10-DA01

言語間差異を活用したWeb情報資源へのアクセスシステムに関する研究

増田 英孝（東京電機大学）

Wikipedia は Web 上のフリー百科事典である。全ての言語版の記事数を合計すると1000万件を超える。またWikipediaは世界のWebサイトのアクセス数ランキングで上位に位置しており、非常の多くの人々に閲覧されている。

このようにWikipediaは膨大な情報を持ち、利用者数も多いが、データの使われ方のほとんどはWebブラウザを介して記事を人が読むことである。Wikipediaデータを解析し、有用な情報を取り出す研究もされているが、Wikipediaデータを解析する手法やプログラムは各々の研究者が独自に開発しており、それ自体が共用されることは少ない。そのためWikipediaデータは貴重な情報を含んでいるが、再利用性が低い。そこで本研究ではWikipediaデータを解析する再利用可能なツールを開発し、Wikipediaから有用な情報を実際に抽出する方法を例示する。

1. 研究の目的と意義

WikipediaはWeb上のフリー百科事典である。日本語版以外にも世界中の多数の言語版が運営されており、2010年12月現在250を超える言語版が存在する。日本語版Wikipediaは約72万件、英語版は約 350 万件の記事を持っている。全ての言語版の記事数を合計すると1000万件を超える。これらの記事は全て一般のユーザによって作成・編集されている。作成する記事の題材は基本的にユーザの自由に任せされているので、通常の辞典には掲載されない話題に関する記事も Wikipedia には多数掲載されている。

Wikipedia データの使われ方のほとんどは、WikipediaのサイトへWebブラウザでアクセスして人が記事を読むことである。しかし Wikipedia データは情報資源として有用そうな性質がいくつかある。

まず機械処理可能な形式でデータが公開されていること。また全ての記事は MediaWiki という共通のWikiシステムによって処理されることを前提に書かれているので文法が統一されている。さらに百科事典サイトを作るという目標の元に記事が書かれているので、文章の文体もある程度は統一されており、俗語や絵文字のような処理しにくい表現が極力排除されている。またページを分類するためのメタデータの付与などもサポートされており、通常の Web ページよりも機械処理可能な情報資源として扱いやすい。

このような性質に着目し、Wikipediaデータを解析して役立つ情報を抽出する研究が行われている。Auerらの研究DBPedia[1]はWikipediaデータからセマンティック Web で利用可能なデータセットを生成している。これにより Wikipedia という 1 サイト内でのみ完結していたデータをセマンティック Web というより広い領域へ接続することを目指している。坂井らの研究[2]はWikipediaのページの分類構造を、図書館の図書の分類構造と組み合わせることで相互に補完されたオントロジーを構築し、情報検索の利便性の向上を狙っている。

Wikipedia を利用した研究が行われている一方で、研究への新しい参入を妨げる部分がある。その最大のものは、Wikipediaのページデータをプログラム上で操作することである。Wikipediaのページデータは MediaWiki で解析可能な形式のタグが付けられたWikiテキストである。Wikiテキストは HTML へ変換することを第一義として設計されており、プレーンテキストよりは組織化されているが、XML やデータベースほど厳密ではない。その結果、仕様に沿った汎用的な解析ツールが開発されず、Wikipedia の研究者は独自のツールを自前で作成して使うことになっている。これは時間と労力を要する作業であり、研究への参入を難しくしてい

る。

この問題を解決するには3つの方法が考えられる。

① ページ作成の段階で構造化のためのデータを埋め込む

② 汎用的なWikiテキスト解析ツールを開発する

③ Wikiテキストを構造化データへ変換する

①のアプローチは MediaWiki に構造化データのための新しい文法を導入し、Wikipediaの執筆者がそれに従って記事を編集することで可能となる。そのような試みとしてSemantic MediaWiki[3]という MediaWiki のエクステンションが開発されている。しかしWikiテキストは容易にWebページを作成することを支援するためのものであり、研究のために新しい文法を導入することに対して、Wikipedia運営者・執筆者・研究者の間で合意が得られるかどうかは不透明であり、現在このエクステンションは公式な Wikipedia には導入されていない。

本論文では②と③のアプローチのもとに開発した成果物を報告し、長短を考察する。またそれらのアプローチで実際にどのようなデータが取得できるようになるのかを例示する。

参考文献

[1] Soren Auer, Chris Bizer, Jens Lehmann, Georgi Kobilarov, Richard Cyganiak, and Zachary Ives. DBpedia: A nucleus for a web of open data. Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference, pp. 715-728, 2007.

