自動更新型WIXファイル生成システムおよびDeep Webに対するアタッチ機構の構築

金岡　慧† 遠山元道††

† †† 慶應義塾大学理工学部情報工学科 〒223-8522 神奈川県横浜市港北区日吉 3-14-1
E-mail: †kei@db.ics.keio.ac.jp, ††toyama@ics.keio.ac.jp

あらまし Web IndeX (WIX) とは, キーワードと URL の組み合わせであるエントリの集合が記述された WIX ファイルを用い, Web ページ内の文章に出現するキーワードに対して, それに対応する URL へのハイパーリンクを生成 (アタッチ) するシステムである. 本研究ではエントリを取得するための設定ファイルを記述することで, WIX ファイルの生成・自動更新を行うシステムを構築した. また検索エンジン結果ページなどのクエリパラメータのみが変化する URL 集合は無限に存在するため, WIX ファイル化が困難となる. そこで本研究では, 大規模な固有名詞のリストを用いた動的 URL を生成する新たなアタッチ機構を提案する. 固有名詞のリストは日本語版 Wikipedia の見出し語一覧、Google・Yahoo 検索急上昇ワードなどをベースとし, また Web ページを形態素解析することで得られる新規語によって構築した.

キーワード Web IndeX 、Web 情報システム、Web スクレイピング、コンテンツ

1. はじめに

著者らは Web における利用者主導による情報資源結合を実現するために, Web IndeX (WIX) システムという情報資源表現形式の提案, 開発を行っている. キーワードと URL の組み合わせであるエントリの集合を XML 形式で記述したものを WIX ファイルという. WIX ファイルを用い, 閲覧中の Web ページに結合することで, Web ページ内の文章に出現するキーワードをそれに対応する URL へのハイパーリンクに変換する. 現在の Web では, Web ページ作成者によって特定のアンカーテキストから特定のページへのリンクが関連付けられるという構造が一般的である. また, Web ページ内のリンクは常に既存の Web ページへしかリンクすることができず, その Web ページ作成後に作成される新たな Web ページへのリンクを作成することは, その Web ページ作成時には決してできない. WIX では, アンカーテキストとリンクを Web ページから独立した「キーワードとリンク先の集合」として扱い, 任意のドキュメントに対してユーザ主導で「結合」することでドキュメント内のキーワードを対応する URL のハイパーリンクに自動的に変換する. その結果, Web ページ作成の時系列という壁を越え, 古い Web ページから新しい Web ページへのリンクも可能となる.

本論文では, WIX ファイルの管理とコンテンツの充実を目的とし, 2 つの提案を行う. 1 つ目として, 設定ファイルに必要なパラメータを記述するだけで WIX ファイルを生成できるシステムを構築した. またこのシステムでは, Web 上に存在するリンク集や単語リストファイル (Web リソース) を利用して生成した WIX ファイルの自動更新を行う. 次に 2 つ目の提案として, WIX ファイルの作成が困難となる Deep Web に対する新たなアタッチ機構を提案する. これは 1 つの大規模な単語リストを用いて動的に URL を生成し, 様々なページに遷移することができる機構である.

本論文の構成は以下の通りである. まず, 2 章で本論文の研究目的について述べる. 3 章で WIX システムの概要を説明する. 4, 5 章で提案システムについて説明する. 6, 7 章で評価・まとめを行う.

2. 研究の目的

近年, Web の普及と共にユーザは検索エンジンを利用して情報検索を行うようになった. ユーザは情報を取得したい単語を検索エンジンに入力し, その検索結果の Web ページ集合の中から必要な情報を得る, といったステップを踏むのが一般的である. したがって Web ページ内で新たに情報を取得したい単語が存在した場合, ユーザは更にその単語を検索エンジンなどにかけなければならない. このような単語が複数存在する場合, ユーザは何回も検索エンジンに単語を入力しなければならず, かなりの負担になってしまうと考えられる.

これに対し, 著者らは Web における利用者主導による情報資源結合を実現するために, Web IndeX (WIX) という情報資源表現形式の提案, 開発を行っている. WIX システムには WIX ファイルというリソースが存在し, システム開発者らが作成したもの, 企業や一般ユーザが作成したものなどがある. 本研究ではエントリを取得するための設定ファイルを記述することで, WIX ファイルの生成を行うことができるシステムを構築することで, WIX ファイルの作成の効率化を目的とした. またこのシステムでは Web リソースを用いて生成した WIX ファイルの自動更新を行う. これによって WIX ファイル作成者の管理の負担を軽減するとともに, WIX ユーザが常に最新の内容の

