社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DSM－48 (5)
2008－QAI－26 (5)
2008／3／6

# HTTP セッションのハンドオーバによる WEB サーバのロードバランス

土居 幸一朗　後藤 滋樹

早稲田大学 〒169-8555 東京都新宿区大久保 3-4-1
E-mail: {doi, goto}@goto.info.waseda.ac.jp

**あらまし** 既存の web ロードバランシング技術は，新しく受信したリクエストを各バックエンドサーバに適切に割り当てることを目的にしている．しかし既存の方法では，何らかの原因でリクエストを配分する際に前提となった状況が変化しても，それに応じて負荷を動的に再配分することが難しい．本研究では，HTTP の標準に定められた部分的リソース取得機能を応用して，リバースプロキシ型のロードバランサを実現する．具体的には，実行中のデータ転送をサーバ間で動的にハンドオーバすることによって，負荷分散の精度を向上する新しい技法を提案する．

**キーワード** ロードバランス，負荷分散，WWW，HTTP，ハンドオーバ

# WEB Loadbalancing with HTTP Session Handover

Koichiro DOI and Shigeki GOTO

Waseda University 3-4-1 Ohkubo, Shinjuku-ku, Tokyo, 169-8555 Japan
E-mail: {doi, goto}@goto.info.waseda.ac.jp

**Abstract** The existing loadbalancing methods are mainly focused on the efficient distribution of the newly incoming requests among the backend servers. These methods cannot be used to redistribute the load when the condition has been changed unexpectedly during the data transmission. This paper proposes a new loadbalancing method which can dynamically handover HTTP sessions during the data transmission based on reverse-proxy loadbalancer. The method can improve the efficiency of the load distribution compared with existing methods.

**Key words** Loadbalancing, WWW, HTTP, partial content, handover

## 1. はじめに

### 1.1. 研究の背景

World Wide Web (WWW) の利用拡大が続いている．その背景には，youtube やニコニコ動画に代表されるビデオ配信サービスの普及や，Ajax のような Rich Internet Application (RIA) と呼ばれる開発手法の進展がある．

一般に，多数のユーザが利用する可能性があったり，応答速度などのユーザに対するサービス性能を向上させたりする必要がある web サイトでは，複数台のバックエンドサーバを利用する構成がとられる．そのようなサイトでは種々のロードバランシング手法を用いて，到来するリクエストを各サーバに分散させる．既存の手法には，DNS ラウンドロビン，L4 スイッチ，あるいはリバースプロキシを利用するタイプがある．これらの技術は，ボトルネックとなりうるノードの存在や，必要なグローバル IP アドレスの数などに違いがあり，それぞれ長所と短所を持っているため，サイトごとの条件を考慮して選択される．

従来の手法においては，配分されたリクエストに対するデータ転送を開始した後には，それを再配分できない．したがって，リクエストがロードバランサに到着した時点で負荷配分を行う前提となっていた条件が後に変化しても，動的に適応することができない．条件が変化する例としては，データ転送が終了したり，ユーザの操作によって中止されたりする場合が挙げられる．また，そもそもネットワーク帯域などの資源消費を事前には予測できない．新たに到来するリクエストをサーバに割り当てることで負荷を分散する手法には，分散精度に限界がある．

そこで，本研究では RFC 2616 (HTTP/1.1) に定められた部分的リソース（partial content）取得機能を応用して，目下実行中のデータ転送をサーバ間でハンドオーバする手法を提案する．以下，提案するバックエンドサーバの切替え手法を HTTP セッションハンドオーバと呼ぶ．

### 1.2. リバースプロキシ型ロードバランシング

本研究の提案手法では，リバースプロキシ型のロードバランシングの構成を前提とする．リバースプロキシとは，クライアントからのアクセスを代理として受信し，web サーバに仲介する役割をするサーバ（図 1）のことである．ここで，リバースプロキシを利用する

webサイトのドメイン名がwww.example.comであるとすると，ネームサーバでwww.example.comに対応するAレコードにロードバランサのIPアドレスを登録しておく．このようにして，クライアントに対してロードバランサへのリクエストを発行させることができる．バックエンドのwebサーバとクライアントはエンド・ツー・エンドの通信を行わないため，バックエンドサーバにグローバルIPアドレスを割当てる必要がなく，プライベートIPアドレスで運用することができる．以上の働きが，ローカルエリアネットワーク（LAN）から外部への接続を中継する役割を持つ通常のプロキシサーバとは逆の構成であるため，リバースプロキシと呼ばれる．

