3P-1

Webアプリケーションにおけるプリフェッチに関する研究

柴田和祈† 高田眞吾†
†慶應義塾大学大学院理工学研究科 †慶應義塾大学理工学部

1 背景

近年，Webは多くの人に利用されており，ネットワークのトラフィックはしきりに増加している．技術の進歩により通信の帯域幅は増加したが，それでも混み合った回線によってはページのロードに時間がかかる[Ossa 2007].

Webアプリケーションの高速化の一技術としてプリフェッチが挙げられる．プリフェッチとは，あらかじめ予測されるページの先読みを行う技術であり，ユーザの待ち時間の短縮を目的としている．

2 既存研究

クライアントサイドにおけるプリフェッチの研究として，[Dahlan 2008]が挙げられる．Dahlanらはユーザの現在のアクションから次のアクションを予測する，非同期プリフェッチを考案した．しかし，クライアント側で実装するためにはブラウザに何らかのソフトウェアを組み込まなくてはならないという問題がある．

サーバサイドにおけるプリフェッチの研究として，[Fisher 2004]と[Ossa 2007]が挙げられる．Fisherらは，サーバサイドにおいてプリフェッチ対象を静的に決定する，リンクプリフェッチを提案している．リンクプリフェッチとは，Webページの先読み対象をブラウザに通知する手法である．

また，Ossaらはサーバ側において，ユーザのアクセス履歴を分析することで動的にプリフェッチ対象を決定し，クライアント側に通知する研究を行った．しかしOssaらの手法では，データの過剰取得からオーバヘッドが生じてしまうという問題が挙げられる．

また，これらの手法は全て，固定である静的なWebページにのみ適応が可能である．しかし現在，静的ページのみで構成されているWebアプリケーションは極少数である．そこで，ユーザからの入力によって変化する動的なWebページにも適応させることに意義がある．

3 提案

本研究では，動的なWebアプリケーションにおけるプリフェッチを提案する．

3.1 概要

ユーザのクリック先は予測できても，ユーザが直接入力する内容までは予測することはできない．そこで，Webページを静的なコンテンツと動的なコンテンツに分離する．静的コンテンツはリンクプリフェッチを用いて先に取得する．動的コンテンツはAjax(Asynchronous JavaScript & XML)の技術を用いて後から補完する．また，提案機構はPHPアプリケーションを対象とする．

Research on Prefetching Method for Web Applications
Kazuki SHIBATA†, Shingo TAKADA†
†Faculty of Science and Technology, Keio University

3.2 提案機構

本研究の提案機構を図1に示す．まず，PHPアプリケーションを静的コンテンツと動的コンテンツに分離する．そして静的コンテンツの部分をプリフェッチで取得し，キャッシュに保存する．実際にフォームに入力がなされ，submitボタンが押された際に，キャッシュに保存した静的コンテンツをロードする．次にその静的コンテンツ内に埋め込まれているJavaScriptが発動し，Ajax処理によって動的コンテンツを取得する．このようにして後から動的処理部分はページに挿入される形となる．

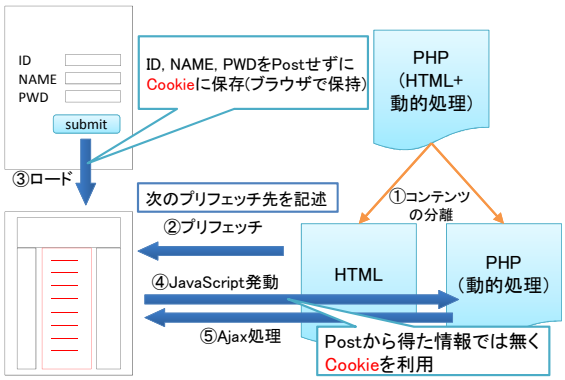

図1: システムの流れ

3.3 静的・動的コンテンツの分離

静的・動的コンテンツの分離は次のように行われる(図2)．PHPアプリケーションにおいて，動的な処理は“<?php”と“?>”に囲まれた部分で行われる．まず静的処理部分と動的処理部分を分離し，それぞれ独立のファイルとして保存する．その際，静的ファイルの方には，元々動的処理が記述されていた部分に“PHP_TMP”という文字列を挿入する．

次に静的ファイル側に，Cookieの処理や，Ajaxの処理を行うためのJavaScriptコードを挿入する．最後に，Ajax処理によって動的処理の結果を取得し，最初に置き換えておいたPHP_TMPを処理結果で置き換える．そうすることで本来得る予定だったページを静的コンテンツと動的コンテンツを別ファイルに分けた形で表示できるようになる．

3.4 データの受け渡し

図1の③，④の処理に当たる，データの受け渡しについて述べる．具体的には，現在のページからキャッシュに格納されている静的コンテンツへのデータの受け渡し(③の処理)，キャッシュページからサーバ側にある動的コンテンツへのデータの受け渡し(④の処理)について述べる．ページ遷移の種類によってデータの受け渡し方法は変わってくる．下記のような例ではidが

