部分評価を応用した動的 Web ページのキャッシュ機構*

竹辺 靖昭

コグニティブリサーチラボ 研究開発部
Yasuaki_Takebe@crl.co.jp

概要

部分評価の手法を応用し，動的 Web ページの生成を高速化するシステムを開発している．このシステムは，Web サイトの開発において広く使用されている PHP という言語で記述されたプログラムを部分評価し，変換結果を Web サーバに配置する．更新の頻度が低いデータのクエリなどは部分評価時に行われ，Web ページを生成する時点では動的な部分のみが実行される．これにより，さまざまな動的 Web ページを生成する負荷を低減することができる．パーソナライズ機能を持つ Web ページに対して実際にこの手法を適用し，評価を行った．

1. はじめに

現在の多くの Web サイトでは，Web サーバ上で動作するスクリプトで，DB からデータを取得し，動的に Web ページを生成する処理が行われている．この Web-DB システムのアーキテクチャの難点として，データが DB で集中的に管理されているため，DB サーバに負荷が集中してしまいやすいということが挙げられる．

この問題を解決するため，動的なコンテンツを Web サーバ側でキャッシュするためのシステム[1]が開発されているが，ページ単位のキャッシュを基本としているものが多く，リアルタイムに変更される情報を含むページやパーソナライズ機能を持つページになどには適用が難しい．また，スクリプト実行系の高速化手法[2]も，DB サーバの負荷の低減を主眼とはしていない．

今回開発中のシステムは，スクリプト言語である PHP のサブセットを対象とした部分評価器と，部分評価によって生成されたスクリプトを Web サーバに配置するシステムからなる．部分評価により，更新の頻度が低いデータへのクエリは部分評価時に行われ，リアルタイムに変更されるデータへのクエリなどのみを行うスクリプトが生成される．部分評価時に行われたクエリの結果は，生成されたスクリプト中に埋め込まれたような形になる．Web ページを生成する際には，このスクリプトが実行されるため，更新の頻度が低いデータへのクエリは行われない．これにより，キャッシュと同様に DB サーバの負荷を低減することができ，リアルタイムに変更される情報を含むページやパーソナライズ機能を持つページに対しても適用が可能である．

以下の節では，まずこのシステムの構成について説明した後，部分評価，スクリプト配信の各システムについて説明する．続いて，このシステムの適用による効果について評価を行う．

2. システムの構成

このシステムは，スクリプト配信システムと部分評価器からなる(図 1)．

スクリプト配信システムは，DB のデータの更新や一定時間の経過などのタイミングにより，部分評価器を呼び出し，部分評価の対象に設定されているスクリプトを変換する．また，変換されたスクリプトを Web サーバに配信する処理も行う．

部分評価器は，3 節で説明するアルゴリズムにより，静的なクエリを実行し，入力されたスクリプトを変換する．

---

* **A Caching System for Dynamic Web Pages Using Partial Evaluation**, Yasuaki Takebe, R&D Division, Cognitive Research Laboratories, Inc., Tokyo.

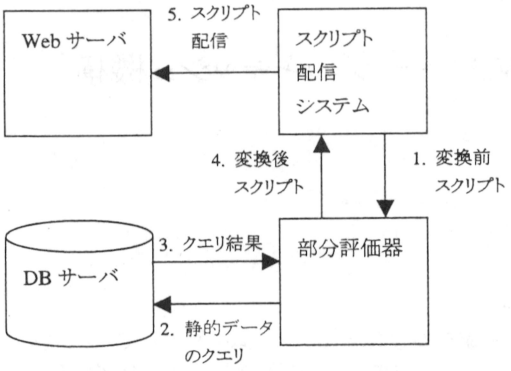

図 1. システムの構成

3. 部分評価器の概要

この節では，部分評価器について説明する．

3.1. PHP の言語仕様

PHP は，動的な Web ページの開発専用に設計されたスクリプト言語である．HTML に埋め込んで記述するようになっているため，Web ページのデザインが行いやすいなどの利点がある．また，PHP は，各種の DB と接続する API を備えており，Web-DB システムの開発が容易になっている．

言語としては以下のような特徴を持つ:

- C 言語に似た制御構造(goto 文はない)
- オブジェクト指向
- スクリプト言語的な特徴
- HTTP との連携

はじめの 2 つについては，既存の言語と似ているため，説明を割愛する．詳細については[3]を参照してほしい．

スクリプト言語的な特徴の例として，可変変数(variable variable)を挙げる．PHP では変数名には必ず'$'をつけることになっているが，$に続けて変数名の代わりに式を書くことにより，その式の評価結果を名前に持つ変数を参照することができる:

```
$b = "a";
$$b = 5;  // $a = 5 と同じ
```

