# Webサーバファームにおける スタンバイ状態を導入したサーバ切り替え方式の開発

藤部修平* 知念賢一* 河合栄治†
赤藤倫久§ 香取啓志§ 山口英* 山本平一*

あらまし

大規模データセンタにおいて、負荷の変動に合わせ動的にサーバ台数を割り当てるシステムを開発した。本システムでは各サーバのコンテンツ保持にキャッシュを用い、積極的にコンテンツの供給を行なう。この際、コンテンツの供給のみ行なうスタンバイ状態を導入し、コンテンツ切り替え処理と負荷分散装置の制御を並列に行なうことで、サーバ追加処理時間を大幅に削減した。本システムのサーバ追加処理時間は数百ミリ秒～数秒程度と非常に短いため、急激な負荷の変動に対しても柔軟に対応することができる。

# The Design and Implementation of a Web Cluster Management System with Hot-Standby Servers

Shuhei Fujibe* Ken-ichi Chinen* Eiji Kawai†
Tomohisa Akafuji§ Keishi Kandori§ Suguru Yamaguchi* Heiichi Yamamoto*

**Abstract**

We have developed a Web cluster management system that controls the number of hosts in a Web cluster dynamically. In our system, where each member of the cluster has a cache mechanism, the server controller puts a new server in hot-standby mode and prepares it for the actual service. In hot-standby mode, a server is fed with corresponding Web site content. Since our system performs the content preparation and the control of the load-balancer simultaneously, it drastically eliminates the time between the detection of increased requests and the addition of the new server. Thus, our system can adapt to broad changes in the request arrival.

## 1 はじめに

WWWサーバへのアクセス集中を回避するサーバ負荷分散の技術として、一つのWebサイトを複数のサーバで運用する技術が一般化している。大規模データセンタにおいては、こうした複数のサーバで構成されたWebサイトを複数収容しサポートする形態が多く見られるようになった。

このようなデータセンタでは、サイトに到来する負荷の最大値を事前に予測し、それにあわせてサイト毎にサーバ台数を静的に割り当てる手法が用いられてきた。しかしこの手法では、負荷の変動に対する柔軟な対応が困難であり、ピーク時以外にはシステム能力が無駄になる。これらの解決策として、データセンタのサーバを全てのサイトで共有し、各サイトのサーバ台数の割り当てを動的に行なう手法が挙げられる。

動的なサーバ割り当てを行なう際、各サーバを過負荷な状況にしないためにサーバの追加処理を迅速に行なわなくてはならない。ところが、既存のシステムはサーバの追加処理に数十秒以上を要するため、これでは不十分だと思われる。

本研究では、リアルタイムに観測した各サーバの負荷データをもとに、動的なサーバ割り当て行なうシステムを開発した。本システムは、サーバの追加

*奈良先端科学技術大学院大学
Nara Institute of Science and Technology
†科学技術振興事業団
Japan Science and Technology Corporation
§朝日放送
Asahi Broadcasting Corporation

処理を行なう際にあらかじめ対象サーバにコンテンツを供給するスタンバイ状態を導入する。これによりサーバの追加処理を迅速に完了させ、急激な負荷の変動に対する柔軟な対応を可能にする。

## 2 動的なサーバ割り当て

本研究の目標は、複数のサイトを収容する大規模なデータセンタにおいて、各サーバ及び各サイトの負荷状態をリアルタイムに観測し、負荷の変動に応じて迅速にサーバの追加を完了させるシステムを開発、運用することである。

### 2.1 要求事項

動的なサーバ割り当てを行なう際、動的な制御によってサービスの質を低下させてはならない。ここでサービスの質の低下とは次のような事象を指す。

- サーバの応答時間の急激な悪化
- サーバダウンによるサービスの停止

実環境での負荷の変動は、不規則かつ急激であるが、動的な制御によってサーバを過負荷な状態にしてはならない。このため、各サーバの処理限界を正確に把握し追加処理を迅速に行なうことが要求される。