WIX ファイルを利用できるようにすることを目的とした．

また検索エンジン結果ページなどのクエリパラメータのみが変化する URL 集合は無限に存在するため，WIX ファイルの作成は困難となる．そこで大規模な単語リストを用いた動的に URL を生成する新たなアタッチ機構を構築することで，先の問題点に対処し，WIX システムにおけるコンテンツの充実を図った．またこれにより，ユーザの Web ブラウジング時のタイピングの負担を軽減することも目的とした．

3. Web IndeX システム

3.1 WIX ファイル

WIX ファイルは XML 形式で記述されたキーワードと URL の組み合わせであるエントリの集合である．エントリには，キーワードとなる見出し語を keyword 要素として，それに対応する詳細情報を示す参照先の URL を target 要素として格納する．また header 要素にファイル概要，作者コメントなど，その WIX ファイル全体についてのメタデータを格納することも可能である．記述例は図 1 のようになる．WIX ファイルは「wikipedia の見出し語一覧」や「Ameba ブログ」などのように，内容がある程度グルーピングされるものが多い．

```
<?xml version="1.0" encoding="UTF-8"?>
<WIX>
  <header />
    <body>
    ...
      <entry>
        <keyword>ザッケローニ</keyword>
        <target>http://ja.wikipedia.org/wiki/ザッケローニ</target>
      </entry>
      <entry>
        <keyword>ジーコ</keyword>
        <target>http://ja.wikipedia.org/wiki/ジーコ</target>
      </entry>
      <entry>
        <keyword>イビチャ・オシム</keyword>
        <target>http://ja.wikipedia.org/wiki/イビチャ・オシム</target>
      </entry>
    ...
    </body>
</WIX>
```

図 1　WIX ファイル記述例 (日本語版 Wikipedia.wix 一部抜粋)

3.2 アーキテクチャ

3.2.1 WIX ライブラリ

WIX ライブラリでは，全ての WIX ファイルの XML テキストをそのまま保存しており，ファイル単位での情報管理を行っている．アタッチの際には全ての WIX ファイルのエントリに対して辞書式マッチングを行うため，WIX ファイルをエントリ単位に分解し，WIX DB に格納する．

3.2.2 WIX DB

WIX DB では，ライブラリで管理している WIX ファイルをエントリ単位に分解し，RDB にタプルとして管理する．WIX ファイルのもつエントリの情報は entry テーブルで管理される．(表 1)．エントリが属する WIX ファイルの id(wid)，エントリの id (eid)，辞書語となる keyword とそれに対応する target を属性として持つ．

表 1　entry テーブル

wid	eid	keyword	target
1	1	芥川龍之介	http://ja.wikipedia.org/wiki/芥川龍之介
1	2	ザッケローニ	http://ja.wikipedia.org/wiki/ザッケローニ
2	3	田中将大	http://ameblo.jp/tanaka-masahiro/
3	5	坂本勇人	http://www.giants.jp/G/player/prof_2756.html
:	:	:	:

3.2.3 Find インデックス

Find インデックスでは，WIX DB の entry テーブルからエントリ情報をメモリ上に展開する．WIX システムでは Aho-Corasick 法に基づくオートマトンを構築し，辞書式マッチングを行う．

3.3 ハイパーリンクの生成 (アタッチ)

WIX システムのクライアントサイドは，FireFox add-on や Chrome Extension などによって実装されている [1]．図 2 は Chrome Extension の例である．ユーザがブックマークボタンをクリックすると，サーバーサイドにおいて閲覧 Web ページと Find インデックスとの辞書式マッチングが行われ，リンク生成済の HTML 文書がレスポンスとして返され，元のページにはなかったハイパーリンクが処理後のページに生成される．このハイパーリンクを生成する処理をアタッチと呼ぶ．これによって，WIX ファイル内の target タグに記述されている URL と結合されたことになる．

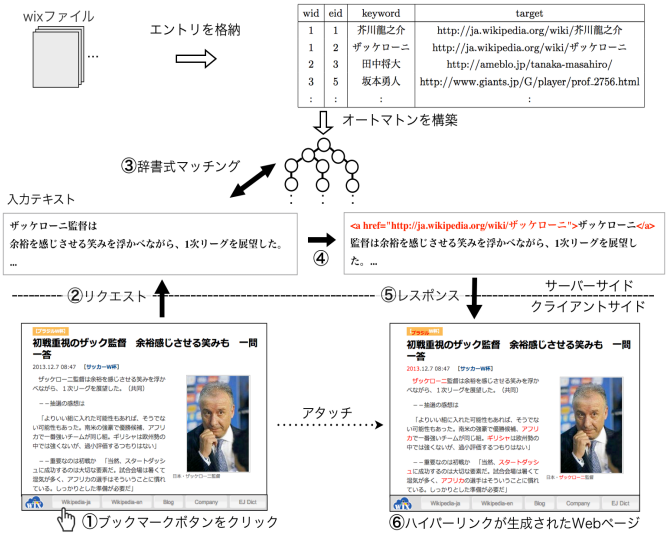

図 2　ハイパーリンクの生成 (Chrome Extension)

