情報がWeb上にいつ現れたかの発見

井上　真大†　　田島　敬史††

† 京都大学工学部情報学科　〒606–8501 京都府京都市左京区吉田本町
†† 京都大学大学院情報学研究科　〒606–8501 京都府京都市左京区吉田本町
E-mail: †inoue@dl.kuis.kyoto-u.ac.jp, ††tajima@i.kyoto-u.ac.jp

あらまし　本論文では，与えられた情報についてその情報がWeb上に最初に現れたのがいつかということを発見する手法を提案する．我々はこの問題を，Webページが与えられた情報を含んでいるかの判定，Webページに明示されたタイムスタンプの検出，与えられた情報を含むページ群のタイムスタンプの分布の分析による初出となるページの推定の三つの段階に分けて考えた．Webページが与えられた情報を含んでいるかの判定に関しては，ドキュメントのセンテンスの包含度という従来の類似度とは異なる新たな尺度を定義し，これに基づいた手法を開発した．また，与えられた情報を含むページ群のタイムスタンプの分布の分析に関しては，分布から情報の種類を判定し，その種類を元に初出となるページを推定する手法を開発した．

キーワード　情報検索，時系列分析

1. はじめに

近年，blogやwikiなどの，HTMLの知識なしに情報をWeb上に発信できるサービスが台頭し，Webを利用する誰もが，特別な知識や技術無しに自分の持てる知識・情報を公開できるようになった．更には，twitter(注1)に始まるより気軽で手軽な情報の発信手段も現れ，今後ますます様々な個人による情報発信が活発になることが予想される．現在では，Webは，世間一般に情報を最も簡単かつ最も早く公開できる場となっていると言える．

しかし，その様な情報発信の個体の増加は同時にWeb上における情報の重複をますます加速させている．実際，現在のWeb上では多くの重複した情報が様々な場所で発信されており，この様な同じ情報の氾濫は，Web上における情報源の特定や，Web上での情報の発生時期の把握を難しくさせている．Web上における情報の発生時期の把握は，Webが最も早く世間一般に情報を公開できる場であることを考えると，世間一般の人々がその情報を知り得た最も早い時期の発見とも言えるため，重要である．しかしながら，現在では，例えば，ある情報を初めにWeb上に公開したページはどこかということや，今となっては有名だが最近までは聞かなかった言葉や意見がWeb上に出現したのはいつか，ということを知ることは困難である．

そこで，本論文では，ある情報に対して，その情報が最初にWeb上に出現したのはどこでありいつであるのかを発見する手法を提案する．本研究では，前提としてWebページが情報発信の最小単位であるとし，また，クエリとしては，複数の異なった情報が含まれているものは考えず完結した一つの情報となっているものを考え，特に，それがセンテンスで表されると仮定している．

我々は，この情報のWeb上の初出発見問題を解決するにあたって，まず情報のWeb上における出現の時系列を取得した．

情報のWeb上における出現の時系列の取得は，Webページを情報発信の最小単位として仮定しているので，Webページが与えられた情報を含むかどうかの判定をする問題と，Webページの公開された日時を取得する問題の2つの問題に分けることができる．本研究ではクエリとして，センテンスで表された情報を取るので，Webページがクエリが示す情報を含むかどうかの判定をする問題に関しては，Webページをドキュメントと見なし，ドキュメントのセンテンスの包含度という尺度を定義し，これに基づいて判定を行う手法を開発した．また，本研究では，Webページ中に明示された公開日や投稿日などのタイムスタンプを，Webページの公開された日時と見なし，Webページ中に明示されたこれらのタイムスタンプを検出する手法を開発した．

次に，初出を求めるにあたって，ここで得られた時系列の最初のタイムスタンプを取ることにより求めることをまず考えたが，完全に正しい時系列を得ることは難しく得られた時系列にはしばしば誤ったタイムスタンプが含まれていたため，単純に時系列の最初を取るのでは明らかに誤ったタイムスタンプをも選択してしまうことがあった．そこで，その様な問題を解消するために，単純に時系列の最初を取るのではなく，時系列全体を分析することによって情報の種類を推定し，それに基づいて正しい初出のタイムスタンプを推定する手法を開発した．

時系列に含まれる個々のタイムスタンプはそれぞれWebページから検出されたものであるので，初出と判定されたタイムスタンプが得られれば，同時に初出の場所，つまり初出のWebページも得ることができる．

本研究では，与えられたクエリでWeb検索して得られた上位100件の結果をデータセットとし，各手法を次のように評価した．まず，Webページの情報包含判定の手法は，これによって得られる包含度でデータセット内のWebページをランク付けしMean Average Precision (MAP) を用いて評価し，Web

(注1)：http://twitter.com/

ページ中に明示されたタイムスタンプの検出の手法は，その精度を求めることによって評価した．また，そのデータセットの中でクエリが示す情報を含んでいる最古の Web ページの日付を人手で導出し，正解の日付として，それと，実験によって最終的に得られた日付とを比較，評価し，時系列の分析を含めた 3 つの手法全体の評価とした．

以降，第 2 節で関連研究について述べ，第 3 節で Web 上における情報の出現の時系列を取得する手法について，第 4 節で得られた時系列から初出のタイムスタンプを判定する方法について説明し，第 5 節で実験結果とその評価を示す．そして，最後に第 6 節で結論を述べる．

2. 関 連 研 究

Web 文書はコンピュータ上で簡単に複製できることもあって，それらの全部或いは一部が再利用されたり，様々な Web 文書を複製して作られるいわゆるスパムサイトも存在している．このような Web 文書の重複や再利用を発見する研究は広く行われており [1], [2]，通常こういった問題には，文書同士，文同士の類似度という尺度が用いられ，これらの類似度を測定する手法としていくつかの手法が研究されている [1]〜[7].

