IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4588

Time-Indexer: a Tool for Extracting Temporal References from Business News

Pawel Jan Kalczynski
Pawel.Kalczynski@utoledo.edu, Information Systems, Management, E-commerce & Sales Dept., College of Business Administration, University of Toledo, 2801 West Bancroft St., Toledo, OH 43606
phone: (419) 530-2258 fax: (419) 530-2290
Department of Management Information Systems, The Poznan University of Economics, Poznan, Al. Niepodleglosci 10, 60-967, Poland

Witold Abramowicz, W.Abramowicz@kie.ae.poznan.pl
Krzysztof Wecel, K.Wecel@kie.ae.poznan.pl
Tomasz Kaczmarek, T.Kaczmarek@kie.ae.poznan.pl
Department of Management Information Systems, The Poznan University of Economics, Poznan, Al. Niepodleglosci 10, 60-967, Poland

ABSTRACT:

The idea behind time-indexing is that documents, apart from for their semantic context, have a temporal context. The context places events described in documents on the time axis. One way of defining temporal contexts is to extract temporal references from documents. The article presents a tool for extracting time (temporal) references from news documents. It employs a set of simple rules and a finite state automaton to compute time indices of documents based on temporal references and publication dates. As distinct from other solutions this tool is based on pattern matching rather than on lexical-syntactical analysis. The paper describes the time indexer and the results of experiments conducted with the tool. The experiment consisted of computing time indices for a collection of business news documents. Preliminary results show that the time indexer produces satisfactory results in terms of its simplicity.

INTRODUCTION

Recent scholarly attempts of utilizing time constraints in document retrieval [Llido 1998] [Abramowicz 2001] [Abramowicz 2002] were based on the idea of event-time periods. A single event-type period is a calendar interval that represents the temporal coverage of information described by a single document. In current literature, the construction of event-time periods is based on the document publication date [Allan 1998] [Swan 2000] or on anchored and unanchored temporal references extracted from the document's content [Aramburu 2001] [Kalczynski 2003]. An anchored temporal reference is a time reference with a known location on the time axis. For example Jan. 1, 2002 and March 1999 are anchored references. Unanchored time references such as "in a month," "five days ago," do not have such location and must be resolved to calendar intervals. A typical event-time period is a contiguous calendar interval [Llido 2001]. Other representations include trapezoidal fuzzy numbers [Abramowicz 2002] and set of intervals [Kalczynski 2002].

Efforts to extract temporal information from documents are also presented in [Schilder 2001]. The significance of such information in the wide area of information systems [Pons 2002] was noticed with the development of temporal databases and data warehouses. The idea of linking time references with events described in documents was expressed in [Setzer 2002]. Time references extracted from texts help to index video content [Salway 2002], build temporal summaries [Allan 2001] and chronicles [Aramburu 1998], or analyze topic evolution [Berlanga 2001].

In general, current solutions to extract temporal information from textual content can be divided into the following areas: systems that annotate time expressions and systems that time-stamp event descriptions or focus on resolving temporal relations between events [Filatova 2001] [Wilson 2000].

In this paper we propose a time indexing tool based on the pattern-matching rather than on text understanding techniques.

TIME-INDEXER

To utilize temporal information included in the textual content stored in an IR system one has to index the documents temporally. This requires connecting extra information denoting its time meaning with each document. Such information will be further referred to as a time index. In order to create a time index for a document one needs to extract time references that occur in the content. As we mentioned earlier, there may be explicit or anchored time references e.g. dates – referred to as "strong references." Weaker or unanchored statements like "last week" or "a month ago" reference time intervals in the past or future but only if they can be properly resolved. In order to resolve a weak reference, a base date (or reference date) is required. This date can be extracted from document meta-data or, if it is not possible, the date is assumed to be the document's publication date. There are also some vague references (e.g.: "some time ago," "in the past"). These are not taken into account in the present implementation of the indexing tool. The temporal context of verb tenses is also not considered. Instead our idea of time indexing focuses on temporal references that may be resolved to calendar intervals.

