Data Quality in Web Archiving

Marc Spaniol, Dimitar Denev,
Arturas Mazeika, and Gerhard Weikum
Max-Planck-Institut für Informatik
Campus E1 4
Saarbrücken, Germany
{mspaniol|ddenev|amazeika|weikum}@mpi-inf.mpg.de

Pierre Senellart
Institut TELECOM
TELECOM Paristech
CNRS LTCI
Paris, France
pierre.senellart@telecom-paristech.fr

ABSTRACT

Web archives preserve the history of Web sites and have high long-term value for media and business analysts. Such archives are maintained by periodically re-crawling entire Web sites of interest. From an archivist's point of view, the ideal case to ensure highest possible data quality of the archive would be to "freeze" the complete contents of an entire Web site during the time span of crawling and capturing the site. Of course, this is practically infeasible. To comply with the politeness specification of a Web site, the crawler needs to pause between subsequent http requests in order to avoid unduly high load on the site's http server. As a consequence, capturing a large Web site may span hours or even days, which increases the risk that contents collected so far are incoherent with the parts that are still to be crawled. This paper introduces a model for identifying coherent sections of an archive and, thus, measuring the data quality in Web archiving. Additionally, we present a crawling strategy that aims to ensure archive coherence by minimizing the diffusion of Web site captures. Preliminary experiments demonstrate the usefulness of the model and the effectiveness of the strategy.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Measurement

Keywords

Web Archiving, Data Quality, Temporal Coherence

1. INTRODUCTION

The goal of Web archiving is to preserve the history of Web sites by repeatedly crawling entire sites and adding versions of both page-contents and page-link structures to an append-only archive. The most well-known endeavor of this kind is the work of the Internet Archive, but national libraries and national archives also have specialized activities along these lines [15]. Capturing the history of digitally born information and preserving the cultural and political zeitgeist of an era offers a potential gold mine for all kinds of media and business analysts, such as political scientists, sociologists, media psychologists, market analysts, and intellectual-property lawyers. For example, one could analyze the effectiveness of political campaigns, search Web sites for prior art that may be relevant in assessing patent applications, or test former versions of Web sites for compliance with country-specific legal rules for Internet media. In essence, a Web archive is a data warehouse for Internet contents, and the quality of the captured data – such as completeness and mutual consistency of a Web site's pages – is decisive for building value-added applications on top of an archive.

Despite the initiatives of the International Internet Preservation Consortium (IIPC)[1] and the Internet Archive[2] in capturing Web contents for future generations, limitations such as storage space, bandwidth, and crawling politeness as well as threats such as Web spam and crawler traps heavily affect the crawling performance and, thus, the quality of the collected data. Current methods are based on snapshot crawls and "exact duplicate" detection [15]. The coherence of data in terms of proper dating and proper cross-linkage is influenced by the temporal characteristics (duration, frequency, etc.) of the crawl process.

Web archiving is commonly understood as a continuous process that aims at archiving the entire Web (broad scope). However, a typical scenario in archiving institutions or companies is to periodically – e.g. monthly – create high quality captures of a certain Web site. These periodic domain scope crawls of Web sites aim at obtaining a best possible representation of a site. A reason for customers having their site archived on a regular basis is, for instance, to guard itself against accusations regarding intellectual property rights, fraud or non-compliance with legal requirements (e.g. EU laws about imprints, terms of use, etc.). Figure 1 contains an abstract representation of such a domain scope crawling process. This Web site consists of n pages $(p_1, \ldots, p_n)$. Each of them consists of several successive versions, indicated by the horizontal lines (e.g., p_n has three different versions in $[t; t']$). Ideally, the result of a crawl would be a complete and instantaneous snapshot of all pages at a given point of time. In reality, one crawl requires an extended time period to gather all pages of a site while being potentially modified in

WICOW'09, April 20, 2009, Madrid, Spain.