通常ドキュメントは複数の事実や知識を含むため，ドキュメントの複製をその度合いで大まかに次の四つのタイプに分けることができる [1]．即ち，(1) 少しの言葉の違いや変更を除いて全く同じである，(2) 多くの，特にあまり一般的でない情報を共に含んでいる，(3) 少しの同じ情報を共に含んでいる，(4) 全く違う，の 4 つである．

[1] では，まずセンテンスとセンテンスの類似度を測る手法を列挙し，二つのドキュメントに含まれる各センテンス同士の類似度を測定することによってドキュメント間の類似度を測定する手法を提案している．また，単語の出現確率や単語同士の共起の確率に注目することにより，より高い精度で類似度が測定できる [4]. [2] では，Web 文書の再利用を検出することを目的として，既知のいくつかのドキュメント間の類似度測定の手法を比較している．

ドキュメントの再利用の検出については，このようにドキュメント同士の類似度という尺度によって行われるべきであるが，本研究では，センテンスで表されたある情報をドキュメントが含んでいるかどうかという判定を行いたいため，ドキュメントのセンテンスの包含度という新たな尺度を考えた．

Web 上の情報の初出発見と似た種類の問題として，情報の新規発見という問題があり，これも幅広く研究されている [8]〜[10]．新規情報の発見は，Web を情報のストリームとして捉え，そこから出力された今まで出力されなかった新しい情報をいかに効率よく正確に発見するかという問題であり，既に Web 上に存在する特定の情報についてその情報がいつ現れたかを発見しようとする本研究とは，発見か探索かという面で異なっている．

[11] では，情報ではなく Web ページに焦点を当て，Web ページをクロールし，新たにクロールされた Web ページの中でどれが本当に新しい（新たに作成された）Web ページかを Web ページのリンク構造を活用することにより発見しようとしている．

3. Web 上における情報の出現の時系列の取得

Web 上における情報発信には実に多種多様な形態が存在し得る．つまり，発信するメディアとしてはテキストや音声，動画が存在し，発信される場所も新たな URI が与えられて発信されることもあれば，既にある Web ページに追加，更新の形で発信されることもある．

しかし，昨今における情報発信では，しばしば blog に始まる Content Management System (CMS) が用いられ，新たな URI を持つ 1 つの Web ページとして発信されることが普通である．また，動画などのテキスト以外のメディアが用いられる場合，テキストだけでは表現できない視覚的な情報を補完するために用いられたり，そうでなくとも通常，動画の内容を示すテキストを伴って現れる．

そこで，本研究では，Web ページを情報発信の最小単位と仮定し，Web ページ内のテキストにのみ着目する．この時，Web 上における情報の出現の時系列の取得は，Web ページが与えられたクエリが示す情報を含むかどうかの判定をする問題と，Web ページの公開された日時を取得する問題の 2 つの問題に分けて考えられる．

3.1 Web ページの情報包含の判定

本論文では，クエリとして与えられる情報は，何かしらの一つの事実や意見を表すような一つの完結した情報であるとしており，一つのセンテンスで表し得るものであると仮定している．そして，実際に本研究ではクエリはセンテンスの形で与えるものとしている．

また，探索の対象である Web ページについてはテキスト部分全体を取り出しドキュメントとして扱っている．理想的には，Web ページの主たる内容の部分だけを取り出しそれをドキュメントとして扱うべきだが，その抽出を完全な精度で行うことは今現在できないこと [12]〜[14]，また本論文が提案する手法では，後述するように，Web ページのメニューや広告などのノイズとなる部分の影響を受けにくいことにより，ここでは Web ページからの内容抽出については考えない．

ここでいうセンテンスやドキュメントとは，生の文や文書ではなく，扱う上でそれらにいくつかの適切な処理を施したものを意味している．具体的には，文はまず空白，句点，ハイフンで区切り単語に分割し，各単語を小文字に置き換え，ステミングを行い，ストップワードを取り除き，単語の並びとして見たものをセンテンスとし，同様にドキュメントは，文書をセグメンテーションし文に分割し，各文に処理を行いセンテンスとし，センテンスの並びとして捉えたものの事である．

こうした時，Web ページのクエリ情報の包含を判定するこの問題は，ドキュメントがクエリセンテンスの情報を含むかという問題に帰着できる．この問題に対し，ドキュメントのクエリセンテンスの包含度という新たな尺度を考えた．

こういった，ドキュメントとドキュメントの比較やドキュメントとセンテンスの比較には tf-idf [6] や，query likelihood [7] などの類似度と呼ばれる尺度が一般的に用いられるが，本問にお

