九州大学学術情報リポジトリ
Kyushu University Institutional Repository

頻度情報を用いたWeb文書群からのテンプレート抽出

野口, 龍太郎
九州大学理学部物理学科

山田, 泰寛
九州大学大学院システム情報科学府

池田, 大輔
九州大学情報基盤センター

廣川, 佐千男
九州大学情報基盤センター

http://hdl.handle.net/2324/2986

出版情報：2004-03-05
バージョン：
権利関係：

KYUSHU UNIVERSITY

DEWS2004 6-B-01

頻度情報を用いたWeb文書群からのテンプレート抽出

野口龍太郎† 山田 泰寛†† 池田 大輔††† 廣川佐千男†††

† 九州大学理学部物理学科 〒 812-0061 福岡市東区箱崎 6-10-1 九州大学情報基盤センター
†† 九州大学大学院システム情報科学府 〒 812-0061 福岡市東区箱崎 6-10-1 九州大学情報基盤センター
††† 九州大学情報基盤センター 〒 812-0061 福岡市東区箱崎 6-10-1 九州大学情報基盤センター
E-mail: †{ryutaro,yshiro}@matu.cc.kyushu-u.ac.jp, ††{daisuke,hirokawa}@cc.kyushu-u.ac.jp

あらまし 大学のシラバスやレシピなど，Web 上には同一テンプレートで記述されたページ群が多数ある．各ページ群に対するテンプレートが分かれば，ページに書かれた個別データを抽出し，データベースとしての活用が期待される．本論文では，Web ページに含まれる n-gram の出現頻度情報だけを用いて効率よくテンプレートを発見するアルゴリズムを提案する．また，これを実装したシステムを用いて Web 上に存在する大学のシラバス，検索エンジンの検索結果について行なった実験結果について報告する．

キーワード Web マイニング，シリーズ型 HTML 文書，テンプレート発見，部分文字列増幅法，ジップの法則

Template Extraction from Web Documents using Substring Amplification

Ryutaro NOGUCHI†, Yasuhiro YAMADA††, Daisuke IKEDA†††, and Sachio HIROKAWA†††

† Department of Physics, Kyushu University Hakozaki 6-10-1, Higasi-ku, Fukuoka 812–8581, Japan
†† Department of Informatics, Kyushu University Hakozaki 6-10-1, Higasi-ku, Fukuoka 812–8581, Japan
††† Computing and Communications Center, Kyushu University Hakozaki 6-10-1, Higasi-ku, Fukuoka 812–8581, Japan
E-mail: †{ryutaro,yshiro}@matu.cc.kyushu-u.ac.jp, ††{daisuke,hirokawa}@cc.kyushu-u.ac.jp

Abstract There are a lot of Web documents written with the same template. Recipes, stuff pages and syllabus pages of universities are typical examples. If we have the templates of these documents, we can extract the contents and can store them into a database. This paper proposes a template detection algorithm using the frequency of frequencies of n-grams in the documents. Experimental results are shown for series of Web documents.

Key words Web mining, a series Web documents, template detection, substring amplification, Zipf's Law

1. はじめに

Web 上には有用な情報が大量にあり，我々は日常生活の中でこれらを利用している．膨大な量の Web 文書群から必要な情報を効率よく得る技術の開発は，現在の情報社会において重要な課題である．HTML を代表とする半構造化データから知識を抽出・統合できれば，Web をデータベースとして利用できるようになる．このような研究は **Web マイニング**と呼ばれ活発な研究が行なわれている．

例えば，大学のシラバス，新聞社の記事，あるいは，検索エンジンの検索結果などのように，同一サイト内の同種の情報は，同じテンプレートを用いて作成されることが多い．このような HTML 文書は**シリーズ型 HTML 文書**と呼ばれる [3], [8], [9]. 特定の Web 文書を異なる多くのサイトから大量に収集するためには，各サイトの個別テンプレートを自動的に特定し，テンプレートを用いて抽出されたデータを統合しなければならない．

シリーズ型 HTML 文書は，新聞記事のように同種の情報が同一のテンプレートを用いて複数のファイルに記述されるシングル・インスタンスと呼ばれるものと，検索エンジンの検索結果のように，一つのファイルに同種の情報が繰り返し記述されるマルチプル・インスタンスと呼ばれるものの 2 種類に分けられる．シングル・インスタンスにおけるテンプレートとは，各ファイルに共通する部分のことを指し，マルチプル・インスタンスにおいては，繰り返し記述される部分を指す．

[2] では，2 つの HTML 文書をそれぞれタグとそれ以外の部分の列に分け，2 つの列から異なる部分を見つけることによりテンプレート部分を特定し，HTML 文書中の必要な情報を抽出している．このようにシリーズ型 HTML 文書から情報抽出するためのプログラムは，ラッパー [6] と呼ばれ近年盛んに研究されて来た [7]．テンプレートとして共通に現れるテキストを除去してインデックスを作り，検索の精度を上げるという研究もある [1].

