# 動的コンテンツのための動的ミラーリング機構の設計および評価

<table>
<tr><td colspan="3">香川大学大学院工学研究科<br>博士前期課程信頼性情報システム工学専攻<br>修　士　論　文</td></tr>
<tr><td colspan="2">修了年度</td><td>平成　　年度（　　年度）</td></tr>
<tr><td colspan="2">指導教員</td><td></td></tr>
<tr><td rowspan="3">審査担当教員</td><td>主査</td><td></td></tr>
<tr><td rowspan="2">副査</td><td></td></tr>
<tr><td></td></tr>
</table>

**香川大学大学院 工学研究科**
**博士前期課程 信頼性情報システム工学専攻**

**河田 光司**

**平成 21 年 3 月 13 日**

# Design and Evaluation of Dynamic Mirroring System for Dynamic Contents

**Abstract** Mirror servers are used to solve high-load problem of Web server. There is a problem that mirror servers are idle when the number of requests of accesss to Web pages provided by them is not so high. To solve the problem is proposed dynamic mirroring system. However, Not only static contents but also dynamic contents are widespread in Web site. Mirroring of dynamic contents is very difficult because there content may vary at every access. In this thesis, the freshness date is introduced for dynamic contents. it proposes and designs dynamic mirroring system for dynamic contents. The system is assumed that the contents can be guaranteed with their freshness date and it is possible to mirror them. It evaluates the number of arrival requests and response time of Web server and mirror server, and freshness of dynamic contents.

**あらまし** Web サーバの高負荷問題を解決する方法の一つとしてミラーリングという手段がある。このミラーリングという手段には、Web ページへのアクセスが少ない時にミラーサーバが遊ぶという問題がある。この問題を解決するために、動的ミラーリング機構の研究を行っている。しかし、Web サイトの中には静的コンテンツだけでなく動的コンテンツも含むものが普及してきている。静的コンテンツのみを扱うミラーリングではミラーリングが困難な Web サーバが増加している。本研究では、動的コンテンツに賞味期限を与え、期限内はコンテンツ内容を保証しミラーリング可能とする動的コンテンツのための動的ミラーリング機構を提案・設計した。さらに、機構を運用したときの Web サーバとミラーサーバのリクエスト到着数および応答時間、動的コンテンツの鮮度の分析・評価した。その結果、クライアントからのリクエスト数が増加しても、各サーバへのリクエスト到着数は閾値 (高負荷状態) を超えず、ある程度のリクエスト数に限られるが賞味期限内の保証されたコンテンツを提供できることが分かった。

**キーワード** 動的ミラーリング、動的コンテンツ、賞味期限、コンテンツの鮮度、コンテンツの分類、負荷分散

# 目次

# 第 1 章

# はじめに

現在、各家庭へのパソコンの普及に伴い、インターネットを利用するユーザが年々増加している。アクセスするクライアントの増加に伴い Web サーバへの負荷は増大する一方である。そこで、Web サーバへの負荷を軽減させるために、ミラーリングという手法が用いられている。ミラーリングとは Web サーバ上にあるデータと同じものをミラーサーバに保持させ、クライアントからのリクエストを分散させて、サービスを提供するという手法である。

従来のミラーリング時のアクセス振分けでは、ミラーサーバを明示的に公開し Web サーバも含め、それらの中からクライアントが選択する方法、1 つのホスト名に対して複数の IP アドレスを定義しアクセスを振り分ける DNS ラウンドロビンや負荷分散装置を用いてミラーサーバを意識させる方法 [1] などが知られている。これらの方法では、ミラーサーバのデータは特定の Web サーバのものになっており、その Web サーバしかミラーリングできない。よって、複数の Web サービス提供者のサーバに対して、ミラーサーバを充分に活用することができない。また、高負荷時以外の時でもミラーサーバを用意しておく必要がありコストがかかる。

この問題を解決するために、我々の研究室では動的ミラーリング機構の研究 [2][3] を行っている。動的ミラーリング機構とは、Web サーバの負荷状況に応じて動的にミラーサーバを提供し、一時的にミラーリングを行うものである。ミラーサーバは複数の Web サーバに対してミラーリングすることができるので、ミラーサーバが Web サーバに対して固定的になることがなく共有できる。高負荷時にのみミラーサーバを提供することによりミラーサーバを予め用意する必要がなくなり、コストがかからなくなる。

また、現状の Web サイトは Web サービスの多様化により、静的コンテンツに加え動的コンテンツを含むものが増えている。一般的に動的コンテンツは静的コンテンツに比べ、Web サーバに対する負荷が高いので、動的コンテンツへのアクセスが増えていくとサーバが高負荷になりやすい。動的コンテンツは常に変化することが多いので、静的コンテンツのように単純なデータのコピーをすることができず、ミラーリングをすることは困難である。従来の静的コンテンツのみを

扱う動的ミラーリング機構では、動的コンテンツをアップロードしている Web サーバへのミラーリングをすることはできない。

本研究では、静的コンテンツだけでなく動的コンテンツに対してもミラーリングを可能とした、動的コンテンツのための動的ミラーリング機構について提案し、その設計を行う。最初に、動的コンテンツの特徴を分析し、ミラーリング可能なものと不可能なものに分類する。その後、動的コンテンツに対応した動的ミラーリング機構に必要な要件を述べ、その機構を設計する。最後に、機構の評価を行う。

本論文の構成は、第 2 章では動的コンテンツの概要を説明し、動的コンテンツをミラーリング可能なものと不可能なものに分類する。第 3 章で動的ミラーリング機構の概要を説明し、必要な要件を述べる。第 4 章では提案する機構を詳細に説明する。第 5 章は第 4 章で設計された機構を評価し、有効性を考察する。第 6 章は論文全体のまとめと今後の課題について言及する。

# 第 2 章

# 動的コンテンツ

本研究で扱う動的コンテンツの特徴および動的コンテンツを分類するときに用いるパラメータについて説明する。

## 2.1 動的コンテンツの特徴

動的コンテンツとはクライアントからのリクエストに応じて、動的に生成される Web コンテンツのことである。具体的には Java や PHP、Perl などの言語で記述され、クライアントからのリクエストに応じて、Web ページ (HTML ファイルや画像ファイルなど) を生成しレスポンスを返す。検索サイト、掲示板、Weblog といったものが動的コンテンツの代表的なものである。

静的コンテンツとは以下の点で違いがある。

- Web サイトの構成
- Web ページの変化の有無
- コンテンツの更新者

Web サイトの構成は静的コンテンツの場合、図 2.1 のように HTTP サーバのみで構成され、動的コンテンツは、図 2.2 のように HTTP サーバとアプリケーションサーバ、データベースサーバで構成されることが多い。静的コンテンツをミラーリングする場合は、HTTP サーバ上にある HTML ファイルをミラーサーバにコピーすることでミラーリングすることが可能であるが、動的コンテンツをミラーリングする場合は、アプリケーションサーバとデータベースサーバが Web ページを生成するので Web ページをそのままコピーすることができない。よって、動的コンテンツをミラーリングするには、生成された Web ページをコピーする仕組みが必要がある。

Web ページの変化は静的コンテンツの場合、クライアントのリクエストごとに Web ページが変化することはなく、同一の内容が返される。動的コンテンツの場合は、クライアントからのリ

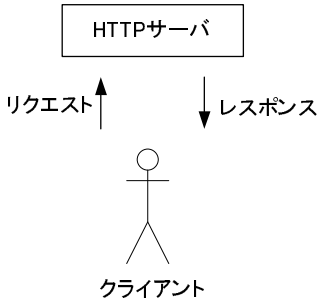


図　2.1 Web サイトの構成 (静的コンテンツ)

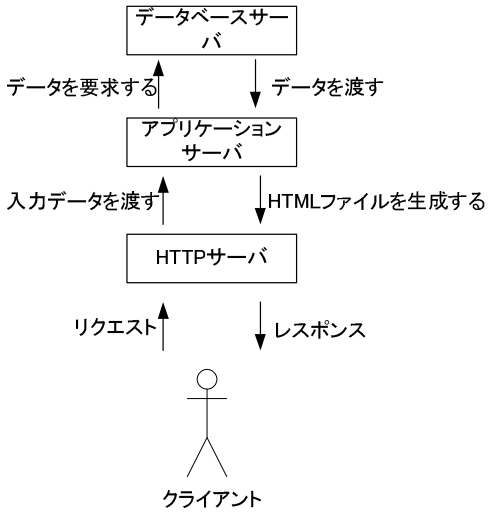


図　2.2 Web サイトの構成 (動的コンテンツ)

クエストごとに Web ページが変化することが多いので、返される内容が常に変化する可能性がある。ミラーリングする場合、静的コンテンツは内容が変化しないので単純に Web ページをコピーすればよい。動的コンテンツは、常に内容が変化しているので、変化を考慮したコピーを行う必要がある。

コンテンツの更新者は静的コンテンツの場合、コンテンツ作成者のみになる。動的コンテンツの場合の更新者は、クライアントがコンテンツの内容を更新する可能性があるので、多くのクライアントが更新者となる。ミラーリングする場合、静的コンテンツはコンテンツ作成者がコンテンツを更新したときに同期をとればよいが、動的コンテンツはクライアントがコンテンツ内容を更新することを考慮したデータの同期が必要になる。

## 2.2 動的コンテンツの分類

動的コンテンツの中で、ミラーリングすることが可能なものについて分析する。動的コンテンツは前節で述べたように、クライアントのリクエストに応じてコンテンツの内容の変化が発生する。また、コンテンツの内容は常に更新されていくことが多いので、ある時点でコピーしたコンテンツは時間が経過していくと新鮮さを失っていく (Web サーバ上のコンテンツと内容が違っていく)。そこで、コンテンツ変化の度合とコンテンツ鮮度の重要性という 2 つのパラメータで動的コンテンツを分類しミラーリング可能なものを分析する。

### 2.2.1 コンテンツ変化の度合

コンテンツ変化の度合とは Web サーバがクライアントからのリクエストを受取った時、動的コンテンツにどのように反映するかということである。分類すると表 2.1 のようになる。

(1) はクライアントからのリクエストが発生しても、データを参照するのみのコンテンツである。リクエスト時に送信した入力データがある場合は、データを絞り込む条件などに利用するのみになる。よって、コンテンツ内容が更新されることはない。一般的にコンテンツ作成者がコンテンツ内容を変更しない限り更新されることはない。(1) において、同一のページを生成するもの

表 2.1 コンテンツ変化の度合

| 変化の度合 | 具体例 |
|---|---|
| (1) 内容を読込むだけで変化しない | 検索、広告表示など |
| (2-a) 内容が定期的に変化 (追記、書換) する | アンケートなど |
| (2-b) 内容が即座に変化 (追記、書換) する | オークション、掲示板、Weblog など |

は静的コンテンツと同様に扱えるのでミラーリング可能である。自動的に複数の種類の Web ページを生成するものは生成されるページを全てコピーし、Web サーバ上の Web ページと同条件でミラーサーバ上のキャッシュを提供できればミラーリング可能になると考える。

(2-a) はクライアントからのリクエストが発生したとき、入力データを即座にコンテンツへ反映しないので、反映されるまでは同一の Web ページが生成されるコンテンツである。リクエスト時に送信した入力データは、定期的にコンテンツへ反映される。よって、コンテンツに反映されるタイミングでデータのコピーを行うことによりミラーリング可能となる。

(2-b) はクライアントのリクエストが発生したとき、入力データを即座にコンテンツへ反映しリクエストの度にコンテンツ内容が更新されるコンテンツである。リクエスト時に送信した入力データは即座に反映されていくので、ある時点でコンテンツをコピーしたとしても、時間が経過すると Web サーバのコンテンツとミラーサーバのコンテンツは違ってくる。よって、コンテンツ内容の鮮度が低くなっていくと考えられる。ミラーリングする上ではコンテンツ内容の鮮度を考慮する必要がある。(2-b) をミラーリングするかどうかは次に説明するコンテンツ鮮度の重要性によって決定する。