いては類似度という尺度を用いるのは不適切である．なぜなら，類似度というものは本質的には比較するもの同士がどれほど同じであるかを測る尺度であり，片方が片方を完全に包含していたとしても，包含する側に余分な情報が含まれていれば，それに応じて類似度は低下してしまう，あるいはするべきである．また，類似度においては，similarity(a, b) = similarity(b, a) という対称律も成り立っているべきである．

しかし，本問においては，比較するものが包含側と被包含側に明確に立場で分離でき，また包含側が被包含側を完全に含んでいるのであれば，包含側にどれほど余分な情報が含まれていようとも，包含度は最大であるべきである．こういった理由から，本問においては包含度という尺度を考え，それを計算するために 2 つの手法を考案し比較した．

1 つは，クエリセンテンスとドキュメントを構成する各センテンスとの word overlap を求め，その最大値を包含度とする手法である．ここで word overlap とはクエリセンテンスと，対象となるセンテンスに共に含まれる単語数（同じ単語が両センテンスに複数存在していれば，重複してカウントする）を，クエリセンテンスの長さ，つまりクエリセンテンスに含まれる単語数で割ることにより求められる値である．明白な欠点として，この手法は，クエリセンテンスが示す情報がドキュメントの複数のセンテンスにまたがって記述されている場合にうまく計算ができないという欠点を持つ．

そこで，もう 1 つの手法では，クエリセンテンスが示す情報が複数の文にまたがって記述されている可能性を考慮し，クエリセンテンスに含まれる単語で，クエリセンテンスとの word overlap が最も高いセンテンスに含まれない単語がドキュメントの他のセンテンスにあれば，そのセンテンスと word overlap 最大のセンテンスとの距離に指数的に反比例させた値を包含度に加算していくという手法をとる．

これら 2 つの手法を順に，MWO (Maximum Word Overlap)，EMWO (Extended Maximum Word Overlap) と呼び，EMWO のアルゴリズムを擬似コードを用いて Algorithm 1 に示す．

この時，ドキュメント *doc* がクエリセンテンス *sen* が示す情報を含んでいるかどうかを，ドキュメントのセンテンスの包含度を計算する関数 f から得られた包含度が，閾値 t 以上であるかどうかで判定するとすると，この関数は f と t の選び方に依存し，include$_{f,t}(doc, sen)$ の形で表せる．

3.2 Web ページ中に明示されたタイムスタンプの検出

Web ページには大まかに分けて，サイトのトップページや blog サイトの投稿の一覧のページのような URI が固定で内容が刻々と変化するページと，ニュースサイトにおける個々の記事のページや blog サイトにおける個々の投稿のページのように，その情報発信に対して新たな URI が与えられその内容が変わることは少ないページの 2 種類が存在する．

前者の Web ページを動的なページ，後者の Web ページを静的なページと呼ぶとすると，本研究では，情報発信の最小単位として Web ページを考えており，ページの一部の修正や追加などより細かい単位での情報発信については扱わないことにしているので，1 つの Web ページ内で刻々と内容が変化する動的なページは評価の対象に含めない．昨今においては情報発信は専ら静的なページによって行われるため，動的なページを対象としないことによる損失は少ない．

さて，Web ページの公開された日時の取得について考えたとき，これを一般的に得る方法は無く，HTTP ヘッダには最終更新日時を示す Last-Modified フィールドが存在するが，これの送信は強制ではなく使用されていなかったりアクセスに応答し

Algorithm 1 EMWO

```
// Doc: target document
// Doc[n]: the n-th sentence of Doc
// L_Doc: the number of sentences in Doc
// Sen_q: query sentence
// L_Sen_q: the number of words in Sen_q
// WordOverlap(s, t): calculate word overlap of two sentecnes

// examine the highest word overlap
val ← 0
for i = 1 to L_Doc do
  if WordOverlap(Doc[i], Sen_q) > val then
    val ← WordOverlap(Doc[i], Sen_q)
  end if
end for

if val > 0 then
  ret ← val
  for i = 1 to L_Doc do
    if WordOverlap(Doc[i], Sen_q) eq val then
      tmp ← val
      for word in Sen_q do
        if not Doc[i] includes word then
          j ← 1
          while i − j ≧ 1 or i + j ≦ L_Doc do
            if (i − j ≧ 1 and Doc[i − j] includes word) or
            (i + j ≦ L_Doc and Doc[i + j] includes word)
            then
              break
            end if
            j ← j + 1
          end while
          if i − j ≧ 1 or i + j ≦ L_Doc then
            tmp ← tmp + 2^−j / L_Sen_q
          end if
        end if
      end for
      if tmp > ret then
        ret ← tmp
      end if
    end if
  end for
  return ret
else
  return 0
end if
```