[4] では，シングル・インスタンスとマルチプル・インスタンスの両方を対象とし，同種の項目を多数含む半構造化文書の集合からテンプレート部分を特定するアルゴリズムを提案している．共通部分特定アルゴリズムは，交代数という計数を用いて，部分文字列の長さ n と頻度の割合 a%を自動的に決定する．この時，長さ n の部分文字列のうち，頻度の上位 a%に含まれるものは，テンプレート部分に出現する．交代数とは，文字列と部分文字列の集合が与えられたとき，入力上でその部分文字列の出現する部分とそうでない部分の境界の総数を表す. [11] では，このアルゴリズムを応用してラッパーの生成を行なっている．

Web 上には，同一のテンプレートを用いて記述されているシリーズ型 HTML 文書が膨大に存在する．しかし，サイトが違えば同種の項目を持つページであっても，テンプレートが異なっている．例えば，Web 上には検索エンジンは数多く存在するが，検索結果のテンプレートはそれぞれ異なっている．このため，多くのテンプレートに対応するためには，高速で人手のかからない手法が求められる．また，特徴的なタグを除去するなどの特定のテンプレートのみに適用できる前処理を必要としない手法が求められる．

我々は [5], [12] において，与えられた文書群から，部分文字列の頻度を数えるだけで繰り返し出現する部分文字列の集合を発見する**部分文字列増幅法**という手法を提案した．得られる部分文字列を含むかどうかで共通テンプレートで書かれたファイルだけをフィルタリングできる．

この手法は，まず全ての長さの n-gram について，入力として与えられた文字列 (の集合) における出現頻度を求める．次に，頻度 f に対し，出現頻度が f である n-gram の種類数 $V(f)$ を求め，$F(f) = f \times V(f)$ により，頻度 f の全ての n-gram の出現総数を求める．f を変化させたとき，$F(f)$ が最大となる頻度 f が共通テンプレートを持つファイルの総数であると推定する．そして，その頻度を持つ n-gram が共通テンプレートを構成していると考える．

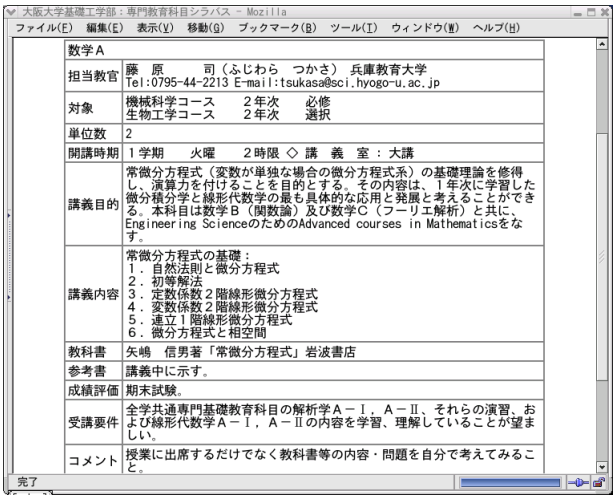
この手法は，入力の長さを n とすると，$O(n)$ 時間で動作し，テンプレートを持たない文書や異なるテンプレートを持つ文書群が与えられた場合にも適用できる．

本論文では，この部分文字列増幅法を用いて，同一のテンプレートを持つ文書群からテンプレートを特定するアルゴリズムを提案する．また，このアルゴリズムを大学のシラバスデータと検索エンジンの検索結果に適用したテンプレート抽出実験について報告する．

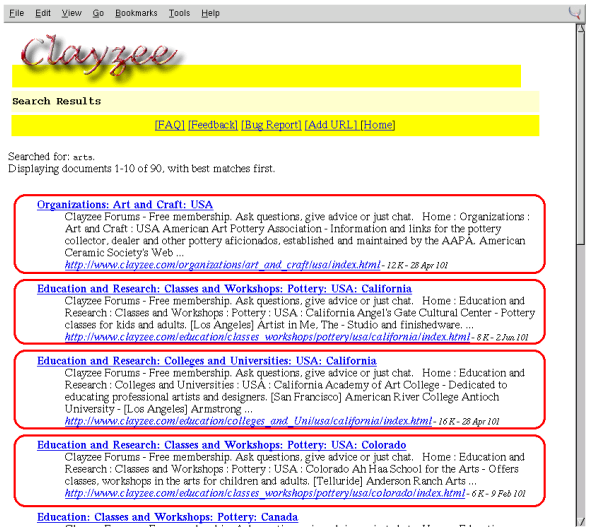
2. シリーズ型 HTML 文書

同一サイトには，同種の情報を提示するため，共通のテンプレートで書かれた文書が多数あることが多い．このような HTML 文書の事を，梅原らは**シリーズ型 HTML 文書**と呼んでいる [8], [9].