### 2.2.2　コンテンツ鮮度の重要性

コンテンツ鮮度の重要性とは、コンテンツ内容の鮮度がクライアントに影響するかという尺度である。例えば、オークションなどの最新情報を得る必要のある即時性をもつコンテンツは鮮度の重要性が高く、逆にブログや掲示板といった最新情報でなくても充分な情報を得られるコンテンツは鮮度の重要性が低いと考えられる。重要性の高いコンテンツをミラーリングすると古い情報をクライアントに与える可能性があるのでミラーリングすべきではない。逆に重要性の低いコンテンツは古い情報でも充分な情報を得られるので、ミラーリング可能であると考える。

本研究では、内容を読込むだけで変化しないコンテンツ、内容が定期的に変化 (追記、書換) するコンテンツ、内容が即座に変化 (追記、書換) するがコンテンツ鮮度の重要性の低いコンテンツをミラーリング可能なコンテンツとする。

# 第 3 章

# 動的ミラーリング機構

動的ミラーリング機構の概要と負荷分散の関連研究・関連技術について説明する。さらに動的コンテンツに対応した機構の要件、実現すべき透過性について説明する。

## 3.1　概要

動的ミラーリング機構とは、Web サービス提供者の運用する Web サーバが高負荷になる時、一時的にミラーサーバを提供し、負荷分散を行う機構 (図 3.1) である。

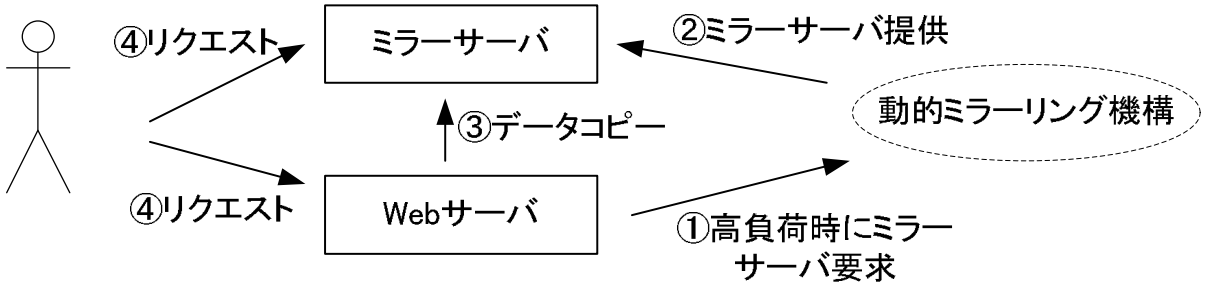


図　3.1 動的ミラーリング機構

Web サービス提供者の運用するサーバが高負荷になる時、動的ミラーリング機構にミラーサーバを要求し、機構からミラーサーバが提供される。その後、ミラーリングするコンテンツがコピーされ、クライアントからのリクエストが分散される。これにより、Web サービス提供者は予めミラーサーバを用意する必要がなく、資源の有効利用ができる。本研究では動的コンテンツがアップロードされている Web サーバに対して、動的ミラーリングを提供できる機構を提案する。次に、動的コンテンツに対応した動的ミラーリング機構に要求される透過性、要件について述べる。

## 3.2　透過性

透過性とは、分散された構成をクライアントに意識させず、あたかも集中型のシステムであるかのように見せることである。透過性の種類を表 3.1 に示す。

表　3.1 透過性の種類

| 透過性 | クライアントに対しての特徴 |
|---|---|
| アクセス透過性 | 常に同一のリソースアクセス方法を提供する |
| 位置透過性 | リソースの位置を意識させない |
| 移動透過性 | 使用中のリソースの移動を意識させない |
| 複製透過性 | リソースが複数に複製されていることを意識させない |
| 並行透過性 | 競合状態を意識させない |
| 障害透過性 | リソースの障害を意識させない |
| 永続透過性 | リソースが永続的または揮発的であるかを意識させない |
| 規模透過性 | システム規模の拡張・縮小を意識させない |

**●アクセス透過性**

クライアントがリソースに対して同一操作でアクセスできることである。動的ミラーリングでは Web サーバ、ミラーサーバどちらに対しても Web ブラウザを用いてアクセスできるのでアクセス透過性は実現されている。

**●位置透過性**

リソースの物理的位置を指定しなくてもクライアントがリソースにアクセスできることである。動的ミラーリングでは、例えばラウンドロビン DNS を採用することにより 1 つのドメイン名で Web サーバとミラーサーバの両方を参照することができる。具体的には図 3.2 のように、クライアントは Web ブラウザに対してドメインを入力する。Web ブラウザは DNS サーバからドメインのアドレス情報を取得する。その後、Web ブラウザがアドレス情報を基に Web サーバもしくはミラーサーバへアクセスする。以上より、位置透過性は実現されている。

**●移動透過性**

この透過性はリソースが移動してもクライアントに影響がないことである。図 3.3 のように、サーバ 1 にあったリソースがサーバ 2 に移動しても、クライアントは意識せずにそのままサーバ 2 のリソースにアクセスすることができる。動的ミラーリングでは Web サーバのコンテンツがミラーサーバに移動して Web サーバ上から消失するといった処理はないので、移動透過性は考慮する必要はない。

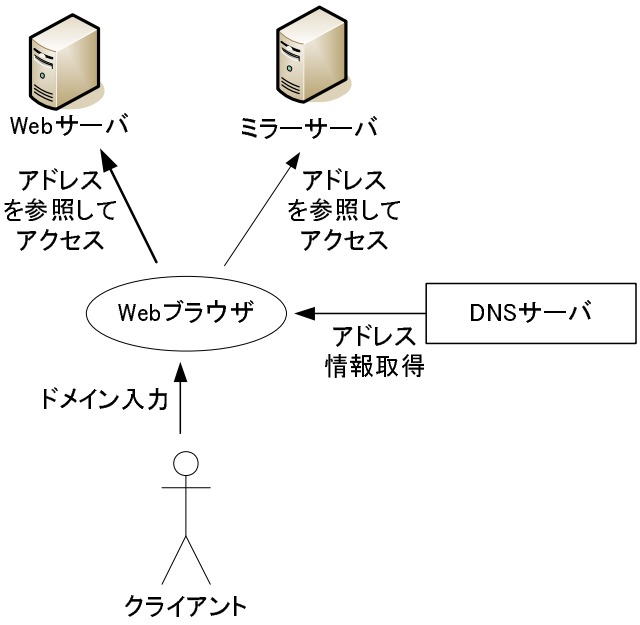


図　3.2 位置透過性

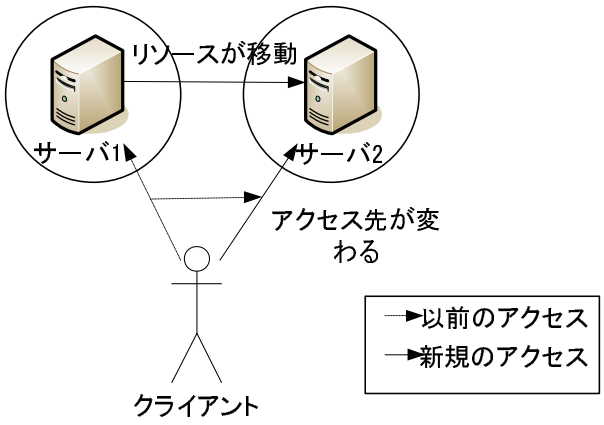


図　3.3 移動透過性

**●複製透過性**

これはクライアントに対してリソースが複数に複製されていることを意識させないことである。静的コンテンツをミラーリングする場合、クライアントは Web サーバにアクセスしてもミラーサーバにアクセスしても同様の静的コンテンツ内容を得られる状態であるので複製透過性を満たしている。しかし、動的コンテンツをミラーリングする場合、クライアントは Web サーバにアクセスした時、最新情報のコンテンツを、ミラーサーバにアクセスした時、古い情報のコンテンツを取得することになるので、異なる内容を取得してしまう可能性がある (図 3.4)。これより、複製透過性を満たしていない。本研究では、完全な複製透過性は満足できないが、動的コンテンツに賞味期限を付加し、その賞味期限内では古い情報であっても十分価値のあるコンテンツであると考え、賞味期限内においてはミラーリングを行う。

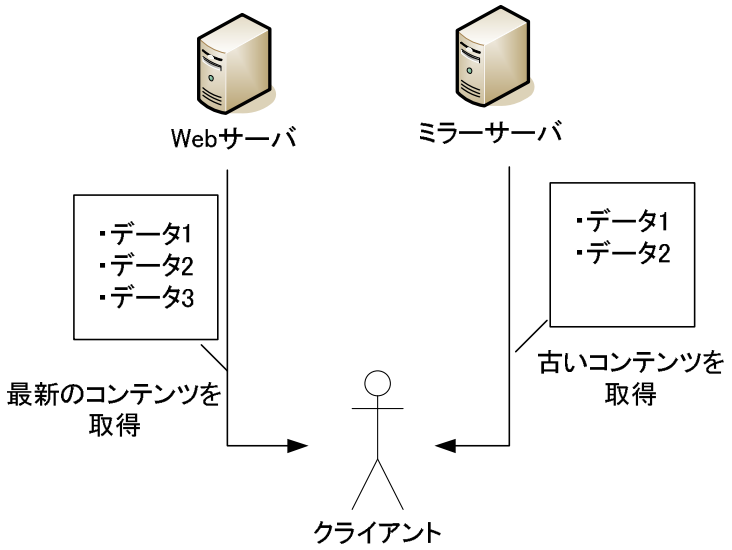


図　3.4 複製透過性

**●並行透過性**

これはリソースを複数のクライアントが処理しているときに、競合状態を意識させないことである。動的コンテンツの場合、複数のクライアントがサーバに対してリクエストを要求し、サーバ側でリクエスト応じたレスポンスの生成を行うので、各クライアントにサーバ側の競合状態を意識させない仕組みが必要である。

**●障害透過性**

リソースに障害が発生しても、クライアントに意識させないことである。動的ミラーリングでは、ミラーリング中、Web サーバ、ミラーサーバに障害が発生したとき、別のミラーサーバに切りかえるなどして、なるべく障害をクライアントに意識させない仕組みが必要である。

**●永続透過性**

これはリソースが永続的な記憶装置にあるか、揮発性の記憶装置にあるかをクライアントに意識させないことである。動的ミラーリングでは、永続的な記憶装置 (ハードディスク) にコンテンツは保存されるので、この透過性を考慮する必要はない。

**●規模透過性**

これはサーバを増やして規模を拡張する処理をクライアントに意識させないことである。動的ミラーリングではクライアントのアクセスによる負荷の大きさによりミラーサーバが自動的に増減することになるので規模透過性を満たしている。

以上より、動的ミラーリング機構は複製透過性、並行透過性、障害透過性を考慮しながら設計する必要がある。

## 3.3　要件

動的ミラーリング機構が動的コンテンツを扱う場合、必要となる機能にはリクエストの振分け、ミラーサーバの管理、ミラーリングするコンテンツの選択、コンテンツのコピーがある。各要件について述べる。

### 3.3.1　リクエストの振分け

リクエストの振分けとは、クライアントからのリクエストを Web サーバまたはミラーサーバに振分ける機能である。振分け方法にはブラウザを利用する方法、Web サーバを利用する方法、中継機器を利用する方法、DNS を利用する方法がある。

**●ブラウザを利用する方法**

ブラウザ上のプログラムでクライアントからのリクエストを振分ける方法である。ミラーリング実行中に Web サーバもしくはミラーサーバから、コピーされた動的コンテンツの情報およびミラーサーバ情報を、クライアントのブラウザに渡し、その情報を基にクライアントがブラウザを操作してアクセスする。

- 利点
  - 機構側に機能を追加する必要がない
  - クライアント毎にアクセス先を振分けることができる
- 欠点
  - ブラウザ上に機能を実装する必要がある

– 各クライアントがミラーサーバ情報を必要とするので情報量が多くなる

**● Web サーバを利用する方法**

Web サーバに振分け機能を組込む方法である。クライアントからのアクセスが Web サーバに来たとき、コピーされた動的コンテンツの情報を基に次回のアクセス先を指定する方法である。

- 利点
  - Web サーバのみに振分け機能を実装すればよい
  - Web サーバで自由にアクセスを振分けられる
- 欠点
  - Web サーバにアクセスが集中し、Web サーバがダウンした場合振分けられない
  - 振分け処理のオーバヘッドが負担になり、さらに高負荷になる可能性がある