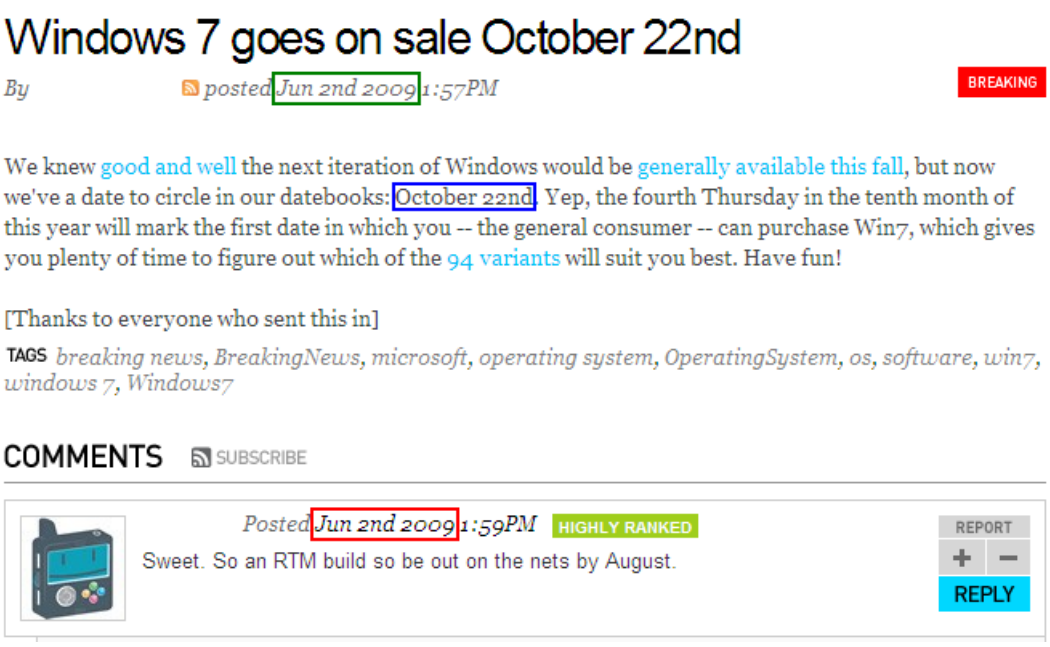

図 1　Web ページに出現する日付表現

た日時が入っていたりと正確な日時が示されていない場合が殆どである．そこで本論文では，Web ページ中に明示されている公開日時や投稿日時などのタイムスタンプを，Web ページの公開された日時と見なし，Web ページ中に明示されたそれらのタイムスタンプを検出する手法を開発した．

この，タイムスタンプの検出は同時に Web ページを動的なページと静的なページに分離することにも繋がる．なぜなら，動的なページにおいてはその性質によりページのタイムスタンプの意味合いが薄く，タイムスタンプが明示されていることは少ないが，一方で静的なページにおいては，blog システムに始まる CMS を用いて作成されることが一般的であり，これらは自動的に投稿日時を付加してくれるため，タイムスタンプが明示されていることが多く，タイムスタンプが明示されている Web ページに着目すれば，それが自然と静的な Web ページに着目することに繋がるからである．

さて，Web ページ中に明示されたタイムスタンプの検出をするためには，タイムスタンプが記されていない Web ページも存在するため，まず Web ページが明示されたタイムスタンプを持っているかどうかを判断する必要がある．しかし，これを Web ページを見て機械的に判断することは難しいため，我々は Web ページ中に現れる全ての日付表現を見て，十分にタイムスタンプらしいと言える日付表現があればその Web ページには明示されたタイムスタンプがあるとして，タイムスタンプらしいと言える日付表現がなければその Web ページは明示されたタイムスタンプを持たないとして判断することを考えた．

これを実現するために，我々は，日付表現がタイムスタンプらしいと考えられる要因や逆にタイムスタンプらしくないと考えられる要因を列挙し，これらに基づく評価関数を設計し，各 Web ページ中に現れる全ての日付表現の中で最も高い評価値を持つ日付表現を選び，その評価値が事前に決めた閾値を越えていたならばそれをその Web ページのタイムスタンプと見なし，閾値より低ければその Web ページ中には明示されたタイムスタンプが存在しないと見なすという手法をとった．

日付表現としては，“Dec 23rd, 2009”, “2009/12/23”, “09.12.23”, “12/23/09” など通常日付として使われる殆ど全ての表現を考慮しており，その抽出には正規表現を用いた．

さて，我々が使用したタイムスタンプらしさを示す要因を以下に列挙する．

a. posted や published など投稿を表す単語が近くに存在する
b. URL に年月（日）が含まれていて，それと一致する
c. HTML タグで直接囲まれている
d. 時刻表現が近くにある
e. 他の日付表現と違う形の表現である
f. 文書の前の方に現れる

文書中に現れる日付表現は殆ど全てが図 1 に示す 3 つの種類のいずれかに振り分けることができる．つまり，“Windows 7 is released on October 22nd ...” の様に文章の一部として現れるもの（図 1 で青で囲まれたもの）と，タイムスタンプを表しているが Web ページ自体のタイムスタンプではないもの（図 1 で赤で囲まれたもの）と，我々が検出しようとしている，Web ページ自体のタイムスタンプを表すもの（図 1 で緑で囲まれたもの）の 3 つである．