HTTP との連携というのは，Cookie や HTML のフォームによってブラウザから渡されたデータを，自動的に PHP の変数の初期値とするというものである．例えば，ブラウザから，

```
http://.../foo.php?n=5
```

という要求が来た場合には，foo.php に記述されているスクリプトにおいて，変数$n の初期値は 5 になる．このため，ある変数の初期値が何かということを PHP のプログラムを見ただけで判断することはできない．

3.2. 部分評価の概要

この部分評価器が対象としている言語仕様は，PHP のうち，以下のものをのぞいたサブセットになっている:

- オブジェクト指向
- 参照
- 可変変数，可変関数(variable function)

部分評価の手法としては，はじめに束縛時解析を行い，その結果を用いて部分評価を行う手法を採用している．部分評価では，部分評価時に計算できることを静的，実行時まで決まらないことを動的という．束縛時解析とは，プログラム中の変数が動的か静的かを判定する処理のことをいう．

手続き型言語の束縛時解析の手法にはさまざまなものが提案されているが[4][5]，この部分評価器で採用しているものは非常に単純なものである．現状では以下のように束縛時解析を行っている:

- 変数はプログラム中のどこでも同じ束縛時(動的または静的)をもつ
- 動的な制御構造の中で代入が行われた変数は動的と解析する

以下に例を挙げる:

```
$i = 0;
if (dyn) {  // 動的な条件式
  $i = $i + 1;
}
```

条件式が動的なため，$i はプログラム全体で動的と解析される．

また，HTTPによって外部から渡される変数は動的と解析する必要があるが，前節で説明したように，これをプログラムの解析によって判断することはできない．そのため，以下のような構文を追加し，変数が動的であることを明示的に宣言できるようにしている:

```
pm_dynamic $var1, $var2, ..., $varn;
```

3.3. DB へのアクセス方法

今回開発しているシステムでは，効果的に部分評価を行うため，PHP にある DB アクセス用の API を変更している．以下でこれについて説明する．

PHP では，以下のような機能を持つ API を組み合わせて DB へのアクセスを行う:

- DB への接続
  DB へ接続し，接続ハンドルを取得する．
- クエリの実行
  接続ハンドルを指定し，クエリを実行する．
  結果セットを取得する．
- 行の取得
  結果セットから行を取得する．

通常，1 つのページ内においては，DB への接続は一度だけ行い，得られた接続ハンドルを使いまわす．以下の例は，MySQL という DBMS にアクセスするプログラムである:

```
$handle = mysql_connect(...);

$result1 = mysql_query(SQL文, $handle);
$row1 = mysql_fetch_array($result1);
$result2 = mysql_query(SQL文, $handle);
$row2 = mysql_fetch_array($result2);
```

しかし，3.2 で説明した手法でこのプログラムを部分評価しようとしても，望ましい結果が得られない．$handle は，DB への接続ハンドルという値をとる．この値はリテラルで表現できないため，部分評価結果に埋め込むことができない．このため，SQL 文が動的な場合には，$handle も動的と解析しなければいけなくなる．$handle はプログラム全体で使いまわされているため，ページ内のクエリ全体が動的になってしまう．

これを避けるため，このシステムでは，DB アクセスを行う API を以下のように変更している:

- DB への接続
  接続する DB の情報を定義する．
- 動的なクエリの実行
  実行時にクエリを実行する．
- 静的なクエリの実行
  与えられた SQL 文が静的であれば，部分評価時にクエリを実行する．
- 行の取得
  結果セットから行を取得する．

プログラムの例は以下のようになる:

```
sql_connect(...);

$result1 = sql_query(SQL文);
$row1 = sql_fetch_array($result1);
$result2 = sql_query_static(SQL文);
$row2 = sql_fetch_array($result2);
```

sql_query は動的にクエリを行う関数であるため，$result1 は動的と解析される．sql_query_static に与えられたパラメタが静的であれば$result2 は静的と解析される．sql_connect は，定数の定義と同等のものと解析されるため，部分評価時にも実行時にも評価が行われる．

これにより，実行時に行うべきクエリを残しつつ，静的なクエリを部分評価時に行うことができる．

3.4. 動的な変数の束縛

動的な Web ページの有効な応用例として，パーソナライズ機能の実現がある．しかし，これまで説明してきた部分評価の手法では，パーソナライズ機能を持つ Web ページを効果的に部分評価することは難しい場合が多い．例えば，Web サイトの訪問者をグループに分類しておき，グループごとに内容を変更するような Web ページを表示するた

