https://wcc.wide.ad.jp/.
2) 神屋郁子, 下川俊彦, 岡村耕二, 河合栄治, 寺田直美, 岡本裕子, 谷崎文義, 赤藤倫久. 経路情報を利用した広域分散配信の実証実験. インターネットコンファレンス 2009 論文集, pp. 83–89, October 2009.