Archiving the Web using Page Changes Patterns: A Case Study

Myriam Ben Saad
LIP6, University P. and M. Curie
4 place Jussieu 75005
Paris, France
Myriam.Ben-Saad@lip6.fr

Stéphane Gançarski
LIP6, University P. and M. Curie
4 place Jussieu 75005
Paris, France
Stephane.Gancarski@lip6.fr

ABSTRACT

A pattern is a model or a template used to summarize and describe the behavior (or the trend) of a data having generally some recurrent events. Patterns have received a considerable attention in recent years and were widely studied in the data mining field. Various pattern mining approaches have been proposed and used for different applications such as network monitoring, moving object tracking, financial or medical data analysis, scientific data processing, etc. In these different contexts, discovered patterns were useful to detect anomalies, to predict data behavior (or trend), or more generally, to simplify data processing or to improve system performance. However, to the best of our knowledge, patterns have never been used in the context of web archiving. Web archiving is the process of continuously collecting and preserving portions of the World Wide Web for future generations. In this paper, we show how patterns of page changes can be useful tools to efficiently archive web sites. We first define our pattern model that describes the changes of pages. Then, we present the strategy used to (i) extract the temporal evolution of page changes, to (ii) discover patterns and to (iii) exploit them to improve web archives. We choose the archive of French public TV channels *France Télévisions* as a case study[1] in order to validate our approach. Our experimental evaluation based on real web pages shows the utility of patterns to improve archive quality and to optimize indexing or storing.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Algorithms, Design, Measurement

Keywords

Web Archiving, Web Page Changes, Pattern

1. INTRODUCTION

Due to the growing importance of the World Wide Web, many national libraries and organizations (*e.g.* Internet Archive[2], European Archive[3]) have the mission of archiving portions of the Web to prevent useful content from disappearing. The major challenge of such organizations is to collect, preserve and enable (possibly far) future accesses to a rich part of the Internet content from around the world. Therefore, future generations of users will be able over time to access and browse off-line a collection of sites even after they have disappeared from the Web. Web archiving is typically performed using web crawlers. Crawlers periodically harvest the web and update the archive with fresh images (or versions). However, maintaining a good quality of the archive is not a trivial task because the web is evolving over time (generating many page versions) and allocated resources are usually limited (storage space, bandwidth, site politeness rules, etc.). The quality of web archives can be assessed by various measures such as freshness [13], sharpness [15], coherence [30], etc. Among those measures, completeness appears to be the most relevant. Completeness is the ability of the archive to contain the largest amount of page versions. To maintain a complete archive (containing all versions of every page), web pages should be captured continuously whenever there is a (small) change in any page. Of course, this is practically infeasible due the limitation of resources and the large number of pages (and sites) to archive. Nonetheless, it is possible to optimize web crawlers in order to capture the largest number of pages and thus the quality of the archive can be significantly improved. For that, archiving systems must predict the behavior of web sites to be collected in order to guess when and at which frequency each page must be visited.

Up to now, there are three major factors that have been considered to learn how often each page should be revisited: (i) the importance/relevance of pages (*e.g.* PageRank, similarity of keywords to user queries, etc.) [14, 24, 35], (ii) the longevity[4] of information [23] and (iii) the frequency of changes [11, 12, 15, 16]. Another factor that has been ignored so far is the *importance of changes* between pages

[1] This research is supported by the French National Research Agency ANR in the CARTEC Project (ANR-07-MDCO-016).

JCDL'11, June 13–17, 2011, Ottawa, Ontario, Canada.
Copyright 2011 ACM 978-1-4503-0744-4/11/06 ...$10.00.

[2] http://www.archive.org/

[3] http://www.europarchive.org/

[4] Longevity [23] is the lifetimes of content fragments that appear and disappear from web pages over time.

versions. It is very frequent that crawlers waste time and space to retrieve page versions with unimportant changes (*e.g.* advertisements, copyright, decoration, navigation bars, etc.). Moreover, the frequency of change used by the main crawler strategies is estimated based on the homogeneous Poisson process [13]. Unfortunately in [6, 18, 27], researchers demonstrate that Poisson model is only valid when the time granularity of changes is longer than one month which is too far to be the common case on the web.

Our work is applied on the archive of the French National Audiovisual Institute (INA) which preserves radio and television channels sites and related pages. These pages are updated frequently (several times per day) and, hence, the Poisson model can not be used as above explained. By monitoring some TV channels sites, we have observed some periodicity of changes over days. For some pages, there are more important changes during the day than in the night and during workdays than in weekends. Therefore, we had the idea to discover periodic patterns from pages. A pattern models the evolution of the importance of page changes over a period of time. We claim that patterns can be a useful tool to predict changes and thus efficiently archive web pages. Based on patterns, the most significantly changed pages can be visited "at the right moment" optimizing the use of resources (bandwidth, storage space, etc.). Patterns have received a considerable attention and were used in various domains such as web mining [4, 26, 32], networking [28, 34], systems [33, 21, 20, 36], etc. They were generally used to predict trends, to simplify data processing and to improve system performance. However, as far as we know, patterns have never been used in the context of web archiving. In this paper, we demonstrate, through a case study, the utility of patterns to improve the effectiveness of web archives. The archive of French TV channels "*France Télévisions*" has been chosen, as a case study, to validate our approach. We first present different steps and algorithms used to discover patterns from collected web pages. Then, we show how discovered patterns can be used to (i) efficiently index or store web pages and (ii) to optimize web crawlers in order to improve archives quality. The major contributions of this paper can be summarized as follows:

- A definition of a pattern model that represents the evolution of pages changes over accurate period of time (fine granularity).
- A description of different algorithms used, through a case study, to discover patterns from pages. Some observations and trends are also reported.
- A pattern-based strategy to optimize (i) archives quality and (ii) pages indexing/storing.
- Experimental results that show the usefulness and the effectiveness of our pattern based approach.

The rest of the paper is organized as follows. Section 2 discusses related works on pattern mining and on web archiving. Section 3 formally defines our pattern model for web pages. Section 4 presents briefly the different steps followed to discover patterns from web pages. In Section 5, an overview of major pattern benefits for archives is given. Section 6 describes a case study where patterns are discovered from French TV channels pages. An approach based on patterns to optimize indexing and storing is proposed in Section 7. Section 8 shows how patterns can be used to optimize web crawling and to improve data quality. In Section 9, conducted experiments based on real patterns are presented and discussed to demonstrate the effectiveness of our approach. Finally, Section 10 concludes.

2. RELATED WORKS

As mentioned in the introduction, our work is related to pattern mining and web archiving. We present below related works in those two areas.

