

Detecting Age of Page Content

Adam Jatowt

Kyoto University

Yoshida-Honmachi, Sakyo-ku,
606-8501 Kyoto, Japan
Phone: +81-75-7535969

adam@dl.kuis.kyoto-u.ac.jp

Yukiko Kawai

Kyoto Sangyo University
Motoyama, Kamigamo, Kita-Ku,
603-8555 Kyoto, Japan
Phone: +81-75-7052958

kawai@cc.kyoto-su.ac.jp

Katsumi Tanaka

Kyoto University

Yoshida-Honmachi, Sakyo-ku,
606-8501 Kyoto, Japan
Phone: +81-75-7535969

tanaka@dl.kuis.kyoto-u.ac.jp

ABSTRACT

Web pages often contain objects created at different times. The information about the age of such objects may provide useful context for understanding page content and may serve many potential uses. In this paper, we describe a novel concept for detecting approximate creation dates of content elements in Web pages. Our approach is based on dynamically reconstructing page histories using data extracted from external sources - Web archives and efficiently searching inside them to detect insertion dates of content elements. We discuss various issues involving the proposed approach and demonstrate the example of an application that enhances browsing the Web by inserting annotations with temporal metadata into page content on user request.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval. I.7.1 [Document and Text Processing]: Document and Text Editing – *version control*.

General Terms

Algorithms, Theory

Keywords

age detection, web archive, metadata, document annotation

1. INTRODUCTION

The Web is evolving very rapidly due to the ease of publishing information. At the same time, the Web is vulnerable to time passage as much new content is created continuously and old content becomes quickly obsolete. It is thus important to distinguish fresh and obsolete content in Web pages. While it is true that many new pages are continually being added and old pages are being deleted from the Web, there exists a large fraction of persisting documents that have long histories, which could be successfully analyzed. Often, main or top pages in Web sites belong to this category. Many such pages contain elements inserted at different time points. Some pages show timestamps or

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WIDM'07, November 9, 2007, Lisbon, Portugal.

Copyright 2007 ACM 978-1-59593-829-9/07/0011...\$5.00.

other temporal metadata informing about the creation dates of content elements. For example, blog entries usually contain dates indicating when they were added, or news articles have annotations attached with information about their age. There are also other more ambiguous ways of expressing content age, for example, by using words like “new” or “latest”.

The main function of temporal metadata is to inform users about the age of page content. This adds temporal context to the content and many times changes its semantics and perception by users. Readers often implicitly utilize this information to better understand the content by correctly locating it on a time scale. It is especially evident in the case of news or other time-sensitive documents.

In practice, however, pages often do not offer any clues about the age of their content. The reader is then left to guess the actual age using her or his knowledge and intuition. In other cases, temporal annotations provided by page authors may sometimes be misleading and should not always be trusted. This often happens in the case of obsolete or abandoned pages. In addition, sometimes, expressions such as: “new” or “recent” may be simply used to attract readers, even if the actual age of the content is quite old. Thus, in general, although temporal annotations are very useful for readers, they may not be always available or should not be always trusted.

The objective of this research is to provide a method for automatic generation of temporal metadata by determining the age of content elements in pages. A page element is considered here as a contiguous area of content such as an image or a piece of text introduced to the page at a given time point in the past. Our proposal is based on searching inside page histories to discover the creation dates of page elements. The creation date of content is estimated as the most probable time point of its insertion into the page that can be estimated from past data. Before the search can start, page histories have to be reconstructed. This is done by automatically selecting and downloading past snapshots of pages from existing Web archival repositories. The collection of such repositories constitutes a large data source that can represent the past state of the Web. We call it the “past Web” in contrast to the current, live Web. From this viewpoint, our work exploits the past Web for enhancing and improving the user’s experience of the current Web.

The detection of content creation dates cannot be effectively done by simply checking last-modified dates provided by Web servers. These tell only the time of the last updating of pages and do not offer any information about the content and sizes of changes. Hence, a whole page may be claimed as fresh, while actually

most of its content could have been published long time ago. In addition, last-modified dates are often unreliable or may not be provided by many servers.

Detecting creation dates of page elements using our approach has many potential applications. First of all, it can be useful when there is no temporal metadata for a page and a user cannot easily estimate the age of the content. Automatically generating temporal metadata could then provide new context that would facilitate content understanding. For example, a user may wish to know the appearance date of a certain statement in a company homepage in order to better understand its meaning or importance. In another example, the user may check when a particular member of a laboratory began to work by detecting when her or his name was added to the list of the laboratory's personnel. Naturally, there can be some delay between the real-world event and its report on the page, yet often a reasonable approximation is achieved by investigating temporal constraints of the content. Knowing the creation date of content may also help to understand vague temporal expressions, like, "three months ago" or "yesterday" by anchoring them in time if they cannot be easily resolved. On the other hand, for page content that already contains temporal metadata, the method can be used to check its credibility. For example, it would be possible to examine whether the content stipulated with the "new" expression is actually fresh.