[1] `http://netpreserve.org`
[2] `http://www.archive.org`

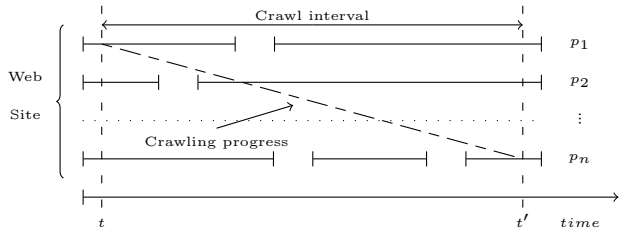

Figure 1: Web site crawling process (domain scope)

parallel, causing thus incoherencies in the archive. The risk of incoherence increases further due to politeness constraints and need for sophisticated time stamping mechanisms.

An ideal approach to Web archiving would be to have captures for every domain at any point in time whenever there is a (small) change in any of the domain's pages. Of course, this is absolutely infeasible given the enormous size of the Web, high content-production rates in blogs and other Web 2.0 venues, the disk and server costs of a Web archive, and also the politeness rules that Web sites impose on crawlers. We therefore settle for the realistic goal capturing Web sites at convenient points (whenever the crawler decides to devote resources to the site and the site does not seem to be highly loaded), but when doing so, the capture should be as "authentic" as possible. In order to ensure an "as of time point x (or interval $[x, y]$)" capture of a Web site, we need to develop an archiving crawler that *ensures coherence of crawls regarding a time point or interval*, and *identifies those contents of the Web site that violate coherence.*

This paper formalizes these intuitive considerations and develops a model for Web archive quality centered around the notion of the coherence of a site capture. We discuss various ramifications of this model, identifying practically feasible approaches, and we present appropriate crawling strategies that aim to maximize the quality of a Web archive within given resource constraints.

The rest of the paper is organized as follows. In Section 2 we give an overview on related research and point out the key aspects that make Web archiving very different from crawling and indexing for search engines. In Section 3, we give a formal definition of our archive coherence model and discuss the underlying assumptions. Based on this model we explain how to measure and (in the subsequent step) assure the coherence of an archive with respect to a time point or time interval. In Section 4, we present the strategies toward optimized archive coherence. Our strategies increase the probability of obtaining largely coherent crawls and are compared with conventional crawling strategies in Section 5.

2. RELATED WORK

The Web is a continuously evolving network of contents and an interconnecting link structure, which requires developing strategies for adopting to change in content, size, topology and use [14]. Hence, the Web might not be understood in a holistic approach of a single discipline, but needs to be best investigated in a combined approach with methodologies adopted from various disciplines. In this aspect, ongoing research in the field of Web science [9] tries to incorporate approaches from various disciplines, such as algorithmics [2], data mining [4], physics [1], sociology [12] and media theory [10]. However, related research is up to now mostly confined to approaches originating from a single discipline. Primarily, research is focused on improving the efficiency of crawlers for Web indexing in search engines or crawler development in general. The contribution of Pant et al. [19] describes the technical process involved in Web crawling. A quite technical overview on the Heritrix crawler is given by Mohr et al.. However, they do not address the issue of coherence [16]. B. E. Brewington and G. Cybenko [3] analyze changes of Web sites and draw conclusions about how often they must be reindexed. The issue of crawl efficiency is addressed by Cho et al. [8]. They state that the design of a good crawler is important for many reasons (e.g. ordering and frequency of URLs to be visited) and present an algorithm that obtains more relevant pages (according to their definition) first. In a subsequent study Cho and Garcia-Molina describe the development of an effective incremental crawler [5]. They aim at improving the collection's freshness by bringing in new pages in a more timely manner. Into the same direction head their studies on effective page refresh policies for Web crawlers [6]. Here, they introduce a poisson process based change model of data sources. In another study, they estimated the frequency of change of online data [7]. For that purpose, they developed several frequency estimators in order to improve Web crawlers and Web caches. In a similar direction goes research of Olston and Pandey [18] who propose a recrawl schedule based on information longevity in order to achieve good freshness. Another study about crawling strategies is presented by Najork and Wiener [17]. They have found out that breadth-first search downloads hot pages first, but also that the average quality of the pages decreases over time. Therefore, they suggest performing strict breadth-first search in order to enhance the likeliness to retrieve important pages first. Research on improving the scalability of a Web crawler in order to crawl 6 billion pages and beyond is presented by Lee et al. [13]. Their findings show that changing the BFS crawling order and designing low-overhead disk-based data structures increase the efficiency of large-scale crawlers. A dedicated survey about the evolution and dynamics of wikis as social networks is done by Klamma and Haasler [11]. Interesting in this paper is the disclosure of social networks based on the hierarchical structure of important and unimportant nodes.