シリーズ型の Web 文書は，1 ページ中のデータの個数によって，シングル・インスタンス文書群と，マルチプル・インスタンス文書群の 2 種類に分けられる．大学のシラバスや新聞社のニュースは 1 ページ中に 1 項目しか書かれていないのでシングル・インスタンスであり，検索エンジンの検索結果やニュース一覧のページは，1 ページ中に複数の項目があるのでマルチプル・インスタンスと見なせる．図 1 (a) はシングル・インスタンスのページの例であり，図 1 (b) はマルチプル・インスタンスの例である．



(a) シングル・インスタンス



(a) マルチプル・インスタンス

図 1　シリーズ型 HTML 文書

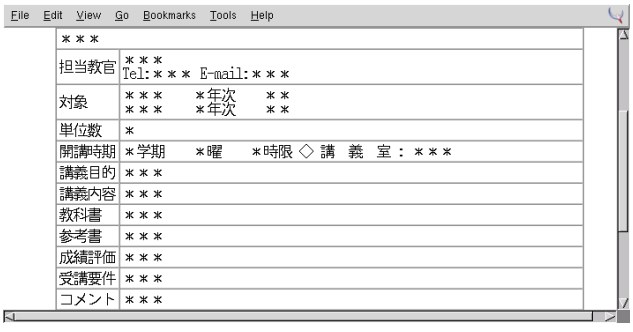


図 2　シラバステンプレート

テンプレートとは，同種の情報を同じ構造を用いて記述するための定型部分のことを指す．シリーズ型 HTML 文書は，テン

プレートの中にコンテンツを記述することにより表現される．例えば，図 2 は，図 1 (a) のテンプレート部分であり，“***” にコンテンツを挿入することにより，シリーズ型 HTML 文書が作成される．

3. 準　　備

アルファベット Σ から有限個の要素 $c_1, \ldots, c_k$ をとり，それらを並べて得られる列 $c_1c_2 \ldots c_k$ のことを Σ 上の**文字列**と呼ぶ．文字列 $w = c_1c_2 \ldots c_k$ の長さは k であるとし，これを $len(w) = k$ と表す．長さ n の文字列を n-**gram** と呼ぶ．

文字列 s に対して，$s = uvw$ を満たす文字列 u, v, w をそれぞれ，s の**接頭辞**，**部分文字列**，**接尾辞**と呼ぶ．文字列 s とその部分文字列 v の関係は反射的，推移的，反対称的な関係であり，順序である．この関係を $\leqq$ と表す．

ここで極大元の定義を示す．順序集合 U の要素 b にたいし $b \leqq x$ かつ $x \in U$ ならば $x = b$ となるとき，つまり U の要素で b よりも大きいものが存在しないとき b を U の極大元という．文字列を要素とし，$\leqq$ を順序とする集合の極大元を**極大文字列**と呼ぶ．

4. ジップの法則と部分文字列増幅法

共通のテンプレートで書かれたシリーズ型の Web 文書群において，文章として表されるコンテンツ部分の文字列はページ毎に異なるので，出現頻度は極端に低いが，テンプレートは入力文書群に共通する定型部分であるため，テンプレートを表す文字列の出現総数は少なくともその文書数以上になる．しかし，例えば「する」のように，一般の文章でも高頻出の単語もあり，単純に文字列の出現頻度だけで，テンプレート部分とコンテンツ部分の識別はできない．

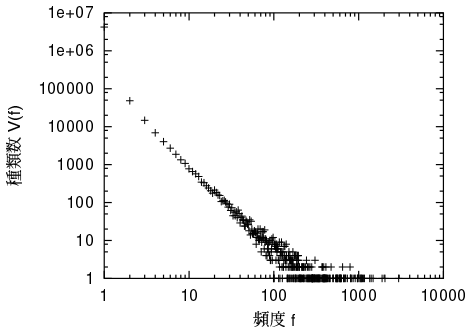
部分文字列増幅法 [5] では，まず，すべての長さの n-gram について，入力として与えられた文字列 (の集合) における出現頻度を求める．次に，頻度 f に対し，出現頻度が f である n-gram の種類数 $V(f)$ を求め，$F(f) = f \times V(f)$ により，頻度 f の全ての n-gram の出現総数を求める．f を変化させたとき，$F(f)$ が最大となる頻度 f が共通テンプレートを持つファイルの総数であると推定する．そして，その頻度を持つ n-gram が共通テンプレートを構成していると考える．

理論的解析は [5] を参照されたいが，部分文字列増幅法で得られるピークが共通テンプレートを持つシリーズ型文書の個数と一致することは，直観的には下のように理解できる．