タイムスタンプを表している日付表現とそうでない，つまり文章中に現れるような日付表現を判別するのは比較的容易で，例えば，タイムスタンプを表している日付表現はそれ自身が意味を持っており孤立していることが多く，日付表現が HTML タグで直接囲まれて存在していることが多い．また，文章中に現れるような日付表現は単に日付だけを示し，時間まで示す，

つまり時刻情報を伴って現れることは稀であるが，タイムスタンプを表す日付表現は時刻情報を伴なって現れることはままある．これらの考えが，要因 c と要因 d の根拠となっている．

一方でタイムスタンプを表している日付表現に対して，それが Web ページ自体のタイムスタンプであるかどうかを判別することは難しく，経験的に要因 a,e,f などの要因を用いている．つまり，Web ページ自体のタイムスタンプは Web 文書の初めの方に現れることが多いこと，Web ページ自体のタイムスタンプは強調の意味も込めて Web ページ内の他の日付表現とは違った日付表現を用いること（Web ページ自体のタイムスタンプは”Nov. 23, 2008” で表されるが，他の日付表現は”08.11.23” など簡略な表現となっている等）がしばしば見られたことなどに拠っている．

上述した要因以外にも，日付表現より上位のタグで class 属性に “comment” を含むものがあればコメントらしいので減点するなどといった細々としたヒューリスティックもいくつか用いており，我々は，これら要因に実験的，経験的に点数付けを行い，その和として評価関数を定義した．

この時，Web ページ w のタイムスタンプを検出する関数は，ここで計算された評価値に対する閾値 t に依存し，$\mathrm{timestamp}_t(w)$ の形で表せ，この関数は Web ページ w に明示されたタイムスタンプがあるならばそれを返し，なければ null を返すものと定義する．

3.3 Web 上における情報の出現の時系列

最終的に我々は明示されたタイムスタンプを持つ Web ページにのみ着目し，これらに対しクエリセンテンスを包含しているかどうかの判定をする．この時，Web 上における情報の出現の時系列は以下の様に定式化でき，上で定義した 2 つの手続きを用いることによりこれを求めることができる．

> $\mathcal{W}$ を Web ページ全体の集合とすると，手続き $\mathrm{timestamp}_t$ を用いて，タイムスタンプを持つ Web ページの集合 $\tilde{\mathcal{W}}$ は次の様に表せ，
>
> $$\tilde{\mathcal{W}} = \{w \mid w \in \mathcal{W} \wedge \mathrm{timestamp}_t(w) \neq \mathrm{null}\}$$
>
> クエリ情報 q の出現の時系列 $\mathrm{TL_q}$ は，手続き $\mathrm{include}_{f,\,t}$ を用いて，以下の様に表せる:
>
> $$\mathrm{TL_q} = \{\mathrm{timestamp}_t(w) \mid w \in \tilde{\mathcal{W}} \wedge \mathrm{include}_{f,\,t}(w,\,q)\}$$

本研究の手法では，以前存在していたが既になくなってしまっている Web ページまでを考慮に入れることは出来ない．これに際して，Web アーカイブなどの仕組みを利用することも考えたが，現在実在する Web アーカイブでは，Web ページの網羅度が完全でないこと，Web ページの保存する間隔が大きいこと，最近の Web ページは閲覧できないことなどにより，結果として利用を見送った．

4. 情報の出現の時系列からの初出判定

情報の出現の時系列から初出のタイムスタンプを手に入れるには，理論的には時系列において単純に一番過去のタイムスタンプを選択すれば良いだけである．しかしながら，実際に前節のステップを経て得られる時系列には，クエリが示す情報を含んでいないのに含んでいると誤って判定された Web ページのタイムスタンプや，Web ページから誤って抽出されたタイムスタンプなどの誤ったタイムスタンプがしばしば含まれていた．もちろん，誤ったタイムスタンプが含まれないように精度重視で時系列を求めることも可能ではあるが，それによって正しい初出のタイムスタンプが失われてしまっては意味がなく，できる限り再現率も上げなければならない．従って，この様なノイズとも言えるタイムスタンプが時系列には含まれ得るということを考慮して，初出判定を行う必要がある．

初めに，情報には，普遍的に正しい情報や Web が存在する以前から存在していた情報と，比較的最近（少なくとも Web ができてから）のある時点に発生し存在している情報の 2 種類あることに注意する．前者の例としては「氷が溶けると水になる」や「1600 年に関ヶ原の戦いが行われた」が挙げられ，後者の例としては「イチローがオールスターで MVP に選ばれた」や「Microsoft が Windows 7 を発売する」が挙げられる．後者のような情報は，何かしらのイベントの発生により生じる情報であり，これをイベント情報と呼ぶ．イベント情報は，その性質上イベントの発生以前には知り得ず，従ってクエリが示す情報がイベント情報である場合，そのイベントのおおよその発生日時が分かればその日時より過去のタイムスタンプは破棄できる．

ここで注意すべき事として，ここで言うイベントとは，その情報が世間一般に知らされるきっかけとなった出来事のことを指しており，情報が示すイベントとは必ずしも一致しない．例えば,「イチローがオールスターで MVP に選ばれた」や「××で地震が発生した」などの情報は，情報が示すイベントが発生して初めて周知される情報であるが,「Microsoft が Windows 7 を発売する」という情報は，Microsoft が Windows 7 を発売するその前から知られていた情報であり，情報が示すイベントと情報が周知されるきっかけとなったイベントは異なっている．