Summarizing, before mentioned related research depicts some of the key challenges to be solved for ensuring crawl coherence. As we have figured out already, related research mostly focusses on aligning crawlers towards more efficient and fresher Web indexes. However, aiming at an improved crawl performance against the background of assured crawl coherence requires a slightly different alignment. Our task therefore is to achieve both: Increase the probability of obtaining largely coherent crawls and identify those contents violating coherence.

3. ARCHIVE COHERENCE MODEL

In this section, we introduce our archive coherence model. We start with basic assumptions and the notation. After that, we introduce our definitions of observable and inducible coherence that will be applied to subsequently quantify coherence. Finally, we express the probability of incoherence.

3.1 Assumptions

In the following, we assume that a Web site to be crawled consists of n Web pages that change over time that occur independent of each other. Time for downloading contents

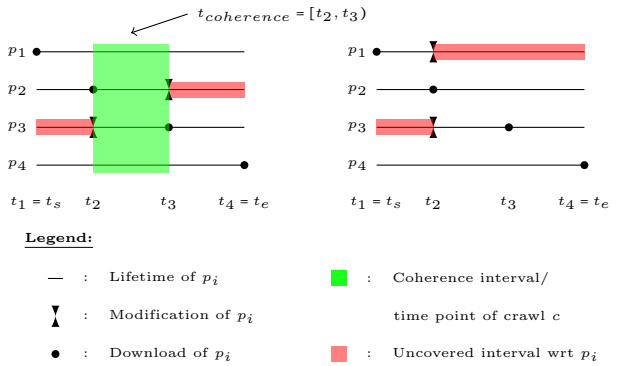

Figure 2: Crawl c containing coherence interval (left) and without coherence interval (right)

is neglected and the time between any two subsequent downloads is equal. Change probabilities are considered to be known for any two pages.

3.2 Notation

We assume that a Web site to be crawled consists of n Web pages numbered $\{p_1, \ldots, p_n\}$. Changes of these Web pages occur per time unit immediately before download and independent of each other according to the probabilities $\lambda_1, \ldots, \lambda_n$ that are associated with the Web page of the corresponding number. We assume that the delay Δt between the downloads of the pages is the same, and the download time is neglected. For convenience $[t_s, t_e]$ denotes the crawl interval, where $t_s = t_1$ is the starting point (download of the first page) of the crawl and $t_e = \Delta t \cdot n = t_n$ is the ending point of the crawl (download of the last page). The time of downloading page p_i is denoted as $t(p_i) = t_j$ having $j \in [1, n]$. In addition, we assume to retrieve the last modified stamps of pages $\mu_1, \ldots, \mu_n$ (having $\mu_i \leq t(p_i)$) upon download. We call the consecutive process of downloading the Web pages $\{p_1, \ldots, p_n\}$ of an entire Web site a crawl c.

3.3 Definitions

The following definitions are based on a common notion of "coherence" applied to the issue of Web archiving and – thus – particularly to Web crawling. Our understanding of coherence refers to the explanations given in the Oxford English Dictionary[3] describing coherence as "the action or fact of cleaving or sticking together", which – in terms of a Web site – results in a "harmonious connexion of the several parts, so that the whole 'hangs together'".

Defintion 3.1. *Coherence*
1. A single Web page is always coherent.
2. The invariance interval $[\mu_i, \mu_i^*]$ of page p_i lies between the last modified time stamp μ_i at time $t(p_i)$ of downloading p_i ($\mu_i \leq t(p_i)$) and the next change μ_i^* following $t(p_i)$.
3. Two or more pages are coherent if there is a time point (or interval) $t_{coherence}$ so that a non-empty intersection among the invariance interval of all pages exists:

$$\forall p_i, \exists t_{coherence} : t_{coherence} \in \bigcap_{i=1}^{n} [\mu_i, \mu_i^*] \neq \varnothing$$

Figure 2 depicts our definition of coherence in a graphical representation of a Web site consisting of four pages $(p_1, \ldots, p_4)$.