We demonstrate the usefulness of detecting content creation dates by showing application for enhanced browsing. It first computes the age of content in accessed pages and then visualizes it to users in the form of temporal annotations. Users can then obtain additional information for understanding the content of viewed pages.

The rest of the paper is organized as follows. In the next section we provide the necessary background. In Section 3 we describe the data preparation process, propose two search methods and discuss several issues related to the concept of age detection. Section 4 demonstrates an application that uses our method and the results of preliminary experiments. In Section 5 we describe the related research. Lastly, Section 6 concludes the paper.

2. BACKGROUND

2.1 Version Management

Version management was an important characteristic of early hypertext systems [23]. For example, it was an intrinsic and fundamental mechanism in Ted Nelson's Xanadu [17] – a pioneering Hypertext system. Versioning was used there to prevent content duplication between document versions. Instead of repeating the same content parts, newer document versions contained links to the content of older versions. Usually, the main purpose of versioning is to force rigorous source tracking so that consecutive changes or their authors can be easily identified [20,23]. This guarantees the rollback option to any previous document version if necessary. However, in the current Web, only certain types of pages (e.g., wikis) enable source tracking. The complete past data is available for such pages and the actual changes that occurred in the past can be easily derived. Recently, there has been also a proposal for automating version management in Web servers [5]. Such servers would automatically preserve previous document versions in case of any content update and would provide standardized access methods to

these versions. Nevertheless, for the majority of page types, no version management is currently provided. Therefore, external, third-party data sources such as Web archives have to be exploited. This approach, however, requires efficient methods for selecting and analyzing retrospective data, which is fragmentary, distributed and sometimes overlapping.

2.2 Web Archives

Our method utilizes data accumulated in Web archives, which contain frozen snapshots of Web documents from the past. Many such archives came to existence along with the proliferation of the Web and the growing awareness of the part it plays in societies. The Internet Archive¹ [10] is the best known public Web archive containing more than 2 petabytes of data. Currently there is, however, no textual search available in this archive despite the massive amount of data it contains. Other dedicated Web archives also exist such as archives containing pages from certain countries, e.g., the Australian Archive² or thematic Web archives related to certain event or topic, such as the September 11 archive³. Besides, there are other repositories of past Web data that could be exploited, such as local caches, site archives, personal repositories or search engine caches. Along with the increasing awareness of the necessity for the preservation of the Web it is expected that more archives with a broader coverage will become available in the future.

Web archiving research faces many nontrivial problems related to the selection, storage and preservation of Web content [13]. Therefore, the Web archiving community has put relatively little attention until now to proposing applications that could successfully exploit the accumulated data. Nevertheless, some researchers have drawn the attention of the Internet community to the benefits of utilizing Web archives [1,2,9,19]. For example, the potential of mining Web archive data for general knowledge discovery and the necessary measures for achieving this goal were discussed in [2,9]. Arms et al. [1] has also reported on building a research library for scientists to study the evolution of the content and structure of the Web. This is an ongoing project that aims at creating an infrastructure for the analysis of data obtained from the Internet Archive.

3. CONTENT AGE DETECTION

3.1 Data Preparation

The snapshot of a page captured at a certain time point, indicated by a timestamp, reflects the past content that the page had at that time point. The data collection step in our approach is responsible for constructing the resource list containing references to page snapshots. During the subsequent search process selected snapshots from the list will be downloaded for content comparison.

Due to resource limitations, Web archives often contain only few past snapshots of pages or they may not contain any snapshots at all for certain pages. Our concept of the data collection process attempts to alleviate this problem thanks to communication with

¹ Internet Archive: <http://www.archive.org>

² Pandora, Australia's Web Archive: <http://pandora.nla.gov.au>

³ September 11 Web Archive: <http://september11.archive.org>

several archival repositories at the same time via a proxy server. The server acts as an intermediary between the archives and the local system, translating each query into the format required for every repository. After receiving a request for the past data of a given page, it queries the archives about snapshots they contain. The archives should then send lists of contained snapshots with their metadata. These are then used for a construction of the resource list containing references to past page snapshots. The references are chronologically ordered in the list according to the timestamps of the snapshots that they refer to. The references provide information about the repositories and timestamps of the snapshots.

An assumption is made here for the existence of the same URL for a page over time. However, in reality, URLs can sometimes change (e.g., due to the change in site topology) while the actual content of the page remains the same. To cope with this problem, a method for detecting changes in URLs over time should be applied. Such a method would necessarily involve an analysis of changes in page context over time and is beyond the scope of this paper.

Finally, the last issue to be mentioned is a problem of data without any timestamps. For example, Yahoo! search engine provides cached snapshots of pages; however it does not attach any timestamps to them. Hence, it is not possible to insert the reference to a snapshot into the resource list without checking the similarity of its content to the content of other snapshots with known timestamps. This would be, however, disadvantageous for our approach as it requires prior downloading of snapshots.