**●中継機器を利用する方法**

クライアントと機構の間に中継機器 (レイヤ 7 スイッチやプロキシ機能を持ったサーバ) を設置し、クライアントのリクエストを振分ける方法である。コピーされた動的コンテンツの情報を中継機器に渡し、中継機器がその情報を基にクライアントのリクエストを振分ける。

- 利点
  - Web サーバ、ミラーサーバに振分け機能によるオーバヘッドがかからない
- 欠点
  - 中継機器にアクセスが集中する
  - 中継機器がダウンした場合、振分け機能が利用できなくなる

**● DNS を利用する方法**

ダイナミック DNS、ラウンドロビン DNS の機能を利用しアクセスを振分ける方法である。機構はミラーリング開始時にコピーされたコンテンツの情報を基に、DNS 上の IP アドレス情報を更新する。さらに、コンテンツの情報が変化したりミラーサーバの状況が変わる度に動的に DNS 上の IP アドレス情報を更新する。

- 利点
  - アクセスが集中する箇所がない
  - 機構側に機能を追加する必要がない

- 欠点
  - DNS の更新時に遅延が生じ、リクエスト振分け処理が遅れる場合がある

以上のような議論があり各方法には一長一短があるので最適な方法は条件・状況によって変わってくる。中継機器を用いた方法は Web サーバ、ミラーサーバに対して振分け機能のオーバヘッドがかかりにくく、仕組みが単純であると考えたので、本研究ではこの方法を採用したときの分析を行っている。

### 3.3.2　ミラーサーバの管理

ミラーサーバの管理とはミラーサーバと Web サーバの関係が多対多であった場合に、各ミラーサーバと Web サーバの対応関係の構成を管理するものである。ミラーサーバの状態の監視やミラーサーバの提供、返却する処理が必要である。また、動的コンテンツを扱う場合、賞味期限が切れるまではミラーリングを実行し続けるなどの処理が必要になってくる。管理方法には集中管理と分散管理がある。

**●集中管理**

各 Web サーバと各ミラーサーバの間に集中管理サーバを置き、Web サーバからのミラーリング要求やミラーサーバからのミラーリング情報などを一括して集中管理サーバに送信する方法である。集中管理サーバは Web サーバとミラーサーバの対応関係やミラーリング中のコンテンツ情報などを考慮して、Web サーバにミラーサーバを割当てる。割当ては全て集中管理サーバで行う。

- 利点
  - 負荷、ミラーリング状況は集中管理サーバにのみ送られるので、通信量が少ない
  - 割当て処理を集中管理サーバでのみ実装すればよい
  - 割当てが完了するまでのステップが短い
- 欠点
  - 集中管理サーバがダウンすると、機構全体がダウンするので、信頼性が低い

**●分散管理**

分散管理は各ミラーサーバをミラーサーバ群ととらえ、各ミラーサーバが周辺のミラーサーバの負荷状況やミラーリング状況などの管理情報を知っている。これより、一つのミラーサーバが持っている管理情報を少なくでき、群全体ではそれらを探索することにより全ての管理情報を網羅することができる。Web サーバはミラーサーバ群に対してミラーリング要求を行う。

- 利点
  - 一つのミラーサーバがダウンしても、他のミラーサーバが補助することができる
  - 各ミラーサーバが持つ管理情報は少ない
- 欠点
  - ミラーサーバ間のトラフィック量が大きくなってしまう
  - トラフィックやステップが膨大にならない管理アルゴリズムが必要である

以上のような議論があり各方法には一長一短があるので最適な方法は条件・状況によって変わってくる。集中管理を用いた方法が一番ミラーサーバ間のトラフィック量が少ないので、本研究ではこの方法を採用したときの分析を行う。

### 3.3.3　ミラーリングするコンテンツの選択

ミラーリングするコンテンツの選択とはWebサービス提供者がアップロードしたコンテンツをミラーリング可能なものと不可能はものに分類し、ミラーリング可能なものの中からミラーリングするコンテンツを選択する機能である。本論文ではミラーリング可能なものと不可能なものに分類する方法は言及しない。評価ではWebサービス提供者が予め手動で分類しているものと仮定する。オンデマンドとプリフェッチが代表的な方法である。

**●オンデマンド**

オンデマンドとは、Webサーバに対してクライアントからのリクエストが到着したとき、リクエストされたコンテンツをミラーサーバにコピーしていく方法である。Webサーバにクライアントからのリクエストが到着する度にコンテンツをミラーサーバにコピーしていく。１度リクエストされたコンテンツはミラーサーバにコピーされるのでクライアントはミラーサーバにリクエストを要求してコンテンツを取得することができる。

- 利点
  - 資源の使用量が少ない
- 欠点
  - 最初のリクエストは必ずヒットしないので、応答時間が長くなる可能性がある
  - ヒット率が低いコンテンツもキャッシュされてしまう

**●プリフェッチ**

プリフェッチとはヒット率の高いと予測されるコンテンツを予めコピーする方法である。この方法は予測するアルゴリズムの精度によって、ヒット率が大きく変動する。コンテンツをすべてコピーする方法と一部をコピーする方法がある。動的コンテンツの場合、クライアントのリクエスト (質問) に応じて、複数の Web ページが生成される。1 つの動的コンテンツであっても、クライアントから複数の種類の質問がある場合、質問毎に別の Web ページが生成される。よって、生成される Web ページ数が膨大になる可能性があるために、すべてコピーする方法を利用するのは難しい。予測する方法としてはアクセス頻度などの統計データを用いた統計的推測やデータ内のリンク先、関連キーワードを用いた連想的推測がある。

- 利点
  - 前もってデータがコピーされるので、ミラーリング開始が早い
  - Web サーバの負荷の低い時にコピーすることができる
- 欠点
  - 精度の高い予測方法が必要になる

以上のような議論があり各方法には一長一短があるので最適な方法は条件・状況によって変わってくる。プリフェッチを用いる方法が一番 Web サーバへの更新のオーバヘッド低く、負荷の低い時にコピーすることもできると考えたので、本研究ではこの方法を採用したときの分析を行う。

### 3.3.4　コンテンツのコピー

コンテンツのコピーとは Web サーバ上の静的・動的コンテンツをミラーサーバにコピーし Web サーバとミラーサーバ間でデータの整合性を保つ処理でである。動的コンテンツをコピーする場合には賞味期限を考慮する必要があり、賞味期限内にコンテンツの同期を行わなければならない。コンテンツのコピー方法は以下のものがある。

**●コンテンツの賞味期限が過ぎる直前に同期**

この方法は賞味期限までコンテンツの同期を待つ方法である。賞味期限まで同期を待つので同期回数は少ない。しかし、賞味期限内においても、Web サーバ上のコンテンツ内容は更新されている可能性があるので鮮度は低くなる。

**● Web サーバ上のコンテンツが変化したときに同期**

この方法は Web サーバでコンテンツが更新する度に、同期する方法である。コンテンツを更新する度に同期を行うので、鮮度は高いままになる。しかし、更新頻度が高いコンテンツの場合は、同期のオーバヘッドが高くなってしまい、Web サーバがさらに高負荷になってしまう。

**●負荷が低い状態のときに同期**

この方法は賞味期限内において、Web サーバの負荷が低い状態になったときに同期する方法である。Web サーバの負荷が低い状態のときに同期を行うので、Web サーバに負担がかからない。ミラーリング中に低い状態にならない場合は、賞味期限を過ぎる直前に同期する方法と同様になる。

以上より、Web サーバ上のコンテンツが変化したときに同期する方法が一番鮮度の高い状態を保てるので、本研究ではこの方法を採用したときの分析を行う。

## 3.4　関連研究・関連技術

本研究の動的ミラーリング機構と負荷分散に関する関連研究・関連技術との違いを記述する。負荷分散に関する関連研究・関連技術には CDN、P2P、Spider Cache、プロキシサーバなどがある。

**● CDN(Contents Delivery Network)**

CDN とはファイルサイズの大きいデジタルコンテンツをネットワーク経由で配信するために最適化されたネットワークのことである。この CDN はコンテンツ配信のための専用ネットワークを構築する必要があり、多くの一般企業やコンテンツプロバイダが提供している [4]。また、CDN 技術を利用して Web サーバのコンテンツを動的にグループ化してグループ単位でミラーリングする方式 [5] などが提案されている。

CDN の仕組みは図 3.5 のようにコンテンツサーバがクライアントのリクエストに対して、コンテンツを所持した最適なエッジサーバの URL をクライアントに通知して負荷分散する。これにより、オリジナルのコンテンツサーバはアクセスが集中せず高負荷になることはない。さらに、クライアントに対して通信距離が近く、低負荷状態のサーバが提供される可能性が高いので、コンテンツを短時間で取得することができる。一般的に CDN は動画ファイルなど大規模データの配信に利用されている。また、CDN は専用ネットワークであるので、専用ネットワーク内の固定されたエッジサーバ間で負荷分散を行うことになり、CDN に参加しない限り一般のサービス提供者は利用することができない。

動的コンテンツをミラーリング対象としている部分が本研究とは大きく違っている。CDN では動的コンテンツにおいて、同じ内容を生成する部分のみコピーすることができるが、クライアントのリクエスト毎にコンテンツ内容が変化する Web ページに関してはコピーすることができない。

**● P2P(Peer to Peer)**

P2P とは複数のコンピュータが相互に接続し、直接ファイルなどを送受信するネットワークの形態である。Winny[6] などの P2P では、ネットワークにコンテンツがアップロードされると、コンテンツは図 3.6 のようにコピーが繰り返され、拡散していく。アップロードしたサーバはコンテンツを管理することが難しくなるのでミラーリングには適さない。本研究は、P2P 技術とは違

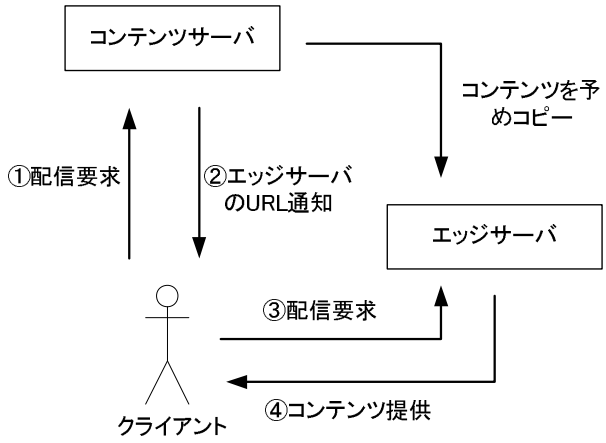


図　3.5 CDN

いミラーサーバ管理機能から指定されたミラーサーバ内でコンテンツのコピーが行われるので、Web サーバがコピーされたコンテンツ全てを把握することができる。よって、コンテンツが拡散していくことはない。

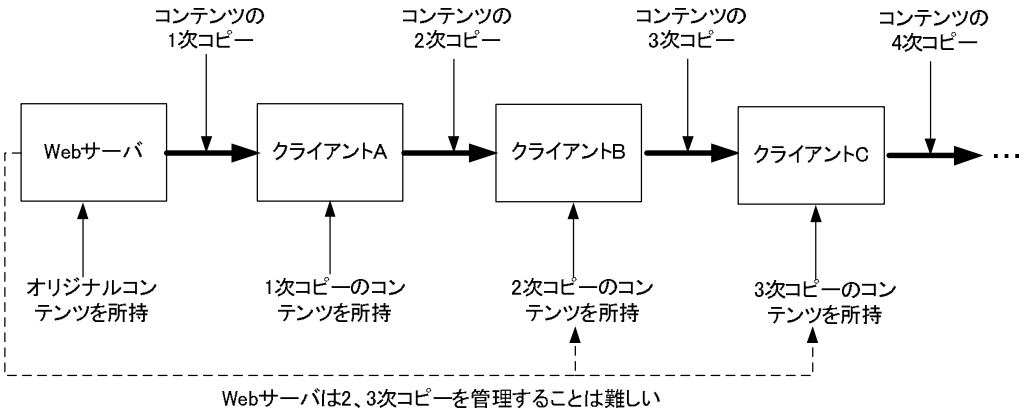


図　3.6 コンテンツの拡散

**● Spider Cache**

Spider Cache はインターネットアクセラレータに分類され、1 度リクエストされた動的コンテンツをキャッシュとして保持する装置である [7]。2 回目以降はそのキャッシュを利用して Web サーバの負荷を軽減する (図 3.7)。 Spider Cache は Web サービス提供者のネットワークに存在している。そして、サービス提供者自身がキャッシュ可能なコンテンツを分類し、各コンテンツ毎にキャッシュの設定を行っている。しかし、Spider Cache の場合、一人のサービス提供者の Web サーバしかキャッシュすることができず、複数のサービス提供者の Web サーバに対してキャッシュ