Pattern Mining. There is a lot of research, publications, and applications where patterns are involved mainly, in the data mining area. Patterns mining is widely used for different applications such as trajectories of objects, weather forecasts, DNA sequences, stock market analysis, etc. It is impossible to give here a complete coverage on this topic but interested readers can refer to [19] for example. The intent of this section is not to describe all available algorithms for pattern discovery. We rather present here some applications, in various domains, where patterns are used to predict trends, simplify data processing and improve system performance. Then, a brief overview of major summarization techniques used to discover patterns is given.

In [26], frequent patterns are discovered from significant intervals of web log data and are used to forecast the user behavior. In the context of network analysis, Sia et al. [28] propose a RSS monitoring algorithm, using user access pattern, to allocate efficiently resources and provide subscribers with fast new alerts. In [34], Risto Vaarandi detects frequent patterns form event log files to help the creation of system profile and the detection of anomalies. In [2] Adar et al. use pattern to describe people's revisitation behavior to understand the relationship between web change and revisitation.

Generally, the methods and algorithms used to discover patterns are developed from several fields such as statistics, data mining, machine learning and pattern recognition. These methods depend on the data type to be treated such as set, sequence, time series, tree, graph, etc. Patterns discovery is based on summarization techniques such as: statistical analysis, association rule, clustering, classification, etc. Statistical analysis are the common methods to give statistical description (trend, periodicity) of pattern based on frequency, mean, median, etc. like in [28]. Association rules are used to discover interesting relations between objects, events, etc. such as in [4]. Clustering, like in [34], is a technique to group together a set of data having similar characteristics. Classification, as used in [10], is the process of mapping a data into one of predefined classes.

To sum up, patterns are widely used in different fields. Depending on the data set, many pattern summarization techniques have been proposed, each with different purpose. However, to the best of our knowledge, patterns had never been exploited in the context of web archiving. In this paper, we show how patterns can be discovered from pages changes and used to improve the efficiency of web archives. Most of existing algorithms search for frequent patterns without any prior knowledge about the form of pattern to be discovered. Here, however, we have an idea about the format of the periodic pattern to be discovered. Our idea is inspired from a long period of monitoring observations. We noticed that time series extracted from the same pages have generally similar change evolution over days. As the patterns we want to discover have a simple format, a very simple statistical analysis (*i.e.* based on changes average) can be used to extract periodic patterns from web pages.

Web Archiving

In recent years, there have been various studies about web archiving. An overview of the main issues involved by web archiving is given by Masanès [22]. In [17], authors address issues concerning the format of information to be stored or indexed in the archive. The specification of the web perimeter was studied in [1, 8]. To improve the freshness of web pages, efficient crawl strategies were proposed in [11, 12] based on a page changes frequency [13]. Brewington and Cybenko [7] estimate how often web sites must be re-indexed for more fresher indexes. A re-crawl scheduling strategy based on information longevity is proposed in [23] to optimize the freshness of web pages. For a better quality of the archive, Cho et al. [14] propose a crawl strategy to download the most important pages first. The importance of pages is defined based on different ways: similarity between pages and queries, number of backlinks over the entire site, rank of a page, etc. Similarly, Castillo et al. [9] download the best ranked pages. Recent studies address the issue of the temporal coherence of a site crawl. Spaniol et al. [30] introduce the notion of temporal coherence and propose a crawling strategy to optimize web sites coherence. In [31], visualization strategies were proposed to help the archivist in understanding the nature of coherence defects. In other studies [15], a framework (SHARC) was implemented to maximize the sharpness of web archives. Sharpness is a quality measure proposed by authors to reflect how much the site's pages have been changed during the crawl.

All those archiving approaches address the challenge: efficiency [1, 17], freshness [7, 11, 12, 23], data quality [14, 31, 15, 30]. Most of web crawler strategies download the most important pages (based on PageRank, etc.) and/or retrieve the most frequently changing pages (based on homogeneous Poisson model). However, the importance of changes between page versions has been ignored so far. Moreover, the homogeneous Poisson model used to estimate changes frequency is not valid for page modified very frequently as in our case study. Similarly to our work, Adar et al. [3] propose models and analysis to characterize the amount of change on the web at finer grain (hourly updates), but they do not propose a method to estimate the importance of changes detected between pages versions. According to them, their change analysis can be used to refine crawler policies by focusing on meaningfull changes, but no strategy has been proposed yet. Though interesting, none of existing archiving approaches, as far as we know, have used patterns and the importance of changes to improve the quality of archives. Using patterns for an optimized web archiving is the core of this paper. We present a case study where patterns are discovered from French TV channels web pages. Then, we demonstrate how patterns can be useful to improve archives quality and pages indexing.

3. PATTERN-BASED MODEL

In this section, we introduce the concept of changes importance. Then a pattern-based model that describes the evolution of page changes over a period of time is defined.

3.1 Page Version Importance

A page version is a capture (or a snapshot) obtained through crawling a page from the web. Given a version of a web page, we define its importance by combining the two following concepts:

1. The importance or the relevance of the pages (*e.g.* PageRank, similarity of keywords to a user query, etc.)

2. The importance of changes between the current version and the last archived one of the same page. The importance of changes between two page versions can be estimated based on the function explained in Section 6.2. This function returns a normalized value between 0 and 1. An importance value near one (respectively near 0) denotes that changes between versions are very important (respectively useless *e.g.* advertisements or decoration).

Definition 3.1.1. ***Version Importance***
Given a version ν_η^i captured from the page P_i, the importance $\omega(\nu_\eta^i)$ of the version is the multiplication of the importance of the page $\omega(P_i)$ by the importance of changes occurred between the current version ν_η^i and the last archived one $\nu_{\eta-1}^i$ of the same page.

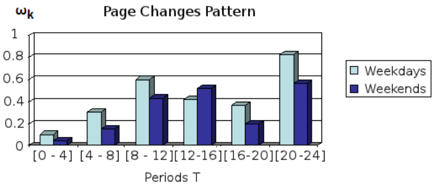
$$\omega(\nu_\eta^i) = \omega(P_i) * ImpCh(\nu_{\eta-1}^i, \nu_\eta^i)$$

By introducing this concept of importance, the strategy of web crawlers can be optimized to download in priority the most relevant pages that have the most important changes since the last archived version.

3.2 Page Change Pattern

A pattern models the evolution of page changes over a period of time (*e.g.* a day) as shown in Figure 1. It represents the change rate (or *importance rate*) over time. It is periodic and may depend on the day of the week and of the hour within a day. Separate patterns can be defined for weekends. We formalize below the definition of pattern based on the importance of changes.