[3] `http://dictionary.oed.com`

The download of a page p_i by the crawler is indicated by a black circle. As described above, we assume that there is a single download per time unit. In our example, the download sequence is given as p_1, p_2, p_3, p_4 occurring at t_1, t_2, t_3, t_4. Even more, all pages exist during the whole crawl interval $[t_1, t_4]$. However, some pages are subject to changes taking place in the crawl interval. In the example on the left hand side of figure 2, page p_2 changes at t_3 and page p_3 changes at t_2. In combination with the crawl sequence mentioned before, this results in a coherence interval spanning from t_2 till t_3 ($t_{coherence} = [t_2, t_3)$). In the example on the right hand side of figure 2, page p_1 and p_3 change at t_2. In consideration of the same crawl sequence as before, this results in an empty coherence interval ($t_{coherence} \in \varnothing$). Since the union of the uncovered intervals of pages p_1 and p_3 spans the whole crawl interval, there is not a single time point that ensures an "as of time point x (or interval $[x, y]$)" capture of this Web site.

From a practical point of view, the definition of coherence introduced before is of limited value only. The key point is simple: A real life crawler is "left-hand side aware", but "right-hand side blind". Since a Web page might change immediately after its download, a crawler can only be certain about the appearance of page p_i within an interval lasting from (if available) this page's last modification date μ_i until its time of download $t(p_i)$ during the course of crawl c. To this end, we now introduce a definition of observable coherence, which allows a crawler to disclose coherence based on the last modified stamp of Web pages.

Defintion 3.2. *Observable Coherence*
Two or more Web contents are observable coherent if there is a single time point $t_{coherence}$ so that there is a non-empty intersection of the intervals spanning the respective download time $t(p_i)$ and the corresponding last modified stamp μ_i retrieved at time of download ($\mu_i \leq t_i$):

$$\forall p_i, \exists t_{coherence} : t_{coherence} \in \bigcap_{i=1}^{n} [\mu_i, t(p_i)] \neq \varnothing$$

Due to the fact that a crawler is "left-hand side aware", but "right-hand side blind" the time point $t_{coherence}$ needs to be carefully selected. In case, a coherence statement is desired about all crawled contents of a Web site a specific case of observable coherence – called measurability – is required. Given only the knowledge obtained through a crawl c spanning the time interval $[t_1, t_n]$, measurability is given only for a single coherence time point ($t_1 = t_{coherence}$).

Lemma 3.3. *Measurable Coherence*
Given a Crawl c measurable coherence can only be given if the crawl is observable coherent. Measurability is only given for $t_1 = t_{coherence}$.

Proof. *Assume the contrary and pick a random page p_i downloaded after p_1 (that means $t(p_1) < t(p_i)$) to become the reference time point of assurance. Without the loss of generality we pick the next page (that means p_2) to become the new reference time point of assurance. Since we require measurability we have to intersect over all measurable coherence intervals. These are (at least) $[\mu_1, t(p_1)]$ and $[\mu_2, t(p_2)]$, having $\mu_2 \leq t(p_2)$. However, when intersecting the intervals with respect to any other point than t_1 (like $t_{coherence} = t_2$ in this case), the intersection is empty by definition since the interval of p_1 ends at $t(p_1) = t_1$.* □