### 2.2 処理時間が変化する要因

サーバの追加処理とは、負荷分散装置上の設定を変更する作業と、サーバの保持するコンテンツを切り替える作業の2点である。負荷分散装置の設定変更に要する時間は、設定のための通信にかかる時間と負荷分散装置が設定を有効にする時間であり、一般に数秒かそれ以下と非常に小さい。コンテンツの切り替えに要する時間は、各サーバがどのようにコンテンツを保持するかに依存する。2つの作業のうち、処理時間の大部分はコンテンツの切り替えに要する時間である。このため各サーバがどのようにコンテンツを保持し、どのように切り替えるかが、サーバの追加処理に要する時間を決定づける。

## 3 既存のシステムとその問題点

サーバの負荷状況に合わせた動的なサーバ割り当てを実現するための試みは他にも存在し、現在も研究が進められている。IBM社の Project eLiza [1] における Oceano では負荷に合わせたサーバ台数の増減のほか、人の手を借りずに OS やコンテンツのインストールを行なう研究が行なわれている [2, 3]。また、日立製作所によって研究が進められている VPDC [4] システムもサーバの負荷状況を監視し、サーバ台数を動的に変化させる。

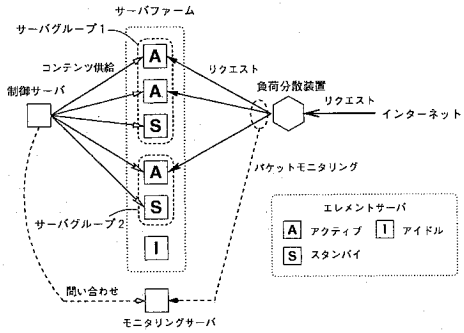


図 1: システム構成図

一方で、これらのシステムの問題点は、サーバの追加処理時間が長いことである。Oceano は、サーバの追加時に OS やコンテンツのインストールも行なうためサーバの追加処理時間は数分と非常に長い。VPDC は、コンテンツの保持形式に NFS を利用し処理時間を短縮させているが、それでも数十秒を要すると報告されている。これらのシステムではコンテンツの切替が完了するまでサービスの開始ができない。

## 4 設計

### 4.1 システム構成

本システムの構成図を図1に示す。本システムは制御サーバ、サーバファーム、モニタリングサーバ、負荷分散装置によって構成される。本システムでは、Web サイトはサーバファームにおけるサーバグループに相当し、各サーバグループは1台以上のエレメントサーバによって構成される。

次に、本システムの構成について個別に説明する。

**負荷分散装置**

全サーバグループへのリクエストは、インターネットを介して負荷分散装置に到来する。負荷分散装置は到来したリクエストを、それぞれのサイトを構成するエレメントサーバに分散させる。また、全エレメントサーバへの通信をモニタリングサーバに対しポートミラーリングしている。

**エレメントサーバ**

エレメントサーバは到着したクライアントからの HTTP リクエストに応え、要求されたコンテンツを返す。エレメントサーバには Chamomile [5] を用いる。

モニタリングサーバ

モニタリングサーバは、全エレメントサーバへ流入、流出するパケットをモニタリングする。また、制御サーバからの問い合わせに対し、現在の全エレメントサーバの負荷情報を返す。モニタリングサーバにはENMA [6] を改造したものを用いる。

制御サーバ

本システムにおいて制御サーバは最も重要な役割を担い、次に示す処理を行なう。

- エレメントサーバへのコンテンツ供給
- モニタリングサーバへの負荷情報の問い合わせ
- サーバ追加、削減の判断
- エレメントサーバへのリバースプロキシ再起動命令の発行