[2] 坂井 哲, 増田 英孝, 清田 陽司, 中川 裕志. Wikipedia と図書館情報資源による調べ方自動提示システム. 情報処理学会 第 72 回全国大会 講演論文集 (3K-5), March, 2010.

[3] Semantic mediawiki. http://www.semantic-mediawiki.org.

2. 当拠点公募型共同研究として実施した意義

(1) 共同研究を実施した大学名

東京大学

(2) 共同研究分野

超大規模データ処理系応用分野

(3) 当公募型共同研究ならではという事項など

本研究では Wikipedia の大規模なデータを高速で処理するために分散処理プラットフォーム Hadoop[4]を使用している。Hadoop は複数台のマシンからなるクラスタを構築して分散処理を実現する。本研究は Hadoop クラスタの実環境として、東京大学情報基盤センターが保有するサーバマシンを使用した。その結果、日本語版 Wikipedia のデータを解析するのに、マスタ 1 台、スレイブ 3 台の構成で、多くの場合数分から十数分で完了することが分かった。

参考文献

[4] Apache Software Foundation. Apache Hadoop. http://hadoop.apache.org.

3. 研究成果の詳細

研究の内容と成果を説明する。また前提となるWikipediaやMediaWikiの知識について説明する。

3.1. Wikipediaのデータの現状

WikipediaはMediaWikiというWiki管理システム上で管理されている。Wikipedia には Web サイトであることと、MediaWiki で運営されていることによる特徴や制約がある。

3.1.1. Wikipediaのページ構造

MediaWiki 上のサイトはページを作成・編集することで成立している。全てのページはIDとタイトルを持つ。MediaWiki 上のサイトに対して、以下のようなタイトルを含むURLから個別のページにアクセスすることができる。

http://ja.wikipedia.org/wiki/タイトル

タイトルはURLに組み込まれる仕様上、一意に識別することが可能であり、サイト内でユニークである。タイトルは以下のように名前空間文字列とページ名で構成されている。

```
名前空間:ページ名
```

ページには役割によっていくつかの種別があり、タイトルの名前空間によって識別される。名前空間を示す文字列は言語版ごとに設定可能であり、以降特に断らない場合は日本語版について説明する。

記事

記事とは Wikipedia の通常のページである。1件の記事に1つの主題を持ち、その事柄を説明する文章が書かれている。記事は他のページと異なり名前空間文字列がなく、ページ名のみでタイトルが構成される。

カテゴリ

カテゴリはページを分類するためのページである。名前空間文字列は Category である。ページ中に以下のようにタグ付けをしたテキストを記述することで、そのページをカテゴリに含めることができる。

```
[[Category:カテゴリ名]]
```

カテゴリは1ページに複数設定することができる。またカテゴリにカテゴリを設定することも可能である。このため Wikipedia のカテゴリ構造は入り組んだグラフ構造となっている。

テンプレート

テンプレートはプログラミング言語におけるメソッドやサブルーチンのような機能を実現するページである。名前空間文字列は Template である。

テンプレートは次のように他のページから呼び出され、文字列を返す。

```
{{テンプレート名
|引数名 1 = 引数 1
|引数名 2 = 引数 2
        …
|引数名 n = 引数 n
}}
```

MediaWiki は呼び出し元のテンプレートの記述を、テンプレートが返した文字列へ置換する。テンプレートは任意の個数の引数を取ることができる。また四則演算や条件分岐、ループなど簡単なプログラミング言語のような機能を備えている。

Wikipedia ではテンプレートはインフォボックスと呼ばれる表を生成するために使われているのが目立つ。しかしテンプレートはどのような文章でも生成することができ、表の生成は用途のひとつに過ぎない。

3.1.2. Wikipedia の配布データ

Wikipedia ではデータをまとめて配布しており、ランセンス下でのダウンロードと使用が許可されている。

配布データは SQL ダンプ形式と、XML 形式の 2 種類がある。配布データは数十種類あるが、大半は SQL ダンプ形式である。これは MediaWiki が使用しているデータベースのテーブルをエクスポートしたもので数 KB から数十 MB の比較的小さなサイズである。内容はカテゴリやリンクなどのページ間の関係である。