3.2 Search

Detecting content creation dates is done by a search for the oldest snapshots containing content elements that occur in the present page version. Selected snapshots from the resource list are fetched so that their content can be compared with the content of the present page version for common elements. In order to detect the creation date of an element, it is necessary to find a pair of consecutive snapshots, in which only the newer one contains the element. For a more formal definition of the creation date let us suppose that there are n snapshots s_1, s_2, \dots, s_n with respective timestamps t_1, t_2, \dots, t_n . s_n is the snapshot of the present page version.

Definition 1. Time point t_k is considered to be the creation date of an element of s_n if the element is contained in snapshot s_k whose timestamp is t_k , while not being contained in snapshot s_{k-1} whose timestamp is t_{k-1} and $t_{k-1} < t_k$.

Change detection has to be thus done between the selected past snapshots and the present page version in order to find creation dates of content elements. To perform the change detection, first the selected snapshot is stripped from any content that was added by repositories⁴. Next, every HTML node in both the selected snapshot and in the present page snapshot is converted to a single line. The algorithm then examines if there are any lines with the same content in the compared documents using *diff* method, which is based on finding longest matching subsequences. Images are compared by analyzing image attributes in their HTML tags.

⁴ For example, cached pages in Google search engine have annotations attached containing information about them.

During the search, a list is maintained containing all the elements whose creation dates have been already detected in the previous comparison steps. This helps to decrease the number of searched elements in the consecutive search steps. Thus the number of nodes to be compared should decrease or at least should remain the same along with the progress of the search process. The search terminates when all the elements that occur in the present snapshot will have their creation dates detected or in case when the system cannot compute creation dates due to the lack of data.

3.2.1 Sequential Search

The naïve approach is to sequentially download snapshots from the resource list starting from the youngest one. Each snapshot is then compared with the present page version until all the content elements will have their creation dates detected. However, such a sequential search is often impractical as it means downloading and comparing large number of snapshots. This is often the case when the elements in the current page version are old and repositories store many past snapshots of the page. Reversing the search direction (starting from the oldest snapshot) would not always help as the actual distribution of content creation dates remains unknown.

Sequential search process is depicted in Figure 1. Page snapshots are visualized as grey rectangles that contain objects (dots). For simplicity, the present page version is composed of only one object, which is marked by a yellow dot. Thus, the search is aimed at finding the time point when the object has been added to the page. The figure shows the search cost in terms of search steps, that is, downloaded page snapshots.

3.2.2 Multi-binary Search

For improving search efficiency a multi-binary search is proposed. First, it looks for the oldest snapshot which contains any element that also occurs in the current page version. Then, it searches for creation dates of other elements utilizing the already downloaded snapshots. Multi-binary search differs from a usual binary search done on an ordered sequence in two aspects. First, it looks for a pair of consecutive snapshots in which only the older one contains a given content element (according to Definition 1). Second, it attempts to detect the creation dates of all page elements in one search process by using multiple binary searches. This minimizes the number of necessary search steps.

The multi-binary search for the case of a page containing one object is displayed in Figure 2. When comparing Figures 1 and 2 it can be seen that multi-binary search is faster than the sequential one. The proposed search method can thus result in considerable cost savings in case of pages with many past snapshots and relatively old content. Figure 3 presents an example of multi-binary search for a page whose current version contains three elements.

Below we describe the multi-binary search process in detail. Let us denote the time period between t_1 and t_n by T . First, the usual binary search is done on T for a snapshot that does not contain any common element with the snapshot of the present page version, s_n . To do this the page snapshot is downloaded that lies closest to the middle point of T (step 1 in Figure 3). This snapshot is then compared with s_n for occurrence of the same content elements. If there are some identical content elements, then the snapshot is fetched that lies closest to the point $1/4*T$ (step 2 in Figure 3). Otherwise, the snapshot closest to the time point $3/4*T$

is downloaded. This continues until the snapshot is found that does not have any common element with the current page version (step 3 in Figure 3). Let us denote the timestamp of such a snapshot by τ . Next, the binary search is made again for the discovery of the youngest snapshot that has completely different content than the one in s_n . However, this time the search is made on the snapshot subsequence constrained by τ and τ' . τ' is the timestamp of the most recently fetched snapshot that provides the assurance that the search target lies between τ and τ' . After the required snapshot is found, the search process continues until the oldest snapshot is detected that contains at least one element which occurs also in the current page version (steps 4, 5 in Figure 3). In this way the creation date of the oldest element is detected ($t_{ins(1)}$ in Figure 3).

When the creation date of the oldest element is discovered, then the search process continues in a similar manner to find the creation dates of the remaining, newer content elements (steps 6, 7 in Figure 3). The snapshots that have been already fetched are now used to determine the constraints for consequent binary searches. In general, the time period between two timestamps t_{left} and t_{right} of any two previously fetched snapshots s_{left} and s_{right} is used as a scope for a binary search if the three following conditions are fulfilled:

- The snapshots are consecutive in the sequence of already fetched snapshots, that is, there have been no other snapshots fetched whose timestamps are included in the period $\langle t_{left}, t_{right} \rangle$.
- There are some snapshots in the resource list that have timestamps in $\langle t_{left}, t_{right} \rangle$ and that have not been fetched yet.
- The numbers of content elements that are common with the present page version are different for both snapshots.