Definition 3.2.1. ***Pattern***
A pattern of a page P_i with an interval length l is a nonempty sequence Patt(P_i) = $\{(\omega_1,T_1)$; ...; (ω_k,T_k);...; $(\omega_{N_T},T_{N_T})\}$, where N_T is the total number of periods in the pattern and ω_k is the average of the importance of changes estimated in the period T_k. The sum of the time periods, $\sum_{k=1}^{N_T} T_k$, is equal to l. We choose l=1 day as the length of the pattern in our case study.



Periods T	[0 - 4]	[4 - 8]	[8 - 12]	[12 - 16]	[16 - 20]	[20 - 24]
Weekdays ω_k	0.1	0.3	0.6	0.42	0.37	0.83
Weekends ω_k	0.04	0.14	0.43	0.52	0.2	0.57

Figure 1: Pattern Example

Example 3.2.1. ***Pattern***
Let Patt(P_1)=$\{$(0.1, [0h-4h]);(0.3, [4h-8h]);(0.6, [8h-12h]); (0.42, [12h-16h]);(0.37, [16h-20h]);(0.83, [20h-24h])$\}$ be a

periodic pattern of the page P_1 for weekdays as shown in Figure 1. 0.1, 0.3, 0,6, 0.42, 0.37 *and* 0.83 *are respectively the average importance of changes for each interval period* $T_1 = [0h - 4h]$,...,$T_6 = [20h - 24h]$. *Also a periodic pattern can be defined separately for weekends as shown in the Figure 1.*

3.3 Time Series

Patterns are discovered from a collection of time series obtained by crawling the page periodically during a long period of time. A time series represents the evolution of the change importance of a specific page during a day.

Definition 3.3.1. ***Time Series***
A time series (Figure 2) of a page P_i with an interval length l=1 day is a nonempty sequence TSeries(P_i) = $\{(\mu_1, T_1)$; ...; (μ_k, T_k);...; $(\mu_{N_T}, T_{N_T})\}$, where μ_k is the importance of changes detected between two versions captured at the begin and the end of the period T_k.

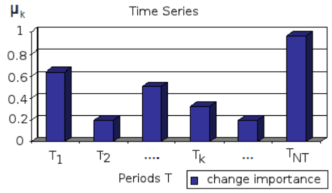


Figure 2: Time Series

4. PATTERN DISCOVERY PHASES

In this section, we present briefly the major phases used to discover patterns from page changes.

In order to discover patterns, page changes are observed and monitored for a long period of time. The framework as shown in Figure 3 illustrates the different phases needed to discover pattern from web pages. It consists of five major modules: The *crawler* periodically downloads web pages; the *web archive* stores and indexes the different versions of pages retrieved by the crawler; the *change detection* module generates delta files describing changes that have occurred between successive page versions; the *change importance estimator* evaluates the importance of changes and represents them by time series; the *pattern discovery* module mines patterns from extracted time series. More details of pattern discovery phases are given below.

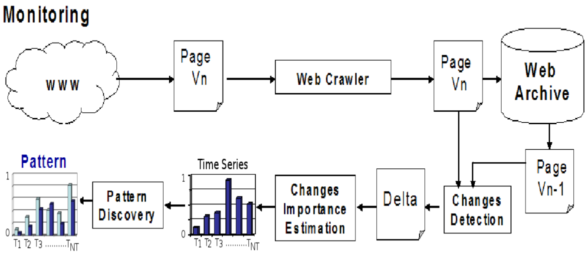


Figure 3: Pattern Discovery Framework

- Pages crawling
  The crawler harvests the web by iteratively downloading new versions of pages. The frequency of the crawler is chosen to be not too long (*e.g* hourly crawl) to do not miss relevant modifications. Each retrieved version is then stored and indexed in the archive.

- Change detection
  Based on the change detection algorithm Vi-DIFF [25], each current version (ν_n) of page is compared with its previously downloaded one (ν_{n-1}). Vi-DIFF uses a visual segmentation algorithm to partition the page into multiple blocks. Then, it detects structural and content changes. Structural changes alter the visual appearance of the page and the structure of its blocks, whereas content changes modify text, hyper-links and images inside page blocks. Then, a delta file describing all these changes is generated. For more details about the Vi-DIFF algorithm, please refer to [25].

- Change importance estimation
  In this phase, the successive delta files generated during a day are evaluated by the function [5] to estimate the importance of changes. This function depends of three major parameters; (i) the importance of each block composing the page, (ii) the importance of changes operations (insertion, deletion, etc.) detected between the two versions and (iii) the amount of change operations occurred on each block. Then, for each page, a time series that represents the evolution of the importance changes during a day is built.

- Pattern discovery
  In this step, the set of time series are analyzed to extract periodic patterns for each page based on statistical analysis (*i.e.* changes average). Patterns are discovered by averaging the importance of changes of time series collected during a long period. Some periodic patterns are defined separately for workdays and weekends. Different pages (of the same site or of different sites) with similar behavior of changes can be grouped to share a common pattern. During an on-line crawl, patterns should be updated periodically to always reflect the current behavior of web pages. Based on discovered patterns, the evolution of page changes can be predicted over accurate periods of time and exploited to optimize web archiving.

5. PATTERN BENEFITS

Our approach based on patterns can improve the effectiveness of web archives at the two following points: (i) optimized pages indexing/storing, and (ii) improved quality of archives by efficient crawling.

5.1 Optimized Storing or Indexing

Due to the limitation of resources such as storage space, bandwidth, site politeness rules, etc., web archive systems must avoid wasting time and space for storing/ indexing some versions with unimportant changes such as advertisements. Based on patterns, one can predict for each page, the period of time when there are important changes on pages. An adequate change importance threshold can be fixed from patterns to decide when it is appropriate to index or store each page. Hence, the optimized index facilitates a fast and an accurate page retrieval from the archive. The speed and the performance in finding relevant/important web pages for a search query will be significantly improved.

5.2 Improved Archive Quality

The role of web crawlers is to decide what page should be refreshed/archived and with which priority. Based on patterns, web crawling can be optimized in such way that it improves the quality of the archive. In this paper, we consider the completeness as the quality measure. Completeness is the ability of the archive to contain the largest amount of useful page versions. Driven by patterns, the web crawler can retrieve the most important versions in order to maintain the archive as complete as possible. An important version is a relevant version of a page that has important changes since the last one archived. Hence, unimportant changes in the page (*e.g.* advertisements, decoration, etc.) can be ignored and useful information is captured by a single crawl, maximizing the use of resources.