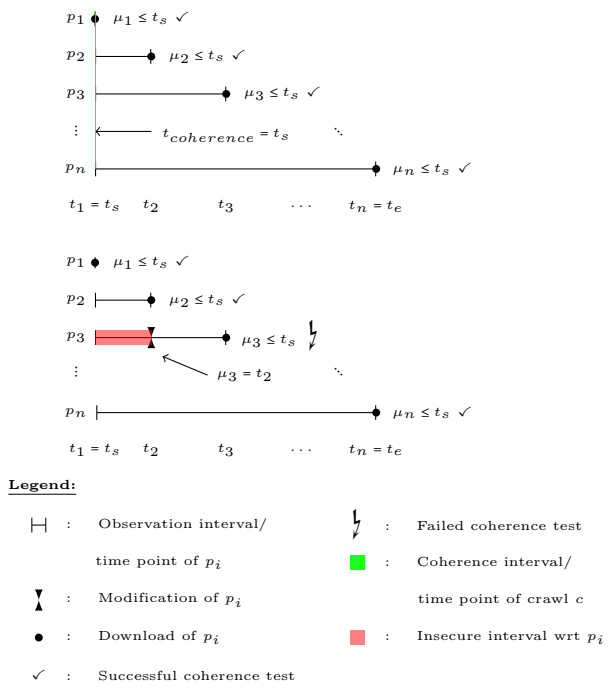

Figure 3: Measurable coherence fulfilled (top) and violation of measurable coherence (bottom)

Figure 3 highlights the concept of measurable coherence (as a specialization of observable coherence) in a graphical representation of a Web site that consists of n pages. Assuming a download sequence $p_1, \ldots, p_n$ spanning the crawl interval $[t_1, t_n]$ the respective observation intervals/time points span from t_1 (wrt p_1) up to $[t_1, t_n]$ (wrt p_n). As a consequence, the risk of a single Web page p_i being incoherent heavily depends on its position during the course of crawl c, which will be subject to further investigation in section 3.4. The top of figure 3 depicts n successful tests of measurable coherence. This results in an assurable coherence statement for the entire Web site valid at time point $t_{coherence} = t_s$. By contrast, the bottom of figure 3 indicates a failed measurable coherence test for page p_3. In this case, page p_3 was modified at t_2, which resulted in an updated last modified stamp ($\mu_3 = t_2$). For that reason, there might not be given an assurable coherence statement for the entire Web site, since the time interval $[t_1, t_2)$ is insecure with respect to page p_3.

In reality and against the background of large Web sites it appears almost impossible to achieve an assurable coherence statement. Though, we might not be able to assure coherence for an entire Web site, we might still be interested in specifying how "coherent" the remaining parts of our crawl c are. For that purpose, we introduce a metric that allows us to quantify the quality of a crawl c.

Defintion 3.4. *Quantifying Measurable Coherence*
First we define an error function $f(p_i)$ that counts the occurring incoherences during the crawl:

$$f(p_i) = \begin{cases} 0 & , \; if \;\; \mu_i \le t_s \\ 1 & , \; else \end{cases}$$

The overall quality of a crawl c is then defined as:

$$q(c) = \frac{\sum_{i=1}^{n} f(p_i)}{n}, \;\; n \ge 1$$

Unfortunately, the reliability of last modified stamps cannot be guaranteed due to missing trustworthiness of Web servers. Hence, the only 100% reliable method is to self create a "virtual time stamp" by comparing the page's etag or content hash with its previously downloaded version. To this end, we introduce an induced coherence measure that allows to gain full control over the contents being compared.

We now apply a crawl-revisit sequence $\Pi(c, r)$, where r is a subsequent revisit of the previously crawled set of Web pages $\{p_1, \ldots, p_n\}$. In this consecutive revisit process we obtain a second (and potentially different) version of the previously crawled pages denoted as $\{\tilde{p}_1, \ldots, \tilde{p}_n\}$. Hence, the crawl-revisit sequence $\Pi(c, r)$ consists of n crawl-revisit tuples $\pi(p_i, \tilde{p}_i)$ having $i \in \{1, n\}$. Technically, the last crawled page p_v having $t(p_v) = n$ is not revisited again, but considered as crawled and revisited page at the same time. Hence, the revisit takes place in the time interval $[t_{n+1}, t_{2n-1}]$. As for visits, the time of downloading page $\tilde{p}_i$ is denominated as $t(\tilde{p}_i) = t_k$ now having $k \in [n, 2n-1]$. In accordance with the definition of the crawl interval, for convenience we denote $[\tilde{t}_s, \tilde{t}_e]$ to be the revisit interval, where $t_e = \tilde{t}_s = t_n$ is the starting point of the revisit (download of the last visited page that is at the same time the first revisited page) and $\tilde{t}_s = \Delta t \cdot (n-1) = \tilde{t}_e$ is the ending point of the revisit (download of the last revisited page). In addition, we define the etag or content hash of a page or an revisited page as $\theta(m)$ having $m \in \{p_i, \tilde{p}_i\}$. Overall, a complete crawl-revisit sequence $\Pi(c, r)$ spans the interval $[t_1, t_{2n-1}]$. It starts at $t_s = t_1$ with the first download of the crawl and ends at $\tilde{t}_e = t_{2n-1}$ with the last revisit download.