4. 自動更新型 WIX ファイル生成システム

4.1 背景と概要

WIX ファイルを作成するには次の手法が挙げられる．

（1） 手動での記述

（2） WIX File Extractor [2] の使用
（3） Web 上のリンク集をクローリング
（4） Web 上の単語リストファイルの使用
（5） ローカルの単語リストファイルの使用

作成方法 1 の手動での記述の場合, ユーザの意図が最も反映された WIX ファイルが作成できるが, 手間や時間といった負荷が大きい. 作成方法 3 のように Web 上のリンク集から HTML パーサーやプログラムを記述することによってエントリを取得することはできるが, 知識のないユーザにとっては非常に困難となる. この問題への解決策として, 藤井が提案した WIX ファイル作成支援システムである WIX File Extractor [2] がある. これはウェブブラウザの拡張機能を用い, リンク集が存在する Web ページ上でユーザがマウス操作で目的のエントリ部分を選択することで, システムがそのエントリまでの X Path 式を用いて WIX ファイルを作成することができる. しかし, 作成した WIX ファイルにノイズが含まれるといった問題点がある. 作成方法 4・5 に関してもプログラムを記述することになるが, リンク集や単語リストファイルごとにプログラムを記述して WIX ファイルを作成するのはとても非効率であると言える.

そこで本研究ではリンク集や Web 上・ローカルの単語リストファイルからエントリを取得する設定ファイルを記述することで WIX ファイルの生成を行うことができるシステムを提案する. これによって知識のない一般ユーザだけでなく, HTML パーサーやプログラムを記述することができるユーザにとってもそれらを記述する手間を省くことができ, 効率的に精度の高い WIX ファイルを作成することができる. 設定ファイルは図 3 に示すように JSON 形式で記述される.

```
{
    "wixFileName" : "…",
    "username" : "…",
    "origin" : "…",
    …
}
```

図 3　設定ファイル

wixFileName では WIX ファイルの名前を定義し, username では作成者名を記述する. origin では以下のいずれかを指定する.

- ”html”：Web 上のリンク集からエントリを取得
- ”webfile”：Web 上に存在する単語リストを使用
- ”localfile”：ローカルの単語リストを使用

またここで, Web 上に存在するリンク集や単語リストファイルを Web リソースと呼ぶ. Web リソースを用いて作成される WIX ファイルは WIX システムにおいて主力コンテンツとなっているものが多い. 例えば Ameba 芸能人・有名人ブログ一覧などといったリンク集や, Wikipedia の見出し語一覧ファイルから作成した WIX ファイルがあげられる. これらの WIX ファイルは元にしている Web リソースの内容が更新されるたびに, その WIX ファイルの内容も更新されるべきである. しかしその更新の確認を, Web リソースから生成された全ての WIX ファイルに対して行うことは負担となる. そこで本システムでは以下の図 4 のようなステップを Web リソースから生成された WIX ファイルに対して定期的に行うことで, その内容を常に最新の状態にすることができる. これによってファイル作成者の負担軽減につながり, ユーザが常に最新の情報を得ることができる.

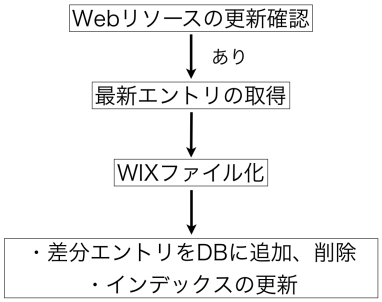

図 4　自動更新処理の流れ

4.2　システム内部仕様

本システムは以下の処理部によって構成される.

（1） Web リソースの更新確認部

Web リソースを元に生成した WIX ファイルの最新更新日時情報は, 以下の表 2 のように管理されている.

表 2　WIX ファイル最終更新日時テーブル

id	wixfile_name	update_date
1	Wikipedia_ja	2013-Dec-19 12:12:32
2	Wikipedia_en	2013-Dec-02 11:07:54
3	ameblo	2013-Dec-02 04:30:06
:	:	:

Web リソースの最新更新日を HTTP ヘッダのメタ情報にある Last-Modified エンティティヘッダフィールドから定期的に取得し, DB の値と比較して更新されていた場合, エントリ取得部に処理が移る.

（2） エントリ取得部

エントリ取得処理を行う. 設定ファイルにおいて記述された”origin”の値によって処理が分岐する.

（3） WIX ファイル生成部

エントリ取得部から受け取ったエントリを元に WIX ファイルの作成を行う.

（4） アップデート処理部

作成した WIX ファイルをライブラリに配置し, DB とインデックスの更新処理を行う.