6. CASE STUDY: FRANCE TV ARCHIVE

We choose the *France Télévisions* (*FranceTV*) archive as a case study to experiment our pattern based approach. We claim that *FranceTV* site is rich enough to reflect the evolution of changes of major radio and TV channels of the French Web. We have monitored, over a period of one month, more than one hundred pages composing the *FranceTV* site. Each page is crawled every hour, every day. More than 3.000 page versions were obtained every day. We first present the characteristics of this site. Then, we describe algorithms used to extract time series and discover patterns from collected pages versions. Some observations and trends deducted from discovered patterns are finally reported.

6.1 Site characteristics

France Télévisions is the French public national television broadcaster. It includes the TV channels: *France 2*, *France 3*, *France 4*, *France 5*, *France 0* and *Sport.FranceTV*. It also includes ''*La 1 ère*" which is the network of radio and television channels operating in France's overseas departments and territories around the world (*e.g.* Réunion, Martinque, etc.). *FranceTV* site contains a vast and a various amount of web pages that have different changes behaviors. There are dynamic pages that change very frequently (several times per day) like news pages. There are also (few) quasi-static pages that change for example only twice a week. For some pages, there are significantly higher changes during the day than during the night. Due to the time zone difference between France and its overseas departments, periods of work activities are different and this affects the behavior of page changes. Hence, *FranceTV* archive is chosen as a case study due to the diversity of its web pages.

6.2 Time Series Acquisition

As mentioned before, pages of *FranceTV* sites were crawled every hour and saved as a new version. Collected versions of the same page, over a day, are grouped and analyzed to extract time series.

6.2.1 Principle

The acquisition phase of time series from pages changes combines two steps, changes detection and change importance estimation, as presented in Section 4. The first step consists in detecting changes between versions $(\nu^i_{\eta-1}, \nu^i_\eta)$ using Vi-DIFF algorithm (*cf.* Section 4) to generate delta files. Then, the changes, described in deltas, are evaluated and are used to extract time series for each page every day.

As detailed in [5], the importance of changes detected in the delta Δ is computed by the following function based on three major parameters:

(i) the importance *Imp(Bk)* of each block composing the page. It can be evaluated based on the method of [29],
(ii) the percentage *PerCh(Bk,Op)* of change operations (insert, delete, etc.) occurred on each block,
(iii) the importance *Imp(Op)* of changes operations. For example, insert can be considered more important than a delete.
The importance of changes *ImpCh*(Δ)

$$= \sum_{i=1}^{N_{Bk}} Imp(Bk_i) * \frac{1}{N_{Op}} \sum_{j=1}^{N_{Op}} Imp(Op_j) * PerCh(Bk_i, Op_j)$$

where
- N_{Op}, N_{Bk} are respectively the number of operations and the number of blocks.
- $Op = \{insert, delete, update\}$
- $\sum_{i=1}^{N_{Bk}} ImpBk_i = 1$

Larger the amount of changes inside important blocks is, higher the estimated importance of changes.

In our case study, the importance of each block was manually estimated. The percentage of changes occurred in each block is computed from the delta file Δ. It represents the amount of change operations detected for each block divided by the total number of elements in the block (text, link, etc.). The importance of operations depends on both the operation type (delete, insert, etc.) and on the type of element (link, image, etc.) affected by the change. In the rest of the paper, we note the importance of an operation *Op* over an element *El* by *Imp(Op,El)*. Furthermore, we would like to give less importance for an advertisement link or image. An advertisement link (or image) has usually its address (or name) that includes the term ''advertisement". Also, we want to consider the distance between compared texts (of two versions) while computing the importance of changes. Thus, the following requirements have been introduced to estimate the importance of an operation *Imp(Op,El)*.

- $Imp(Op, Text) = distance(Text(\nu_{n-1}), Text(\nu_n))$
- $Imp(Op, El) = \begin{cases} \alpha \quad if \quad El \quad is \quad Advertisement \\ 1 \quad else \end{cases}$

where
$El = \{link, image\}$ and $Op = \{insert, delete, update\}$
-The textual distance computes the proportion of distinct words between the two compared texts.
- If an advertisement element (link or image) is modified, a low value α is given to the importance of an operation (*e.g.* α=0.1). Otherwise, the importance of an operation $Imp(Op, El)$ is equal to 1.

6.2.2 Time Series Acquisition Algorithm

We use Algorithm 1 to extract time series from collected page versions. A new time series is created every day for each page. At each period (line 3-6), the changes between successive versions of the same page are detected, evaluated and then added to time series. The change detection (*detectChanges()* in line 4) is done by calling Vi-DIFF algorithm (*cf.* Section 4). A pseudo code of the function that estimates the importance of change in the delta (line 5) is depicted by algorithm 2. The importance of changes, as shown in algorithm 2, is computed by averaging the percentage of changes in each block (line 5), the importance of change operation (line 6-11) and the importance of block (line 16). At

Algorithm 1 Time Series Acquisition

```
Input:
P_1, P_2,..., P_n - list of pages
ν^i_1, ν^i_2, ..., ν^i_{N_T} - versions of page P_i
Output:
TSeriesList - time series of pages
Begin
1. for each page $P_i$ i=1...,$n$ do
2. TSeries=newTimeSeries()
3.     for each version $\nu^i_k$ of period $T_k$ k=1...,$N_T$ do
4.         $\Delta$=DetectChanges($\nu^i_{k-1}$,$\nu^i_k$)
5.         $\mu$=EstimateChangeImportance($\Delta$)
6.         TSeries.add($\mu$,$T_k$)
7.     end for
8. TSeriesList.add($P_i$,TSeries,date)
9. end for
10. Return TSeriesList
End
```

the end of algorithm 1, a collection of time series describing the change importance of each page is obtained for a given date (line 8-10). Figure 4 shows four time series samples obtained from *Sport.FranceTV.fr* page during weekdays. It clearly appears that there is similar periodicity of changes over the four days. Furthermore, the periodicity of changes for some pages may depend on the time of the day.

Algorithm 2 Change Importance Estimation

```
Input:
$\Delta$ - delta file
Output:
importance - estimated changes importance
Begin
for each block Bk in $\Delta$ do
    ImpBk=getBlockImportance()
    sum=0
    for each operation Op over an element El $\in$ Bk do
        PerCh=getPercentageChanges(Op,El,Bk)
        if (El is Text) then
            ImpOp=distance(Text($\nu_{n-1}$),Text($\nu_n$))
        else if (El is advertisement) then
            ImpOp=$\alpha$
        else
            ImpOp=1
        end if
        end if
        sum=sum+ImpOp*PerCh
    end for
importance=importance+(sum/$N_{op}$)*ImpBk
end for
Return importance
End
```

6.3 Pattern Discovery

The strategy used to discover patterns from already collected time series is described in this Section.