Defintion 3.5. *Inducible Coherence*
Two or more Web contents are inducible coherent if there is a time point $t_{coherence}$ between the visitation of pages $t(p_i)$ and the subsequent revisits $t(\tilde{p}_i)$ where the etag or content hash of corresponding pages ($\theta(m)$ having $m \in \{p_i, \tilde{p}_i\}$) has not changed:

$$\forall p_i, \exists t_{coherence} : \theta(p_i) = \theta(\tilde{p}_i) \wedge t_{coherence} \in \bigcap_{i=1}^{n} [t(p_i), t(\tilde{p}_i)]$$

Figure 4 highlights the functioning of inducible coherence applied to a Web site consisting of n pages. We assume a download sequence $p_1, \ldots, p_n$ spanning the crawl interval $[t_1, t_n]$ and an inverted subsequent revisit sequence $\tilde{p}_n, \ldots, \tilde{p}_1$ spanning the revisit interval $[t_n, t_{2n-1}]$. Like with measurable coherence, the risk of a single Web page p_i being incoherent heavily depends on its position in the crawl-revisit sequence $\Pi(c, r)$, which will be subject to further investigation in section 3.4. The left upper part of figure 4 depicts n successful tests of inducible coherence. This results in an assurable coherence statement for the entire Web site valid at time point $t_{coherence} = t_n$. By contrast, the lower left section of figure 4 indicates a failed inducible coherence test for the crawl-revisit tuple $\pi(p_3, \tilde{p}_3)$. In this case, page p_3 was modified elsewhere between $t(p_3) = t_3$ and $t(\tilde{p}_3) = \tilde{t}_{n-3} = t_{2n-3}$, which results in a failed inducible coherence test. Different from tests for measurable coherence (cf. figure 3), we are in this case (due to non-existing or non-reliable last modified stamps) not able to determine the exact time point of modification. To this end, we are only able to discover a boolean result because of a failed etag or hash comparison for the crawl-revisit tuple $\pi(p_3, \tilde{p}_3)$. The whole interval is flagged as insecure, even though, the modification might have taken place far