### 4.2 エレメントサーバのサービス手法

#### 4.2.1 コンテンツ保持方式

本システムでは各エレメントサーバがどのようにコンテンツを保持し、グループ変更時にどう切り替えるかが非常に重要となる。

本システムでは各エレメントサーバはキャッシュを用いてコンテンツを保持する。全てのコンテンツは制御サーバが集積して保持し、各エレメントサーバはリバースプロキシとしてクライアントからのリクエストに応える。ローカルのディスク、又はメモリ上のキャッシュを利用できるため、ネットワーク経由のコンテンツを利用するNFSと比べ、クライアントへの応答速度は高速である。

エレメントサーバは、自身が保持していないコンテンツへリクエストを受けた時、そのコンテンツを制御サーバに問い合わせて取得することができる。この機能により、全てのコンテンツを保持していなくても、グループの変更直後からサービスを開始することができる。全てのコンテンツを保持しなくても良いことから、記憶装置の使用量が少ない点でローカルディスクに全てのコンテンツを保持する方式よりも優れている。コンテンツの切り替えは、問い合わせ先の制御サーバのポート番号を変更することで行なう。各エレメントサーバはグループ毎に設定ファイルを用意しておき、グループ変更時には設定ファイルを変更してリバースプロキシを再起動させる。この命令は必要に応じて制御サーバが発行する。

リバースプロキシの再起動は数秒で完了する。また、起動後にコンテンツを転送する必要がないため切り替え作業は容易といえる。しかし、コンテンツを全く持たない状態でサービスを開始させると、到来するリクエストの全てを制御サーバに問い合わせるため、制御サーバの負荷を急激に増加させてしまう。本システムにおいて、制御サーバのダウンはシステム全体に深刻な影響を与えるため、このような事態は避けなくてはならない。

表 1: エレメントサーバの状態

| 状態 | コンテンツ供給 | リクエスト |
|---|---|---|
| アクティブ | 受ける | 応える |
| スタンバイ | 受ける | 応えない |
| アイドル | 受けない | 応えない |

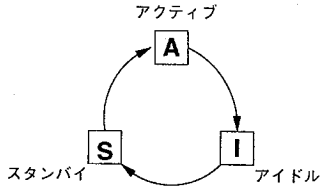


図 2: エレメントサーバの状態遷移

本システムでは制御サーバが新規コンテンツを積極的に各サーバに供給する。これは、新規コンテンツの投入やコンテンツの更新を確実に伝え、古いキャッシュをクライアントに返さないための機構 [7] である。

#### 4.2.2 エレメントサーバの3状態

本システムでは、前述のコンテンツ供給機構を用いグループ追加前のエレメントサーバにあらかじめコンテンツの供給を行なう。これをスタンバイ状態と呼ぶ。スタンバイ状態のエレメントサーバは制御サーバからコンテンツを供給されるが、クライアントからのリクエストは到来しない。エレメントサーバの状態を、コンテンツ供給の有無とリクエストの到来の有無により3つに分ける。それらを表1に示す。サーバグループはアクティブ状態のエレメントサーバとスタンバイ状態のエレメントサーバによって構成される。

スタンバイ状態を経てあらかじめコンテンツを蓄える事で、アクティブ状態となった直後の制御サーバへの問い合わせを軽減させることができる。エレメントサーバの3状態は常に一方向に遷移する (図2)。