Intuitively, a calendar comprises a finite number of time units (e.g. years, quarters, months, days) referred to as time granularities [Goralwalla 2001]. Time granularities are essential in analyzing temporal context of text documents because apart from the time intervals, they also define the importance (or weight) of references. For instance "last month" is less precise than "two days ago." In the proposed solution we assume that "day" is the finest granularity.

The proposed time indexer is based on meta-rules which represent patterns (rules) containing temporal references. Meta-rules comprise words (e.g. "ago" or "since") and variables (e.g. "[month]" or "[weekday]"). The variables are translated into words in the run-time. For instance "[number] [granularity] ago" is a meta-rule that enables finding multiple patterns in the document content like "2 weeks ago," "three months ago,", and "seven days

ago."

Extraction of temporal references is a three-stage process. At the first stage, the textual content and reference date of the analyzed document are identified. This is performed by a dedicated parser that extracts necessary elements from the HTML code. As a result a set of XML documents emerges. The document has the following tag-hierarchy:

<DOC>
<REFDATE>
<TITLE>
<STORY>

At the second stage, sentences are extracted from documents' contents. We apply a simple set of sentence-parsing rules without employing semantic text analysis. The mechanism checks whether the punctuation mark is a part of floating point number (e.g.: 2.5), name (e.g.: Pawel J. Kalczynski) or one of the common abbreviations (e.g. U.S.). In general we assume that sentences do not end with any of the common abbreviations.

At the third stage, temporal references are extracted from sentences by means of meta-rules. The references are translated into symbolic representations which may be easily converted to calendar intervals. The proposed method of representing temporal references extracted from documents is described below.

TIME INDICES

The symbolic notation that we apply to represent temporal references builds upon the notation proposed in [Llido 2001]. The notation is based on dates, integer numbers and symbols representing temporal granularities: d – day, w – week, m – month, q – quarter, h – half and y – year. For instance "5d" denotes five days and "3m" three months. In addition to granularities we use symbols, which represent document publication dates on different granularity levels. And so, *rdd* denotes the actual publication date, *rdw* stands for the first day of the week, in which the document was published, *rdm* – the first day of the month, *rdw* – quarter, etc. For example if rdd=12/17/02 (Tuesday), then rdw=12/16/02, rdm=12/1/02, rdq=10/1/02, rdh = 7/1/02, and rdy=1/1/02.

Such a simple notation and two operators ("+" and "-") enable translating temporal references extracted from document contents into time intervals. For example "last week" may be translated into [rdw-1w,rdw-1d], and "next month" into [rdm+1m,rdm+2m-1d]. The main advantage of this notation is that it is human-understandable. This enables manual definition of meta-rules.

In the defined model, each reference has three properties: beginning of the interval, end of the interval, and the granularity level. For instance the reference "over the next two weeks" found in the document content would be resolved by the meta-rule "over the next [number] [granularity]" into {rdd, rdd+2w-1d, w}. Each reference is described by the interval and the granularity level. The granularity level is necessary to assess the importance (weight) of the extracted reference.

Finally, the symbolic representation of temporal references is translated into absolute calendar intervals. As a result a time index $T=\{(b_i, e_i, g_i)\}$ i=0..n is created for each document, where n denotes the number of temporal references in the document content, b_i and e_i denote respectively the beginning and end of a single time interval such that $b_i \le e_i$, and $g_i \in \{d,w,m,q,h,y\}$ represents the granularity level of the reference i.

In this way temporal references are stored in a structured format as records in a database. Below we describe the time indexer rules in more detail.

RULES

A single time indexer rule is a function that translates time-reference phrases extracted from a sentence into a symbolic representation. The meta-rules were created manually based on simple heuristics.

At the first stage, we manually searched for patterns in a small group of documents and built appropriate meta-rules. Then we processed the whole collection of documents with this small set of rules. Sentences containing specific words, which indicate potential time reference (e.g. "weeks" or "January") were marked as suspicious. The small fraction of suspicious references was processed manually and new meta-rules were added. Then we cleared the "suspicious" attribute and started the procedure all over again. At the time we finished this paper, the procedure was still in progress due to the large number of suspicious sentences. Sample time-indexing meta-rules are given below.

```
yesterday → rdd-1d;rdd-1d

back in [month] [year] →
[year]+[month]m-1m;[year]+[month]m-1d

next [granularity] →
rd[granularity]+1[granularity];
rd[granularity]+2[granularity]-1d

today → rdd;rdd

year-ago [number] quarter →
rdy-1y+[number]q-1q;rdy-1y+[number]q-1d

in [month] [year] →
[year]+[month]m-1m;[year]+[month]m-1d
```