さて，我々は，様々なクエリについて実験することにより，イベント情報を示すクエリから得られる時系列はそのイベントの発生日時付近においてしばしば顕著に山となることを発見した．これは，情報において速報性は一つ重要なファクターであることを考えればごく自然な結果であると言える．

そこで，我々は，与えられたクエリから得られた時系列を分析して有意な山が存在し，その山より過去にタイムスタンプが殆ど存在しなければ，与えられたクエリがイベント情報を示すクエリだと仮定し，その山の最も古い麓のタイムスタンプを初出のタイムスタンプとして判定した．そして，イベント情報を示すクエリでないと判定されたクエリについては，任意の時期においてクエリが示す情報が Web 上に出現し得るので，単純に最も古いタイムスタンプを初出のタイムスタンプとして判定した．

ここで，有意な山の判定とその山より過去にタイムスタンプが殆ど存在しないという二つの判定が必要であるが，本研究では，様々なクエリで実験した結果に基づき，ある日付の Web

クエリ	全て	クエリ情報を含む	明示されたタイムスタンプを持つ	タイムスタンプを持ちクエリ情報を含む
Ichiro Suzuki was named All-Star Game MVP	100	68	57	45
Windows 7 is released on October 22nd	100	93	62	60
street view invades privacy	100	66	87	61

表 1　本研究で用いたクエリと，それらのデータ（数字は Web ページ数を表す）

ページの数が時系列に含まれる全 Web ページ数の $\frac{1}{10}$ 以上であれば有意な山であると判定し，この日付から，その日付における Web ページが存在する限り日付を一日ずつ過去にしていった結果の日付を山の麓と定義し，その山の麓の日付より過去の日付の Web ページの数が時系列に含まれる全 Web ページ数の $\frac{1}{10}$ 以下であればその山より過去のタイムスタンプは殆ど存在しないとして判定した．

5. 実　　験

前節の結果より，真に正しい初出の日時を手に入れるためには全 Web ページを網羅する必要があるが，それは現実的には到底不可能であり，本研究では，与えられたクエリを Yahoo![注2] の Web 検索 API である Yahoo! Search BOSS[注3] を用いて Web 検索した上位 100 件の結果をデータセットとして用いた．

少なくともイベント情報に関しては上位 100 件を見れば初出の日付を導出するのに十分であることが，実験的に判明している．これは，イベント情報を即座に発信するようなサイトは往々にして有用なサイトであり，検索ランキングにおいて上位に現れがちであることを考えれば，自然である．

そして，我々は上位 100 件のそれぞれの Web ページについてクエリが示す情報を含んでいるかの判定と，明示されたタイムスタンプの検出を人手で行い，これを正解データとした．

表 1 に，本研究で用いたクエリとそのデータを示す．各クエリは次のような意図が根底にあり選ばれている．

Ichiro Suzuki was named All-Star Game MVP
イベント情報であり，クエリが示すイベントの発生によって知られる情報
Windows 7 is released on October 22nd
イベント情報であり，クエリが示すイベントの前から知られている情報
street view invades privacy
イベント情報でなく，任意の時期において発生し得る情報

Web ページの情報包含判定の手法は，それによって得られた包含度で各クエリのデータセットを包含度最大のものからランキングを行い，以下の式で定義される Mean Average Precision (MAP) を用いて評価した．

> あるクエリに対して，正解データの数が N であり，上から i 番目のランクの正解データのランクを $rank_i$ で表すとすると，Average Precision (AP) は以下のように表せ，
>
> $$\mathrm{AP} = \frac{1}{N}\sum_{i=1}^{N}\frac{i}{rank_i}$$
>
> 各クエリに対して求めた AP の平均 (mean) が MAP である．

AP の値は，N 個の正解データがちょうど上位 N 個にランキングされた時に 1 となり，0 から 1 の値をとる．

検索結果のランキングをそのまま用いたものを baseline として，MWO と EMWO のそれぞれで包含度を求めてランキングした場合の MAP の値を表 2 に示す．

	baseline	MWO	EMWO
MAP	0.821	0.894	**0.896**

表 2　情報包含判定の手法毎の MAP

表 2 より，MWO を使った場合，MAP は baseline より約 9%向上することが分かるが，EMWO は MWO から約 0.2%の向上と殆ど変わらないことが分かる．しかしながら，これは，EMWO が他のセンテンスを考慮することによって MWO と殆ど評価値が変わっていないためではなく，EMWO が他のセンテンスを考慮することによって，適切に評価値が上げられている Web ページもあれば，違うコンテキストで違う意味になってしまっている他のセンテンスの単語を見てしまうことにより，誤って評価値を上げてしまっている Web ページも存在するので，これら二つの結果が相殺しあってしまっているためである．

特に，“Ichiro Suzuki was named All-Star Game MVP” のクエリにおいては，EMWO が MWO より低い値となってしまったが，これは “MVP” という単語が，違うコンテキストで，つまり別の “MVP” を指して存在している事が多かったためである．EMWO の手法に関しては，今後改良の余地があると言える．