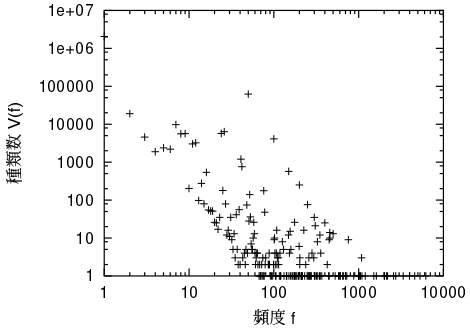
文書群が与えられたとき，部分文字列の頻度 f と，その頻度を持つ部分文字列の種類数 $V(f)$ をプロットすると図 3(注1) のようなグラフが得られる．

テンプレートを持たない自然言語文においては，多くの点が直線上に並び，この分布はベキ分布と呼ばれる．これは，自然言語文についてのジップの法則として知られている．ところが，同一のテンプレートを持つ半構造化文書群では，テンプレート

(注1)：長さ 1 から 30 までの部分文字列の頻度を全て数え，グラフでは両軸が対数軸となっている．



(a) 自然言語文におけるベキ分布



(b) 共通テンプレートの文書群でのベキ分布からの乖離

図 3　n-gram の頻度と種類数

を構成する文字列（その長さを n とする），ならびにその部分文字列（このような部分文字列は $O(n^2)$ 個存在する）が全て文書の個数だけ現れる．従って，文書の個数 f のところで，$V(f)$ は自然なベキ分布から乖離した点となる．

ベキ分布における f と $V(f)$ の関係は，$\log V(f) = b - a \log f$ あるいは $f^a \times V(f) = e^b$ と表すことができる．後者は，$f^a \times V(f)$ が頻度 f に依らず一定ということを表している．従って，ベキ直線からの乖離が，後者の表現では x 軸からの乖離として現れる．共通テンプレートを持つ文書群についての図 3(b) のデータに対して，簡単のため $a = 1$ として $F(f)$ のグラフを描くと図 4 のように，極端なピーク $F(f)$ を持つ頻度 f がある．最大のピークは入力ファイル数のところだが，その倍数の頻度のところでも，低いピークが現れている．

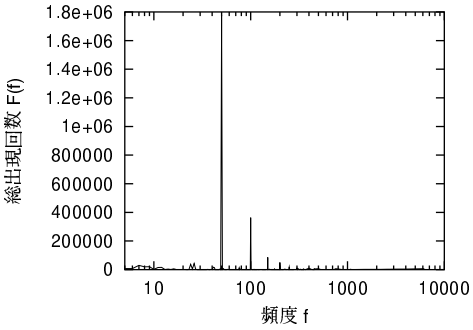


図 4　頻度 f と総出現数 $F(f)$

5. テンプレート抽出アルゴリズム

本節では，入力文書群からテンプレートを特定するためのテ

ンプレート抽出アルゴリズムについて述べる．アルゴリズムは，テンプレートを構成する部分文字列の集合を出力する．この集合の要素を**テンプレート要素**と呼ぶ．図 5 は，実装したテンプレート抽出アルゴリズムの擬似コードである．入力に対する前処理として，あらかじめ改行を削除した．

```
procedure Template_Detection
    (var D:  a set of documents):  set of strings
  var
    f, f_p:  integer;
    V, F:  ハッシュ表;
    T:  set of strings; /* テンプレート要素の集合 */
  begin
    V := Count(D); /* 部分文字列の頻度をカウント */
    for f ∈ keys(V)
        F(f) := f × |V(f)|;
    end;
    f_p := FindPeaks(F); /* 最大のピークを抽出 */
    T := Reconstruct(V(f_p)); /* 極大文字列の抽出 */
    report (T);
  end;
```

図 5　テンプレート抽出アルゴリズム

まず $Count(D)$ で，入力文書群 D に対し部分文字列の頻度をカウントする．V はキーに頻度，値にその頻度を持つ部分文字列の集合からなるハッシュ表である．テンプレート抽出アルゴリズムでは，長さ 1 から 10 までの部分文字列をカウントする．そして，それぞれの頻度に対し，$F(f) = f \times |V(f)|$ を求める．F はキーに頻度 f，値に $f \times |V(f)|$ を持つハッシュ表である．次に，$FindPeaks(F)$ において最大ピークとなる f_p を求める．ただし，頻度 2 以下にピークがある場合はそれを除外し，その次に大きい頻度を f_p とする．最後に，$Reconstruct()$ において頻度 f_p を持つ複数の文字列をまとめて極大文字列を作る．

6. 実　　験

本節では Web 上に存在する HTML 文書群に対し，前節で述べたテンプレート抽出アルゴリズムを適用した 2 種類の実験結果を述べる．実験に用いた計算機の仕様は，CPU: Pentium4 3.06GHz, 主記憶容量: 2GByte である．

初めの実験は，シングル・インスタンスを持つ文書群に対する実験として，国内の 4 つの大学から 10 種類のシラバスページ群を入力として与えた．各データに対し，あらかじめ同じテンプレートで記述されていることが分っている HTML 文書を 10 ファイル程度与え，それらのページ群の共通テンプレートを求めた．