4.3 システム外部仕様

4.3.1 リンク集を用いた WIX ファイルの作成

リンク集を用いて WIX ファイルを生成するには図 5 のような設定ファイルを記述する．また使用するフィールドの概要を表 3 に示す．

```
{
    "wixFileName" : "…",
    "username" : "…",
    "origin" : "html",
    "crawling" : [{
        "url" : "…",
        "selector" : "…",
        "keyword" : {
            "val" : "…",
            "find" : "…",
            "trim" : ["…"]
        },
        "next" : {…}
    }]
}
```

図 5　リンク集を用いる設定ファイル

表 3　フィールド一覧 (*は必ず記述が必要となるフィールド)

フィールド名	型	概要	
crawling*	array	リンク集からエントリを取得するのに必要な以下のフィールドを記述	
url*	string	起点となる URL を記述	
selector*	string	起点となる URL から抽出したい要素の CSS セレクタを記述	
keyword	string	keyword をどのようなオプション (val, find, trim) で抽出するかを記述	
val	string	selector で指定したタグの属性を指定	
find	string	selector で抽出したタグの子要素を CSS セレクタで指定	
trim	array	”blank”	空白削除
		”bracket”	括弧と括弧内の文字列の削除
		string	記述された正規表現, 文字列を削除
next	object	起点のページからの遷移がある場合に使用	

設定ファイルで記述されたパラメータを元に, エントリ取得部においてエントリの取得を行う．リンク集からのエントリの取得には, 起点となるページの URL と取得したいエントリが存在する CSS セレクタを必ず指定する必要がある．処理の流れは以下のようになる．

（1） ”url”, ”selector”の処理

指定された URL にリクエストを送り, HTML 文書を取得．指定された CSS セレクタを用いて取得した HTML 文書をパース．CSS セレクタで指定されたタグの href 属性を target 要素として取得．

（2） ”keyword”の処理

指定がない場合, 抽出したタグのテキストノードを keyword 要素として取得．”val”, ”find”が記述されている場合, 抽出したタグの属性もしくは子要素のテキスト部分を keyword 要素として取得．”trim”が記述されている場合, 取得した keyword に対して”trim”で指定された処理を行う．

（3） next フィールドの有無

next フィールドが記述されている場合, 取得した target を”url”として処理 (1) に戻る．記述されていない場合, 取得した keyword と target のエントリ集合を WIX ファイル生成部へ渡す．

設定ファイルの記述例を図 6 に示す．例に挙げた Web ページにおいて, CSS セレクタを指定しただけでは取得される keyword に空白やアルファベットといったノイズが入る．これに対し, 設定ファイルにおいて keyword に対する”trim”を記述することで, 取得される keyword のノイズを除去することができる．また図 7 に起点ページからの遷移があるリンク集の例とその設定ファイルを示す．

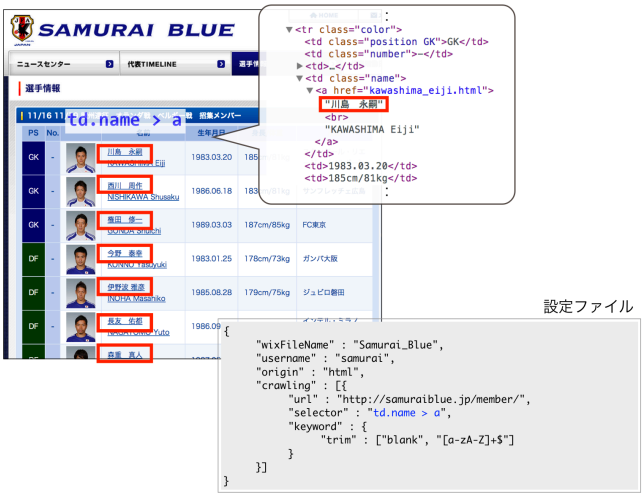

図 6　リンク集を用いる設定ファイル記述例 (ページ遷移なし)

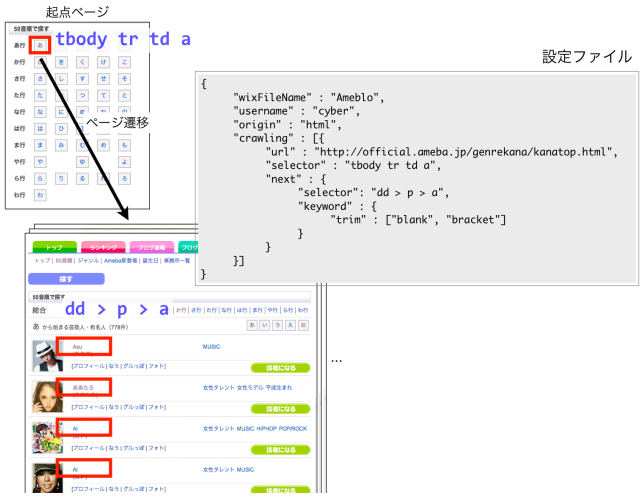

図 7　リンク集を用いる設定ファイル記述例 (ページ遷移あり)