することは不可能である。本研究の動的ミラーリング機構は複数のサービス提供者の Web サーバに対してミラーリングが可能である。

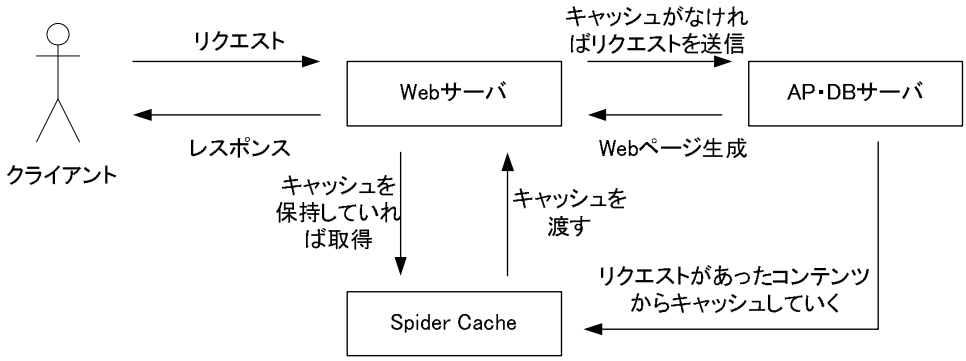


図　3.7 Spider Cache

**●プロキシサーバ**

プロキシサーバとはクライアントのコンピュータに代わって、代理としてインターネットとの接続を行なうコンピュータのことである。代理で接続した後に、コンテンツをキャッシュしておく。2 回目以降の接続からはキャッシュを所持していればキャッシュを返し、所持していなければ Web サーバから取得するようになる。クライアントのリクエストに応じてキャッシュしていくので pop 型のシステムになる。pop 型のシステムの場合、キャッシュが行われるまで負荷分散ができない。ヒット率の低いコンテンツが多くキャッシュされる可能性がある。また、キャッシュしたコンテンツを Web サーバ自身が把握できないといった問題がある。本研究の動的ミラーリング機構では、Web サーバ自身がミラーリングするコンテンツを選択し、賞味期限を決めてコピーすることが可能なので、コピーしているコンテンツを把握することができる。

# 第 4 章

# 動的コンテンツのための動的ミラーリング機構の設計

前章で述べた要件・透過性を考慮しながら、機構の構成、処理の流れについて記述する。

## 4.1　構成

本研究で提案する動的ミラーリング機構は以下の目的を満たせるように設計する。

- 動的コンテンツに対して賞味期限を付加し、できる限り鮮度の高い状態のコンテンツをクライアントに提供する。
- 機構から一時的にミラーサーバを提供し、一般のサービス提供者の Web サーバに対してミラーリングを行う。Web サーバとミラーサーバの両方とも高負荷状態に陥らないようにする。

提案する動的コンテンツのための動的ミラーリング機構の構成概念図を図 4.1 に示す。

動的ミラーリング機構は Web サーバ、ミラーサーバ、リクエスト振分け機能、ミラーサーバ群管理機能で構成される。図 4.1 では Web サーバ A はミラーサーバ群管理機能からミラーサーバ a、ミラーサーバ b が提供されミラーリングを行っている。Web サーバ B はミラーサーバ群管理機能からミラーサーバ b、ミラーサーバ c が提供されミラーリングを行っている。クライアントはリクエスト振分け機能にリクエストを送信し、適切なサーバからレスポンスを受け取っている。

機構内の各サーバ、各機能について説明する。

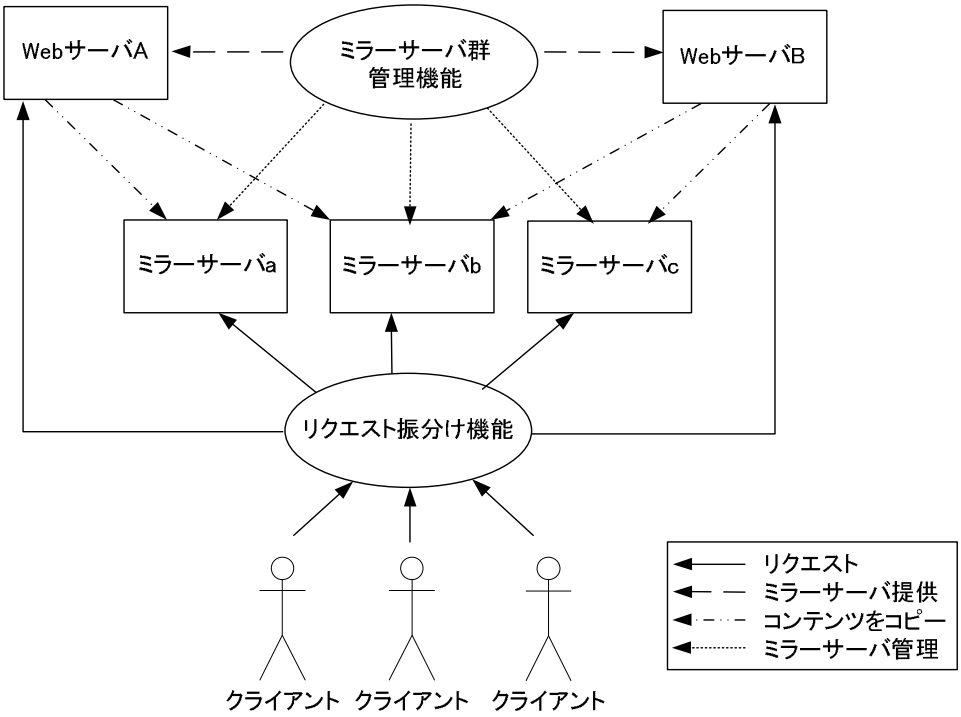


図　4.1 構成概念図

**● Web サーバ**

一般のサービス提供者が所持する Web サーバである。Web サーバが高負荷になる時、ミラーリングが開始する。この Web サーバ上にアップロードされている動的・静的コンテンツがミラーリングの対象になる。ミラーリング中はミラーリングしているミラーサーバの情報、ミラーリング中のコンテンツの情報を所持している。Web サーバは以下の処理を行う。

- サーバ負荷を計測し、高負荷になるときミラーサーバ管理機能にミラーサーバを要求する
- 低負荷状態が続くと、ミラーサーバ管理機能にミラーサーバを返却する
- 動的コンテンツの中からミラーリング可能なものを分類し、可能なものの中からミラーリングするコンテンツを選択する
- 選択したコンテンツをミラーサーバにコピーし、その後、更新間隔毎にコンテンツの同期処理を行う

**●ミラーサーバ**

ミラーサーバは一般のサービス提供者が所持する Web サーバに対してミラーリングを行うサーバである。Web サーバ上の動的・静的コンテンツがコピーされている。1 台のミラーサーバが複

数台の Web サーバのミラーリングを行う場合もある。ミラーリング中は、Web サーバの情報とミラーリング中のコンテンツの情報を所持する。ミラーサーバは以下の処理を行う。

- サーバ負荷を計測し、高負荷になると新たにミラーサーバを追加する
- 更新間隔毎にミラーリング中のコンテンツの同期処理を行う

**●リクエスト振分け機能**

リクエスト振分け機能とはクライアントからのリクエストを Web サーバまたはミラーサーバへ振分けるものである。ミラーリング中のコンテンツ情報およびクライアントのリクエストの種類 (最新情報が必要なリクエストもしくは古い情報でもすぐに欲しいリクエスト) によって振分け先を変更する。

**●ミラーサーバ管理機能**

ミラーサーバ管理機能は Web サーバからのミラーリング要求に対し、ミラーサーバを割当てる。また、ミラーサーバの状態を監視する。ミラーサーバの負荷状況、コンテンツのミラーリング状況を基に、Web サーバにとって最適なミラーサーバを提供する。

## 4.2　処理の流れ

機構における処理の流れを述べる。処理にはミラーサーバの提供、ミラーサーバの返却、クライアントからのリクエストの振分け、コンテンツの同期がある。

### 4.2.1　ミラーサーバの提供

ミラーサーバが Web サーバに提供されるまでの流れを図 4.2 に示す。図 4.2 では Web サーバへミラーサーバ a が提供されている。

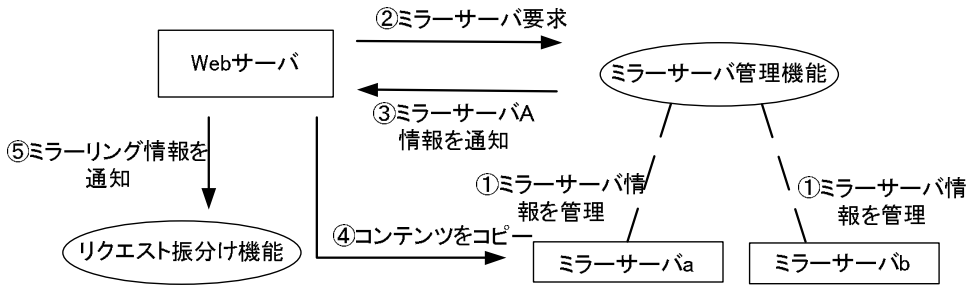


図　4.2 ミラーサーバ提供の流れ

①ミラーサーバ管理機能は常にミラーサーバの状況を監視している。

②Web サーバが高負荷になる時、ミラーサーバ管理機能にミラーサーバを要求する。

③ミラーサーバの状況を基に最適なミラーサーバを選択し Web サーバにミラーサーバ情報を送信する。

④ミラーリングするコンテンツを選択し、提供されたミラーサーバにコピーする。

⑤ミラーサーバのアドレス、ミラーリング中のコンテンツ情報を通知し、ミラーリングを開始する。

### 4.2.2　ミラーサーバの返却

ミラーリングが終了しミラーサーバが機構へ返却されるまでの流れを図 4.3 に示す。図 4.3 ではミラーリング中のミラーサーバ a が返却されるまでの流れになる。ミラーリングが終了すると自動的にミラーサーバ上のコンテンツは賞味期限が切れて消去される。

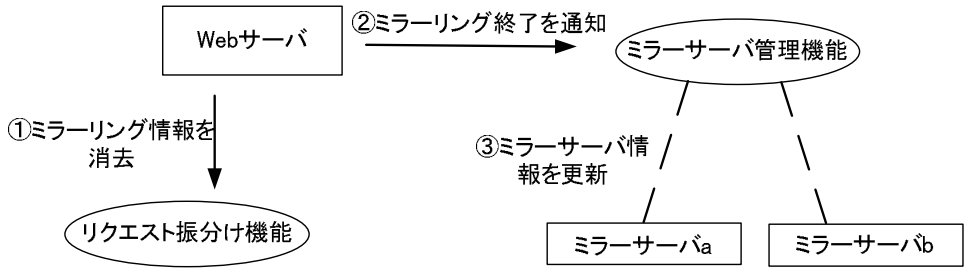


図　4.3 ミラーサーバ返却の流れ

①Web サーバの負荷が低負荷状態になると、ミラーリング情報を消去し、振分け処理を中止する。

②管理機能へミラーリング終了を通知する。

③連絡後、ミラーサーバの情報 (割当て終了) を更新する。以上よりミラーリングが終了する。

### 4.2.3　クライアントからのリクエストの振分け

ミラーリング実行中のクライアントからのリクエストの振分けの流れを図 4.4 に示す。クライアントが最新情報を要求する場合と古い情報でも可とする場合が存在する。

①コンテンツをリクエストする。

②最新情報を求めるリクエストは Web サーバへ振分ける。古い情報でも可とするリクエストの場合でも、要求するコンテンツのコピーがミラーサーバ上に存在しなければ Web サーバへ振分ける)