Using these conditions the content elements in the page have their creation dates detected starting from the oldest to the youngest one.

Sometimes, it may happen that the creation date of the oldest element cannot be detected due to insufficient data. In such a case the search continues for discovering the creation dates of the remaining elements using sequences of binary searches as described above.

Special care should be taken for dealing with content modifications. Modification of an element could be simply considered as a pair of consecutive operations of deletion and insertion made on the same content part. However, it would be advantageous to recognize relatively insignificant changes (e.g., grammatical change, word order change) and continue the search for detecting the actual creation date of the element. In order to do so we set up a threshold that is used in each comparison step. Elements in compared snapshots are considered to be same if their similarity is higher than the specified threshold value. The similarity is defined here as the ratio of the number of common terms to the number of all terms occurring inside a node in the past and in the present snapshot of the page.

We assume continuous occurrences of elements in pages. In other words, we neglect the cases when a given object was first deleted from a page and then it was added to the page again. Without this assumption, one would have to download and analyze all past snapshots from the resource list. In some cases, however, multi-binary search can still correctly detect the actual creation date of

such reoccurring elements. The standard sequential search, on the other hand, will always fail, unless the reverse sequential process is used.

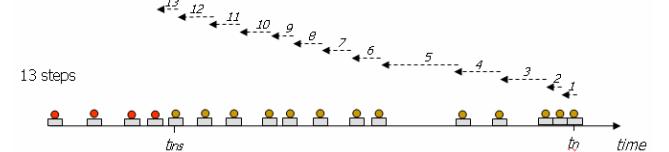


Figure 1. Sequential search for one content element represented by a yellow dot (t_{ins} is the element's creation date; t_n is the present time).

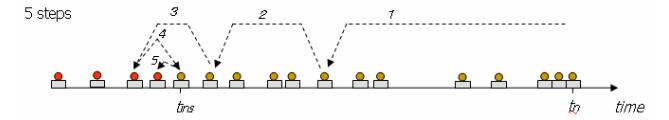


Figure 2. Multi-binary search for one content element represented by a yellow dot (t_{ins} is the element's creation date; t_n is the present time).

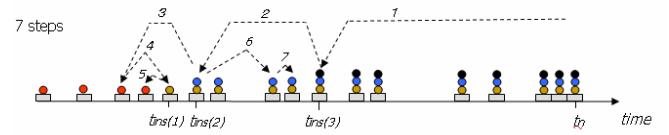


Figure 3. Multi-binary search of three content elements represented by yellow, blue and black dots ($t_{ins(1)}, t_{ins(2)}$ and $t_{ins(3)}$ are creation dates of the elements; t_n is the present time).

3.3 Discussion

There will be usually a certain error involved with the age detection process as demonstrated in Figure 4. As explained before, the search process finds the oldest page snapshot that contains the element. Let us denote its timestamp by t_k . However, the actual time point of the element's insertion is somewhere between t_{k-1} and t_k , where t_{k-1} denotes the youngest page snapshot that does not contain that element. Let us suppose that the actual history of the page is depicted in the bottom timeline in Figure 4. We can see that the result provided by the search process contains some error due to insufficient data. In order to decrease the maximum possible error, the middle point of $\langle t_{k-1}, t_k \rangle$ could be considered as the element's creation date. However, assuming t_k to be the creation date is a safer choice as it provides the assurance that the actual age of the element is not smaller than the length of the period $\langle t_k, t_n \rangle$.

As there is no additional information concerning the exact timing of the content insertion, a uniform probability distribution of this event is assumed here for the period $\langle t_{k-1}, t_k \rangle$. This probability could be modified in the case when some other information is available concerning the addition of the element to the page. For example, for this purpose, we could utilize the average change frequency and change degree of the page.

The error shown in Figure 4 depends on the length of the time period between t_{k-1} and t_k , hence, the average error of the detection process should be smaller the higher is the number of past snapshots. The amount of available past data is thus a main factor influencing the precision of the age determination process. However, the increase in the number of available past snapshots

causes higher cost of the search process. This cost could be decreased by not downloading duplicate, consecutive snapshots if snapshot checksums were provided as metadata⁵.

It should be also noted here that content migration may occur within Web sites. Thus, it could happen that a certain object had been published earlier on another page in the same site. The actual creation date of the object in the site would be thus older than the one detected using our method. As the history reconstruction of the entire Web site is a costly process, we neglect the issue of content migration here. In [4] a tool was reported for detecting the content migration among pages within Web sites.

Finally, a certain element could have been published in a page in a different form. For example, some concept could have been expressed in the page using different sentences or an image could have assigned different attributes. Accurately detecting the real date of the first occurrence of such an object is nontrivial and often requires a deeper semantic comparison of content. In the current research we use only a threshold value in the change detection process to detect simple content transitions.