4.3.2 単語リストファイルを用いた WIX ファイルの作成

Web 上・ローカルに存在する単語リストファイルから WIX ファイルを作成するには, 図 8, 図 9 のような設定ファイルの記述が必要になる. それぞれ"uri", "filepath"の値として参照する単語リストファイルの URI またはパスを指定し, "format"の部分にはファイルの形式を記述する. "prefix"の値には, 単語と結合することで URL を形成する文字列を記述する. 図 10 に Web 上の日本語版 Wikipedia の見出し語一覧ファイルを用いた WIX ファイルの生成を行う設定ファイルを例として挙げる.

```
{
    "wixFileName" : "…",
    "username" : "…",
    "origin" : "webfile",
    "resource" : [{
        "uri" : "…",
        "format" : "…",
        "prefix" : "…"
    }]
}
```

図 8　Web 上の単語リストファイルからの WIX ファイルの作成

```
{
    "wixFileName" : "…",
    "username" : "…",
    "origin" : "localfile",
    "resource" : [{
        "filepath" : "…",
        "format" : "…",
        "prefix" : "…"
    }]
}
```

図 9　ローカルの単語リストファイルからの WIX ファイルの作成

5. Deep Web に対するアタッチ機構

5.1 背景と概要

従来の WIX システムにおいて, 動画や画像などのコンテンツやポータルサイトの検索結果ページなどが遷移先となるような WIX ファイルは存在しなかった. そのような WIX ファイルを作成するとなると, コンテンツが存在する URL や検索結果ページ内の個々の URL を target 要素, それに対応するキーワードを keyword 要素として格納した WIX ファイルを生成することとなる. しかしそのような URL は膨大に存在するため, WIX ファイル化するとなるとそれぞれのキーワードに対して

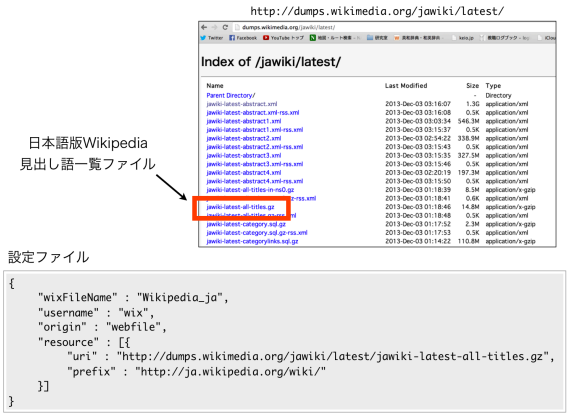

図 10　日本語版 Wikipedia 単語リストファイルを用いる設定ファイル

その膨大な数のエントリを記述することになってしまい, 作成が困難となる. またそれらの中からいくつかを選ぶということになっても, ユーザによってその選定の指標は異なることから, 一意に定めることができないといった問題も発生する. (図 11)

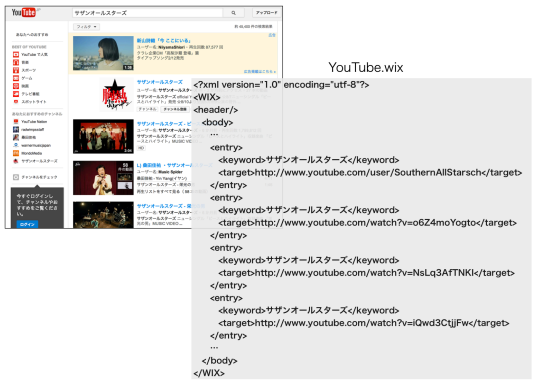

図 11　WIX ファイル化が困難となる例 1

また各コンテンツとそれに対する URL が一覧表示されている検索結果ページ自体を WIX ファイル化するとなると, その URL は入力されうるキーワードの数だけ存在することになり, 先と同様に作成は困難となる. 以下の図 12 に Google 検索結果ページ集合を WIX ファイル化する例を取り上げる.

図 12　WIX ファイル化が困難となる例 2

しかし検索エンジン結果ページなどの URL は全て共通してクエリパラメータのみが変化し, URL からクエリパラメータを除いた部分 (以下 prefix と呼ぶ. 表 4 参照) は変化しない. このように Web 上に存在しているが検索エンジンのクローラーがインデックス化することのできない領域にある文書や Web ページなどのことを Deep Web という. 例えばポータルサイトでのキーワードに関する検索結果ページや, amazon や youtube のように検索窓にキーワードを入力して得られる結果ページなどである. この特徴を利用し, 固有名詞から成る 1 つの大規模な単語リストと prefix を結合することで動的 URL を生成する, WIX ファイルを使ったアタッチとは異なる新たなアタッチ機構を構築した. prefix は以下の表 4 のように格納されている.