③最新の情報をクライアントに返信する。

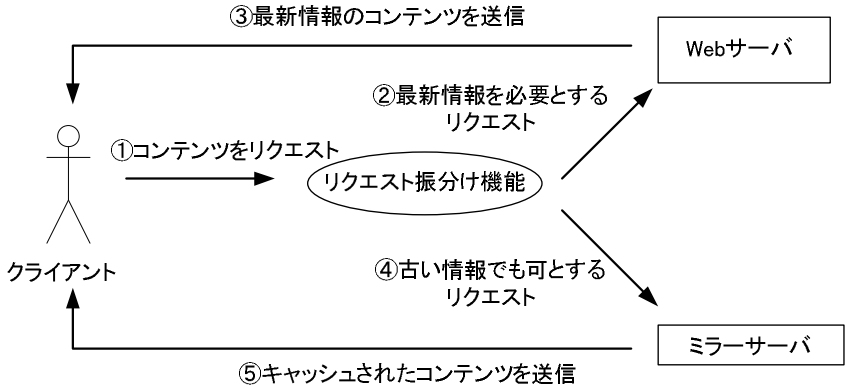


図　4.4 クライアントからのリクエストの流れ

④古い情報でも可とするリクエストはミラーサーバに求めるコンテンツのコピーが存在すればミラー　　サーバへ振分ける。

⑤コピーされたコンテンツをクライアントに返信する。

### 4.2.4　コンテンツの同期

コンテンツの同期とはミラーリング実行中に更新間隔毎にコンテンツの同期をとる処理である。これによりコンテンツの新鮮さを保つことができる。図 4.5 にミラーリング実行中のコンテンツの同期の流れを示す。

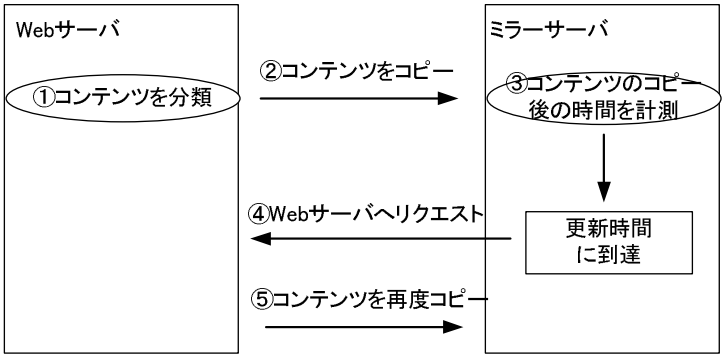


図　4.5 コンテンツの同期の流れ

①コンテンツをミラーリング可能なものと不可能なものに分類する。
(今回の分析では予め分類されている)

②ミラーリング開始時、ミラーリング可能なコンテンツをミラーサーバへコピーする。

③コピーしたコンテンツのコピー後の経過時間を計測する。

④コピー後の経過時間が指定した更新時間 (更新間隔) になると Web サーバへコンテンツのリクエストを送信する。

⑤リクエストを受取ると、再度、最新のコンテンツをミラーサーバへコピーする。

# 第 5 章

# 評価

前章で設計した動的コンテンツのための動的ミラーリング機構の特性について分析を行い、その後、機構が動作したときの性能を評価する。

## 5.1　データモデル

設計した動的ミラーリング機構の特性について分析する。特にクライアントのリクエスト、更新間隔 (更新間隔毎にコンテンツを同期する) が変化したときのコンテンツの鮮度、各サーバのリクエスト到着数および応答時間の変化について分析する。データモデルを図 5.1 に示す。

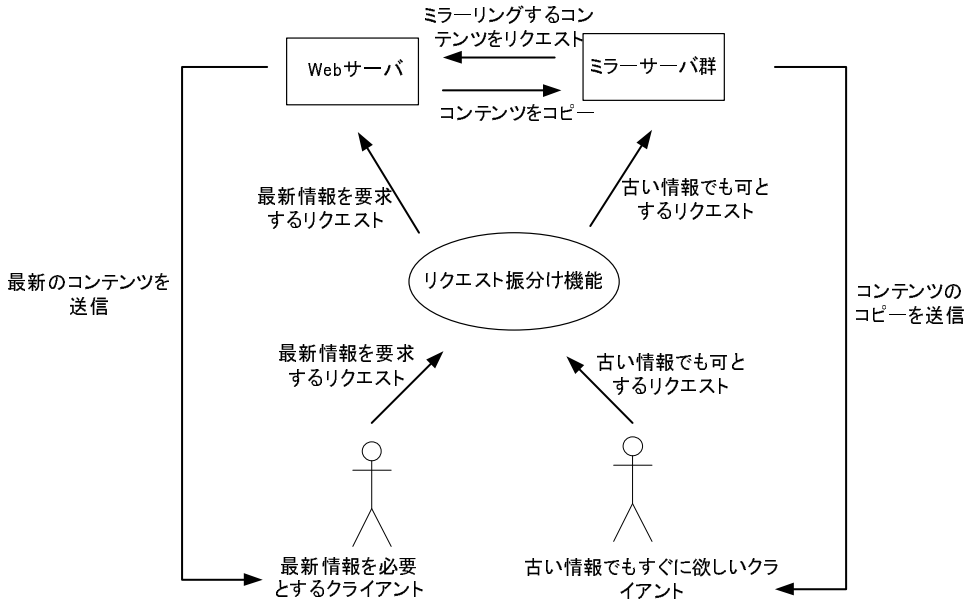


図　5.1 データモデル

データモデルでは、クライアントは最新情報を必要とするクライアントと古い情報でもすぐに欲しいクライアントに分類される。最新情報を必要とするクライアントはリクエスト振分け機能を介し、Web サーバから最新のコンテンツを取得する。古い情報でもすぐに欲しいクライアントはリクエスト振分け機能を介し、応答時間の早いミラーサーバからコンテンツのキャッシュを取得する。ミラーサーバ群は指定した更新間隔で Web サーバへコンテンツのリクエストを送信し、リクエストが到着すると Web サーバは再度ミラーサーバへコンテンツをコピーする。

データモデルで用いられるパラメータはクライアント側のものとサーバ側のものに分けられる。クライアント側のパラメータを表 5.1 に示す。クライアント側の各パラメータについて説明する。

表 5.1 クライアント側のパラメータ

| パラメータ名 | 変数名 | 単位 |
|---|---|---|
| リクエスト (最新情報) | $N_{latest-dynamic}, N_{latest-static}$ | 単位時間あたりリクエスト数 |
| リクエスト (古い情報) | $N_{old-dynamic}, N_{old-static}$ | 単位時間あたりリクエスト数 |
| 最新情報を得るまでの応答時間 | $T_{latest}$ | 秒 |
| 古い情報を得るまでの応答時間 | $T_{old}$ | 秒 |
| 得られた情報の鮮度 | $F$ | 値 |

リクエスト (最新情報) とは最新情報を必要とするクライアントのリクエスト数のことである。$N_{latest-dynamic}$ は動的コンテンツを要求するリクエスト数、$N_{latest-static}$ は静的コンテンツ要求するリクエスト数である。

リクエスト数 (古い情報) は古い情報でもすぐに欲しいクライアントのリクエスト数のことである。$N_{old-dynamic}$ は動的コンテンツを要求するリクエスト数、$N_{old-static}$ は静的コンテンツ要求するリクエスト数である。

最新情報を得るまでの応答時間とは最新情報を必要とするクライアントが Web サーバへコンテンツを要求してから得られるまでの時間のことである。

古い情報を得るまでの応答時間とは古い情報でもすぐに欲しいクライアントがミラーサーバへコンテンツを要求してから得られるまでの時間のことである。

得られた情報の鮮度とは、ミラーサーバから取得したコンテンツの新鮮さを表す。値が高いほどコンテンツは新しく値が低いほどコンテンツは古い。1.0 を最大値とし、0.0 を下回ると賞味期限が切れている。

次に、サーバ側のパラメータを表 5.2 に示す。 サーバ側の各パラメータについて説明する。

コンテンツの賞味期限とはコンテンツの価値を保証する時間のことである。ミラーリングする静的コンテンツ数および動的コンテンツ数とはミラーリングを行うコンテンツの数である。$C_{dynamic}$

表 5.2 サーバ側のパラメータ

| パラメータ名 | 変数名 | 単位 |
|---|---|---|
| コンテンツの賞味期限 | $T_{freshness-date}$ | 秒 |
| ミラーリングする動的コンテンツ数および静的コンテンツ数 | $C_{dynamic}, C_{static}$ | 値 |
| コンテンツのコピー後の経過時間 | $T_{copy}$ | 秒 |
| コンテンツの更新間隔 | $T_{update}$ | 秒 |
| Web サーバのコンテンツ処理時間 | $T_{web-process}$ | 秒 |
| ミラーサーバのコンテンツ処理時間 | $T_{mirror-process}$ | 秒 |
| ミラーサーバ数 | $M$ | サーバ台数 |
| Web サーバへのリクエスト到着数 | $L_{web}$ | 値 |
| ミラーサーバへのリクエスト到着数 | $L_{mirror}$ | 値 |

は動的コンテンツ、 $C_{static}$ は静的コンテンツの数である。コンテンツのコピー後の経過時間はミラーサーバにコンテンツをコピーした後の時間である。コンテンツの更新間隔とはミラーサーバ上にあるコンテンツを更新し Web サーバ上のコンテンツと同期する間隔のことである。Web サーバおよびミラーサーバのコンテンツの処理時間とは、クライアントの 1 回のリクエストに対してのコンテンツを生成するまでの処理時間のことである。Web サーバへのリクエスト到着数とは Web サーバに対して単位時間に到着するリクエストの総数のことである。ミラーサーバへのリクエスト到着数とはミラーサーバに対して単位時間に到着するリクエストの総数のことである。リクエスト到着数が多いほど、サーバの負荷が高い。

## 5.2 モデルの分析

前節で述べたパラメータを用いて、各サーバへのリクエスト到着数および応答時間、コンテンツの鮮度と賞味期限の関係について分析する。

### 5.2.1 Web サーバへのリクエスト到着数および応答時間

Web サーバへのリクエスト到着数および応答時間を分析する。Web サーバへのリクエスト到着数は以下の数式のようになる。
ミラーリング開始前

$$L_{web} = N_{latest-dynamic} + N_{latest-static} + N_{old-dynamic} + N_{old-static} \tag{5.1}$$

ミラーリング開始後

$$L_{web} = N_{latest-dynamic} + N_{latest-static} + (1-\alpha)(N_{old-dynamic} + N_{old-static}) \\ +M(C_{dynamic} + C_{static})a(t) \tag{5.2}$$

$$\begin{pmatrix} a(t) = 1 & (t = nT_{update}) & (n = 1, 2, 3 \cdots) \\ a(t) = 0 & (t \neq nT_{update}) & (n = 1, 2, 3 \cdots) \end{pmatrix}$$

ミラーリング開始前は全てのリクエストが Web サーバで処理されるので (5.1) 式のようになる。ミラーリング開始後は最新情報を要求するリクエストは全て Web サーバへ割当てられ、古い情報を要求するリクエストは $\alpha$ の割合でミラーサーバへ割当てられるので、クライアントからのリクエスト到着数は

$$N_{latest-dynamic} + N_{latest-static} + (1-\alpha)(N_{old-dynamic} + N_{old-static})$$

となる。古い情報を要求するリクエストがミラーサーバへコピーされたコンテンツを全く要求しない場合は $\alpha = 0$、全て要求している場合は $\alpha = 1$ となる。一般的に

$$\alpha = \frac{C_{dynamic} + C_{static}}{max(C_{dynamic}) + max(C_{static})}$$

である。$max(C_{dynamic})$、$max(C_{static})$ はミラーリングできるコンテンツの最大数のことである。$max(C_{dynamic})$ はミラーリングできる動的コンテンツの最大数、$max(C_{static})$ はミラーリングできる静的コンテンツの最大数となる。また、コンテンツが同期する時、ミラーサーバから Web サーバへのリクエストが存在する。よって、全ミラーサーバが更新間隔ごとにコンテンツの数だけリクエストを送信するので、$M(C_{dynamic} + C_{static})a(t)$ が加算され、(5.2) 式になる。