### 4.3 動的なサーバ割り当て制御

動的なサーバ割り当てを行なう本システムの全体の制御の流れを図3に示す。制御サーバは起動後、次のような手順でシステムの制御を行なう。

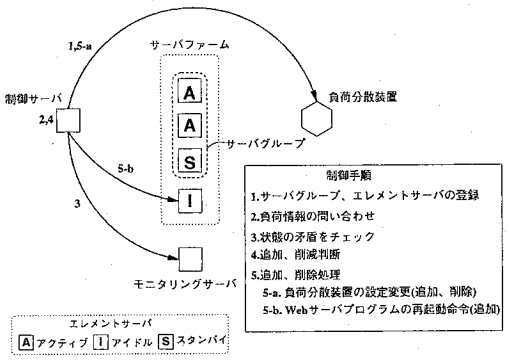


図 3: システム制御の流れ

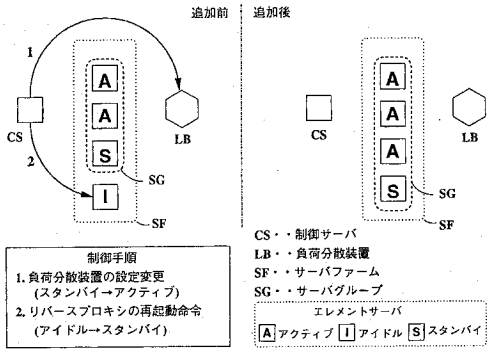


図 4: グループ追加時の処理

1. サーバグループ、エレメントサーバを負荷分散装置に登録
2. モニタリングサーバへの負荷状況の問い合わせ
3. 状態の矛盾をチェック
4. 追加、削除の判断
5. 追加、削減処理
6. 2へ戻り繰り返す

以下、それぞれの処理について詳細に述べる。

#### 4.3.1 サーバグループ、エレメントサーバの登録

制御サーバは起動直後、まず負荷分散装置と通信しサーバグループとエレメントサーバの登録を行なう。

#### 4.3.2 負荷情報の問い合わせ

制御サーバは、モニタリングサーバから全エレメントサーバの負荷情報を取得する。本システムでは、負荷情報として各エレメントサーバの現在までの総コネクション数を取得し、前回取得した値との差分をとることでコネクション到着率を推測する。

#### 4.3.3 状態の矛盾のチェック

制御サーバは各エレメントサーバの状態を把握している。制御サーバは、モニタリングサーバから取得した情報により、負荷分散装置の振舞いを知ることができる。負荷分散装置の振舞いが制御サーバの想定と異なっていた場合、原則として負荷分散装置の設定を変更し制御サーバの想定する状態に合わせる。

#### 4.3.4 追加、削減の判断

事前の実験によってエレメントサーバ1台あたりの処理限界を静的な値として求めておく。この1台あたりの処理限界値をもとに、追加、削減のしきい値をそれぞれ設定する。

**追加判断**

追加の判断は、グループ内に1台でも追加のしきい値を上回るサーバが存在する時に行なう。

**削減判断**

削減の判断は、グループ全体の負荷の総和が削減のしきい値を下回った時のみ行なう。削減のしきい値は、追加のしきい値の半分以下の小さな値に設定するためサーバの割り当て台数は増加しやすく減少しにくい。これはサーバの割り当て台数が振動する事を防ぐための配慮である。

#### 4.3.5 追加処理

制御サーバがあるサーバグループに対して追加判断を行なった時、制御サーバは次の2つの作業を実行する (図 4 )。

1. 負荷分散装置上の設定を変更し、現在スタンバイ状態のエレメントサーバを負荷分散対象に加える
2. アイドル状態のサーバを走査し、Webサーバプログラムを再起動させる (スタンバイとなる)

2は1に比べて時間を要するが、1と2では異なるエレメントサーバを対象とするためそれぞれの作業を独立して実行できる。本システムではこれらを並列に実行して追加処理に要する時間を短縮する。

#### 4.3.6 削減処理

制御サーバがあるサーバグループに対して削減判断を行なった時、制御サーバは負荷分散装置上の設定を変更し、現在アクティブ状態のエレメントサーバを1台アイドル状態にする。

## 5 実験

### 5.1 負荷の変動に対する追従特性

本システムが急激な負荷の変動に追従できることを確認するため、負荷の変動に対する追従特性測定の実験を行なった。

表 2: 実験条件

| | |
|---|---|
| 負荷分散装置 | ExtreamNetworks 社製 Summit1i |
| 処理限界値 | 100 コネクション/ 台 |
| 追加しきい値 | 処理限界値の 70% |
| 評価値 | 過去 30 秒分の移動平均 |