次に，Web ページ中に明示されたタイムスタンプの検出の手法の評価について述べる．まず，Web ページからタイムスタンプを検出する際，以下の 3 つの誤りが生じ得ることに注意する．

1. タイムスタンプのある Web ページで誤ったタイムスタンプを検出する
2. タイムスタンプのない Web ページで誤ったタイムスタンプを検出する
3. タイムスタンプのある Web ページでタイムスタンプがないと判定する

(注2)：http://www.yahoo.com/
(注3)：http://developer.yahoo.com/boss/

	$t=40$	$t=50$	$t=60$
EV1	19.3	**16.3**	17.7
EV2	32.0	**26.0**	26.3

表 3　タイムスタンプ検出手法の閾値毎の評価値

クエリ	正解の日付	出力された日付	イベント情報と判定
Ichiro Suzuki ...	2007/07/10	2007/07/10	yes
Windows 7 ...	2010/11/11	2011/11/11	yes
street view ...	2007/06/01	2007/06/01	no

表 4　クエリ毎の正解と最終的な結果

ここで，上記 1. と 2. の誤りは誤ったタイムスタンプを出力し時系列においてノイズとなる分，3. の誤りと比べて重い誤りであると言える．そこで，我々はこれら 3 つの誤りについて，全て等しい重みで計算する方法と，1. と 2. の誤りを 3. の誤りの 2 倍の重みで計算する二つの方法で評価した．前者の評価方法は，単純に精度で評価していることに他ならない．ここでは，前者の評価方法を EV1，後者の評価方法を EV2 と呼ぶことにする．

表 3 に，タイムスタンプの検出においてそれぞれ違う閾値を用いた場合の，EV1 と EV2 の値を示す．EV1 では全ての誤りを 1 点として，EV2 では 1. と 2. の誤りを 2 点，3. の誤りを 1 点として和を求め計算している．いずれの評価も値が低いほど良い検出方法と言える．

これらの実験結果を踏まえて，我々は $f = \mathrm{EMWO}$, $t = 0.75$ の時の $\mathrm{include}_{f,t}$ 関数と，閾値 $t = 50$ の時の timestamp 関数を用いて，各クエリが示す情報の出現の時系列を求めた．

我々は，これらの得られた時系列を分析することによって最終的な解となる日付を得た．表 4 に，正解の日付，出力された日付，またクエリがイベント情報と判断されたかを示す．ここでは正解データに基づき，データセットの中でクエリが示す情報を含んでいる最古の Web ページの日付を正解の日付としている．

表 4 より分かるように，3 つのクエリ全てにおいて，正しく正解の日付が出力されている．特に，“Ichiro Suzuki was named All-Star Game MVP” と “Windows 7 is released on October 22nd” については，出力された時系列には正解の日付より前の日付が存在していたが，時系列を分析することによってイベント情報であると判定し，そのようなノイズに惑わされず正しい初出の日付を出力できている．

6. 結　　論

我々は，与えられた情報についてその情報が Web 上に出現したのはいつかということを発見することを目標として，本論文でそれを解決するための手法を提案した．そのためには，Web ページの公開日時と個々の Web ページが与えられた情報を含んでいるかということを判定する必要があり，その両方の手法を提案した．しかしながら，それらの手法は共に 8 割強程度の精度に収まり，情報の Web 上出現の完全な時系列を取得することはできなかった．

そこで，情報にはある一時より知られることとなった情報とそうでない情報の二種類あることに留意し，時系列から情報がその二種類のどちらに所属するかを判定し，特に前者の場合において情報の出現しえた日時を推定することにより，robust に情報の初出の日時を検出することを可能とした．

本研究では 3 つのクエリで実験を行い，そのいずれにおいても最終的な結果として正しい解を得ることが出来た．今後，より多くの実験による評価を行う予定である．

また，今後より多くのデータに対して実験を行った場合，本研究の提案手法で高い精度を得るためには，Web ページの公開日時を検出する手法と Web ページがクエリ情報を含んでいるかの判定をする手法の両手法において更なる精度の向上が必要となることが予想される．これらの手法の精度の向上はつまり，より正確な情報の Web 上出現の時系列を得られることを意味し，単に情報の初出を判定するだけでなくその情報が時間の推移によりどう現れたかを見ることにより，情報の Web 上での広がりや流行りを知ることができ，マーケティングなどにも活用できる．

Web ページの公開日時の検出の改善については，将来 Web ページの発信された正しい日時をクライアント側に伝えるような仕組みができれば，それに頼るということが考えられる．今後，ますます一回の情報発信につき一 URI ということが主流になれば，こういった仕組みの登場はますます望まれる．

Web ページのクエリ情報の包含判定の改善については，本研究では全ての単語を同じ重み付けで扱ったが，これを重要な単語とそうでない単語で異なった重み付けをすることにより，より尤もらしい判定を行うことが考えられる．

通常，こういった重み付けにおいてはコレクションと呼ばれる多種多様かつ膨大な量の文書を用意し，そこでの単語の出現確率などに基づいて，重み付けがなされることが多い．しかし，本研究では上位 100 件の結果だけを見ており，偏った内容かつ少量の文書集合であるため，単に単語の出現確率だけで単語の重要度を判定することは難しいように思われる．