Web サーバからの応答時間 (最新情報を得る時間) は以下の数式で表される。

$$T_{latest} = T_{web-process} \times L_{web} \tag{5.3}$$

Web サーバを待ち行列 (行列は 1 列) とし、この待ち行列の処理が完了するまでの時間を Web サーバで発生する待ち時間とする。このとき、1 回のコンテンツ処理時間は $T_{web-process}$、行列の長さはリクエスト到着数であるので $L_{web}$ である (図 5.2)。よって、Web サーバで発生するコンテンツ処理時間は (5.3) 式となる。

更新間隔 $T_{update}$ とリクエスト (古い情報)$N_{old-dynamic} + N_{old-static}$ を変化させたときの Web サーバへのリクエスト到着数の総数は図 5.3、図 5.4 のようになる。この分析の条件は表 5.3 のようになる。

図 5.3 の結果より、更新間隔が長くなるほどリクエスト到着数が $\frac{1}{n}$ のオーダで減少している。これは更新間隔が長くなるほど計測時間中の更新回数が少なくなるからである。更新間隔が短いほ

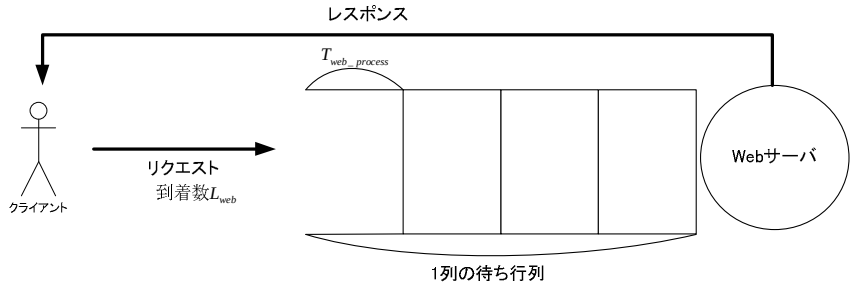


図 5.2 Web サーバ上の待ち行列

表 5.3 Web サーバへのリクエスト到着数および応答時間を分析する時の条件

| パラメータ | 数式 | 値 |
|---|---|---|
| リクエスト (最新情報) | $N_{latest-dynamic} + N_{latest-static}$ | 1000 |
| リクエスト (古い情報) | $N_{old-dynamic} + N_{old-static}$ | 0〜2000 |
| Web サーバのコンテンツ処理時間 | $T_{web-process}$ | 0.01 |
| ミラーサーバのコンテンツ処理時間 | $T_{mirror-process}$ | 0.01 |
| 更新間隔 | $T_{update}$ | 1〜10 |
| $\alpha$ | $\alpha$ | 0.5 |

どコンテンツの鮮度は高くなるが、その分 Web サーバへのオーバーヘッドが高くなってしまう。

また、図 5.4 の結果より、リクエスト (古い情報) が増加すると、リクエスト到着数が $n$ のオーダで増加している。リクエスト (古い情報) が $n$ のオーダで増加していくので、同様にリクエスト到着数も $n$ のオーダで増加している。

図 5.3、図 5.4 と同様の条件で、更新間隔 $T_{update}$ とリクエスト (古い情報) $N_{old-dynamic}$+$N_{old-static}$ を変化させたときの最新情報を得るまでの応答時間は図 5.5、図 5.6 のようになる。

図 5.5 の結果をみると、リクエスト到着数の時と同様に、更新間隔が長くなるほど応答時間が $\frac{1}{n}$ のオーダで減少している。また、図 5.4 の結果もリクエスト到着数と同様にリクエスト (古い情報) が増加すると、応答時間が $n$ のオーダで増加している。これは応答時間がリクエスト到着数に比例しているからである。Web サーバの負荷が比較的低い場合、高負荷にならない範囲で更新間隔を短くすることにより鮮度を高くすることができる。

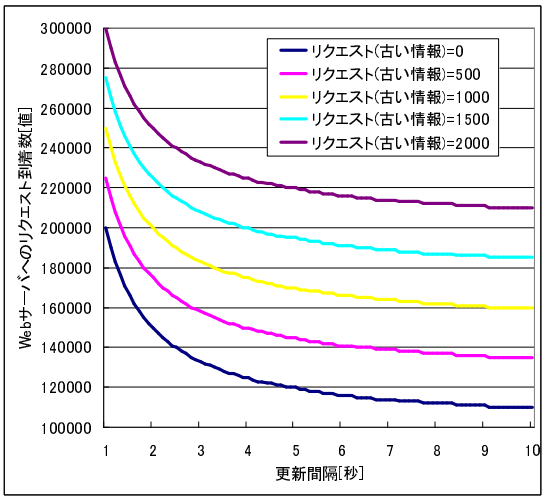


図　5.3 Web サーバへのリクエスト到着数 (X 軸=更新間隔)

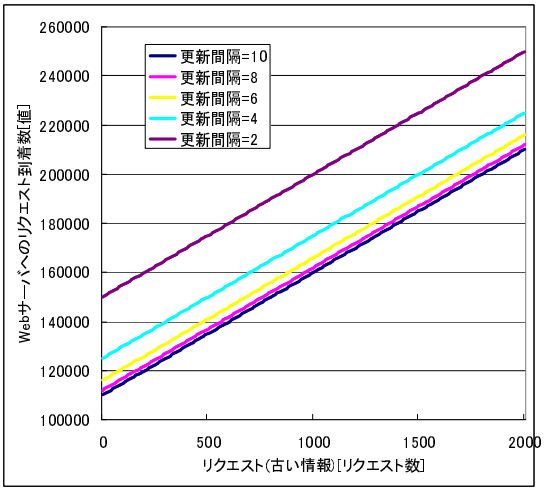


図　5.4 Web サーバへのリクエスト到着数 (X 軸=リクエスト (古い情報))

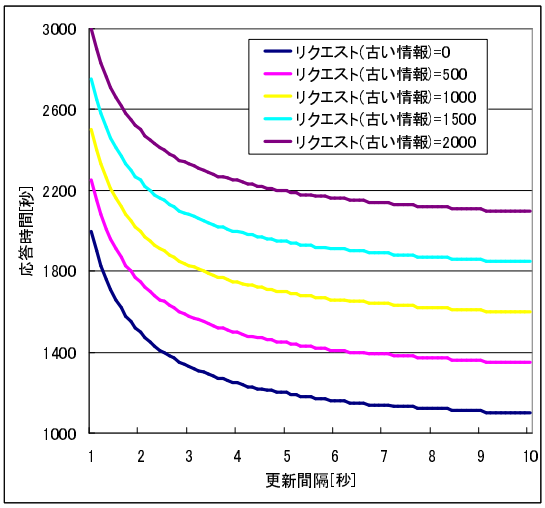


図　5.5 Web サーバからレスポンスを得るまでの応答時間 (X 軸=更新間隔)

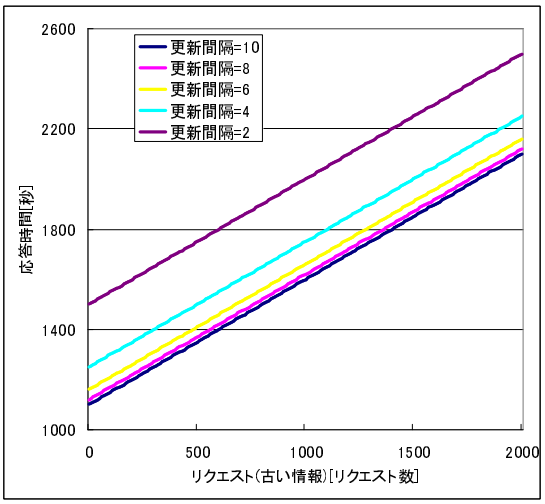


図　5.6 Web サーバからレスポンスを得るまでの応答時間 (X 軸=リクエスト (古い情報))

### 5.2.2　ミラーサーバへのリクエスト到着数および応答時間

ミラーサーバへのリクエスト到着数および応答時間を分析する。ミラーサーバへのリクエスト到着数は以下の数式のようになる。

$$L_{mirror} = \frac{\alpha(N_{old-dynamic} + N_{old-static})}{M} \tag{5.4}$$

ミラーリング開始前は全リクエストを Web サーバ側で処理するので $L_{mirror} = 0$ である。ミラーリング開始後は古い情報を要求するリクエストがミラーサーバに振分けられる。そのリクエ

ストはサーバ台数分だけ分散されるので (5.4) 式のようになる。$\alpha$ は Web サーバへのリクエスト到着数の係数 $\alpha$ と同様である。また、ミラーサーバ群全体では

$$M \times \frac{\alpha((N_{old-dynamic} + N_{old-static})}{M} = \alpha((N_{old-dynamic} + N_{old-static})$$

となる。

ミラーサーバからの応答時間 (古い情報を得る時間) は以下の数式で表される。

$$T_{old} = T_{mirror-process} \times L_{mirror} \tag{5.5}$$

ミラーサーバを待ち行列 (行列は 1 列) とし、この待ち行列の処理を完了するまでの時間をミラーサーバで発生する待ち時間とする。このとき、1 回のコンテンツ処理時間は $T_{mirror-process}$、行列の長さはリクエスト到着数であるので $L_{mirror}$ である (図 5.7)。よって、Web サーバで発生するコンテンツ処理時間は (5.5) 式となる。

リクエスト (古い情報)$N_{old-dynamic} + N_{old-static}$ を変化させた時のミラーサーバへのリクエスト到着数の総数と古い情報を得るまでの応答時間は図 5.8、図 5.9 のようになる。この分析の条件は表 5.4 である。

リクエスト到着数と応答時間の両方ともリクエスト (古い情報) が増加すると、$n$ のオーダで増加している。リクエスト到着数はリクエスト (古い情報) に $\frac{\alpha}{M}$ を乗算したものであるので $n$ のオーダで増加する。応答時間はリクエスト到着数に比例しているので同様に $n$ のオーダで増加する。

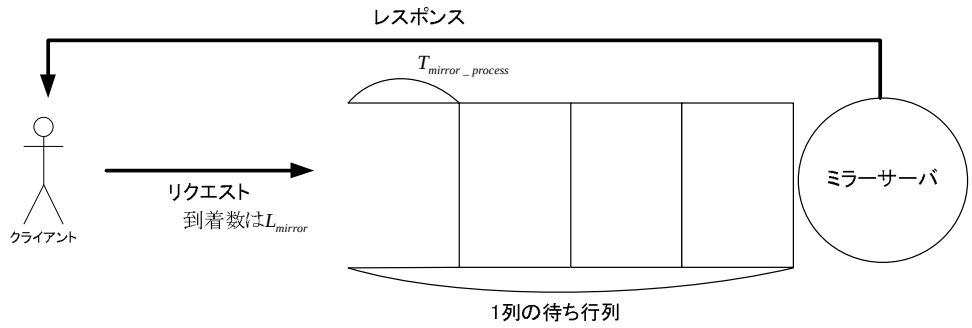


図 5.7 ミラーサーバ上の待ち行列

表 5.4 ミラーサーバへのリクエスト到着数および応答時間を分析する時の条件

| パラメータ名 | 数式 | 値 |
|---|---|---|
| リクエスト (最新情報) | $N_{latest-dynamic} + N_{latest-static}$ | 1000 |
| リクエスト (古い情報) | $N_{old-dynamic} + N_{old-static}$ | 0〜2000 |
| Web サーバのコンテンツ処理時間 | $T_{web-process}$ | 0.01 |
| ミラーサーバのコンテンツ処理時間 | $T_{mirror-process}$ | 0.01 |
| $\alpha$ | $\alpha$ | 0.5 |