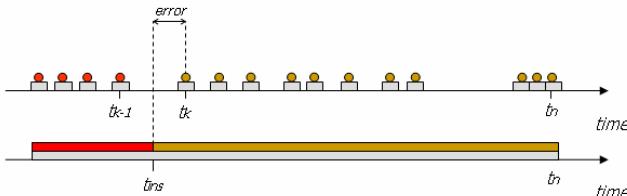


Figure 4. Error of creation date detection due to fragmentary data.

4. BROWSING ENHANCEMENT

We describe here the browsing enhancement that utilizes the proposed method. The system was implemented using C# and the Microsoft .NET 2.0 Framework. It works as a standalone application; however, it could be also implemented as a toolbar for standard Web browsers. A user can request the age detection of an arbitrary page during Web browsing by pressing the “start age detection” button. Upon such a request data is automatically retrieved from archival repositories and the detection of content creation dates is performed. We used here the Internet Archive, Google and MSN caches as data sources. In case when page snapshots could not be downloaded, the system waited short time and tried again. If the next trial failed, it proceeded to fetch the other selected snapshot. In addition, threading was used for decreasing the time cost.

The user has a choice of three search types: sequential, sequential reverse and multi-binary. She or he can also specify the threshold level for dealing with trivial content modifications. The default value of this threshold was set to 1.

The system visualizes content creation dates by attaching annotations to the right corners of red frames that surround content elements by using JavaScript (Figure 5). The size of annotations is kept at the minimum level guaranteeing their readability and at the same time leaving the original outlook of a

⁵ In general, for a sequence of snapshots $s_1, \dots, s_i, \dots, s_m$ that have identical content, only the oldest and the latest snapshots (i.e., s_1 and s_m) should be added to the reference list.

page changed as little as possible. Those content elements for which creation dates could not be found (due to the insufficient amount of data) are annotated with an expression “before $t_{\text{oldest_snapshot}}$ ”, where $t_{\text{oldest_snapshot}}$ denotes the timestamp of the oldest snapshot.

Sometimes the response time can be quite long especially in case of pages whose content is old and for which there are many past snapshots available. To alleviate this problem we decided to add a dynamic visualization mode in which the annotations are gradually visualized during the search process. The progress of the search is also displayed at the top of the page as a timeline indicating page snapshots that have been already downloaded (Figure 6). The user can then stop the search process at any time by pressing “stop age detection” button. This could be done, for example, when the creation date of given content of interest has been already displayed in a page or when there are too many snapshots left to be fetched. When “start age detection” button is pressed again, the browser returns the original view of the page.

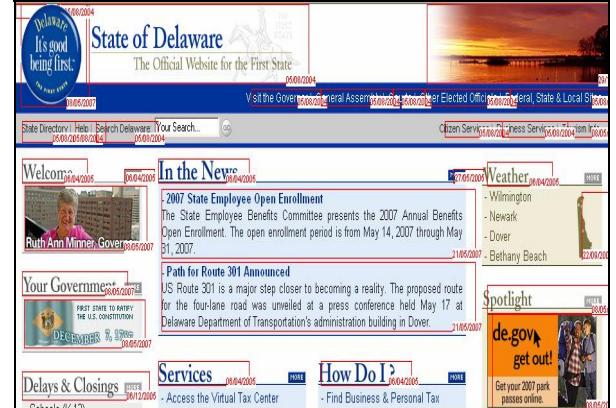


Figure 5. Example of annotated page, <http://www.delaware.gov>

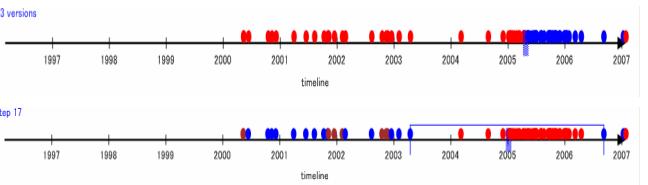


Figure 6. Visualization of search progress for sequential search (top timeline) and for multi-binary search (bottom timeline).

In addition, the browser shows the creation date of the whole page estimated as the timestamp of the oldest snapshot in the resource list. Besides, the average creation date for the whole content is displayed. It is computed by considering the relative sizes of elements in pages (Equation 1).

$$t_{ins}^{avr} = \sum_{i=1}^M \frac{n(i)}{N} * t_{ins}(i) \quad (1)$$

Here, t_{ins}^{avr} is the average creation date of page content, $n(i)$ is the number of terms inside an element i , $t_{ins}(i)$ is the creation date of element i , N and M are the number of terms and the number of elements in the whole page, respectively. The creation dates of pages and the average creation dates of their content help users to assess the general freshness levels of information found in the pages.

The proposed browsing enhancement can be used for arbitrary pages for which there are any past snapshots available in selected Web archives. It helps to assess the age of content fragments of interest, which can be helpful in case when there is no other temporal information provided. Besides, the system can be used for judging the trustworthiness of existing temporal annotations on pages.