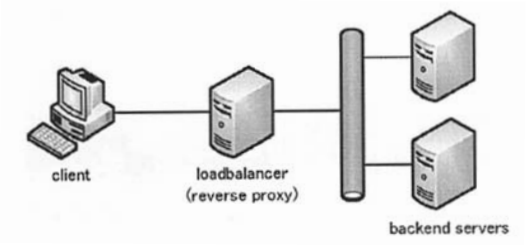


図 1 リバースプロキシ型ロードバランサ

リバースプロキシは，アプリケーション層において，クライアントからのアクセスを中継する．したがって，データ転送に際しては，クライアントからリバースプロキシの間，そして，リバースプロキシからバックエンドサーバの間で，合計2本のTCPセッションが確立される．

リバースプロキシ型のロードバランシングでは，バックエンドサーバを複数台利用する．クライアントからのリクエストを受信した時点で，何らかの基準（後述）で選択したバックエンドサーバに対してデータ転送を中継する．サーバ選択基準には，単純に順次サーバを割当てていくラウンドロビンのほか，サーバの応答時間やCPU負荷などの情報に基づいて負荷の小さいサーバを選択する手法がある．リバースプロキシ型では，すべてのトラフィックがロードバランサを通過するため，ロードバランサの処理能力やネットワーク帯域がシステムのボトルネックとなりやすいという欠点がある反面で，バックエンドサーバやクライアントの状況を把握しやすいという利点がある．

## 2. 提案手法

### 2.1. 部分的リソース要求機能

提案するバックエンドサーバの切替え機構をHTTPセッションハンドオーバと呼ぶ．HTTPセッションハンドオーバでは，前述のように，RFC 2616で規定された部分的リソース（partial content）要求機能を利用する．部分的リソース要求とは，HTTPのリクエストにRangeフィールドを付加した要求のことである．クライアントはRangeフィールドによって，対象リソース全体のうち必要とするデータ部分の開始オクテット数とオフセットの組を指定することができる．部分的リソース要求機能が実装されたwebサーバは，Rangeフィールドを含むGETリクエストを受信すると，要求されたリソースの指定部分をクライアントに対して返送する．この機能を利用することで，webブラウザでファイルのダウンロードを再開する場合などに，すでに受信したデータを重複して再受信せずに済むため，クライアント，サーバともに効率のよい通信を行うことができる．

### 2.2. メッセージシーケンス

HTTPセッションハンドオーバにより，バックエンドサーバを切替える際のシーケンスを説明する．以下，ドメイン名がwww.example.comであるwebサイトから，ファイルを取得すると仮定する．

(1) クライアントがネームサーバに対してwww.example.comのIPアドレスを問い合わせる．
(2) ネームサーバはロードバランサのIPアドレスを回答する．
(3) クライアントはロードバランサに対して通常のGETリクエストを発行し，ファイルを要求する．
(4) ロードバランサは使用するバックエンドサーバを選択し，そのサーバにリクエストを発行する．
(5) バックエンドサーバがデータ転送を開始する．ロードバランサは，バックエンドサーバからのデータをクライアントへ中継する．

ここで，データ転送が完了する前に，何らかの原因によりバックエンドサーバ間で負荷の偏りが生じたとする．

(6) ロードバランサは，現在転送を受けているバックエンドサーバとの接続を切断する．
(7) 相対的に負荷の小さいサーバを選び出し，部分的リソース取得要求によって，未転送の部分の転送を要求する．
(8) 新しく転送されたサーバがロードバランサへのデータ転送を開始する．ロードバランサはクライアントへデータを中継する．
(9) 再び不均衡が生じた場合は，(6)に戻る．
(10) クライアントに全データを転送し終えたら，クライアントとの接続を切断する．

このうち，(6) ～ (8) が HTTP セッションハンドオーバによるサーバ切替え手順である．これによってデータ転送中にバックエンドサーバが切替えられた場合でも，クライアント上の web ブラウザなどは，一切関知する必要がない．あたかも同一のサーバから継続的にデータを取得したのと同等の処理することができる．以上の手順を図 2 にまとめる．