めの処理の概要は以下のようになると考えられる:

```
$group = 個人情報からグループを取得;
$query = $group を使用した SQL 文;
$result = sql_query($query);
...
```

個人情報は訪問者によって異なるため，部分評価時に決めることはできない．したがって動的となる．個人情報に依存する$group，$query，$result も全て動的となり，部分評価が全く行えない．

今回開発しているシステムでは，動的であっても，いくつかの値をとることがわかっている変数にアノーテーションを行う構文を追加し，こうした場合にも部分評価が行えるようにしている．アノーテーションを行う構文は以下のような形式をもつ:

```
pm_bind ($var : val_1, val_2, ..., val_n) {
    stmt
}
```

この構文により，*stmt* において$*var* が val_1〜val_n のどれかの値をとることを部分評価器に示す．以下，この構文を bind 文という．bind 文が現れると，部分評価器は以下の条件を満たす変数について *stmt* 内だけで再度解析を行い，可能ならば静的と解析する:

- bind 文の先頭で死んでおり，かつリテラルで表現可能な値をとる変数
- bind 文の先頭および bind 文から脱出するブロックで死んでいる変数

部分評価器は，この束縛時解析の結果を使い，bind 文を以下のような形に部分評価する．

```
if ($var == val_1) {
    stmt'[$var/val_1]
} else if ($var == val_2) {
    stmt'[$var/val_2]
} else if (...) {
    ...
} else {
    stmt'
}
```

ここで，*stmt'*[$*var*/$val_i$]は，$*var* が val_i をとるとして *stmt* を部分評価した結果であり，*stmt'* は *stmt* をそのまま部分評価した結果である．

リテラルで表現可能な変数で bind 文から脱出するブロックで死んでいない変数が bind 文内では静的と解析された場合，脱出時点でとっている値をその変数に代入する文を変換結果に挿入する．例えば，以下の bind 文

```
pm_bind ($i : 1) {
    $i = $i + 1;
    echo $i;
}
echo $i;
```

これは次のように変換される:

```
if ($i == 1) {
    echo 2;
    $i = 2;  // 脱出前に値を代入
} else {
    $i = $i + 1;
    echo $i;
}
echo $i;
```

4. スクリプト配信システム

2001/11/30 現在，スクリプト配信システムの実装はまだ完了していない．最初の実装では，スクリプト配信システムは，以下の機能を実現する予定である:

- 各 Web ページに設定された一定時間ごとに部分評価器を呼び出す
- 結果を Web サーバに配信する

将来的には，データが更新されたタイミングで部分評価器を呼び出す機能なども実装する予定である．

5. 評価

今回開発しているシステムを適用することにより，動的な Web ページを生成する負荷がどの程度低減されるかを実験するため，例題として図 2 のようなデザインの Web ページを生成するスクリプトを作成し，部分評価前と後での比較を行った．この Web ページは，ニュースを提供するサイトのトップページにパーソナライズ機能を持たせたものという想定である．認証を行うと，その訪問者が選択したジャンルのニュースの一覧を見ることができる．

この Web ページでは，パーソナライズ機能を実現するため，Cookie の情報をもとにアクセスしている人を識別し，DB に登録されたユーザ情報から興味あるジャンルを取得し，各ジャンルのニュースの一覧を DB から取得するという処理を行っている．スクリプトの行数は 218 行になった．

5.1. 部分評価器の能力

Web ページ生成の処理のうち，各ジャンルのニュースの取得は部分評価時に行うことができるはずである．しかし，今回実装した部分評価器では，普通にスクリプトを書いた場合には，このようには部分評価されないことが多いと思われる．

うまく部分評価するために気をつけなければいけない点としては以下のようなものがある:

- 一時変数を同じ名前にしない
- 動的な制御文の中ではできるだけ変数への代入を行わない

2 点目の注意点のため，例えば以下の仮想的なコードような処理はうまく部分評価できない:

```
if (認証が行われていたら) {
    個人情報を取得;
    $personal = パーソナライズされたニュース;
    $personal の情報を表示;
} else {
    $default = デフォルトのニュース;
    $default の情報を表示;
}
```