In future, we plan to add an additional mode which will enable a user to select a specific area of page content for age detection. The system would then detect the creation dates of content elements that are inside HTML nodes covering the selected area. In this mode the data collection would be the same as in the standard approach. However, the search process would be different since it would consider the current page version as containing only the selected HTML nodes.

4.1.1 Experiments

Below we present the results of preliminary experiments conducted in order to test the proposed concept of detecting content creation dates and to compare the effectiveness of both search methods. The experiments were done on May 21st, 2007 on a computer with Windows XP operating system equipped with 3GHz processor and 3GB memory.

Table 1 shows the results of using both sequential and multi-binary search methods for several example pages. For each page we calculated the number of steps, that is, the number of downloaded past snapshots and the required time cost for both search methods. These are shown in the 2nd and 3rd columns in Table 1 (time cost is expressed as the number of seconds in brackets). In addition, we computed the creation dates of pages and the average creation date of page content. Table 1 presents also the total number of available snapshots and the estimated creation dates of youngest and oldest elements in pages. Below, we discuss some of the results.

The first page (<http://www.google.com/intl/en/about.html>) is a page that provides basic information about Google Inc. and its products. After detecting the creation dates of its content, we found that the section's title "For Site Owners" was the oldest content, while the advertisements of "Docs and Spreadsheets" and "Google Groups" were the youngest elements. Most of the page content was rather old (average creation date - 02/12/05) indicating that the page did not change much in the recent past.

The second page (<http://www.dmoz.org>) is the main page of the well-known Open Directory Project (ODP). Most of the listed categories (top-level categories) were the oldest elements in this page. The youngest element was the counter of sites, editors and categories ("4,830,584 sites - 75,151 editors - over 590,000 categories") shown at the bottom of the page. In general, the page had very old content (average creation date of content - 14/06/2000). This is because the published category listings generally did not change over time. There are many past snapshots of the page (327 snapshots) hence the sequential search took relatively much time (124 seconds). Multi-binary search was much better in terms of fetched snapshots (28 snapshots), yet its time cost was only about 25% shorter than the one for the sequential search (93 seconds).

For CIKM 2004 conference homepage (<http://ir.iit.edu/cikm2004>) there were only 13 past snapshots available. Only the snapshots for the year 2004 are provided by the Internet Archive. We think

that perhaps the number of visiting users was low in the subsequent years and the page content did not change much. This could be the reason why the Internet Archive does not contain few versions of the page. The youngest content for this page (creation date - 09/10/04) was the link to "Student Award" and "Conference Program" pages, while the rest of the page, such as links to "Organizing Committee" or "Paper Submission" had the oldest creation dates (before 12/02/04).

The third page (<http://www.worldchanging.com>) is a blog site listed in Yahoo! directory⁶ (Society and Culture > Relationships > Dating > Blogs). It is interesting to notice that the average age of the page content is relatively young compared to the total age of the page. As it is a blog site, its past content is stored in the site's archive. For this page, the sequential search produced final results much faster and with the smaller number of downloaded snapshots. The oldest elements on the page were annotated with "contact" and "designers logo". The youngest element, on the other hand, was the content of the latest post titled "The Week in Sustainable Mobility (5/20/07)" with the annotation "Mike Millikin, 20 May 07". Using our method it is thus possible to some extent examine the trustworthiness of temporal annotations inside page content.

For the pages <http://www.businessweek.com>, <http://www.yahoo.com> and <http://www.stanford.edu> the sequential search process took quite a long time (489, 859 and 231 seconds, respectively) due to the large number of snapshots (964, 912 and 631 snapshots, respectively). The multi-binary process required 28 snapshots and 91 seconds for <http://www.yahoo.com>, 65 snapshots and 444 seconds for <http://www.businessweek.com> and 20 snapshots and 54 seconds for <http://www.stanford.edu>. Usually, the multi-binary search required fewer snapshots for downloading and it was faster than the sequential search process.

The last two pages, <http://www.delaware.gov> and <http://www.state.nj.us> are home pages of Delaware and New Jersey states. Because of their relatively rich content the multi-binary search process took relatively long time, although it required fetching much smaller number of snapshots.

In addition, as many popular pages contain lots of fresh content and the Internet Archive often does not provide the most recent data (usually there are no snapshots available for the last 6 to 12 months⁷) we think that the combination of both search methods should produce good results. In such a hybrid approach, first the sequential search is used for downloading the earliest two to three snapshots (e.g., snapshots obtained from search engine caches). Next, the system starts the multi-binary search utilizing the previously fetched snapshots.

⁶ Yahoo! Directory: <http://dir.yahoo.com>

⁷ According to Internet Archive there are plans to substantially decrease this delay in the near future.

Table 1. Results of content creation date detection for example pages using both search methods (dates are in dd/mm/yy format).