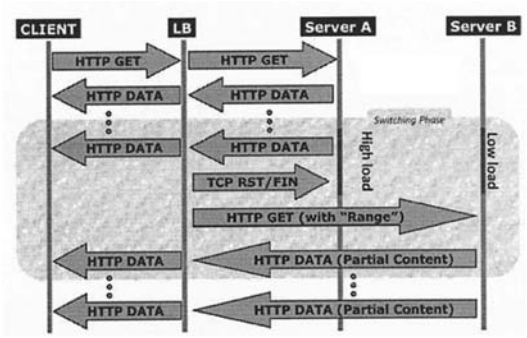


**図 2 提案手法のメッセージシーケンス**

## 3. プロトタイプの作成

提案手法に基づいて，プロトタイプを作成した．実装にあたっては，プログラミングの容易さを重視し，Ruby 言語の TCPSock ライブラリを利用した．

Ruby 言語などのスクリプト言語は，コンパイル型かつ低級言語の C 言語に比べて動作速度が遅い．したがって，本プロトタイプは動作速度の面で C 言語などによるチューニングの余地が存在する．逆に言えば，本プロトタイプが十分高速に動作すれば，提案手法が実用上問題ない速度で動作すると結論付けることができる．

ここでは純粋なオーバヘッドを計測するために，バックエンドサーバの切替えを引き起こすトリガーは手動操作とした．バックエンドサーバからクライアントへデータ転送が行われている最中にキーボードを打鍵すると，あらかじめ定められたサーバ間で HTTP セッションハンドオーバが行われるように実装した．

## 4. 評価実験

### 4.1. 互換性

#### 4.1.1. 目的と手順

提案手法で利用する部分的リソース取得機能は RFC 2616 に定められている．しかし，その実装については"SHOULD"（望ましい）とされており，"MUST"（必須）ではない．そのため，web サーバによっては部分的リソース取得機能が実装されておらず，提案手法を利用できない可能性がある．

そこで，web サーバとして利用されることの多い代表的なソフトウエアについて互換性を確認した．選定したサーバソフトウエアは，Netcraft の統計[3]により，インターネット上で利用されているシェアの大きなサーバ上位 5 位のうち，特定の blog サイトの運用が主な目的となっている 1 つを除いた 4 本である．4 つの合計シェアは 88.01%である．

#### 4.1.2. 結果

表に示すように，調査対象としたすべての web サーバソフトウエアについて，部分的リソース取得機能が実装されており，提案手法を利用可能であることが確認できた．

**表 1 提案手法の互換性**

| ソフトウエア名称/開発主体 | 互換性 |
|---|---|
| Apache Http Server/Apache Software Foundation | あり |
| Internet Information Service (IIS) 6.0/Microsoft | あり |
| Sun Java System Web Server/ Sun Microsystems | あり |
| lighttpd/ Jun Kneschke | あり |

### 4.2. オーバヘッド

#### 4.2.1. 目的と手順

提案手法では，2 つの点でオーバヘッドが生じる可能性がある．片方は，バックエンドサーバからクライアントへのデータ転送をロードバランサで中継することによる処理コストである．もう 1 つは，HTTP セッションハンドオーバによりバックエンドサーバを切り替える際に生じる時間的なコストである．

まず，中継による処理コストを調べるため，異なるサイズのファイルに対して，ロードバランサを経由した場合と経由しなかった場合について，ダウンロードに必要な時間を計測した．ファイルサイズは 2 MB から 4096 MB (4 GB) の間で変化させた．

次に，HTTP セッションハンドオーバによるコストを計測するために，ロードバランサによる切替えを行った場合，ロードバランサを使用するものの切替えを行わない場合，そしてロードバランサを利用しない場合の 3 通りで，1 GB のファイルのダウンロードに必要な時間を各々計測した．切替えを行う場合の回数は 1 ～3 回とした．

いずれの計測でも，ロードバランサのプロトタイプと httpd は同一マシンで動作させて実験を行った．サーバとして使用したマシンの詳細を表 2 に示す．

表 2 サーバマシンの詳細

| | |
|---|---|
| Linux kernel | 2.6.23 (Fedora 8) |
| CPU | Intel Core 2 Duo E6850 |
| メモリ | 2.0 GB (DDR2-800) |
| NIC | Intel 82566DC Gigabit Ethernet |
| httpd | Apache/2.2.6 |
| Ruby | 1.8.6 |