XML 形式のデータは各ページの情報や内容を収録したものである。日本語版の場合は jawiki-latest-pages-meta-current.xml というファイルが配布されており、この中に全ページの内容が収録されている。

XML ファイルはリスト 1 のように、siteinfo 部分と page 部分に分かれている。siteinfo 部分にはサイト情報や名前空間文字列の定義などが書かれている。ページ部分は個別のページの情報が書かれている。

```
<mediawiki>
  <siteinfo>
サイト情報、名前空間定義
  </siteinfo>
  <page>1ページ目のデータ </page>
  <page>2ページ目のデータ</page>
      ……
  <page>nページ目のデータ</page>
</mediawiki>
```

リスト1. XMLデータファイルの構造

3.1.3. XMLファイルの利用法

Wikipedia で配布されている XML ファイルは MediaWiki のインポート機能で取り込み、データベースに復元することができる。しかしこのインポート機能は低速であり、日本語版の XML ファイルをインポートするのに数日かかる。これを解決するためにxml2sqlやMWDumperといった専用の変換ツールでSQLダンプへ変換しMediaWikiのデータベースへ登録する方法が公式で説明されている。データベースに格納されるデータは生の Wiki テキストであり、それを解析するには別のプログラムが必要である。MediaWikiはWikiテキストを解析するためのパーサを含んでいる。通常パーサの解析結果は最終的な出力となる HTML 文書として取得できる。しかしMediaWikiはPHPで実装されているため、パーサを使用するプログラムも PHP に制限される。またHTMLへ変換されたデータは元の Wiki テキストが含んでいたデータが失われている。例えば呼び出し元のテンプレートの記述は戻り値に置換されるため、HTML文書では元のWikiテキストにどのテンプレートが記述されていたかの痕跡が失われることがある。

このようにMediaWikiはWikiテキストをWebブラウザを通して編集し、HTML文書として表示することに特化しており、蓄積したデータを第三者が再利用する仕組みに乏しい。そこで本研究では Wikipedia データの再利用性を高めるために 2 つのアプローチを提案する。ひとつは XML 形式のデータファイルから、特定の情報を抽出するツールを開発することである。もうひとつは MediaWiki のコードを流用することで、パーサの出力結果のオブジェクトをHTML文書ではなく、オブジェクトの内容を永続化したような形式の XML 文書へ変換することである。

3.2. HadoopによるWikipediaデータの解析

Wikipedia データの再利用性を高めるひとつ目のアプローチは、XML形式のデータファイルから、特定の情報を抽出するプログラムを開発する事である。本研究ではプログラムの実装に、Apacheが開発するオープンソースの分散コンピューティングプラットフォームのHadoopを使用する。

3.2.1. Hadoopによる解析の利点

XML ファイルからページ間のリンク情報を抽出するプログラムがあれば、MediaWiki やデータベースを介さずに直接目当ての情報が抽出できるため、時間のかかる前処理がいらず、操作も簡単である。

そのようなプログラムの実装に Hadoop を使用する利点はいくつかある。Hadoopは分散処理環境を構成するマシンの台数を自由に設定し、必要に応じて処理能力を拡張することができる。数百台からなる分散環境を構築し、数TB規模のデータを処理することも可能である。2011年1月現在、日本語版Wikipediaのデータを記録したXMLファイルのサイズは約7GBほどであり、数台のマシンからなるHadoop環境で十分に処理できる。また今後 Wikipedia の記事がより増加したとしても、マシンの台数を増やすことで対応できる。

Hadoop 自身は Java 言語で実装されていて、ユーザによるプログラミングも基本的は Java で行う。しかしHadoop Streamingという、他のプログラミング言語を Hadoop 環境で利用する仕組みも提供されている。これはHadoopと他のプログラムの間で標準入出力によってデータのやり取りを行うもので、標準入出力をサポートするプログラムであれば使用できる。