URL	Sequential search	Multi-binary search	Page creation date	Average creation date	All snapshots	Oldest element	Youngest element
http://www.google.com/intl/en/about.html	67 (17s)	23 (15s)	10/05/00	02/12/05	74	15/06/01	22/05/07
http://www.dmoz.org	387 (124s)	28 (93s)	25/01/99	14/06/00	389	before 25/01/99	21/05/07
http://ir.iit.edu/cikm2004	13 (3s)	6 (3s)	12/02/04	05/04/04	13	before 12/02/04	09/10/04
http://www.worldchanging.com	2 (18s)	20 (120s)	06/12/03	01/03/07	367	16/05/07	22/05/07
http://www.yahoo.com	964 (489s)	28 (91s)	17/10/96	28/05/06	1015	03/10/99	21/05/07
http://www.businessweek.com	912 (859s)	65 (444s)	31/10/96	23/03/07	928	15/08/00	21/05/07
http://www.stanford.edu	631 (231s)	20 (54s)	13/01/97	20/02/07	641	10/02/99	21/05/07
http://www.commonlaw.com	35 (6s)	7 (7s)	12/04/97	08/04/02	35	20/05/98	13/05/07
http://www.delaware.gov	236 (321s)	23 (257s)	18/04/01	27/03/06	302	24/03/04	21/05/07
http://www.state.nj.us	198 (336s)	40 (204s)	06/03/97	14/06/06	689	30/11/04	19/05/07

5. RELATED RESEARCH

The proposed idea of automatically estimating the creation dates of content elements of pages and annotating them with these dates is new and, to the best of our knowledge, it has not been pursued in research yet.

In the early hypertext systems, document versions were preserved and stored within the systems [6,11,17,20]. It was relatively easy for users to manually browse and search inside document versions, especially when their number was low. Our approach is based on the usage of external data sources, as the majority of Web documents are not subject to any automatic versioning. This means that the data is not only fragmentary but is also distributed and sometimes overlapping among different repositories. Although recently some applications were proposed to effectively visualize page histories using their archived snapshots [8,22], still, a user has to rely on her or his perception if she or he wishes to know the age of page content. An example of such an application is a browsing application for Web archives which we have previously proposed [8]. It allows for a customized traversing of the link structure of the past Web as well as it draws user attention to changes by animating added and deleted page content.

This work is related to the research about estimating the level of obsoleteness of pages [3,18,21]. Toyoda and Kitsuregawa [21] recently proposed an algorithm for detecting newly created pages based on sets of Web snapshots. They used a recursive definition of a page novelty computed in a similar way to the PageRank algorithm by utilizing link structure. In another research, Bar-Yossef et al. [3] estimated levels of decay of pages using a random surfer model in order to measure probabilities of accessing abandoned pages. Nunes et al. [18] proposed complementing the traditional HTTP header-based dating technique by utilizing information about the age of page neighborhood. In this paper we approach the problem of freshness evaluation with a finer granularity and from a different direction

as we focus on content elements inside pages and use archived data.

Some work [14,15,16] has been recently done on measuring the persistence and availability of page copies inside the repositories of major search engines and the Internet Archive. The objective was to estimate the possibility of reproducing the latest versions of Web sites in case of server crashes or other accidents that cause loss of Web data and to provide efficient tools for this purpose. Although, we utilize the same kind of data, our aim and method are different. We propose the idea of dynamically creating temporal metadata for content elements on pages to provide additional information to Web users. To do this we introduce searching mechanisms inside page histories. Lastly, we attempt at reconstructing the entire histories of Web pages rather than their latest versions.

Discovering and annotating temporal expressions in natural language is another related research area (e.g., [7,12]). Usually, absolute reference points (e.g., dates, timestamps) are required for the correct disambiguation of relative temporal expressions in text. When no such information is provided, the document creation date is used in order to anchor ambiguous expressions in time. Our method has the potential to improve this research by providing temporal constraints for content age in pages.

6. CONCLUSIONS

In contrast to many traditional document types, Web pages often contain elements created at arbitrary time points. However, usually, it is difficult to know their creation dates. In this paper, we have proposed a method for automatic discovery of the creation dates of page content. This is a retrospective approach that utilizes past snapshots of pages obtained from external, archival repositories in order to dynamically reconstruct page histories at user's request. The histories are then efficiently searched for time points when content elements were observed for

the first time. In result, it is possible to assign approximate creation dates to page content.

This type of temporal representation provides additional context to page content that can be used, for example, for improving document understanding, disambiguation of temporal expressions or temporal search. We have discussed advantages of estimating temporal metadata of pages and demonstrated its usefulness by the example of an application for enhanced browsing of the Web that annotates the content of pages with its creation dates.

7. ACKNOWLEDGEMENTS

We would like to thank reviewers for their valuable comments. This research was supported by the MEXT Grant-in-Aid for Scientific Research in Priority Areas entitled: Content Fusion and Seamless Search for Information Explosion (#18049041, Representative Katsumi Tanaka) and by the Informatics Education and Research Center for Knowledge-Circulating Society (Project Leader: Katsumi Tanaka, MEXT Global COE Program, Kyoto University) as well as by the MEXT Grant-in-Aid for Young Scientists B entitled: Information Retrieval and Mining in Web Archives (#18700111).