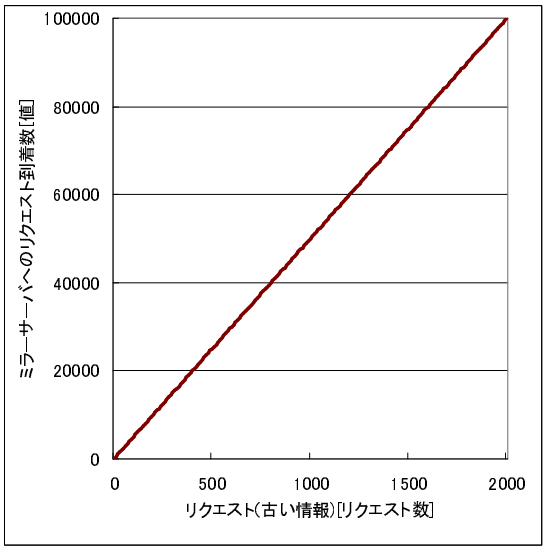


図 5.8 ミラーサーバへのリクエスト到着数

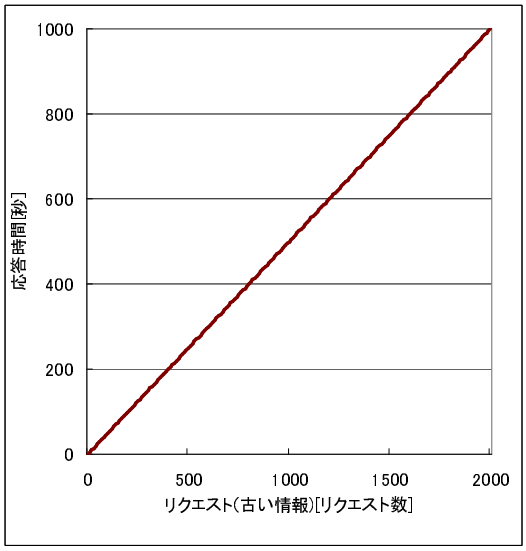


図 5.9 古い情報を得るまでの応答時間

### 5.2.3　コンテンツの鮮度

コンテンツの鮮度は賞味期限が長くなるほど高く。コピー後の経過時間が長いほど低くなる。よって、以下の数式のようになる。

$$F = 1 - \frac{T_{copy}}{T_{freshness-date}} \tag{5.6}$$

鮮度は 1 に近いほど高く、0 に近いほど低い。負数になると賞味期限を過ぎている。また、コンテンツの鮮度を保証することができる経過時間の最大値が更新間隔になるので、(5.6) 式をを変形し $T_{copy} \leq T_{freshness-date}(1-F)$ の式を満たす $T_{copy}$ の最大値が $T_{update}$ となる。つまり、保障する鮮度と更新間隔の関係は $T_{update} = T_{freshness-date}(1-F)$ となる。$T_{freshness-date} = 5.0$ の時の、保障する鮮度と更新間隔の関係を図 5.10 に示す。図 5.10 を見ると、保障鮮度を高くすると更新間隔はで短くなっている。よって、保障鮮度を高くすると更新間隔が短くなり Web サーバへの

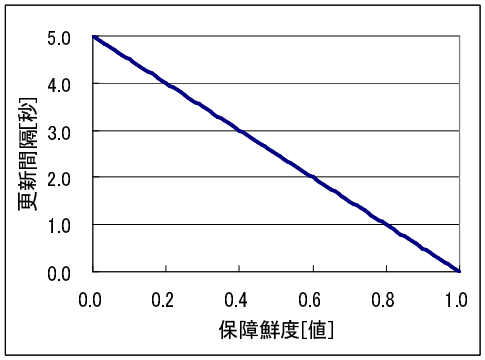


図　5.10 保障する鮮度と更新間隔の関係 ($T_{freshness-date} = 5.0$)

オーバヘッドが大きくなる。

## 5.3　機構の性能評価

提案する機構の性能評価を行う。Web サーバへのリクエストを増加させ、閾値 (高負荷状態) に達した時、ミラーリングを開始する。ミラーリング開始後、再度閾値に達した場合、以下の手法をで各サーバへのリクエスト到着数を分散させる。

1. Web サーバへのリクエスト到着数が閾値に達した場合、更新回数を少なくする (更新間隔を長くする)
2. ミラーサーバへのリクエスト到着数が閾値に達した場合、ミラーサーバ数を増やす
3. 更新回数を少なくすることができない場合、全てのリクエストを全サーバへ均等に分散する

1 は更新回数を少なくすることにより、Web サーバに対する更新のオーバヘッドを小さくする手法である。2 はミラーサーバ台数を増やすことにより、ミラーサーバへのリクエストを分散させて負荷を小さくする手法である。3 は更新回数を少なくすることができない場合に、Web サーバへ振分けられるリクエストを強制的にミラーサーバへ分散させ、負荷を小さくする手法である。3 では最新情報を取得できないクライアントが発生する。しかし、サーバダウンを防ぐことが最優先であるので 3 の方法を実行する。

また、複数のミラーサーバがコンテンツを更新し Web サーバ上のコンテンツと同期をとる時、同期する方式には以下のものがある。今回は各方式について分析し比較を行う。

- 同時更新方式
- 賞味期限を考慮しないタスキリレー方式

- 賞味期限を考慮したタスキリレー方式

同時更新方式は全てのミラーサーバ、Web サーバに対して同時に更新を行う方式である。ミラーサーバ数が増加するほど Web サーバに対して同期のオーバヘッドが生じる。タスキリレー方式は 1 台ずつタスキ上にサーバをつなげ階層的にコピーする方法である。ミラーサーバの Web サーバに対するリクエストが 1 台分だけになるので更新によるオーバヘッドが小さい。しかし、各サーバ間での更新において遅延が発生するために、全てのミラーサーバの更新が終了するまで時間がかかる。賞味期限を考慮しないタスキリレー方式は賞味期限を考慮せず最初の 1 台のみが賞味期限内に更新する。賞味期限を考慮するタスキリレー方式は全ミラーサーバの更新が終了する時間が賞味期限以内になるように更新する。

分析ではクライアントからのリクエストを増加させる。ミラーリング開始後、最新情報を要求するクライアントと古い情報を要求するクライアントが $\beta$ の割合で分散される。このときの Web サーバおよびミラーサーバへのリクエスト到着数、応答時間、コンテンツ鮮度を評価する。各分析の条件は表 5.5 である。

### 5.3.1　同時更新方式

同期更新方式で機構を実行し $\beta = 0.7$ の時、リクエスト到着数は図 5.11、応答時間と鮮度は図 5.12 のようになる。さらに、$\beta = 0.3$ 時のリクエスト到着数は図 5.13、応答時間と鮮度は図 5.14 のようになる。

表　5.5 分析条件

| パラメータ | 値 |
|---|---|
| ミラーリング開始の閾値 | 10000 |
| リクエスト数 | 0〜40000 |
| ミラーサーバ数 | 初期値 1.0 |
| 賞味期限 | 1.0 |
| Web サーバのコンテンツ処理時間 | 0.2 |
| ミラーサーバのコンテンツ処理時間 | 0.1 |
| サーバ間のコピーにかかる時間 | 0.01 |
| 更新時に生じるミラーサーバ側の負荷 | 100 |
| $\alpha$ | 1.0 |
| $\beta$ | 0.7 or 0.3 |
| 計測時間 | 1.0 |

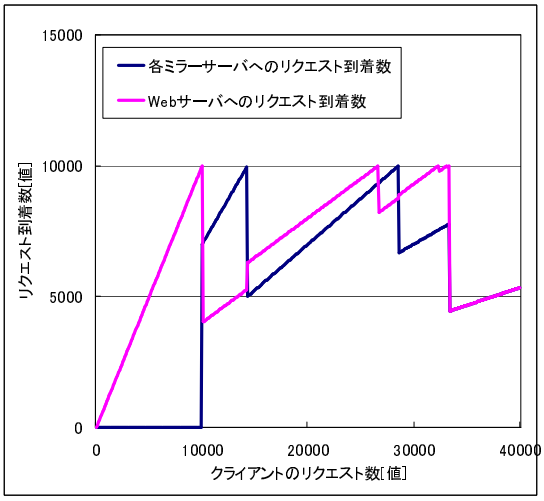


図　5.11 リクエスト到着数の変化 ($\beta = 0.7$)

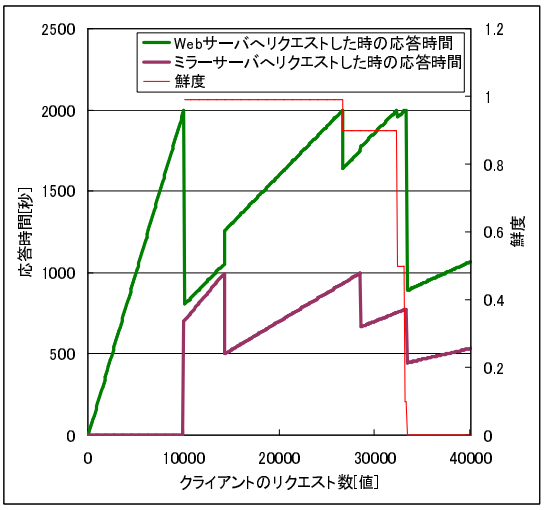


図　5.12 応答時間と鮮度の変化 ($\beta = 0.7$)

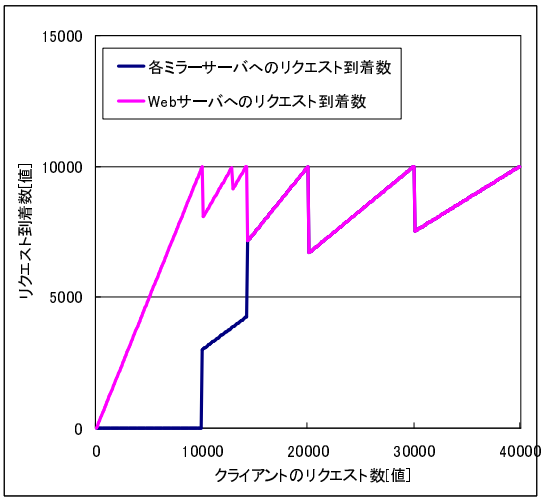


図　5.13 リクエスト到着数の変化 ($\beta = 0.3$)

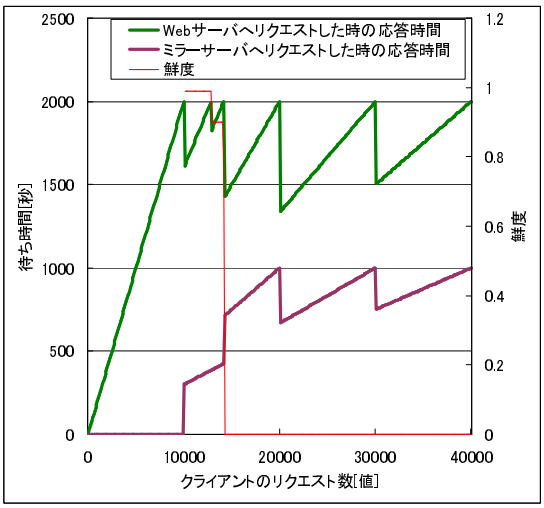


図　5.14 応答時間と鮮度の変化 ($\beta = 0.3$)

$\beta = 0.7$ の場合、クライアントの 3 割が最新情報を、7 割が古い情報をリクエストする。図 5.11、図 5.12 の結果を見ると、リクエスト数が 10000 になった時、ミラーリングが開始されている。リクエスト数が 14000、28500 になった時、ミラーサーバ数が増加している。ミラーサーバ数が増加することで、ミラーサーバへのリクエスト到着数、応答時間が減少し、Web サーバの更新のオーバヘッドが上昇している。加えて、リクエスト数が 26600、32300、33300 の時、更新間隔が長くなり、Web サーバのリクエスト到着数、応答時間が減少している。しかし、更新間隔が長くなったので、コンテンツの鮮度が減少している。リクエスト数が 33400 の時、これ以上更新間隔を長くできなくなり、強制的にクライアントのリクエストを全サーバへ分散している。

また、応答時間はミラーサーバのほうが Web サーバに比べて短い。これはミラーサーバのコ

ンテンツ処理時間が、Web サーバのコンテンツ処理時間より短いからである。ミラーサーバはコピーした HTML ファイルを返す処理時間のみだが、Web サーバは HTML ファイルを生成する処理時間も必要となるので Web サーバのほうがコンテンツ処理時間が長い。さらにミラーサーバ数を増やすことにより応答時間を短くすることができる。

以上の結果より提案機構でクライアントのリクエスト数が 33500 までは、賞味期限内の保障された動的コンテンツをミラーリングすることができている。ミラーリング開始後、7 割のクライアントが古い情報をリクエストするようになったので、7 割減少したリクエスト到着数だけミラーリングのオーバヘッドにまわすことができている。