表 4　prefix テーブル

id	name	prefix
1	Google	http://www.google.co.jp/search?&q=
2	Yahoo	http://search.yahoo.co.jp/search?p=
3	amazon	http://www.amazon.co.jp/s/ref=nb_sb_noss_2?field-keywords=
4	youtube	http://www.youtube.com/results?search_query=
:	:	:

単語リストは Wikipedia の見出し語, Google・Yahoo 検索急上昇ワードなどをベースとし, また Web ページを形態素解析することで得られる新規語によって構築した. Web ページは, Google ニュースや Yahoo ニュースなどのニュースページを対象とした. つまり 1 つの単語リストと prefix との結合を行うことで動的に URL を生成し, 様々なページに遷移することができる新たなアタッチ機構である.

5.2　単語リスト DB とアタッチ機構

Deep Web に対するアタッチ機構では, 単語リスト DB を用いてアタッチを行う. 通常の WIX ファイルを用いたアタッチはキーワードに対応する URL との結合処理を行うが, Deep Web に対するアタッチ機構ではツールバーのボタンと単語リスト DB のキーワードを元にアタッチを行い, アタッチされたキーワードがユーザによってクリックされた際, そのキーワードと押されていたツールバーのボタンの情報を元に, URL を動的に生成し, 遷移することができる. 図 13 にその様子の例を示す.

5.3　ベースとなる単語リストの構築

本研究では日本語 Wikipedia の見出し語から固有名詞を抽出し, ベースとなる単語リストとした. Wikipedia は世界最大規模のコンテンツ量を誇る Web 事典であり, 幅広い分野に関する単語を網羅している. 日本語版の単語総数は 2014 年 1 月時点において 246 万語にのぼる. 4 章において提案した自動更新型 WIX ファイル生成システムを用いて, Wikipedia の見出し語一覧ファイルがアップロードされると, ベース単語リストに用いている Wikipedia の見出し語一覧との差分更新を行う仕様となっている.

Wikipedia の他に, Google・Yahoo 検索急上昇ワードから得られるトレンドワードに着目し, 新規語の追加を行った.

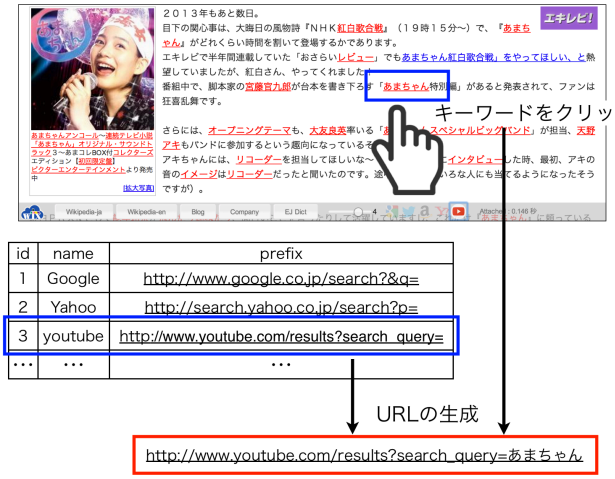

図 13　単語リストを用いたアタッチ

5.4　形態素解析による新規語の抽出

ベースとなる単語リストに加え, Web ページを形態素解析することで上記の単語では網羅しきれない新規語の抽出を行った. 形態素解析には, オープンソースの形態素解析エンジンである MeCab(注1) を使用した.

5.4.1　メインコンテンツ部分の抽出

Web ページのテキスト部分が解析対象になるが, その全てを解析対象としてしまうとメニューバー, ナビゲーションメニュー, 広告部分といったいわゆるノイズと呼ばれる部分まで含んでしまうこととなる. 一般的に, ノイズ部分は Web ページのコンテンツの 40〜50%を占める. (図 14)

図 14　Web ページ上のメインコンテンツとノイズ

本研究では Fei Sun らが提案した手法 [3] を用いて Web ページのメインコンテンツ部分の抽出をまず行い, そこから得られるテキストに対して形態素解析を行った. 更に, Web ページのヘッダー情報の内, タイトル・キーワード・ディスクリプションといったメタ情報に記述されている内容も抽出することで, 新

(注1)：形態素解析エンジン MeCab, 京都大学情報学研究科および日本電信電話株式会社コミュニケーション科学基礎研究所 共同研究ユニットプロジェクト, http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html

規語の抽出を図った.