2 番目の実験は，マルチプル・インスタンスを持つ文書群に対する実験として，23 通りの検索エンジンの検索結果からのテンプレート抽出を行った．この時，1 つの HTML ファイル中に繰り返し出現するテンプレートを発見する．

6.1　大学シラバスページ

シラバスとは，授業に関して担当教官，開講時期，講義内容等を記述した文書を指す．多くのシラバスは，1 文書中に 1 つの授業に関する情報を記述するシングル・インスタンスであるものが多く，1 つのフィールドに講義目的など情報の種類を表す属性名と，その内容を表す属性値との対を用いて記述されている．属性名はファイルに共通しており，属性値はファイル毎に異っている．このため，図 6 では左の丸く囲った部分が属性名，右の丸く囲った部分が属性値にあたる．

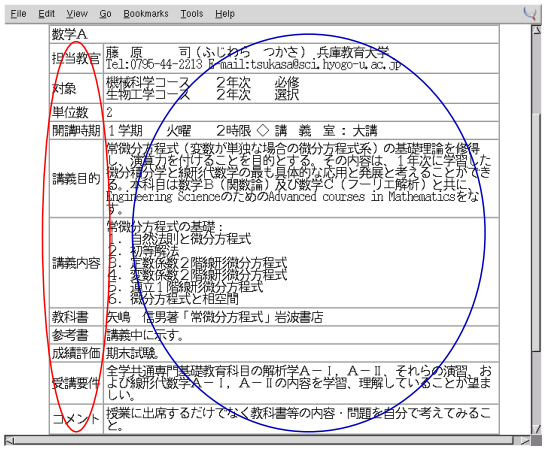


図 6　シラバスデータ

シラバスにおいて，属性名やそれらを表構造として記述する部分等，各々の文書に共通する枠組みがテンプレートにあたる．図 6 を記述している HTML ソースを図 7 に示す．＊＊＊の部分に属性値などの情報が埋め込まれてコンテンツとなり，あわせて一つのレコードをなす．

10 大学から 10 件程度のシラバスデータに対し，テンプレート特定アルゴリズムを適用したところ，ほとんどの実験において，ピークはファイル数と一致していた (表 1).

ID7 と ID8 のデータについては，ピークがファイル数と異なっていた．ID7 については，9 ファイルの内に，2 種類のテンプレートが存在し，それぞれを，7 ファイル，2 ファイルとして使われていた．そのため，7 ファイルのテンプレート上の部分文字列がピークに影響した．現在の実装では，最大の $F(f)$ をもつ f をピークとして定義している．このため，入力に複数のテンプレートが含まれた場合，ある 1 つのテンプレートしか特定することができない. [5] では，複数の異なるテンプレートを持つ文書群を入力として与えたとき，それぞれのテンプレートを持つ文書数と同じ f にピークが出現することが報告されている．よって，テンプレート特定アルゴリズムでは複数のピークを選択することで解決できると考える．

ID8 のデータで見つかった文字列は “> < /td>”, “> <td width=” と “padding:0mm 0mm ” の 3 種類だけだった．このデータはワープロ文書から生成された HTML 文書で，細かい表示設定などのためファイルサイズが大きかった．そのような細かい設定を記述する文字列が高頻度 (1 ファイル中約 1000 回) で現われ，その設定部分がテンプレートとして抽出されてしまった．

<table width=”600” border=”0” cellspacing=”0” cellpadding=”0”><tr><td><table border=”1” cellspacing=”0” cellpadding=”3” width=”100%”> <tr><td colspan=”2” class=”table-bg1”> ＊＊＊ </td></tr><tr> <td class=”table-bg2” nowrap width=”10%”> 担当教官 </td> <td width=”90%”> ＊＊＊
Tel:＊＊＊ E-mail:＊＊＊ </td></tr><tr> <td class=”table-bg2” nowrap> 対象 </td><td> ＊＊＊ ＊年次 ＊＊
 ＊＊＊ ＊年次 ＊＊ </td></tr><tr><td class=”table-bg2” nowrap> 単位数 </td> <td> ＊ </td></tr><tr><td class=”table-bg2” nowrap> 開講時期 </td> <td> ＊学期 ＊曜 ＊時限 ◇ 講 義 室 : ＊＊＊ </td></tr><tr> <td class=”table-bg2” nowrap> 講義目的 </td><td> ＊＊＊ </td></tr> <tr><td class=”table-bg2” nowrap> 講義内容 </td><td> ＊＊＊ </td></tr> <tr><td class=”table-bg2” nowrap> 教科書 </td><td> ＊＊＊ </td></tr> <tr><td class=”table-bg2” nowrap> 参考書 </td><td> ＊＊＊ </td></tr> <tr><td class=”table-bg2” nowrap> 成績評価 </td><td> ＊＊＊ </td></tr> <tr><td class=”table-bg2” nowrap> 受講要件 </td><td> ＊＊＊ </td></tr> <tr><td class=”table-bg2” nowrap> コメント </td><td> ＊＊＊ </td></tr></table> </td></tr></table>