6.3.1 Principle

Our approach used to discover pattern is based on statistical summarization technique (*i.e.* change average). It consists on averaging the importance of changes of each time series collected during a long period. In other words, the importance of changes (at each period) of a pattern is the sum of the changes importance of all time series (at the same period) divided by the total number of collected time series (of the same page). Separate patterns can be learned for a specific day of the week or for weekends. For the pages that change few times per week (or per month), a weekly (or monthly) pattern can be defined.

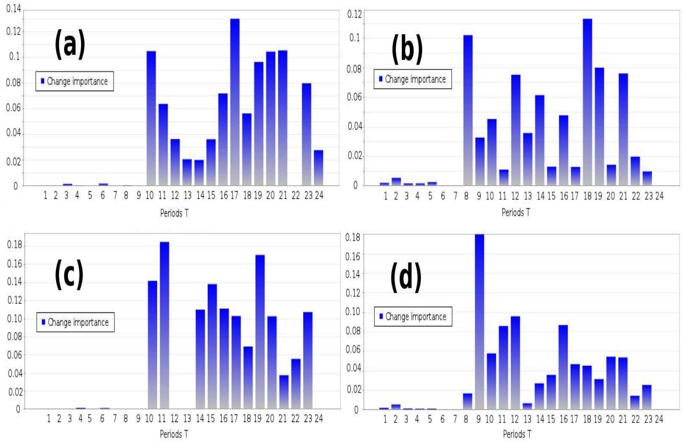


Figure 4: Samples of Time Series

Assume that TS=$\{TSeries_1,\ TSeries_2,...,\ TSeries_{N_{Tseries}}\}$ is the list of time series of the page P_i collected during a long period. Let $Patt(P_i)$=$\{(\omega_1,T_1);\ ...;\ (\omega_k,T_k);...;\ (\omega_{N_T},T_{N_T})\}$ be the pattern to discover from TS. The average importance of changes ω_k defined in $Patt(P_i)$ at the period T_k is computed as follows.

$$\omega_k = \frac{\sum_{j=1}^{N_{Tseries}} \mu_k}{N_{Tseries}}$$

where

- $TSeries_j(\mathrm{P}_i) = \{(\mu_1,T_1);\ ...;\ (\mu_k,T_k);...;\ (\mu_{N_T},T_{N_T})\}$.
- μ_k is the importance of changes of $TSeries_j(P_i)$ at the period T_k.
- $N_{Tseries}$ is the total number of collected times series of P_i.

6.3.2 Pattern Discovery Algorithm

A pseudo code of the pattern discovery algorithm is depicted by Algorithm 3. For each page P_i, a new pattern is mined from the collection of time series. The average importance of changes ω_k is computed for each period T_k of every pattern (line 3-9). At the end of the algorithm, the collection of discovered patterns of all pages is returned (line 11-13).

Figure 5 shows samples of patterns obtained from the following pages; (a) *programmes.France3.fr*, (b) *La1ère.fr*, (c) *Sport.FranceTV.fr* and (d) *documentaire.France5.fr.*

6.3.3 Observations

We have observed from some patterns significant changes during the day (respectively during the night) and during the weekdays (respectively during weekends). Some pages (e.g. *"La 1 ère.fr"*, etc.) have similar changes importance during the day and during the night. This is due to the time zone difference between France and its overseas departments that affects the periods of work activities. Separate patterns

Algorithm 3 Pattern Discovery

```
Input:
P_1, P_2,..., P_n - list of pages
TSeries_1, TSeries_2,..., TSeries_{N_{Tseries}} - times series of P_i
Output:
Patterns - list of discovered patterns
Begin
for each page $P_i$ i=1...,$n$ do
    Patt=newPattern()
    for each period $T_k$ k=1,..$N_T$ do
        imp=0
        for each $TSeries_j$ of $P_i$ j=1,..,$N_{Tseries}$ do
            imp=imp+ TSeries.getImportance($T_k$)
        end for
        $\omega$=imp/$N_{Tseries}$
        Patt.add($\omega$,$T_k$)
    end for
    Patterns.add($P_i$,Patt)
end for
Return Patterns
End
```

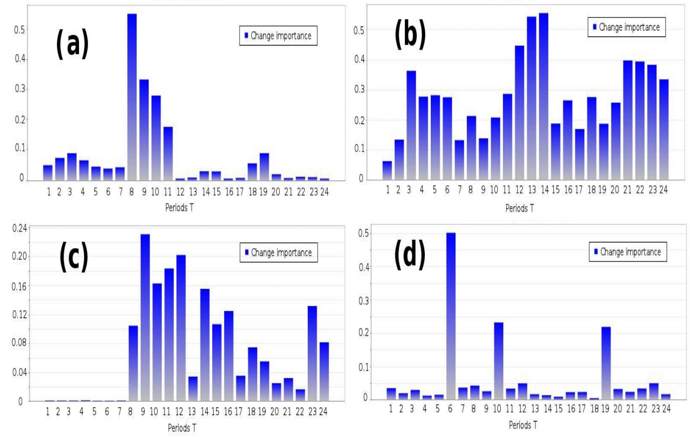


Figure 5: Samples of Discovered Patterns

are also learned for some pages at specific days (holidays) because their change behaviors differ from usual. In addition, we have observed that the pages at the highest level in the site (*e.g.* home pages) have more significant changes than the ones at the deepest levels in the site, which remain quasi static.

6.3.4 Shared Patterns

From obtained patterns, it clearly appears that some pages (*e.g.* "France2.culture.fr" and "France3.culture.fr") have the same template and share similar content information. Also, they present similar change behaviors. Hence, a common pattern can be defined for such pages. Other pages with different content belonging to the same site (or to a different site) have similar changes evolution. Such pages can also be grouped to share a common pattern. Patterns can be learned according to the depth of the page in the site. For instance, a common pattern can be defined for some pages at the deepest level because they change rarely. Further study should be done to learn an efficient methods to create automatically a collection of pages that share common patterns.

7. PATTERN-BASED INDEXING

We show in this section how patterns can be exploited to optimize page indexing.

7.1 Principle

An appropriate change importance threshold is fixed to decide if a retrieved version should be indexed or no. Hence, archive systems avoid wasting time and space for indexing some versions with unimportant changes. As the pattern gives an idea of the evolution of change importance over time, accurate value of threshold δ can be learned for each page. One way to fix the index threshold δ of each page is to compute the average of changes importance of its pattern.