5.4.2 形態素解析による新規語の抽出アルゴリズム

新規語を抽出する既存研究は数多く存在する. それらは沢井ら [4] のように Web 検索を利用した名詞のみで構成される複合名詞の抽出や, 伊藤ら [5] のように品詞結合規則や外部辞書データを用いた複合名詞の抽出手法が多い. それに対し今回提案する手法では, 個々のウェブページを対象に, そのウェブページにおいて出現頻度の高いキーワードに焦点をあて, 複合名詞だけでなく, 名詞以外の品詞からも始まる新規語の抽出を図る.

まずメインコンテンツとメタ情報抽出処理後のウェブページのテキスト部分を得る. この際, テキスト部分は全て結合して得るのではなく, HTML のタグを利用することで文章の切れ目や単語の分かれ目を認識する. このようにして得たテキスト部分に対し, MeCab を用いて形態素解析を行う. 名詞は対象外とした品詞結合規則 (動詞と助動詞の結合など) に当てはまる形態素は予め結合する. 隣接する形態素の出現確率の差を用いて以下のように表す.

$$diffP = |P(i) - P(i+1)|$$

全ての形態素間の $diffP$ を算出し, 値が小さく, かつ $P(i)$ と $P(i+1)$ が大きいものを選定し, 閾値 t とする. 助詞などのようなストップワードは出現確率が高くなることから, ストップワードとそれ以外の品詞の形態素間の出現確率の差を利用することで単語の切り出しを行っていく. 出現確率が等しい, つまり $diffP$ が 0 になる, もしくは先に選定した t となるような形態素を結合していく.

6. 評　　価

6.1 自動更新型 WIX ファイル生成システムの評価

6.1.1 評 価 方 法

本システムの有用性を評価するため, 22 種類のリンク集をもとにエントリー数の異なる WIX ファイルを作成し, その際に記述した設定ファイルで用いた機能に関するデータと適合率を取得した. 作成する際, 取得するエントリの再現率は 100%で固定とし, その上で適合率を 100%に近づけることを目的とした. なお, 本評価においての再現率と適合率は以下のように定義する.

$$\text{再現率} = \frac{\text{期待通り取得できたエントリの総数}}{\text{取得したいエントリの総数}} \times 100 \quad (1)$$

$$\text{適合率} = \frac{\text{取得したいと期待していたエントリの総数}}{\text{取得したエントリの総数}} \times 100 \quad (2)$$

6.1.2 結果および考察

評価実験に用いた 22 種類のリンク集のうち 1 種類を除いて, 適合率 100%の WIX ファイルを生成することが出来た. その設定ファイルに関するデータを表 5 に示す.

表 5 に示す 21 種類の WIX ファイルのうち, 約 7 割が URL と CSS セレクタの記述のみ, もしくはデフォルトのキーワード編集機能 (空白・括弧の削除) を用いて適合率 100%の WIX ファイルを生成することができた. その他のリンク集に関しては, 正規表現を用いて特定のノイズを除去する, またオプション機能として find フィールドを設定することで, 適合率 100%の WIX ファイルを生成することができた. WIX ファイルを生成する既存の手法では, キーワードの編集やノイズを除去することは困難であったが, 本システムでは空白や括弧の削除, 正規表現によるノイズの除去が可能であり, 期待通りの WIX ファイルを生成することができたのだと考える.

また表 5 に示されるように, 設定ファイル自体の記述行数は平均で 14 行と, HTML パーサーやプログラムを記述するよりもはるかに少ない記述で WIX ファイルを生成することができた.

22 種類のうち日本図書館協会の全国の図書館の公式ホームページのリンク集(注2)では適合率 100%の WIX ファイルを生成するために正規表現を数多く記述した結果となった. 原因としては, 該当 Web ページの HTML の構造がリンク集のみを特定できる構造でなかったため, CSS セレクタでキーワード部分の指定をしただけではエントリ対象外のノイズが除去できず, 各ノイズに対応する正規表現を記述しなければならなかったためである. しかしこのように設計がしっかりとなされていない Web ページは稀であることから, 本システムの設定ファイルを用いることで再現率と適合率が 100%となる WIX ファイルを作成することができ, 有用性があると言える.

6.2 Deep Web に対するアタッチ機構の評価

6.2.1 評 価 方 法

本機構によってアタッチされたキーワードの精度の評価を行う. 評価の対象となる Web ページはジャンルを問わずユーザ 6 人によって全 25 種類を選定してもらい, アタッチされたキーワードの再現率は 3 式のように求めた.

$$\text{再現率} = \frac{\text{期待通りアタッチできたキーワードの総数}}{\text{アタッチされると考えられるキーワードの総数}} \times 100 \quad (3)$$

6.2.2 結果および考察

表 6 に評価実験を行った Web ページにおいてアタッチされたキーワードの再現率の分布を示す. この表より約 8 割の Web ページにおいてアタッチされると期待されるキーワードに対してアタッチが行えたことがわかる. 一方再現率が 71〜80 %となった Web ページが存在したが, これはその Web ページの分野がマイナーな単語を数多く含んでおり, 本提案で構築した単語リスト DB では網羅できなかった単語が数多く存在していたためである. 再現率が 81〜90 %の Web ページも, その Web

(注2)：http://www.jla.or.jp/, 参考・抜粋, 2014 年 1 月 17 日アクセス.