まとめるとHadoopでWikipediaデータを解析す

る利点は以下のようになる。

●解析プログラムがあれば前処理なしで即座に使える

●Java 以外の言語も使用可能

●膨大な Wikipedia のデータ量に対応可能

3.2.2. Hadoop プログラムの実装

Hadoop は MapReduce と呼ばれる分散処理のアルゴリズムを採用している。図 1 は MapReduce の処理の流れを図示したものである。map と reduce という 2 段階の工程があり、それぞれ入力として key と value のデータの組を受け取り、出力として新しい key と value を出力する。

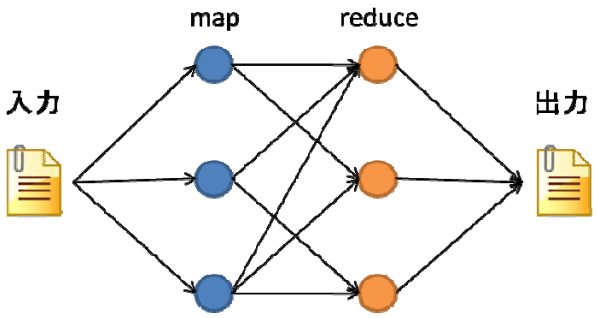

図 1. MapReduce の処理の流れ

リスト 1 に示したように、Wikipedia の XML ファイル内では 1 ページ分のデータが 1 個の page 要素として表現されている。Wikipedia データを Hadoop 上で処理するには、map への入力を XML ファイルから切り出した page 要素の文字列とする。処理対象のファイルをどのように切り出して map へ与えるかは、InputFormat というクラスで定義する。Hadoop には TextInputFormat というテキストファイルを 1 行ずつ切り出して map へ与えるクラスが用意されている。XML ファイルから page 要素を切り出すには TextInputFormat を継承し、page 要素の開始タグから終了タグまでを 1 行ずつ取り出し記録しておいてから、まとめて map へ与える。page 要素の構造はリスト 2 のようになっている。リビジョンとはページの版のことであり、XML ファイルには各ページの最新リビジョンの情報が記述されている。

```
<page>
 <title>タイトル</title>
 <id>ページ ID</id>
 <revision>
  <id>リビジョン ID</id>
  <timestamp>編集日時</timestamp>
  <contributor>
   <username>編集者名</username>
   <id>編集者 ID</id>
  </contributor>
  <comment>コメント</comment>
  <text xml:space="preserve">
     記事本文
  </text>
 </revision>
</page>
```

リスト 2. page 要素の構造

page 要素そのものは DOM や SAX などの既存の XML 解析 API で解析すればよいが、text 要素の中には生の Wiki テキストが記述されている。MediaWiki は PHP で実装されており、他の言語によるパーサは提供されていない。そこで正規表現などで Wiki テキストを解析して必要な情報だけを抽出する。例えば MediaWiki ではページに対して言語間リンクという特別なリンクを設定できる。これは他の言語版に存在する同じ事柄について書かれたページへのリンクである。例えば日本語版の「ウサギ」の記事には次のような言語間リンクが設定されている。

```
[[en:Rabbit]]
```

タイトルの前にコロンで区切って書かれているのは言語プレフィックスという文字列で、言語名を示す。この場合の en は英語版を示すプレフィックスである。言語間リンクを収集するには正規表現で 2 重のブラケットを取得し、言語プレフィックスでリンクが始まっているかどうかを調べればよい。どのような言語プレフィックスがあるかは「全言語版の統計」のページで調べられる。この

ように収集すると、「ウサギ」には次のような言語間リンクがあることが分かる。

[[de:Kaninchen]]	ドイツ語
[[fr:Lapin]]	フランス語
[[ru:К р о л и к и]]	ロシア語

いれらは、いずれも各国の言葉で「ウサギ」を示す言葉である。そのため言語間リンクを翻訳辞書のように使うこともできる。

3.3. MediaWikiを利用したWikipediaデータの構造化

Hadoopによる Wikipedia 解析の手法が、XML ファイルから特定の情報を抽出することに特化したプログラムを書いたのに対して、もうひとつのアプローチはページ全体を XML 文書という構造的なデータに変換することである。