#### 4.2.2. 結果

ファイルサイズとファイル転送時間の関係を図 3 に示す．4 MB 以下のファイルでは，ロードバランサを経由しないダウンロードの方が，ロードバランサを経由する場合に比べて，わずかに必要時間が短かった．しかし，4 MB 以上ではロードバランサを経由する場合の方のダウンロード時間が短くなる場合がある．したがって，音楽や映像のファイルのような比較的サイズの大きなファイルに対しては，ほかの要素の寄与度に比べて，提案手法によるオーバヘッドは十分に小さいと言える．

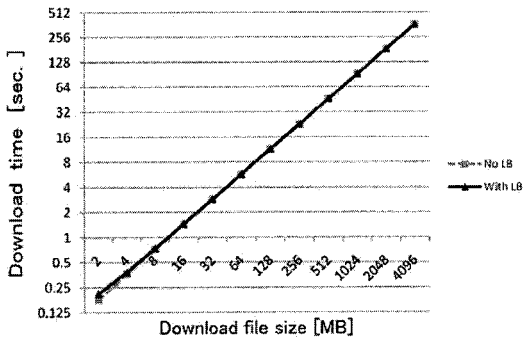


図 3 ファイルサイズとファイル転送時間の関係

次に，切替え回数とファイル転送時間の関係を図 4 に示す．ロードバランサを経由しない場合とハンドオーバを行う場合とで，ダウンロードに必要な時間に違いは生じなかった．試行によっては，ハンドオーバを行った場合でロードバランサを経由しない場合のダウンロード時間を下回る場合もあった．よって，提案手法によるハンドオーバによるコストは十分に小さいことがわかった．

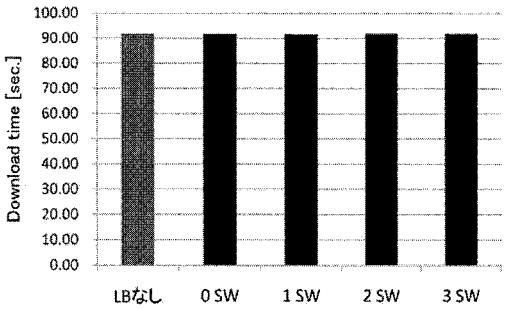


図 4 切替え回数とファイル転送時間の関係

### 4.3. 性能改善の例

#### 4.3.1. 目的と手順

Web サイトによって，ボトルネックとなりうる要素は異なる場合が多い．したがって，ロードバランシング手法を適用するにあたっても，ボトルネック要素を考慮した構成を工夫する必要がある．そのため，ある構成において，ある性能指標の改善があったからといって，一般的に web サイトの性能改善に資する手法を提案できたと主張することは難しい．

そこで，本実験では，web サーバに対してロードバランシングを利用する実測データに基づいて，提案手法による性能改善の例を示す．同時接続には Apache Foundation のベンチマークツール[4]を利用した．

#### 4.3.2. 結果

クライアント接続数と転送速度の関係を図 5 に示す．棒グラフが 1 接続あたりの転送速度，折れ線グラフが同時に接続しているクライアントの合計スループットを表す．同時接続数が 5 に達した時点で合計スループットは頭打ちとなる．

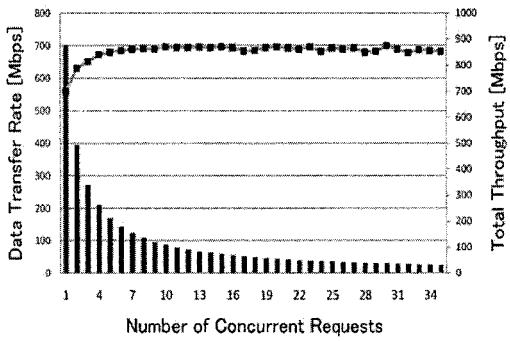


図 5 クライアント接続数と転送速度の関係

つまり，同時接続数が 5 となるまでは，同時接続数

表 3 HTTP セッションハンドオーバによる性能改善の例

| HO 前のクライアント数 | HO 後のクライアント数 | 平均速度の改善率 [%] | 最低速度の改善率 [%] |
|---|---|---|---|
| 1 台:3 台 | 2 台:2 台 | 18.70 | 31.14 |
| 2 台:5 台 | 3 台:3 台 | 8.29 | 22.56 |
| 3 台:5 台 | 4 台:4 台 | 5.93 | 19.11 |
| 4 台:6 台 | 5 台:5 台 | 3.69 | 16.05 |