図 7　シラバステンプレートの HTML ソースの一部

表 1　大学シラバスの実験結果

ID	URL	ファイル数	サイズ (KB)	ピーク	計算時間 (秒)
1	ccs.cla.kobe-u.ac.jp/System/syllabus.html	13	56	13	66.4
2	eduinfo.sci.hiroshima-u.ac.jp/syllabus/science/H9/	10	44	10	11.6
3	es6.es.osaka-u.ac.jp/sch/curri/syllabus/mechanical-science.html	11	92	11	271.1
4	hesvr.rche.kyushu-u.ac.jp/syllabus/sbdepart.cgi?p=2000.0&i=00	10	44	10	43.1
5	rihe.hiroshima-u.ac.jp/	11	68	11	74.6
6	siss.hiroshima-u.ac.jp/japanese/2002/sci/64012.html	11	56	11	124.4
7	www-old.es.osaka-u.ac.jp/syllabus/gs13/ims.ms/	9	50	7	21.6
8	www-old.es.osaka-u.ac.jp/syllabus/s14/be.htm	12	3476	12035	5177.2
9	www.chem.sci.kobe-u.ac.jp/syllabus/	11	48	11	33.0
10	www.econ.kobe-u.ac.jp/sirabas/siragk/03gk1	10	220	10	700.9

つまり，シングル・インスタンスのテンプレートでなく，マルチプル・インスタンスのテンプレートが抽出された．この実験では，シングル・インスタンスのテンプレートを見つけることを目的としているため，入力文書数より大きいピークを無視することにより，求めるテンプレートが特定できると考える．

次に，テンプレート抽出アルゴリズムによって抽出した部分文字列が，実際にテンプレート上に現われるかどうか，あらかじめ，人手で教官の氏名や授業内容などのコンテンツ部分と思われるところを目で見て取り除き，テンプレート部分を特定しておいたものと，アルゴリズムによって抽出したものを比較した．

ID1 では，

<!DOCTYPE HTML PUBLIC”-//W3C//DTD HTML4.01 Frameset //EN”><HTML lang=”ja”><HEAD><meta http-equiv=” Content=Type”content=”text/html; charset=Shift_JIS” ><TITLE> シラバス〜

のような人手で準備したテンプレート上に出現する部分文字列を抽出できた．また，このようなタグの部分だけでなく，“■授業のテーマと目標” のような文字列も抽出した．多くのデータにおいて，属性名はテンプレートの一部として抽出されていた．これらは，シラバスのメタデータと考えられる [3].

テンプレート抽出の問題点として，コンテンツ中の単語の頻度がピークと偶然に一致したため，テンプレートの一部として誤って認識された場合がある．逆に，テンプレートの一部が，コンテンツ部分にも現れ，そのためにファイル数より大きくなり，テンプレートとして認識されない場合もあった．

図 8 は，ID2 のデータに対して入力ファイル数を増やしていったときの，計算時間の変化である. [5] では，接尾辞木を用いて $O(n)$ 時間でテンプレート文字列特定ができることを示してしている．現在の実装は接尾辞木を用いないが，入力サイズに対し線形時間でテンプレート文字列が特定できるていることが分かる．

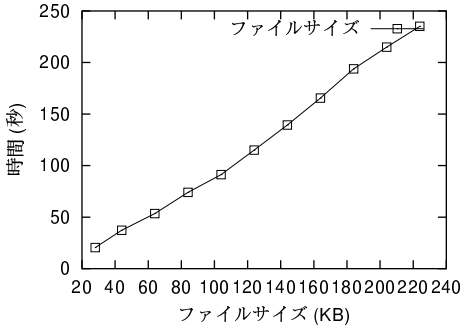


図 8　計 算 時 間

6.2　検索エンジンの検索結果

検索エンジンに検索語を入力して得られる HTML ファイルは，検索結果の一件一件が一つのレコードであるとみなすと，ヒットした件数だけレコードが繰り返し出現するマルチプルインスタンスである．

ここでは 23 種類の検索エンジンに百数十の検索語を入力し，検索結果の HTML ファイルを得た．検索結果のヒット件数が著しく小さい場合は，レコード数が少なくなり，繰り返し部分の出現頻度も小さくなるため，アルゴリズムが働きにくいと予想される．また，ヒット件数の小さい検索結果はファイルサ