3.3.1. Wikipediaデータの構造化の利点

Wikiテキストの文法は基本的には平易だが、テンプレートやテーブルの記述などで複雑な書式になる場合がある。そのようなWikiテキストに対して、常に正確にマッチする正規表現を書くことは労力がかかる行為である。ページを XML 文書化することで、DOM や SAX などの既存の XML 解析 API を通した操作のみで Wikipedia データにアクセス可能となる。

また生のWikiテキストでは、カテゴリや言語間リンクの設定のような記事内容そのものではなくメタデータとしての情報も本文と混在して記述されている。XML 文書化する際にメタデータを本文と分離すれば、それを解析するプログラムはより単純かつ明快になる。

さらに XML 解析 API が用意されていればプログラミング言語の選択も自由となる。

3.2.3. Wikipediaデータの構造化の実装

MediaWikiは標準では図2のような流れでWikiテキストをHTML文書化している。

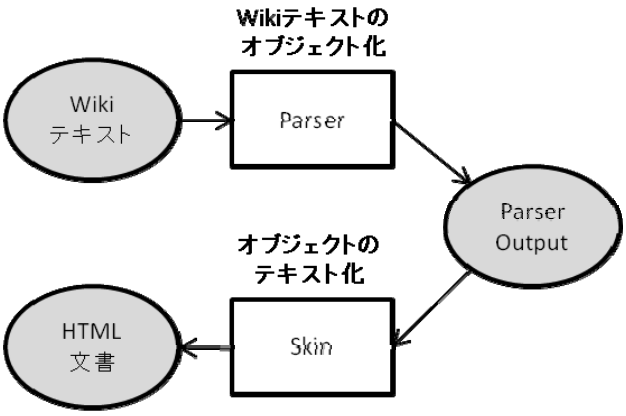

図2. MediaWikiによる処理の流れ

Parser は MediaWiki を構成するクラスであり、生の Wiki テキストを解析し、ParserOutput というクラスのオブジェクトへ変換する機能を持っている。ParserOutput は HTML 文書へ変換された本文のほか、言語間リンクなどのメタデータを保持する。Skin によって ParserOutput を最終的な出力結果へ加工する。2011 年 1 月現在、WikipediaではVectorという名前のSkinが適用されている。つまり Wikipedia を Web ブラウザで閲覧した際に表示される HTML 文書を Vector が作っている。ちなみに Parser が出力する HTML は XHTML になっており、それを XML 文書に組み込んでも不正な XML 文書になることはない。

Wikiテキストを XML 文書化するには、Parser を一連の流れから切り離し、自作のプログラムから呼び出せるようにする。リスト3のプログラムをMediaWiki のホームディレクトリに置くことで、そのプログラム内でParserを使える。

```
<?php
require_once
('maintenance/commandLine.inc');
require_once
('includes/parser/Parser.php');
$parser = new Parser();
?>
```

リスト3. Parserを呼び出すプログラム

maintenance とは MediaWiki のメンテナンス用

のプログラムが格納されているディレクトリである。commandLine.inc には、通常は Web 上から呼び出されて使われる MediaWiki を、コマンドラインから呼び出すための前処理が書かれている。

Parser クラスのオブジェクトである parser に対して、Wiki テキストとタイトル、オプションを引数として与えると、リスト 4 のように ParserOutput オブジェクトが得られる。

```
$parser = new Parser();
$options = new ParserOptions();
$title = Title::newFromText('Hoge');
$output = $parser->parse($text, $title, $options);
```

リスト 4. Parser による Wiki テキスト解析

ParserOutput は本文の他に言語間リンクなどのメタデータに対するアクセサメソッドがある。例えばカテゴリは getCategories()、言語間リンクは getExternalLinks()で取得できる。これらのメソッドを利用して ParserOutput オブジェクトからデータを取り出し、XML 文書を生成する。リスト 5 のプログラムによって、リスト 6 のような XML 文書が取得できる。

なお en、de のような言語プレフィックスは MediaWiki が使用するデータベースに格納されているものが認識される。またテンプレートは呼び出し元ではテンプレート名と引数しか書かないため、呼び出し元の Wiki テキストだけでは Parser は期待通りに動作しない。それらはいずれも予め MediaWiki が使用するデータベースに格納しておく必要がある。言語プレフィックスのテーブル内容は SQL ダンプとして公開されているが、テンプレートは 3.1.3 節で説明したように、変換ツールで SQL ダンプへ変換しテーブルへ格納する必要がある。