The variables represent different values depending on what side (input or output) of the rule they are located. For example [month] may represent "January" on the input side and "1" on the output side, [granularity]: "quarter"→"q," [year]: "2001"→"1/1/2001," and [number]: "second"→"2." The input and output values for all variables are stored in separate structures, which we call dictionaries. A fragment of a dictionary for [weekday] variable is given below.

```
<DICTIONARY name="weekday">
<Mapping output="1">
<Token>Monday</Token>
<Token>Mon.</Token>
</Mapping>
<Mapping output="2">
<Token>Tuesday</Token>
<Token>Tue.</Token>
</Mapping>
   ...
</DICTIONARY>
```

The analysis of suspicious sentences resulted in about 100 meta-rules so far but also showed weaknesses of the proposed notation. Relatively strong references like "in the end of July" or "in the beginning of the fiscal year" could not be objectively resolved by the proposed mechanism.

Implementation of the time pattern-matching mechanism to extract temporal references from the documents' contents is based on the finite-state automaton (FSA).

FINITE-STATE AUTOMATON

The actual mechanism that searches for patterns in document contents is implemented using a finite-state automaton. The definition of automaton states:

A finite-state automaton (FSA) is a quintuple $M = (Q, \Sigma, \delta, i, F)$, *where Q is a finite set of states,* $i \in Q$ *is the initial state,* $F \subseteq Q$ *is a set of final states,* Σ *is a finite alphabet and* $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$ *is the transition function. The transition function* δ *may be extended to* $\hat{\delta} : Q \times \Sigma^* \to 2^Q$ *to accept words over* Σ *as arguments.*

In our implementation the alphabet consists of all tokens extracted from meta-rules and dictionaries. The transition function states whether the automaton should change its state after reading a particular input symbol. Out of many possible transitions, only one is arbitrarily chosen. If there are no transitions that match current input and current state (fail event) the automaton refers to the failure function. Given the last available state, the failure function changes the state of the automaton to the initial-state or other state in accordance with the previous input.

In our implementation the algorithm of Aho-Corasick, a multi-pattern

matching automaton, was adapted [Aho 1975] to serve as a time-indexing tool. The algorithm works in two stages. At the first stage the automaton is constructed according to the set of meta-rules. At the second stage, the automaton is applied to process the input stream of tokens. As distinct from A-C, words, not characters, act as tokens in our time indexer.

In addition, a single state of the automaton may represent a particular word (e.g. "ago") or a variable (e.g. [month]). As distinct from A-C, our automaton produces output only for the last state. This means that only the longest patterns (in terms of the number of tokens) are recognized. For instance "in January" and "in January 2004" are two different references but classical A-C would find three references: "in January" in both strings and "in January 2004" in the second.

The algorithm for the indexing automaton can be expressed in the following pseudo-code [Aho 1975]:

```
state←0
 for i←1 to n do
 while g(state, a_i)=fail
 do state←f(state)
 state←g(state, a_i)
 if output(state)≠empty then
 write output(state)
```

Where:

- `n` is the number of processed tokens
- `g(state,token)` is the transition function
- `fail` is a fail event (no transition found)
- `f(state)` is the failure function
- `output(state)` is the output function that returns raw time indices.

The automaton algorithm is efficient as it enables matching multiple patterns at a time. It has a linear complexity [Aho 1975].