表 2　検索エンジンの検索結果におけるヒット件数とピーク

ID	URL	ヒット件数	ピーク
1	www.clayzee.com/cgi-bin/mysearch.cgi	10,10,10	10,10,10
2	sagasu.jr.chiba-u.ac.jp:8080/cgi-bin/gaibu-osaka	20,20,20	20,20,20
3	search.yahoo.com/bin/search	5,7,2	16,16,15
4	www.ael.org/scripts/samples/search/query.idq	10,10,10	10,10,10
5	www.doe.mass.edu/search/search.asp	10,10,10	5,5,5
6	www.mcrel.org/resources/currdata/index.asp	36,30,21	35,29,20
7	www.edpolicy.gwu.edu/cgi-bin/texis/webinator/ncbe/sitesearch/	10,10,10	10,10,10
8	www.brassring.com/cgi-bin/texis/vortex/articlesearch	10,10,10	10,10,10
9	www.growingnd.com/search/search_results.asp	10,10,10	10,10,10
10	www.headhunter.net/JobSeeker/Jobs/JobResults.asp	25,25,25	12,12,12
11	www.jobs.com/JobSearch/Results.asp	25,25,25	12,12,12
12	search.yahoo.com/bin/search	20,20,20	20,20,20
13	search.news.com.au/cgi-bin/htsearch	10,10,10	10,10,10
14	www.newssynthesis.com/cgi-bin/apexec.pl	20,20,20	19,19,19
15	search.cnnfn.com/query.html	10,10,10	10,10,10
16	searchenginewatch.internet.com/cgi-bin/search/hyperseek.cgi	25,25,25	25,21,24
17	mentalhelp.net/mhn/htgrep9.cgi	50,50,50	48,49,49
18	search.yahoo.com/bin/search	3,4,1	16,13,12
19	www.ninds.nih.gov/find_people/search.htm	10,10,10	10,10,10
20	allhealthnet.com/cgi-bin/site_searcher.cgi	20,20,20	20,20,20
21	search.sandybay.com/searchpages	7,7,7	6,6,6
22	www.junkbusters.com/cgi-bin/search	58,56,50	57,55,49
23	www.osopinion.com/perl/search.pl	10,10,10	10,10,10

イズが小さくなる傾向があったため，検索エンジンごとにファイルサイズの大きい順に上位 3 ファイルに対して実験を行った (表 2).

HTML ファイルは文書の構造や体裁などの要素を定義する HTML タグ部分と通常の文章の部分にわかれている．これを利用し，前処理としてあらかじめ HTML タグ部分のみを取り出し，1 つのタグをアルファベット Σ の要素とみなして，HTML ファイルを Σ 上の文字列とした．

検索エンジンより得られた検索結果の HTML の一部を図 9 に示す．この検索結果ではヒット件数すなわちレコード数が 10

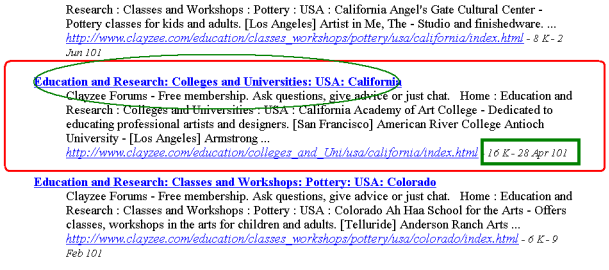


図 9　検索エンジンの検索結果

であった．ピークの頻度も 10 と一致した．テンプレートとして抽出した部分文字列は以下の 3 つであった．

"<P><DT>",
"<STRONG></STRONG></A></DT><DD>
<CITE>",
"</A><FONT SIZE=-1></FONT></CITE></DD>",

検索結果の HTML ソースファイルのなかから，一つのレコードにあたる部分を抜き出したものを図 10 に示す．これは図 9 において中央の四角い囲みに対応する．

ヒット件数と本手法で得られたピークの頻度の関連を分類し，表 3 に示す．

表 3　ヒット件数とピークの分類

ヒット件数とピーク f_p との関連	該当検索エンジン数
ヒット件数と一致	12
ヒット件数より 1 少ない	5
ヒット件数の半分	3
その他	3

- ヒット件数と一致 (1,2,4,7,8,9,12,13,15,19,20,23): 約半数の検索エンジンでヒット件数とピークが一致していた．標準的・簡素な検索結果の HTML であれば，マルチプルインスタンスの構造部分をとらえることができた．
- ヒット件数より 1 少ない (6,14,17,21,22): HTML ソースファイルにおいて，レコードの終わりの部分とレコードの始まりの部分が連続していたため，レコードとレコードが隣り合う部分が長い部分文字列となってヒット件数より 1 少ない回数だけ出現しており，これがピークを構成していた．
- ヒット件数の半分 (5,10,11): 検索結果を 2 色に塗り分けて表示してあるため，2 つのレコードで一つの繰り返しとしてとらえ，ヒット件数の半分でピークが現れた．
- その他 (3,16,18,): カテゴリ型検索エンジンが 1 種類あったが，カテゴリでのヒットを表示する部分と，サイトでのヒットを表示する部分が入れ子になっており，ヒット件数をとらえることが難しかった．