表 5　生成できた WIX ファイルの設定ファイルに関するデータ

ホームページ名	起点ページ数	遷移	空白、括弧削除	正規表現	オプション機能	ファイル行数	取得エントリ数
映画.com	1	なし	-	-	-	9	552
FC Barcelona	1	なし	-	-	-	9	26
EXILE 公式 HP	1	なし	-	-	-	9	14
慶應義塾豆百科	1	なし	-	-	-	9	100
SKE48 公式 HP	1	なし	-	-	-	9	67
Ameba 芸能人・有名人ブログ	1	あり	○	-	-	16	11774
中日ドラゴンズ	1	なし	○	-	-	12	73
広島東洋カープ	1	あり	○	-	-	16	83
NMB48 公式 HP	1	なし	○	-	-	12	65
文部科学省 大学公式 HP リンク集	4	なし	○	-	-	24	1136
金融庁 リンク集	1	なし	○	-	-	12	94
乃木坂 46 公式 HP	1	なし	○	-	○	13	32
SAMURAI JAPAN	1	なし	○	-	○	13	36
横浜 DeNA ベイスターズ	5	なし	○	-	○	41	90
読売ジャイアンツ	1	なし	○	-	○	13	104
阪神タイガース	2	なし	○	○	-	18	91
ソフトバンクホークス	1	なし	○	○	-	12	114
SAMURAI BLUE	1	なし	○	○	-	12	23
日本図書館協会 図書館公式 HP リンク集	1	あり	○	○	-	20	1634
楽天イーグルス	1	あり	○	○	○	18	170
上場企業一覧リンク集「日本企業」	21	なし	○	○	○	12	3551
日本ハムファイターズ	1	あり	○	○	○	18	85

ページにおいて主題となっている単語にはアタッチがされていたが, その他アタッチ処理が行われなかった単語も存在する結果となった. これら Wikipedia のタイトルや Google, Yahoo トレンドワード, ニュースページを解析するだけでは網羅することができない単語へのアタッチを可能にするには, 今後ニュースページ以外の HTML 文書に対しても形態素解析を行うことで単語の切り出しを行っていく必要があると考える. その際, 固有名詞を抽出することができるより精度の高いアルゴリズムの導入が必要となると考える.

表 6　実験に用いた Web ページにおけるアタッチされたキーワードの再現率ごとの分布

	本システムによる再現率						
再現率 (%)	0〜60	61〜70	71〜80	81〜90	91〜99	100	計
Web ページ数 (個)	0	0	1	4	7	13	25

7. まとめと結論

今回 1 つ目に提案した自動更新型 WIX ファイル生成システムは, Web リソースを用いて作成した WIX ファイルの管理負荷の軽減, および最新 WIX ファイルの利便性に貢献する機構となった.

また 2 つ目に提案した Deep Web に対するアタッチ機構では, 246 万語を誇る日本語版 Wikipedia の見出し語一覧, Google・Yahoo 検索急上昇ワードなどをベースの単語リストとし, 加えて新規語抽出アルゴリズムによってベース単語リストでは網羅できない新規語の抽出を行うことで, 大規模単語リストとそれを用いたアタッチ機構を構築した. これにより遷移できるページのコンテンツが増え, WIX がユーザにとってより良いシステムになると考える.

文　献

[1] 林 昌弘, 青山 峻, 朱 成敏, 遠山 元道 (慶應義塾大学) ”WIX システム (1) ユーザインターフェース”, データ工学ワークショップ, DEIM2011. 2011.

[2] 藤井 洋太郎, 遠山 元道 (慶應義塾大学) ”WIX システムにおけるコンテンツ作成支援” 日本 DB 学会論文誌, Vol.11, No.1, pp.7-12, June 2012

[3] Fei Sun, Dandan Song, and Lejian Liao ”DOM Based Content Extraction via Text Density”, SIGIR’11, July 24-28, 2011, Beijing, China.

[4] 沢井 康孝, 山本 和英 (長岡技術科学大学 電気系) ”Web 検索を用いた複合名詞同定”, 言語処理学会 第 14 回年次大会 発表論文集 2008 年 3 月

[5] 伊藤 直之, 西川 侑吾, 田村 直之, 中川 修, 新堀 英二 ”品詞結合規則と外部辞書データを用いた複合名詞の生成”, FIT2009(第 8 回情報科学技術フォーラム)