が増加するに従って，サーバリソースの利用効率が向上する一方で，5 接続を超えてからは利用効率が向上していない．その場合は，別の 5 接続数以下のサーバにデータ転送をハンドオーバすることによって，全体で利用効率を向上させることができる．

図 5 の結果をもとに，2 台のバックエンドサーバ同士でクライアント接続数が 5 台以下となるように調整した場合の平均転送速度と最低転送速度の増加率を表 3 に示す．平均速度と最低速度を，それぞれ最大で 18.7%，31.1%向上させられることが分かった．

## 5. まとめと今後の課題

### 5.1. まとめ

既存の web ロードバランシング手法では，実行中のデータ転送を再配分することが出来ない．そこで，本研究では，リバースプロキシ型のロードバランシング手法を元に，データ転送中であっても，相対的に負荷の高いサーバから負荷の低いサーバへと移し替えられる機構を提案した．

切替え機構には RFC 2616 に定められた部分的リソース取得機能を利用した．HTTP によるデータ転送をバックエンドサーバ間で移し替えることから，提案機構を「HTTP セッションハンドオーバ」と名づけた．Web ブラウザに対しては，1 台のサーバからデータを転送し続けるのと同様に振る舞うため，提案手法を利用する上で，カスタマイズされたブラウザや特殊なプラグインを利用する必要はない．

Ruby 言語の TCPsock ライブラリを利用してプロトタイプを製作した．まず，プロトタイプを利用して各種 web サーバとの互換性を確認した．シェアの上で 88%を超える web サーバで提案手法が利用できることが分かった．また，ロードバランサがデータ転送を中継することによるコスト，HTTP セッションハンドオーバによるコストをそれぞれ計測し，それぞれによるオーバヘッドが十分に小さいことを確認した．さらに，ベンチマークツールを利用して，クライアントによる同時接続数と web サーバのデータ転送速度の関係を実測し，それに基づいて，実際に性能が改善される例を示した．

### 5.2. 今後の課題

提案手法は単純なデータファイルの転送を前提としている．そのため，動的に生成されるデータに対して適用することはできない．Web アプリケーションなどに対して提案手法を適用する上では，アプリケーションで個別に対応する必要がある．ただし，動的に生成される HTML のように，ごく小さなファイルに対しては，既存の手法によって，新しいリクエストをサーバに配分することで十分な負荷配分が実現できるため，負荷を動的に再配分する意義は大きくない．

実サイトに提案手法を適用する上では，切替えトリガーを自動化する必要がある．ロードバランサにおいて，各サーバに対する接続数や，サーバからの応答時間を監視して，一定の差が生じた段階でハンドオーバを行うというような実装が考えられる．どの実装が適切であるかは，個々の web サイトの構成に依存する．また，切替えタイミングの設定については，頻繁な切替え（フラッタリング）が生じないような工夫が必要である．

本研究で示した性能改善の例は，特定のボトルネック要素を想定せずに，クライアント数を分散させることを目標として性能向上を図った．また，ダウンロード時のデータ転送速度のみを性能指標として採用した．しかし，Web サイトの構成や，着目する性能指標によっては，同時接続数に対する性能変化の傾向が異なる可能性がある．したがって，対象サイトに対して十分なベンチマークを行った上で，適切なトリガーを実装する必要がある．

## 文　献


[1] K. Doi and S. Goto, “Load Balancing with HTTP Session Handover”, Core University Seminar, Jeju, Korea, Oct. 2007.

[2] RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1, http://www.ietf.org/rfc/rfc2616.txt

[3] Netcraft, “October 2007 Web Server Survey”, http://news.netcraft.com/archives/2007/10/11/october_2007_web_server_survey.html

[4] Apache Foundation, “ab - Apache HTTP server benchmarking tool”, http://httpd.apache.org/docs/2.0/programs/ab.html