ヒット件数が 10 件未満と著しく小さかった検索エンジン 2

<P><DT><A HREF=”http://www.clayzee.com/education/colleges_and_Uni/usa/ california/index.html”><STRONG>Education and Research: Colleges and Universities: USA: California</STRONG></A></DT><DD>Clayzee Forums - Free membership. Ask questions, give advice or just chat.   Home : Education and Research : Colleges and Universities : USA : California Academy of Art College - Dedicated to educating professional artists and designers. [San Francisco] American River College Antioch University - [Los Angeles] Armstrong ...
 <CITE><A HREF=”http://www.clayzee.com/education/colleges_and_Uni/usa/ california/index.html”>http://www.clayzee.com/education/colleges_and_Uni/ usa/california/index.html</A><FONT SIZE=-1> - 16 K - 28 Apr 101</FONT> </CITE></DD>

図 10　レコード 1 件分の HTML ソース

種類では，ピークとヒット件数との関連性は見られなかった．

7. おわりに

部分文字列の頻度を数えることで，同一のテンプレートを持つシリーズ型 HTML 文書群からテンプレートを特定するアルゴリズムを提案した．実験では，シングルインスタンス，マルチプルインスタンスの文書ともに，アルゴリズムによって抽出した文字列がテンプレート上に出現する文字列であることを確認した．

7.1 今後の課題

実験において，テンプレート部分がどのくらい特定できたか，あらかじめテンプレート部分を特定しておいたものと，アルゴリズムによって抽出したものを人手で比較したが，再現率や精度，適合率を用いて数値で評価を行なうことが今後の課題として挙げられる．

他にテンプレートを特定後，テンプレートに挟まれる部分を抽出することによりシリーズ型 HTML 文書から情報の抽出を行なうことが今後の課題である．また，この研究は，Web シラバス情報収集エージェント [10] における，シラバスページからの情報抽出に応用する予定である．

本論文において実験に使用したプログラムは，計算機の主記憶容量を大量に消費してしまうため，入力データのサイズが 10MB 弱に制限されてしまった．さらに多くのシリーズ型文書群やそのほか大きなデータを扱うためには，より精巧で緻密な実装を要する．

文　献

[1] Z. Bar-Yossef and S. Rajagopalan, Template Detection via Data Mining and its Applications, Proc. of *11th International World Wide Web Conference*, 2002.

[2] V. Crescenzi, G. Mecca and P. Merialdo, ROADRUNNER: Towards Automatic Data Extraction from Large Web Sites, Proc. of *27th International Conference on Very Large Data Bases*, pp. 109–118, 2001.

[3] S. Hirokawa, E. Itoh, T. Miyahara, Semi-Automatic Construction of Metadata from A Series of Web Documents, Proc. of *16th Australian Joint Conference on Artificial Intelligence*, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2903, pp. 942–953, 2003.

[4] D. Ikeda, Y. Yamada and S. Hirokawa, Eliminating Useless Parts in Semi-structured Documents using Alternation Counts, Proc. of *the Fourth International Conference on Discovery Science*, Lecture Notes in Artificial Intelligence, Vol. 2226, pp. 113–127, 2001.

[5] 池田大輔, 山田泰寛, 廣川佐千男, 部分文字列増幅法による共通パタン発見アルゴリズム, 情報処理学会論文誌「数理モデル化と応用」, 2003. (採録決定)

[6] N. Kushmerick, D. S. Weld and R. B. Doorenbos, Wrapper Induction for Information Extraction, Intl. Joint Conference on Artificial Intelligence, pp. 729–737, 1997.

[7] A. H. F. Laender and B. A. Ribeiro–Neto, A Brief Survey of Web Data Extraction Tools, ACM SIGMOD Record, Vol. 31, No. 2, pp. 84–93, 2002.

[8] 梅原雅之, 岩沼宏治, 永井宏和: 事例に基づく HTML 文書から XML 文書への半自動変換, 人工知能学会論文誌, Vol. 16, No. 5, pp. 408–416, 2001.

[9] 梅原雅之, 岩沼宏治, 鍋島英知: 事例に基づくシリーズ型 HTML 型文書の意味論理構造の自動認識 人工知能学会論文誌, Vol. 17, No. 6, pp. 690–698, 2002.

[10] 山田信太郎, 松永吉広, 伊東栄典, 廣川佐千男, Web シラバス情報収集エージェントの試作, 電子情報通信学会論文誌, J86-D-I, No. 8, pp. 566–574, 2003.

[11] Y. Yamada, D. Ikeda and S. Hirokawa, Automatic Wrapper Generation for Multilingual Web Resources, Proc. of the 5th International Conference on Discovery Science, Lecture Notes in Computer Science, Springer-Verlag, Vol. 2534, pp. 332–339, 2002.

[12] 山田泰寛, 池田大輔, 廣川佐千男, 構造的類似性を持つ半構造化文書における頻度分析, 第 2 回情報科学技術フォーラム, 一般講演論文集第 2 分冊, pp. 59–60, 2003.