The index threshold of the page P_i is the sum of all change importance ω_k in $Patt(P_i)$, divided by the number of period N_T.

$$\delta(P_i) = \frac{\sum_{k=1}^{N_T} \omega_k(Patt(P_i))}{N_T}$$

where Patt(P$_i$) = $\{(\omega_1,T_1)$; ...; (ω_k,T_k);...; $(\omega_{N_T},T_{N_T})\}$
The threshold δ can also serve to optimize storing in case of a limited storage space. This threshold can be adjusted with respect to allocated resources by using machine learning algorithms for example.

7.2 Pattern-based Indexing Algorithm

As depicted by algorithm 4, if the importance of changes between the current version and the last stored one (line 5-7) is higher than the threshold δ, the page version is indexed in the archive. Otherwise, the estimated importance is cumulated over time (line 8-10). Whenever this cumulated importance of changes becomes higher than the fixed threshold, the current version is indexed/stored (line 5-6).

Algorithm 4 Pattern-based Indexing

```
Input:
$\delta(P_1)$, $\delta(P_2)$,..., $\delta(P_n)$ - index thresholds of P_i
$\nu_1^i$, $\nu_2^i$,..., $\nu_k^i$ - versions of page P_i
Begin
1. imp=0
2. for each new version $\nu_\eta^i$ of $P_i$ do
3.     $\Delta$=DetectChanges($\nu_{\eta-1}^i$,$\nu_\eta^i$)
4.     $\mu$=EstimateChangeImportance($\Delta$)
5.     if $\mu$ >=$\delta(P_i)$ or imp>=$\delta(P_i)$ then
6.         index the new version $\nu_\eta^i$ of $P_i$
7.         imp=0
8.     else
9.         imp=imp+$\mu$
10.    end if
11. end for
End
```

Figure 6 shows an example of threshold used to index the page *France4.fr*.

8. PATTERN-BASED CRAWLING

We have described in Section 6.3 how patterns are discovered from web pages. In this section, we show how patterns can be exploited to optimize the strategy of web crawlers.

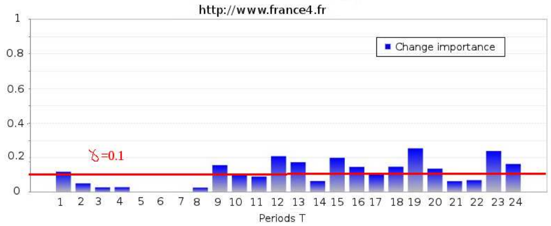


Figure 6: Example of Indexing Threshold

8.1 Principle

The pattern-based strategy consists in downloading, in priority, the most important pages to improve the effectiveness of the archive. The scheduler orders the pages to be crawled based on an urgency (or a priority) which depends on the corresponding pattern.

The urgency *U(P_i,t)* of downloading the page P_i at current time t is

$$U(P_i, t) = \omega(P_i) * \omega_k * (t - t_{lastRefresh})$$

where
- Patt(P$_i$) = $\{(\omega_1,T_1)$; ...; (ω_k,T_k);...; $(\omega_{N_T},T_{N_T})\}$.
- t is the current time (t $\in T_k$).
- ω_k is the average of change importance defined by the pattern for the period T_k.
- $\omega(P_i)$ is the the page's importance.
- $t_{lastRefresh}$ is the last time of refreshing the page P_i.

Only the M first pages with the highest urgency are captured at each period T_k.

8.2 Pattern-based Crawling Algorithm

The pseudo code of the pattern-based crawler is depicted by Algorithm 5. At each period T_k (line 2-5), the urgency of refreshing each page is computed then added to the list *crawlListPages* in descending order. After that, the crawler downloads the M top pages with the highest urgency value. For each page, the current version and the last archived one are compared to build the delta file (line 9-11). The changes occurred in the delta are then evaluated. The result of the importance of change is used to maintain patterns up-to-date. In fact, the average change importance ω_k of patterns should constantly be reevaluated to reflect the real web. If more (respectively less) changes are detected in period T_k, the *Update* function (line 13) recomputes the average importance of changes ω_k of patterns. Also, the value of $\omega(P_i)$ is periodically recomputed because the importance of pages (*e.g.* PageRank) changes over time in the web.

9. EXPERIMENTAL RESULTS

We have conducted experiments to evaluate the efficiency of our pattern-based approach in improving the quality of the archive. For that, patterns discovered from *FranceTV* web pages were exploited to describe the behavior of page changes and used to optimize web crawling. In particular, the completeness rate obtained by different crawl strategies is compared. The completeness of an archive is the proportion of captured versions divided by the total amount of changes that occurred on web sites. In particular, we distinguish a weighted and not weighted completeness measure. The weighted completeness considers the importance of versions (*cf.* Section 3.1), whereas the not weighted one only counts the number of versions without considering their importance.

Algorithm 5 Pattern-based Crawler

```
Input:
P_1, P_2,..., P_n - list of pages
Patt(P_1), Patt(P_2),..., Patt(P_n) - patterns of pages

Begin
1. for each period $T_k$ do
2.   crawlListPages=newList()
3.   for each page P$_i$, i=1...,$n$ do
4.     compute $U(P_i, t) = \omega(P_i) * \omega_k * (t - t_{lastRefresh})$
5.     crawlListPages.add(P$_i$,$U(P_i, t)$)/*descending order*/
6.   end for
7.   for i=1...,M do
8.     P$_i$=crawlListPages.selectPage(i)
9.     $\nu_\eta^i$=downloadPage(P$_i$)
10.    $\nu_{\eta-1}^i$=getLastVersion(P$_i$)
11.    $\Delta$=detectChanges($\nu_{\eta-1}^i$,$\nu_\eta^i$)
12.    $\mu$=EstimateChangesImportance($\Delta$)
13.    Update(Patt(P$_i$),$\mu$,$T_k$)
14.    $t_{lastRefresh}$ = t
15.  end for
16. end for
End
```

As it is impossible to capture exactly all page changes occurred on web sites to measure the completeness, we have simulated the changes importance of web pages based on patterns obtained from *FranceTV*. The performance of different crawl strategies within a controlled test environment are evaluated. Experiments written in Java were conducted on PC running Linux over a 3.20 GHz Intel Pentium 4 processor with 1.0 GB of RAM. At the begin of each experiment, each page is described by a real pattern that has been discovered from *FranceTV* web pages. The importance of changes of each page is generated according to defined patterns. In addition the following parameters are set: the number of pages per site, the duration of simulation, the number of periods in patterns N_T, the number of allocated resources M. The first experiments were performed to see the impact of the number (or the length) of periods defined in patterns on the effectiveness of the pattern-based strategy. It exists a clear trade-off in deciding how long should be each period T_k defined in pattern. If the length of periods is too long (*i.e.* there are few periods), the discovered pattern may be inaccurate and hence our strategy may be less efficient. Figure 7 shows the completeness obtained by our pattern-based strategy under resource constraints M={10%, 20%, 30%} with respect to the number of periods (N_T) defined in patterns. The length of each period T (in hours) and the corresponding periods' numbers N_T defined in patterns are summarized in the following table.