3.4. 提案手法の比較

Wikipedia の再利用性を高めるアプローチとして、Hadoop による解析と、Wiki テキストの XML 文書化の手法を提案した。2 つの手法の長短をまとめたものが表 1 である。

表 1. 提案手法の比較

	Hadoop 解析	XML 文書化
出力結果	基本的にテキスト	XML 文書
プログラミング言語	基本的に Java 他も使用可能	PHP のみ
データの前処理	不要	必要
Wiki テキスト解析能力	なし	MediaWiki 標準パーサ
分散処理	可能	不可能

2 つの手法には長短がある。Hadoop による解析はプログラミング次第で様々なデータが抽出できるので自由度が高い。また分散処理による高速な解析が期待できる。しかし Wiki テキストの解析に当たって MediaWiki 標準のパーサが使えない。またテンプレート呼び出しのように、データベースに予め格納されたデータが必要な機能が使われている Wiki テキストの扱いが困難である。

XML 文書化の手法は、得られる出力結果が XML 文書という構造的なデータのため再利用性がとても高い。また標準のパーサが利用できるため間違いなく Wiki テキストを解析できる。一方でデータの前処理や XML 文書化のプログラムは分散処理できないため、今後 Wikipedia のデータが増大するに従って時間がかかるようになると考えられる。

```
echo "<?xml version=¥"1.0¥" encoding=¥"UTF-8¥"?>¥n";
echo "<page>¥n";
echo "<title>".$output->getTitleText()."</title>¥n";
echo "<body>¥n".$output->getText()."¥n</body>¥n";
echo "<categories>¥n";
foreach ($output->getCategoryLinks() as $category) {
   echo "<category>$category</category>¥n";
}
echo "</categories>¥n";
echo "<interlangs>¥n";
foreach ($output->getLanguageLinks() as $link) {
    echo "<interlang>$link</interlang>¥n";
}
echo "</interlangs>¥n";
echo '</page>';
```

リスト 5. XML 文書生成プログラム

```
<?xml version="1.0" encoding="UTF-8"?>
<page>
  <title>タイトル</title>
  <body>
    <p>本文</p>
  </body>
  <categories>
    <category>カテゴリ 1</category>
    <category>カテゴリ 2</category>
    <category>カテゴリ 3</category>
  </categories>
  <interlangs>
    <interlang>de:言語間リンク</interlang>
    <interlang>en:言語間リンク</interlang>
    <interlang>fr:言語間リンク</interlang>
  </interlangs>
</page>
```

リスト 5. XML 文書例

4. これまでの進捗状況と今後の展望

本共同研究の本来の目的は、言語間の差異を活用したWeb 情報資源の拡充、アクセス方法、情報抽出と組織化に関する研究を行うことであった。しかしWikipediaのデータの再利用性が低く、本来の研究を進めるに当たって障害となることが分かった。そこで本論文ではWikipediaのデータの再利用性を高め、柔軟かつ容易にアクセスするための手法を提案した。これらの手法は本共同研究だけでなく、Wikipedia を対象とした研究全般において共通して使用できると考えている。今後はより再利用性を高める手法を整備するとともに、本来の目的である言語間の差異を活用したWeb 情報資源の拡充という課題に取り組む。

5. 研究成果リスト

(4) 国内会議発表

[1] 森竜也, 増田英孝, 清田陽司, 中川裕志. Wikipediにおける言語間の差異マイニング, 情報処理学会 第 72 回全国大会 講演論文集 (5ZN-6), March, 2010.

[2] 森竜也, 増田英孝, 清田陽司, 中川裕志 . Wikipedia を活用した言語間差異比較システムの提案, 第2回データ工学と情報マネジメントに関するフォーラム DEIM2010,(A5-5), March, 2010.

(5) その他（特許, プレス発表, 著書等）

[3] 森竜也. Wikipedia を解析！Hadoop によるMapReduce処理を試す, SoftwareDesign, 2010年3月号, pp.114-117.

[4] 森竜也. 分散処理プラットフォーム Hadoop による Wikipedia データの解析, Wikimedia Conference Japan 2010, http://meta.wikimedia.org/wiki/Wikimedia_Conference_Japan_2010, November, 2010.