Given a document d, a set of content-parsing rules X, a set of time-reference-extracting rules Y, a set of time-index-elements weighting rules W and a document reference date t_0, the Time Indexer I = (d, X, Y, W, t) returns the time index T of the document d. [Kalczynski 2002]

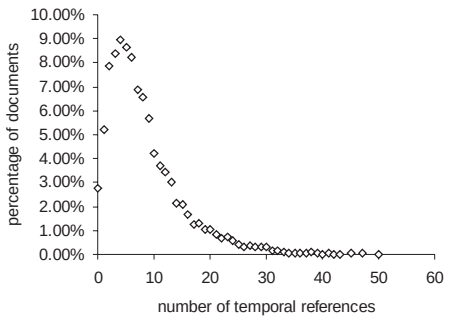
Experiment

For the purpose of the experiment we chose CNN.com/business news documents – one-page HTML documents with mostly textual content, abstracts, navigation bars, logos, banners and semantic hyperlinks. Approximately 7,000 documents that contained significant textual content (bigger than 1KB) were acquired, with reference dates, extracted from their titles, ranging from February 2, 2001 to June 7, 2001.

The content was easily extracted thanks to the source-specific HTML-comment tags that wrapped the actual story. After pre-processing, the documents were stored in a simple XML-format. Most of the documents were from 1kB to 5kB in size. The sentence parser found about 161,000 sentences.

Based on the defined meta-rules, the time indexer created over 1,500 different rules (patterns) and found nearly 57,000 temporal references in the collection. This means that an average business news document contains 8 references. The distribution of the number of references in documents is illustrated in figure 1.

Figure 1. Distribution of the number of temporal references in documents



For 2.78% of documents the automaton did not produce any temporal references. 35% of all sentences in the collection contained at least one temporal reference. This indicates that temporal information is a significant part of business news documents.

Table 1 presents distribution of granularities in temporal references.

Table 1. Temporal granularities in business news

granularity	% docs
days	40%
years	24%
quarters	17%
months	13%
weeks	5%
halves	1%

It appears that 40% of all extracted references were the most precise in terms of the proposed model. Most of these references were weekdays. Thus, an average news document has 2-3 precise references, which refer to particular calendar days. Other references are less precise yet they still contain temporal information that may be used when searching for or analyzing business news documents.

With regards to document publication dates we divided temporal references into past, present and future. Past references are represented by intervals whose right boundary is before the publication date. Present intervals contain the publication date. The remaining references are future references. Table 3 presents percentages of future, present and past references in the analyzed collection.

Table 3. Past, present and future references

Past	Present	Future
42%	40%	18%

It is significant that 18% of all extracted references refer to the future. In terms of news archives such references will be less important than past references, because they address the unknown.

Preliminary verification shows that overall recall for the automaton is 55% and overall precision is 68%. However for the lowest granularity alone (days) recall = 73% and precision = 82%. Adding more rules and adjusting the existing ones should lead to better results.

The automaton is capable of precise indexing but it cannot find all references noticed by human-indexers.

Application to temporal retrieval

Information, especially business information, is relevant to its consumers only if they can put it in an appropriate temporal context. In the face of the information overload problem, precise information retrieval is essential for efficiency of knowledge workers [Zantout 1999]. The use of temporal references improves precision of information retrieval, as it provides additional constraints to be used in user queries. [Aramburu 1998] [Abramowicz 2002].

We define temporal retrieval as a two-stage retrieval. At the first stage user temporal needs expressed as time constraints are compared with temporal indices of documents. A document is considered relevant in terms of time constraints if any of its references intersect with user temporal needs [Abramowicz 2002]. This procedure narrows the answer set by rejecting irrelevant documents, i.e. such documents that do not match time constraints of the query. In the second stage a traditional keyword-based query is executed on the resulting subset.

Precision of information retrieval is generally defined as number of relevant documents retrieved (R) divided by the number of all documents retrieved (N).

$$P = \frac{R}{N}$$

The first step of the temporal retrieval reduces the number of documents

(N) returned by the query because documents irrelevant in time (hence generally irrelevant) will not be returned by the system. The number of relevant documents in the system's response remains the same, while the number of irrelevant ones is decreased. This leads to a better precision of the keyword-based retrieval performed at the second stage.

CONCLUSIONS

This article presents a pattern-matching tool for extracting time references from textual content. The tool has a satisfying performance in terms of its simplicity. Further improvement of the IR parameters of the tool and utilization of temporal context of information can lead to enhancement of a keyword-based information retrieval system. The time indexer uses a set of meta-rules and a finite state automaton. Yet, even with such a simple approach, without involving techniques of shallow text processing and text understanding, significant results can be obtained.