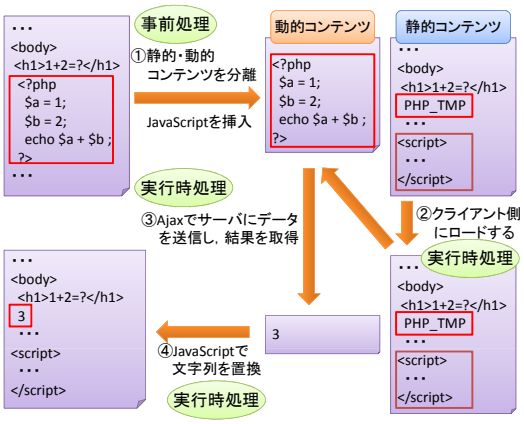

図 2: コンテンツの分離とその後の処理の流れ

a.php に POST で渡され，mode と p が b.php に GET で渡される．

```
<form method="POST" action="a.php">
  <input type="text" name="id" />
  <input type="submit" value="送信"/>
</form>
 ・・・
<a href="b.php?mode=1&p=3">リンク</a>
```

ボタンクリックによるページ遷移：

1. Onclick が発生した際に，Form 部品（input, textarea 等）の name 属性と value 属性を探索
2. DOM(Document Object Model) を用いて，本来サーバに送るべきデータを取得
3. 取得したデータを Cookie に保存（有効期限はページ遷移直後）
4. ページ遷移後，cookie から値を取得し，Ajax でサーバに送信

リンククリックによるページ遷移：

1. Onclick が発生した際に，href 属性の “?” 以降である GET 変数を分析し，送るべきデータを取得
2. 取得したデータを Cookie に保存（有効期限はページ遷移直後）
3. ページ遷移後，cookie から値を取得し，Ajax でサーバに送信

3.5 プリフェッチの手順

リンクプリフェッチという機能を利用することで，Web ページの先読み対象をブラウザに通知することが出来る．リンクプリフェッチを用いたプリフェッチの手順は次のようになる．

1. Apache のログを解析し，ユーザがどのページからどのページへ遷移しやすいのかを分析する
2. .htaccess において，ファイルごとにプリフェッチ先を指定する
3. .htaccess によって，リクエスト毎のレスポンスにリンクヘッダが付け加えられる
4. リンクプリフェッチを利用して，リンクヘッダで指定したファイルをプリフェッチする
5. Cron によって，指定した時間ごとに.htaccess を書き換えるスクリプトを動作させる

このようにすることで一定時間ごとにプリフェッチ対象をユーザアクセスに応じて切り替えることが出来る．また，以下に.htaccess の記述例を示す．

```
<Files "a.html">
Header set Link "<b.html>; rel='prefetch'"
Header set Link "<c.html>; rel='prefetch'"
</Files>
```

この場合，a.html というファイルのリクエストが来た際，そのレスポンスとして b.html，c.html の二つをプリフェッチしてほしいという情報をリンクヘッダに加えることを意味する．

4 評価結果

複数ページから成るショッピングサイトに対して本提案機構を適用した．Firebug を用いて変換前後のアプリケーションのロード時間を計測した．測定結果はそれぞれ 10 回の計測の平均を取った．評価環境として，対象アプリケーションのあるサーバと同一ネットワーク内にある PC と，異なったネットワーク内にある PC でそれぞれ計測を行った．そのうち，それぞれの PC による測定結果を表 1 に示す．

表 1: ロード時間の比較

ネットワーク	変換前	変換後		短縮率
	全体	静的	全体	
同じ	540ms	254ms	417ms	22.8%
異なる	1320ms	499ms	1060ms	19.7%

表 1 の例では，変換後のアプリケーションは変換前と比べ，全体のロード時間が削減されている．また，静的部分が早めに表示されるので，ユーザの感じる待ち時間は軽減されると考えられる．

5 まとめ

動的な Web アプリケーションにおけるプリフェッチを提案した．本提案機構は主に静的・動的コンテンツの分離，データの受け渡し，プリフェッチ処理の 3 つから成り立つ．

参考文献

[Dahlan 2008] A.A.Dahlan, T.Nishimura: “Implementation of Asynchronous Predictive Fetch to Improve the Performance of Ajax-Enabled Web Applications”, 10th International Conference on Information Integration and Web-based Applications and Services, pp.345-350 (2008).

[Fisher 2004] D.Fisher, G.Saksena: “Link Prefetching in Mozilla: A Server-Driven Approach”, 8th International Workshop on Web Content Caching and Distribution, pp.283-291 (2004).

[Ossa 2007] B.de la Ossa, J.A.Gil, J.Sahuquillo, A.Pont: “Improving Web Prefetching by Making Predictions”, 3rd EuroNGI Conference on Next Generation Internet Networks, pp.21-27 (2007).