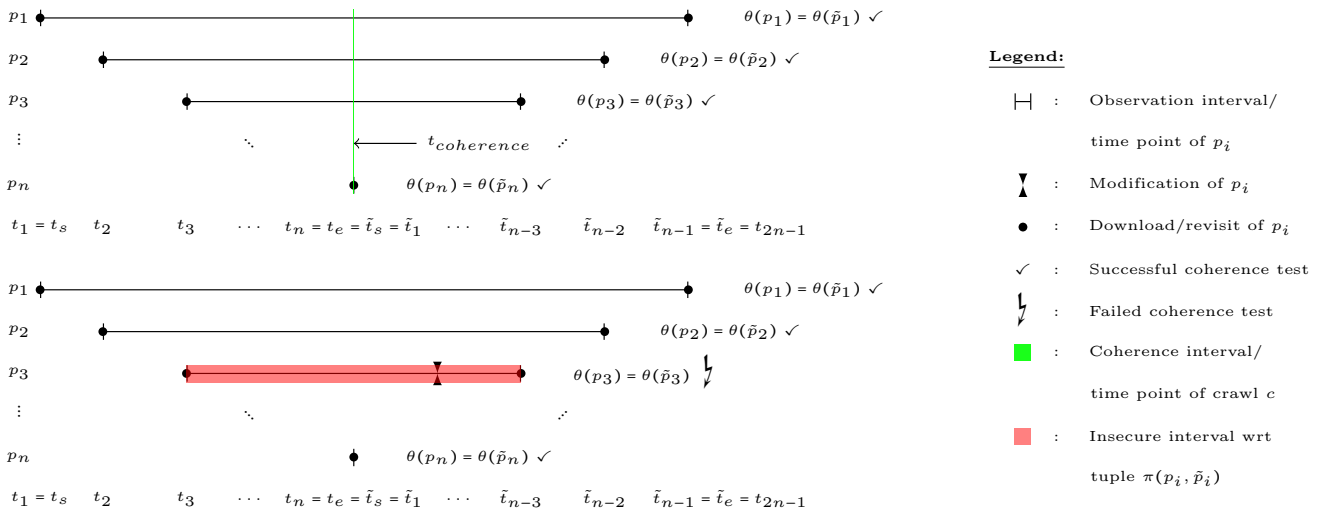

Figure 4: Inducible coherence fulfilled(top) and violation of inducible coherence (bottom)

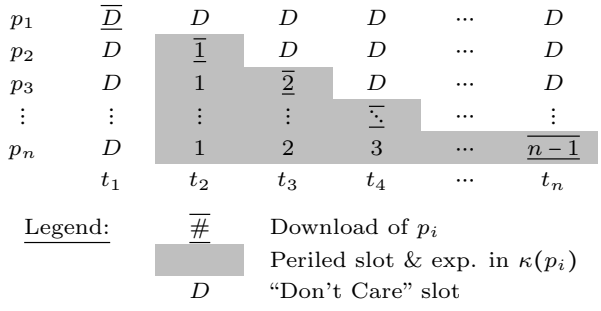

Figure 5: Periled slots in measurable coherence

beyond the aspired coherence time point ($t_{coherence} = t_n$). Thus, despite being coherent from a global point of view for $t_{coherence} = t_n$, a real life crawler might not be able to figure this out (cf. figure 4 for details). Consequently, there might not be given an assurable coherence statement for the entire Web site, since there is an insecure time interval with respect to the crawl-revisit tuple $\pi(p_3, \tilde{p}_3)$.

Likewise for measurable coherence, in reality and against the background of large Web sites it is almost unfeasible to achieve an assurable coherence statement for an entire Web site based on inducible coherence. Though, we might still be interested in specifying how "coherent" the remaining parts of our crawl c are. For that purpose, we introduce a metric that allows us to express the quality of a crawl c.

Definition 3.6. *Quantifying Inducible Coherence*
First we define an error function $f(\pi(p_i, \tilde{p}_i))$ that counts the occurring incoherences for crawl-revisit tuple $\pi(p_i, \tilde{p}_i)$ of the crawl-revisit sequence $\Pi(c, r)$:

$$f(\pi(p_i, \tilde{p}_i)) = \begin{cases} 0 & , \; if \; \theta(p_i) = \theta(\tilde{p}_i) \\ 1 & , \; else \end{cases}$$

The overall quality of a crawl c is then defined as:

$$q(c) = \frac{\sum_{i=1}^{n} f(\pi(p_i, \tilde{p}_i))}{n}, \; n \geq 1$$

Given the previous definitions we are able to evaluate the quality of a crawl c. Since we intend to increase the overall quality, we examine the probability (and thus the risk) of crawling incoherent contents.

3.4 Probability of Incoherence

The probability of a single page p_i being incoherent with respect to the reference time point or time interval $t_{coherence}$ is an important parameter to consider when scheduling a crawl. Incoherence occurs, when a page p_i is subject to one or more modifications μ_i' that are in "conflict" with the ongoing crawl, which is either based on measurable coherence (cf. section 3.3) or on inducible coherence (cf. section 3.5) with respect to the reference coherence time point $t_{coherence}$.