#### 5.1.1 実験方法

本システムに対し、徐々に増加するリクエストを与える。あらかじめ設定した処理限界値を超える事なくサーバを追加し、負荷に追従できるかを確認した。実験条件を表 2 に示す。

クライアントから発行するリクエストレートは 6 秒おきに 10 ずつ増加させ、1 分間で 100 上昇させる場合 (実験 1)、6 秒おきに 5 ずつ増加させ、1 分間で 50 上昇する場合 (実験 2) の 2 種類を実施した。

この実験では 1 リクエストを 1 コネクションとして扱ったため、リクエストレートとコネクションレートは等しい。

#### 5.1.2 結果、評価

実験 1、実験 2 の結果をそれぞれ図 5、6 に示す。

図 5 では本システムは問題なく負荷に追従できることがわかる。図 6 では、最初のサーバ追加の際に一時的に限界値付近までリクエストレートが増加しているが、限界値を越えない運用ができている。

本システムでは、実環境における負荷のスパイクに敏感に反応してサーバ台数が振動するのを防ぐため、追加と削減の判断に移動平均を用いている。この移動平均の影響により、今回の実験のように一方的に上昇し続ける負荷に対しては、追加の判断が遅れる可能性がある。これには単純に追加のしきい値を下げるか、過去の傾向から上昇傾向を読みとり、追加判断に用いることで対処できる。

### 5.2 サーバ追加処理の制御応答時間

本システムのサーバ追加処理が迅速であることを確認するため、サーバの追加判断から、処理が完了し実際に新しく割り当てられたサーバがサービスを開始するまでの時間を算出した。

#### 5.2.1 実験方法

クライアントからサーバグループに大量のリクエストを発行してサーバの追加を起こさせる。次に、制御サーバのログとエレメントサーバのログを用い、制御サーバが追加判断を行なった時刻と、新しくアクティブになったエレメントサーバに初めてクライアントからのリクエストが到着した時刻の差分を算出した。

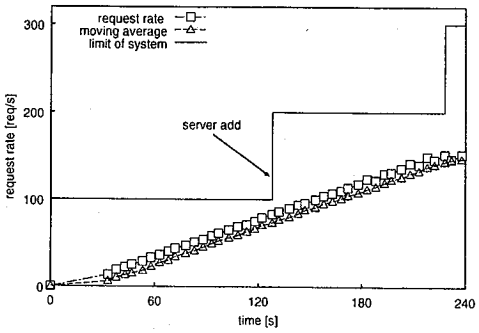


図 5: 負荷の変動に対する追従特性 1

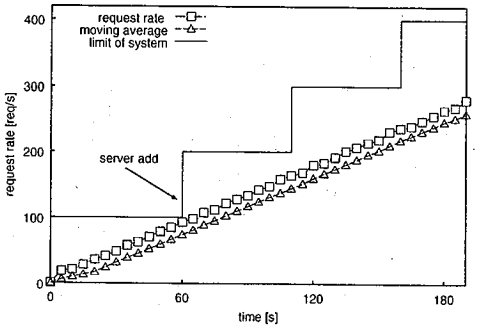


図 6: 負荷の変動に対する追従特性 2

#### 5.2.2 結果、評価

サーバ追加時間の累積密度分布を図 7 に示す。試行回数は 204 回である。この結果から、全体の 90%近くが 200 ミリ秒以内に収まっており、本システムはサーバ追加処理を迅速に完了することがわかる。

### 5.3 運用実験

本システムを 2002 年 8 月 8 日から同 21 日に開催された高校野球インターネット中継において、単一グループ、エレメントサーバ 5 台、1 台あたりの能力の限界を 500 コネクション/秒という条件で運用した。1 日の総アクセス数が最も多かった 19 日の運用結果を図 8 に示す。