REFERENCES

[Abramowicz 2001] Abramowicz W, Kalczynski PJ, Wecel K (2001) Time Consistency Among Structured and Unstructured Contents in the Data Warehouse. In Managing Information Technology in a Global Economy, Proc. of IRMA 2001, Toronto. Idea Group Publishing, pp 815-818

[Abramowicz 2002] Abramowicz W, Kalczynski PJ. Wecel K (2002) Filtering the Web to Feed Data Warehouses, Springer-Verlag London

[Aho 1975] Aho A, Corasick M (1975) Efficient String Matching: An Aid to Bibliographic Search. Communications of AC, Vol 18(6)

[Allan 1998] Allan J, Papka A, Lavrenko V (1998) On-Line New Event Tracking, ACM Proceedings of the ACM SIGIR'98 Conference, pp 37-45

[Allan 2001] Allan J, Gupta R, Khandelwal V (2001) Temporal Summaries of News Topics, ACM Proceedings of the SIGIR'01 Conference, September 9-12, 2001 New Orleans, Louisiana

[Aramburu 1998] Aramburu M, Berlanga R (1998) A Retrieval Language for Historical Documents, Proceedings of the DEXA-2001 Conference, Springer Verlag, pp 216-225

[Berlanga 2001] Berlanga R, Perez J, Mrambury M, Llido D (2001) Techniques and Tools for the Temporal Analysis of Retrieved Information, Proceedings of DEXA'01, Springer-Verlag Berlin Heidelberg, pp 72-81

[Filatova 2001] Filatova E, Hovy E (2001) Assigning Time-Stamps to Event-Clauses. In Proceedings of ACL-EACL 2001, Workshop for Temporal and Spatial Information Processing, Toulouse, pp 88-95

[Goralwalla 2001] Goralwalla I, Leontiev Y, Ozsu M, Szafron D (2001) Temporal Granularity: Completing the Puzzle, Journal of Intelligent Information Systems 16, pp 41-63

[Kalczynski 2002] Kalczynski PJ (2002), PhD Thesis, The Poznan University of Economics, Poland

[Kalczynski 2003] Kalczynski P, Abramowicz W, Wecel K, Kaczmarek T (to appear in 2003) Time-Indexer: a Tool for Extracting Temporal References from Business News in M. Khosrow-Pour (ed.) Information Technology and Organizations: Trends, Issues, Challenges and Solutions, Idea Group Inc.

[Llido 2001] Llido D, Berlanga R, Aramburu M (2001) Extracting Temporal References to Assign Document-Event Time Periods, DEXA 2001 Conference Proceedings, Mayr H et al. (Eds) Springer Verlag, LNCS 2113 , Berlin Heidelberg, pp 62-71

[Pons 2002] Pons A, Berlanga R, Rumz-Shulcloper J (2002): Temporal - Semantic Clustering of Newspaper Articles for Event Detection Pattern Recognition in Information Systems (PRIS2002), Ed. ICEIS Press, pp 104-113

[Salway 2002] Salway A, Tomadaki E (2002) Temporal Information in Collateral Text for Indexing Moving Images, LREC 2002 Workshop on Annotation Standards for Temporal Information in Natural Language

[Schilder 2001] Schilder F, Habel C (2001) From Temporal Expressions to Temporal Information: Semantic Tagging of News Messages. In Proceedings of ACL'01 Workshop on Temporal and Spatial Information Processing, Toulouse, France, pp 65-72

[Setzer 2002] Setzer A, Gaizauskas R (2002) On the Importance of Annotating Event-Event Temporal Relations in Text, In proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002), Las Palmas, Canary Islands, Spain, Workshop on Annotation Standards for Temporal Information in Natural Language

[Swan 2000] Swan R, Allan J (2000) Automatic Generation of Overview Timelines, Proceedings of SIGIR'00, pp 49-56

[Wilson 2000] Wilson G, Mani I (2000) Robust Temporal Processing of News. In Proceedings of the 38th Meeting of the Association of Computational Linguistics (ACL 2000), Hong Kong, pp 69-76

[Zantout 1999] Zantout H, Marir F (1999) Document Management Systems from Current Capabilities toward Intelligent Information Retrieval: An Overview, International Journal of Information Management (19), pp 471-484