$\beta = 0.3$ の場合、クライアントの 7 割が最新情報を、3 割が古い情報をリクエストする。図 5.13、図 5.14 の結果を見ると、リクエスト数が 10000 になった時、ミラーリングが開始されている。リクエスト数が 12800 の時、更新間隔が長くなり、Web サーバのリクエスト到着数、応答時間が減少している。更新間隔が長くなったのでコンテンツの鮮度が減少している。また、リクエスト数が 14200 付近の時、これ以上更新間隔を長くできなくなり、強制的にクライアントのリクエストを全サーバに分散している。強制的に分散させた後、リクエスト到着数が閾値になるとミラーサーバ台数を増加し、分散させるサーバ数を増やしている。

Web サーバとミラーサーバとの応答時間の差は、$\beta = 0.7$ の時と比べてミラーリング開始直後から大きくなっている。

$\beta = 0.3$ の条件では $\beta = 0.7$ の時に比べて、Web サーバのリクエスト到着数が閾値に達するのが速い。これは、$\beta = 0.7$ の時に比べて最新情報を要求するクライアントが多いからである。最新情報を要求するクライアントが多いほど、ミラーサーバへリクエストが分散されない。しかし、Web サーバとミラーサーバとのリクエスト到着数の差が大きいので、Web サーバとミラーサーバの応答時間の差が大きく、ミラーサーバは Web サーバに比べて短い時間でコンテンツを取得することができる。

### 5.3.2　賞味期限を考慮しないタスキリレー方式

同時更新方式では、ミラーサーバ数が増加すると Web サーバに対する更新のオーバヘッドが大きくなる問題がある。この問題を解決する方法にはタスキリレー方式がある。次に、賞味期限を考慮しないタスキリレー方式を用いた分析を行う。賞味期限を考慮しないタスキリレー方式で機構を実行した時の $\beta = 0.7$ 時のリクエスト到着数を図 5.15、応答時間と鮮度を図 5.16 に示す。

$\beta = 0.7$ の場合、クライアントは 3 割が最新情報を、7 割が古い情報をリクエストする。図 5.15、図 5.16 の結果を見ると、リクエスト数が 10000 になった時、ミラーリングが開始する。リクエスト数が 14200、25700 になった時、ミラーサーバ数が増加している。ミラーサーバ数が増加することで、ミラーサーバへのリクエスト到着数、応答時間が減少している。加えて、リクエスト数が 30000 の時、更新間隔が長くなり、Web サーバのリクエスト到着数、応答時間が減少している。

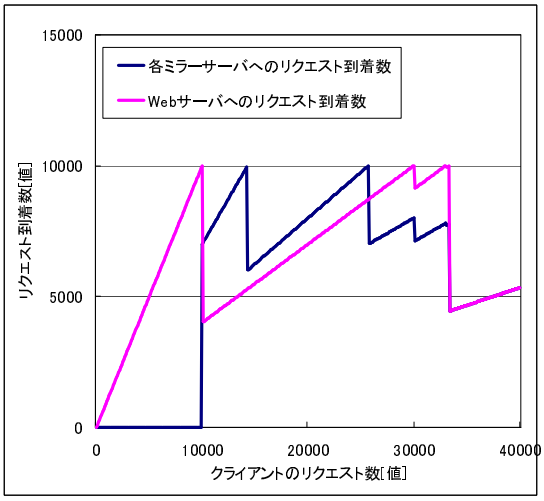


図　5.15 リクエスト到着数の変化 ($\beta = 0.7$)

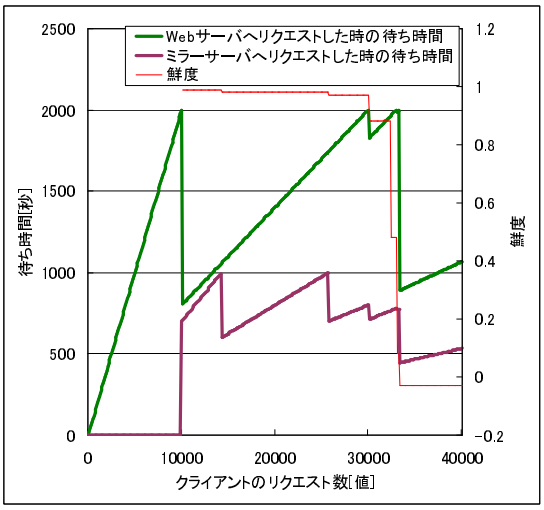


図　5.16 応答時間と鮮度の変化 ($\beta = 0.7$)

しかし、更新間隔が長くなったので、コンテンツの鮮度が減少している。リクエスト数が 34000 の時、これ以上更新間隔を長くできなくなり、強制的にクライアントのリクエストを全サーバへ分散している。しかし、強制的に分散させている時、コンテンツの鮮度が負数になっているので、保障されたコンテンツを提供できない状態になっている。

同時更新方式と比べて、コンテンツを保障したミラーリングができるリクエスト数が多い。同時更新方式がミラーサーバ数が増加したときに、増加した分だけオーバーヘッドがかかるのに対し、この方式はミラーサーバ数が増加しても更新のオーバヘッドが増加しない。よって、ミラーリングができるリクエスト数が多くなったと考えられる。また、同時更新方式と比べてオーバヘッドがかからない分、Web サーバとミラーサーバ間の応答時間差も小さくなっている。しかし、この方式は賞味期限を考慮しないので、リクエスト数が 34000 以降、鮮度が負数になってしまい、保障されないコンテンツが提供されている。これはコンテンツを更新する時に遅延が発生するのが原因である。賞味期限が切れる直前にミラーサーバが更新を行うと、全ミラーサーバが更新されるまで時間がかかり、鮮度が負数になってしまう。負数にならない対策が必要になる。

### 5.3.3　賞味期限を考慮するタスキリレー方式

前節で述べた鮮度が負数になる問題を解決する方法として賞味期限を考慮するタスキリレー方式が考えられる。これは全ミラーサーバの更新を賞味期限内に終了させるために、遅延時間分だけ早めに更新を行うものである。賞味期限を考慮するタスキリレー方式で機構を実行した時の $\beta = 0.7$ 時のリクエスト到着数を図 5.17、応答時間と鮮度を図 5.18 に示す。

図 5.17、図 5.18 の結果を見ると、リクエスト数が 10000 になった時、ミラーリングが開始する。リクエスト数が 14200、25700 になった時、サーバ数が増加している。サーバ数が増加するこ

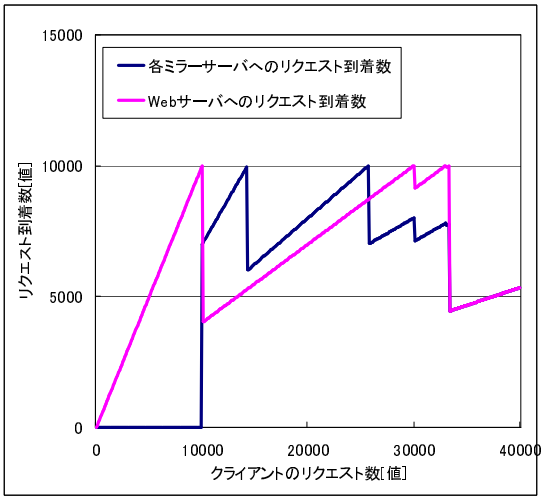


図　5.17 リクエスト到着数の変化 ($\beta = 0.7$)

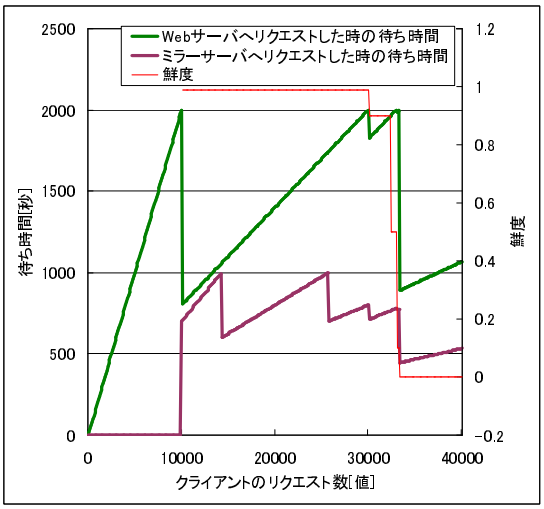


図　5.18 応答時間と鮮度の変化 ($\beta = 0.7$)

とで、ミラーサーバへのリクエスト到着数、応答時間が減少し、Web サーバの更新のオーバヘッドが上昇している。加えて、リクエスト数が 29900 の時、更新間隔が長くなり、Web サーバのリクエスト到着数、応答時間が減少している。しかし、更新間隔が長くなったので、コンテンツの鮮度が減少している。最後に、リクエスト数が 34000 の時、これ以上更新間隔を長くできなくなり、強制的にクライアントのリクエストを全サーバに分散している。また、強制的に分散させた後でも鮮度は負数になっていない。

この方式は賞味期限を考慮しないタスキリレー方式と違い、リクエスト数が 34000 以降でも、鮮度が負数になっていない。これは各サーバ間の同期による遅延を考慮して、早めにコンテンツの同期を行うからである。少し早めに同期を行うので、賞味期限を考慮しないタスキリレー方式に比べて、少し更新間隔が短くなり、更新回数が増えている。しかし、少数回であるので、ほとんど Web サーバへ影響がない。

以上の分析より、賞味期限を考慮するタスキリレー方式が、賞味期限を過ぎず、更新のオーバヘッドも小さいので理想的な方法であることがわかった。また、提案する機構がリクエスト数の増加している状態において、ミラーリング実行した場合、リクエスト到着数が分散され、Web サーバが高負荷状態に陥りにくいことがわかった。各サーバへのリクエスト到着数は閾値を超えず、ある程度のリクエスト数に限られるが賞味期限内の保証されたコンテンツを提供することができている。提案する機構はリクエスト数の増加に対して有効性がある。

# 第 6 章

# おわりに

## 6.1　まとめ

本研究では、動的コンテンツに対応した動的ミラーリング機構について提案し、提案した機構を設計し評価した。最初に動的コンテンツを分析してミラーリング可能なコンテンツとミラーリング不可能なものに分類した。そして、動的コンテンツに対応した場合の動的ミラーリング機構に必要な要件について考察し、要件を考慮して機構を設計をした。最後に、提案機構を実行したときの Web サーバおよびミラーサーバのリクエスト到着数、応答時間、コンテンツの鮮度について評価を行った。その結果、クライアントからのリクエスト数が増加しても、各サーバへのリクエスト到着数は閾値を超えず、ある程度のリクエスト数に限られるが賞味期限内の保証されたコンテンツを提供できることが分かった。

## 6.2　今後の課題

今後の課題は、Web サービス提供者が動的コンテンツをアップロードしたときのコンテンツの自動分類する方法の考察や様々な動的コンテンツの鮮度、賞味期限の特徴を分析する必要がある。また、クライアントのリクエストに対する最適な振分け方法やコンテンツの同期方法を考える必要がある。

# 謝辞

本研究において、ご多忙の中、貴重な時間を割いて丁寧な指導をして下さった最所先生に感謝いたします。ならびに、研究中、適切な助言や励ましをいただいた、研究室の後輩にも感謝いたします。また、研究面において、研究室の先輩に適切な助言や励ましをいただきました。誠に有難うございました。

# 参考文献

[1] Tony, Bourke,「サーバの負荷分散技術」, オライリー・ジャパン, 2001 年.

[2] 入江正行,「Web サーバの負荷状況に応じた動的ミラーリング機構」, 香川大学工学部, 2005 年.

[3] 河田光司,「動的ミラーリングにおける分散管理を用いたミラーサーバの割当て」, 香川大学工学部, 2006 年.

[4] アカマイ,http://www.akamai.co.jp/ .

[5] 藤田範人, 榎本敦之, 岩田淳,「CDN における選択的コンテンツの動的ミラーリング方式とその性能評価」, 電子情報通信学会技術研究報告, Vol.101, No.440(20011114),pp.33-40,IA2001-24.

[6] 金子勇,「Winny の技術」アスキー 2005 年,

[7] Spider Cache,http://www.macnica.net/warp/spidercache.html .