運用実験より、本システムが実環境の負荷変動に追従し設計通りサーバの自動割当を行なったことを確認できた。また、この時サーバの追加に要した時間は 2 秒程度であった。この結果は前節の実験結果と大きく異なるが、これは使用した負荷分散装置やその設定が異なるためである。負荷分散装置は、設

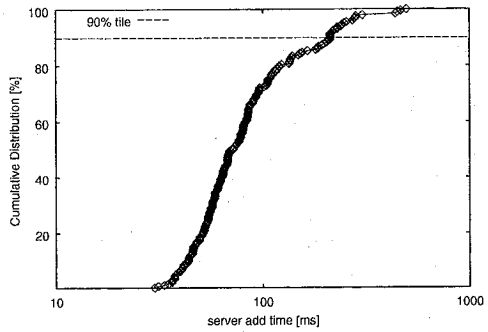


図 7: サーバ追加時間の累積密度分布

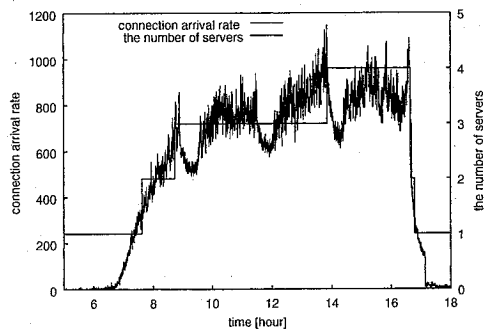


図 8: 運用実験におけるサーバ台数の変化

定変更の命令を受けてから実際に反映させるまでの時間が、ベンダや機種により異なる。

## 6 おわりに

複数の Web サイトを収容する大規模データセンタにおける、動的なサーバ割り当て方式として、スタンバイ状態を導入したサーバ切り替え方式を開発した。本システムのサーバ追加時間はおよそ 200 ミリ秒と迅速である。また、運用実験により、本システムは実環境の急激な負荷変動においても確実に追従し、動的なサーバ割り当てが可能な事を確認できた。

現在は、スタンバイ状態の導入によって、グループ追加直後の制御サーバへの問い合わせをどの程度削減できたかの定量的な評価ができていない。スタンバイ状態の定量的な評価は今後の課題である。

また、より効率良くスタンバイ状態のサーバにコンテンツを供給する手法や、各サーバグループに割り当てるエレメントサーバ台数の最大値決定手法について、今後検討していく予定である。

## 謝辞

CKP(サイバー関西プロジェクト) の方々には運用実験に際し負荷分散装置をはじめ多くの機材を貸与して頂きました。また、本学情報コミュニケーション講座の三野敦史氏、西岡宗一郎氏には本論文執筆にあたり多くの実験を補助して頂きました。あわせて深く感謝致します。

## 参考文献

[1] Autonomic Computing. URL:http://www.research.ibm.com/autonomic.

[2] K.Appleby, S.Fakhouri, G.Goldszmidt L.Fong, M.Kalantar, S.Krishnakumar, D.P.Pazel, J.Pershing, , B.Rochwerger. Oceano - sla based management of a computing utility. *IEEE International Symposium on Integrated Network Management*, 2001.

[3] OceanoProject. URL:http://www.research.ibm.com/oceanoproject/.

[4] 吉村裕, 垂井俊明, 庄内亨, 河辺峻, 杉江衛. Web アクセス集中に対応したサーバ自動割当制御. 電子情報通信学会論文誌, Vol. J85-D-1, No. 9, pp. 866–876, 2002.

[5] Chamomile. URL:http://iplab.aist-nara.ac.jp/~eiji-ka/chamomile/index.html.

[6] ENMA. URL:http://enma.aist-nara.ac.jp.

[7] 西馬一郎, 河合栄治, 知念賢一, 山口英, 山本平一. 通知によるコンテンツ一斉公開機構を用いた WWW クラスタシステム. 情報処理学会論文誌, Vol. 43, No. 11, pp. 3439–3437, 2002.