N_T	2	3	4	6	8	12	24	30	40	60	80	120
T(h)	12	8	6	4	3	2	1	$\frac{4}{30}$	$\frac{1}{10}$	$\frac{2}{30}$	$\frac{1}{20}$	$\frac{1}{30}$

As we can see on Figure 7, the completeness (under resources constants 10%, 20% and 30%) increases with the number of periods defined in pattern. When more periods are defined in the pattern (*i.e.* short period length), better completeness is achieved by our pattern-based strategy.

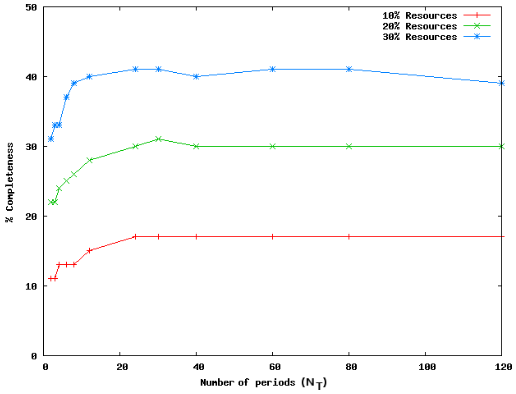


Figure 7: Pattern Time Periods

The achieved completeness remains quasi-constant for patterns having a number of periods higher than 24, *i.e.* a period length shorter than one hour. Even if the period length of the pattern is too short (less than 30 minutes), the achieved completeness is not furthermore improved. Hence, the length of periods defined in patterns should be around one hour (or shorter) to reach the best completeness rate.

Other experiments were conducted to evaluate the total completeness obtained by our *Pattern* strategy compared to the following crawl policies: *Relevance* [14] downloads firstly the most important pages (*i.e.* based on PageRank) in a fixed order, *Frequency* [12] selects pages to be crawled according to their frequency of changes estimated by a Poisson model [13], *SHARC* [15] repeatedly retrieves the entire site and ensure that the most changing pages are downloaded close to the middle of the capture interval and *Coherence* [30] repeatedly downloads the entire sites by starting with the pages that have the lowest probability to cause incoherence in the archive. All experiments that have been conducted to evaluate those strategies are done under the same conditions (*i.e.* a maximum of M pages can be captured at each period T). Figure 8 shows the completeness obtained by the different strategies with respect to the percentage of allocated resources M. In this experiment, the importance of changes (or versions) is not considered. Our pattern based strategy depends on the rate of changes instead of the average of the changes importance. Figure 9 shows the weighted completeness achieved by the different crawl strategies. It presents the percentage of the importance of versions (*cf.* Section 3.1) obtained with respect to allocated resources. On both figures, we can see that the completeness increases with the number of allocated resources. Moreover, it is clear that *Pattern* strategy outperforms the other approaches *Relevance*, *SHARC*, *Coherence* and *Frequency*. It improves the completeness of archives by around 10 % and the weighted completeness by around 20 % in case of limited resources. Through experiments, we demonstrated that the pattern-based strategy (based on the concept of importance or not) improves significantly the quality of the archive. Nonetheless, *Pattern* strategy, which considers the importance of changes, enables to donwload more important versions (20 % better) compared to other strategies. Hence, it is the best strategy to avoid wasting time and space by retrieving usefulness page versions.

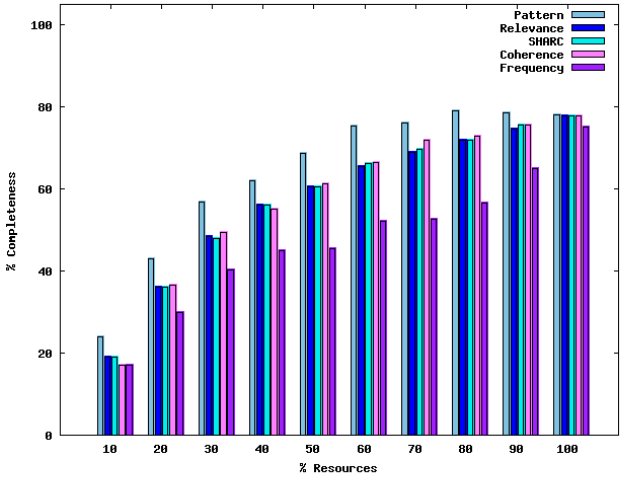


Figure 8: Completeness

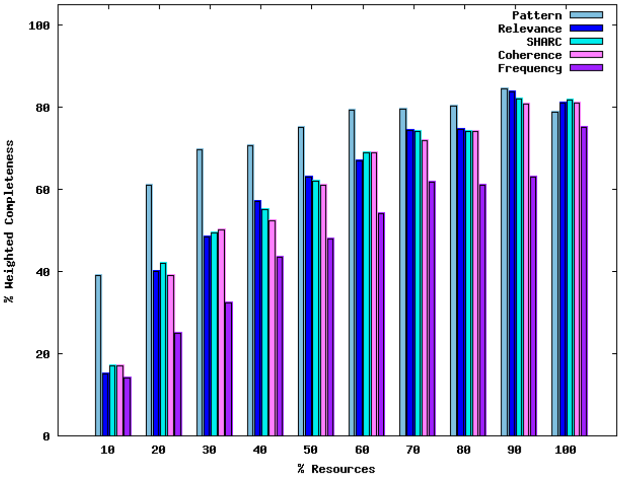


Figure 9: Weighted Completeness

10. CONCLUSION AND FUTURE WORK

The major challenge of web archiving institutes is to provide a rich archive and to preserve its quality. In this paper, we proposed a novel approach based on patterns (i) to efficiently index/store web pages and (ii) to improve archives quality. The quality of an archive is defined by the completeness measure which ensures that the archive contains the largest amount of useful pages versions. As far as we know, this work is the first to use patterns in the context of web archiving. Through a case study, we presented algorithms used to discover patterns from French Televisions channels pages. A method using patterns is proposed to optimize page indexing and storing. Also, we proposed a crawl strategy based on pattern to improve the quality of archives. Evaluation experiments were performed based on real patterns obtained from *FranceTV* pages. These experiments show that our approach outperforms its competitors, obtaining a completeness gain up to 20 % in case of limited resources. This improvement is also due to considering the importance of changes between versions which has been ignored so far. As a future direction, we intend to explore other pattern summarization techniques such as association rules to predict the behavior of some unexpected changes on web pages. Also, we plan to run our pattern strategy over a larger number of web pages of French National Institute (INA). Another future research direction is to create a method that automatically determines collections of web pages that share a common pattern. Finally, we are also interested in studying the impact of pattern strategy on other quality measures of archives such sharpness, freshness, etc.