8. REFERENCES

- [1] Arms, W.Y., Aya, S., Dmitriev, P., Kot, B. J., Mitchell, R. and Walle, L. Building a research library for the history of the web. *Proceedings of the Joint Conference on Digital Libraries*, pages 95-102, Chapel Hill, NC, USA. ACM Press, 2006.
- [2] Aschenbrenner, A. and Rauber, A. Mining web collections. In [13], 153-174.
- [3] Bar-Yossef, Z., Broder, A. Z., Kumar, R. and Tomkins, A. Sic Transit Gloria Telae: Towards an understanding of the web's decay. *Proceedings of the 13th International World Wide Web Conference*, pages 328–337, New York, USA. ACM Press, 2004.
- [4] Davis, P., Maslov, A. and Phillips, S. Analyzing history in hypermedia collections. *Proceedings of the 16th Conference on Hypertext and Hypermedia*, pages 171-173, Salzburg, Austria. ACM Press, 2005.
- [5] Dyreson C. E., Lin H.-L., and Wang Y. Managing versions of web documents in a transaction-time Web server. *Proceedings of the 13th International World Wide Web Conference*, pages 422-432, New York, USA. ACM Press, 2004.
- [6] Haake, A. and Hicks, D. VerSE: Towards hypertext versioning styles. *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia*, pages 224-234, Washington DC, USA. ACM Press, 1996.
- [7] Han, B. and Lavie, A. A Framework for resolution of time in natural language. *TALIP Special Issue on Spatial and Temporal Information Processing*, 3(1), 2004.
- [8] Jatowt, A., Kawai, Y., Nakamura, S., Kidawara, Y. and Tanaka, K. Journey to the past: proposal of a framework for past web browser. *Proceedings of the 17th ACM Conference on Hypertext and Hypermedia*, pages 135-144, Odense, Denmark. ACM Press, 2006.
- [9] Jatowt, A. and Tanaka, K. Towards mining past content of web pages, *New Review of Hypermedia and Multimedia, Special Issue on Web Archiving*, 13(1), pages 77-86. Taylor & Francis, 2007.
- [10] Kimpton, M. and Ubois, J. Year-by-year: from an archive of the Internet to an archive on the Internet. In [13], pages 201-212.
- [11] Maioli, C., Sola, S., and Vitali, F. Wide-Area distribution issues. In Hypertext Systems. *Proceedings of ACM SIGDOC '93*, pages 185-197, Kitchener, Canada. ACM Press, 1993.
- [12] Mani, I. Recent developments in temporal information extraction. *Proceedings of Recent Advances in Natural Language Processing Conference*, pages 45-60, Borovets, Bulgaria, 2003.
- [13] Masanes, J. (ed.). *Web archiving*. Springer Verlag, Berlin Heidelberg, Germany, 2006.
- [14] McCown, F., Diawara, N. and Nelson, M.L. Factors affecting website reconstruction from the web infrastructure. *Proceedings of the Joint Conference on Digital Libraries*, pages 39-48, Vancouver, Canada. ACM Press, 2007.
- [15] McCown, F. and Nelson, M.L. Evaluation of crawling policies for a web-repository crawler. *Proceedings of the 17th Conference on Hypertext and Hypermedia*, pages 157-168, Odense, Denmark. ACM Press, 2006.
- [16] McCown, F., Smith, J.A. and Nelson, M.L. Lazy preservation: reconstructing websites by crawling the crawlers. *Proceedings of the 8th International Workshop on Web Information and Data Management*, pages 67-74, Arlington, VA, USA. ACM Press, 2006.
- [17] Nelson, T. H. *Literary Machines*, edition 87.1, Sausalito Press, 1987.
- [18] Nunes, S., Ribeiro, C. and David, G. Using Neighbors to Date Web Documents. *Proceedings of the 9th ACM International Workshop on Web Information and Data Management*, Lisbon, Portugal. ACM Press, 2007.
- [19] Rauber, A., Aschenbrenner, A. and Witvoet, O. Austrian online archive processing: analyzing archives of the world wide web. *Proceedings of the 6th European Conference on Digital Libraries*, pages 16–31, Rome, Italy. Springer, 2002.
- [20] Shipman F. M. and Hsieh H. Navigable history: a reader's view of writer's time: Time-based hypermedia. *New review of hypermedia and multimedia*, vol. 6, pages 147-167. Taylor & Francis, 2000.
- [21] Toyoda, M. and Kitsuregawa, M. What's really new on the web?: identifying new pages from a series of unstable web snapshots. *Proceedings of the 15th World Wide Web Conference*, pages 233-241, Edinburgh, Scotland. ACM Press, 2006.
- [22] Viégas, F., Wattenberg, M. and Dave, K. Studying cooperation and conflict between authors with history flow visualizations. *Proceedings of the SIGCHI Conference on Human Factors in computing systems*, pages 575-582, Vienna, Austria. ACM Press, 2004.
- [23] Vitali, F. Versioning hypemedia. *ACM Computing Surveys* 31(4): 24. ACM Press, 1999.