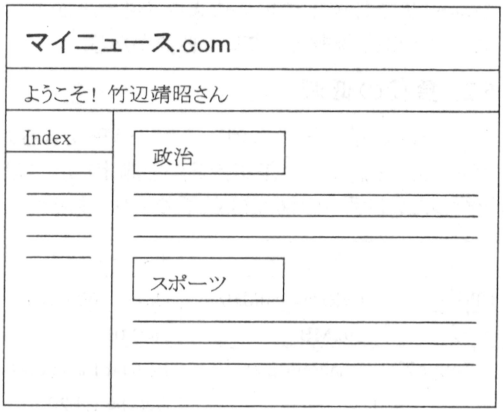

図 2. Web ページの例

認証が行われているかどうかはブラウザからアクセスが行われている時点でしか判断できないため，この if 文は動的になる．デフォルトのニュースは一種類だけなので，直感的には静的に評価が行われてもよさそうであるが，動的な if 文の中で代入が行われているため，$default のような変数は動的と判定されてしまう．

結果的には，上記の点に注意し，ユーザが選択しているジャンルを bind 文によって制限することによって，目標のとおりに部分評価を行うことができる．仮想的なコードは以下のようになる:

```
if (認証が行われていたら) {
    個人情報を取得;
    パーソナライズされたジャンル一覧を取得;
} else {
    デフォルトのジャンル一覧を設定;
}

for ($i in ジャンル一覧) {
    $genre = ジャンル一覧[$i];
    pm_bind ($genre : 政治, ...) {
        そのジャンルのニュースを検索し表示;
    }
}
```

ただし，今回の実験では，部分評価器の実装を行った本人がスクリプトを記述しているため，一般

のユーザでも満足のいく結果を得られるかどうかについては今後評価していく必要がある.

5.2. 負荷の低減

部分評価前と後のスクリプトで, 処理できるリクエスト数にどれぐらい差が出るかを測定した. 実験を行ったシステムの構成は以下の通りである:

	クライアント	サーバ
CPU	Celeron/450MHz	Celeron/600MHz
メモリ	192MB	384MB
ソフトウェア	Windows 2000	RedHat Linux 6.2J Apache 1.3.20 MySQL 3.23.44 PHP 4.0.6 APC 1.1.0pl1

クライアントから, マイクロソフトの Web Application Stress[6]というツールを使い, 10 スレッドで間隔を空けずに 1 分間の連続アクセスを行った. 部分評価前と後の処理リクエスト数および平均レスポンス時間は以下のようになった:

	部分評価前	部分評価後
リクエスト数	1860	5390
レスポンス時間	314m 秒	110m 秒

ともに約 3 倍弱の性能を達成できていることがわかる.

なお, 実験を行った時点ではスクリプト配信システムが実装できていないため, 部分評価器での変換および Web サーバへの配置は手動で行った. 実際の運用においては, 適当なタイミングで部分評価をやり直す必要があるが, この実験では部分評価を行う負荷は考慮されていない.

6. まとめ

部分評価の手法を応用し, 動的な Web ページを生成する負荷を減らすシステムを開発した. 実験により, パーソナライズ機能を持つ動的 Web ページに対しても一定の効果を確認することができた. ただし, 部分評価器に関しては改良すべき点が多い. [5]などの手法を取り入れ, さまざまなプログラムを効果的に部分評価できるようにしていくとともに, 現時点ではサポートしていないオブジェクト指向などの PHP の言語仕様についても考慮していく必要があるだろう.

ポータルサイトでのパーソナライズ機能に関しては, キャッシュ機構を備えた専用のシステムがあるが[1][7], それらを用いて Web サイトの開発を行うためには, 独自のフレームワークに従わなければいけない.

このシステムを用いることにより, 従来からの PHP での開発のノウハウを生かしながら, パーソナライゼーション機能を含むさまざまなサービスを提供できるようになることが期待できる.

謝辞 このシステムの開発は, IPA の平成 13 年度未踏ソフトウエア創造事業のプロジェクトとして行っています. 湯淺 PM をはじめとする未踏関係者の方々には, さまざまな機会において有益な助言をいただきました. また, 富士通株式会社の上田健之氏には, 商用の言語処理系の開発について助言をいただきました. ここに感謝します.

参考文献

[1] Oracle9iAS Web Cache, http://www.oracle.co.jp/9i/9ias/ http://www.oracle.co.jp/9i/9ias/cache/

[2] APC, http://apc.communityconnect.com/

[3] Stig Sæther Bakken, et al. PHP Manual, http://jp.php.net/manual/en/

[4] N. D. Jones, C. K. Gomard, and P. Sestoft. *Partial Evaluation and Automatic Program Generation*. Prentice Hall, 1993.

[5] Luke Hornof, Jacques Noyé. *Accurate Binding-Time Analysis for Imperative Languages: Flow, Context, and Return Sensitivity*. In PEPM '97, pp. 63-73, 1997.

[6] Microsoft Web Application Stress Tool, http://webtool.rte.microsoft.com/

[7] Jetspeed, http://jakarta.apache.org/jetspeed/