11. REFERENCES

[1] S. Abiteboul, G. Cobena, J. Masanes, and G. Sedrati. A First Experience in Archiving the French Web. In *Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, 2002.

[2] E. Adar, J. Teevan, and S. T. Dumais. Resonance on the web: web dynamics and revisitation patterns. In *Proceedings of the 27th international conference on Human factors in computing systems*, Boston, MA, USA, 2009.

[3] E. Adar, J. Teevan, S. T. Dumais, and J. L. Elsas. The web changes everything: understanding the dynamics of web content. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, Barcelona, Spain, 2009.

[4] S. Baron and M. Spiliopoulou. Monitoring the evolution of web usage patterns. In *Lecture Notes in Computer Science*, pages 181–200. Springer, 2004.

[5] M. Ben Saad and S. Gançarski. Using visual pages analysis for optimizing web archiving. In *EDBT/ICDT PhD Workshops*, 2010.

[6] B. Brewington and G. Cybenko. How dynamic is the web? In *World Wide Web conference (WWW'2000)*, pages 257–276, 2000.

[7] B. E. Brewington and G. Cybenko. Keeping up with the changing web. *Computer*, 33(5):52–58, 2000.

[8] D. J. C. Lampos, M. Eirinaki and M. Vazirgiannis. Archiving the greek web. In *4th International Web Archiving Workshop (IWAW04)*, Bath, UK, 2004.

[9] C. Castillo, M. Marin, A. Rodriguez, and R. Baeza-Yates. Scheduling algorithms for web crawling. In *LA-WEBMEDIA '04: Proceedings of the WebMedia & LA-Web 2004 Joint Conference 10th Brazilian Symposium on Multimedia and the Web 2nd Latin American Web Congress*, pages 10–17, 2004.

[10] H. Cheng, X. Yan, J. Han, and C. wei Hsu. Discriminative frequent pattern analysis for effective classification. In *ICDE*, pages 716–725, 2007.

[11] J. Cho and H. Garcia-Molina. The Evolution of the Web and Implications for an Incremental Crawler. In *VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases*, 2000.

[12] J. Cho and H. Garcia-Molina. Effective page refresh policies for web crawlers. *ACM Trans. Database Syst.*, 28(4):390–426, 2003.

[13] J. Cho and H. Garcia-Molina. Estimating frequency of change. *ACM Trans. Interet Technol.*, 3(3), 2003.

[14] J. Cho, H. Garcia-molina, and L. Page. Efficient crawling through url ordering. In *Computer Networks and ISDN Systems*, pages 161–172, 1998.

[15] D. Denev, A. Mazeika, M. Spaniol, and G. Weikum. Sharc: framework for quality-conscious web archiving. *Proc. VLDB Endow.*, 2(1):586–597, 2009.

[16] J. Edwards, K. McCurley, and J. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 106–113, 2001.

[17] D. Gomes, A. L. Santos, and M. J. Silva. Managing duplicates in a web archive. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, 2006.

[18] D. Gruhl, R. Guha, D. Liben-nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW '04*, pages 491–501, 2004.

[19] J. Han, H. Cheng, D. Xin, and X. Yan. Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, 15:55–86, 2007.

[20] Z. Li, Z. Chen, S. M. Srinivasan, and Y. Zhou. C-miner: Mining block correlations in storage systems. In *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*, pages 173–186, 2004.

[21] Z. Li, S. Lu, S. Myagmar, and Y. Zhou. Cp-miner: a tool for finding copy-paste and related bugs in operating system code. In *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, pages 20–20, 2004.

[22] J. Masanès. *Web Archiving*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[23] C. Olston and S. Pandey. Recrawl scheduling based on information longevity. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, pages 437–446, 2008.

[24] S. Pandey and C. Olston. User-centric web crawling. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 401–411, 2005.

[25] Z. Pehlivan, M. Ben Saad, and S. Gançarski. Vi-diff: Understanding web pages changes. In *21st International Conference on Database and Expert Systems Applications (DEXA'10)*, Bilbao, Spain, 2010.

[26] K. Saxena and R. Shukla. Significant interval and frequent pattern discovery in web log data. *CoRR*, abs/1002.1185, 2010.

[27] K. C. Sia, J. Cho, and H.-K. Cho. Efficient monitoring algorithm for fast news alerts. *IEEE Transactions on Knowledge and Data Engineering*, 19:950–961, 2007.

[28] K. C. Sia, J. Cho, K. Hino, Y. Chi, S. Zhu, and B. L. Tseng. Monitoring RSS feeds based on user browsing pattern. In *ICWSM '07: International Conference on Weblogs and Social Media*, 2007.

[29] R. Song, H. Liu, J.-R. Wen, and W.-Y. Ma. Learning block importance models for web pages. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 2004.

[30] M. Spaniol, D. Denev, A. Mazeika, G. Weikum, and P. Senellart. Data quality in web archiving. In *WICOW '09: Proceedings of the 3rd workshop on Information credibility on the web*, pages 19–26, 2009.

[31] M. Spaniol, A. Mazeika, D. Denev, and G. Weikum. "catch me if you can": Visual analysis of coherence defects in web archiving. In *9th International Web Archiving Workshop (IWAW 2009) : Workshop Proceecdings*, pages 27–37, Corfu, Greece, 2009.

[32] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explor. Newsl.*, 1:12–23, January 2000.

[33] T. Ueda, Y. Hirate, and H. Yamana. Exploiting idle cpu cores to improve file access performance. In *ICUIMC '09: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, pages 529–535, 2009.

[34] R. Vaarandi. A data clustering algorithm for mining patterns from event logs. In *IEEE IPOM'03 Proceedings*, pages 119–126, 2003.

[35] J. L. Wolf, M. S. Squillante, P. S. Yu, J. Sethuraman, and L. Ozsen. Optimal crawling strategies for web search engines. In *Proceedings of the 11th international conference on World Wide Web*, WWW '02, pages 136–147, 2002.

[36] L. H. Yang, M. L. Lee, and W. Hsu. Efficient mining of xml query patterns for caching. In *Proceedings of the 29th international conference on Very large data bases - Volume 29*, VLDB '2003, pages 69–80, 2003.