そこで，別の手法として，特に包含度の高い複数の文書を正解（クエリ情報を包含している）文書と仮定して，包含していると見なされたセンテンスによく現れる単語を重要視し，そのセンテンス付近によく現れるクエリセンテンスに含まれない単語をクエリセンテンスと密接に関係していると仮定して，クエリセンテンスに何らかの形で含め，クエリセンテンスを重み付けし拡張するといった Psuedo Relevance Feedback の考え方を用いた手法などが考えられる．

本研究では，データセットとして，検索結果の上位 100 件の結果だけを見ている．3 節の定義より，理論上は全ての Web ページ集合を持ち出せば良いが，Web ページ全体の集合はあまりにも膨大で完全に集めるのは著しく困難な事，また仮に Web ページ全体の集合があったとしても，それら全てに本手法を適用していては時間計算量が膨大になる事により，実際には検索結果の上位を見るというような処置に頼らざるを得ない．そこ

で，上位何件まで見れば十分であるかという問題が発生するが，5 節で述べた通り，イベント情報に対しては上位 100 件を見ることにより，十分に正しい結果を得ることが出来たが，イベント情報でない情報や，またイベント情報であっても短いクエリや抽象的なクエリの場合，上位 100 件の内に正しい最古のページが含まれているとは決して限らない．それを踏まえ，まず上位 100 件をデータセットとして，時系列を取得し，それを観察することにより，更に検索結果を取得するか否かを動的に判定するといった対策が考えられる．

また，本研究では，情報発信の最小単位として Web ページを考え，静的なページにおいてはページ内の追加や削除といった事は行われないとしたが，実際には静的なページにおいても多少の更新や削除がされることがある．従って，厳密にはそれらを考慮に入れるべきであるが，これらの変更のうち初出を求める上で影響を与えるほどの変更はかなり少ないだろうという推測の下，本研究ではこれらを考慮にいれなかった．もし，これらページの変移を考えるのであれば，情報の追加に関しては，Web アーカイブなどを利用して過去のページを参照し，過去のページに今求めている情報が無ければ，少なくともページの作成時点ではこの情報は記述されいていなかったことが分かり，この情報の出現日時としてこのページの作成日時を加えてはいけないことが分かる．一方で，情報の削除に関しては，既に情報が削除されてしまっているため，現時点での Web ページを見る限りでは求めている情報が記述されていないと判定されてしまい，もしこれら情報の削除を完全に把握するのであれば，全 Web ページについてその可能性が存在するため，それは非常に困難であるといえる．従って，例えば求めている情報を現時点で含んでいる Web ページからリンクが張られているページでその情報を含んでいないページに的を絞って情報の削除を模索するといった処方が考えられる．

文　献

[1] D. Metzler, Y. Bernstein, W. B. Croft, A. Moffat, and J. Zobel, “Similarity Measures for Tracking Information Flow”, Proc. of CIKM, pp. 571-524, 2005

[2] M. Bendersky and W. B. Croft, “Finding Text Reuse on the Web”, Proc. of WSDM, pp. 262-271, 2009

[3] Y. Bernstein and J. Zobel, “A Scalable System for Identifying Co-derivative Documents”, Proc. of SPIRE, pp. 55-67, 2004

[4] D. Metzler and W. B. Croft, “A Markov Random Field Model for Term Dependence”, Proc. of SIGIR, pp. 472-479, 2005

[5] N. Balasubramanian, J. Allan, and W. B. Croft, “A Comparison of Sentence Retrieval Techniques”, Proc. of SIGIR, pp. 813-814, 2007

[6] S. E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau, “Okapi at TREC”, Proc. of 1st Text REtrieval Conf, pp. 21-30, NIST, 1992

[7] C. Zhai and J, Lafferty, “A Study of Smoothing Methods for Language Models Applied to Information Retriebal”, Proc. of SIGIR, pp. 334-342, 2001

[8] I. Soboroff, “Overview of the TREC 2004 Novelty Trec”, Proc. of 13th Text REtrieval Conf, NIST, 2004

[9] J. Allan, G. Doddington, J. Yamron, and Y. Yang, “Topic Detection and Tracking Pilot Study: Final Report”, Proc. of DARPA, pp. 194-218, 1998

[10] J. Allan, C. Wade, and A. Boilvar, “Retrieval and Novelty Detection at the Sentence Lecel”, Proc. of SIGIR, pp. 314-321, 2008

[11] M. Toyoda and M. Kitsuregawa, “What’s Really New on the Web? Identifying New Pages from a Series of Unstable Web Snapshots”, Proc. of WWW, pp. 233-241, 2006

[12] L. Yi, B. Liu and X. Li, “Eliminating Noisy Information in Web Pages for Data Mining”, Proc. of SIGKDD, pp. 296-305, 2003

[13] S. H. Lin and J. M. Ho, “Discovering Informative Content Blocks from Web Documents”, Proc. of SIGKDD, 2002

[14] N. Kushmerick, “Learning to remove Internet advertisement”, Proc. of Autonomous Agents, 1999