3.4.1 Conflict Probability in Measurable Coherence

A conflict of measurable coherence is given if:

$$\exists \mu_i' : \mu_i' \in [t_s, t(p_i)] \tag{1}$$

That means, a page has been modified at least once since the start of the crawl t_s and the time of downloading this page $t(p_i)$. Given a page's change probability λ_i and its download time $t(p_i)$, the probability of conflict $\kappa(p_i)$ is given as:

$$\kappa(p_i) = 1 - (1 - \lambda_i)^{t(p_i) - t_s} \tag{2}$$

Figure 5 shows in a graphical representation the pages of a Web site $p_1, \ldots, p_n$ (vertically) to be crawled spanning the crawl interval $[t_1, t_n]$ (horizontally). Given a crawl ordering from top to bottom of pages p_i to be downloaded, the diagram differentiates between those slots where a change of page p_i affects the coherence of crawl c and others that do not. The result is a set of concatenated slots – different in size – that represents (overall) the risk of a crawl being affected by changes. Even more, the length of each slot can be understood as the magnitude of the exponent of $\kappa(p_i)$ in equation 2. As can be easily seen, this risk of conflict exponentially increases with the time of download t_i.

3.4.2 Conflict Probability in Inducible Coherence

A conflict of inducible coherence occurs if:

$$\exists \mu_i' : \mu_i' \in [t(p_i), t(\tilde{p}_i)] \tag{3}$$

That means, a page has been modified at least once since its download during the crawling phase $t(p_i)$ and its revisit $t(\tilde{p}_i)$. Given a page's change probability λ_i, its download time $t(p_i)$ and its revisit time $t(\tilde{p}_i)$, the probability of conflict

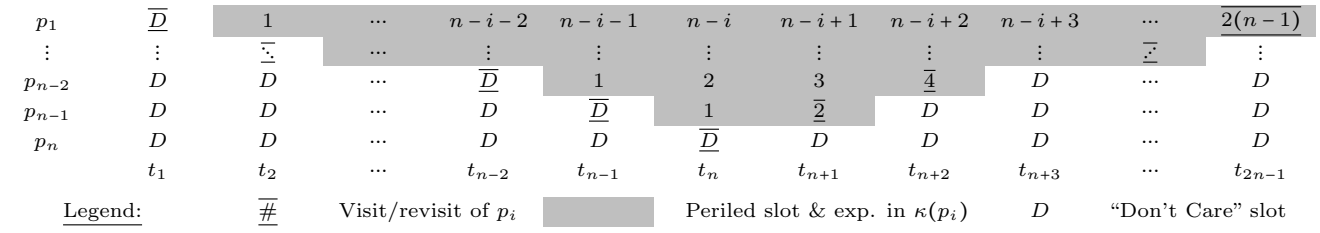

Figure 6: Periled slots in inducible coherence

$\kappa(p_i)$ is given as:

$$\kappa(p_i) = 1 - (1 - \lambda_i)^{t(\tilde{p}_i) - t(p_i)} \quad (4)$$

Potentially conflicting slots in applying inducible coherence are shown in figure 6. In this example, a crawl ordering from top to bottom $(p_1, \ldots, p_n)$ and revisits from bottom to top $(p_{n-1}, \ldots, p_1)$ is being applied. Likewise in the previous case, the illustration differentiates between those slots where a change of page p_i affects the coherence of crawl c and others that do not. Again, the result is a set of concatenated slots – different in size – that represents (overall) the risk of a crawl being affected by changes. However, since the crawl c now takes twice the time, each periled slot is now double in size. That means, the necessity of applying inducible coherence increases the risk of conflict in the exponent of $\kappa(p_i)$ by a magnitude of 2 (cf. equation 4).

As a consequence, from the previous observations we can identify two factors that influence the potential incoherence of a page p_i with respect to the reference coherence time point $t_{coherence}$: Page p_i's change probability λ_i and its download (and revisit) time $t(p_i)$ (and $t(\tilde{p}_i)$). Hence, we will now introduce coherence optimized crawling strategies incorporating both factors.

4. COHERENCE IMPROVED CRAWLING

Conventional archiving crawlers are based on a priority-driven variant of breadth-first-search (BFS) crawling. Even more, they do not incorporate any knowledge about Web sites that have been archived before. However, since information about change probabilities can be derived, our crawling strategy makes use of this "untapped" resource. In addition, there is no revisit concept for "virtual time stamping" in conventional implementations of archiving crawlers.

4.1 Crawling for Measurable Coherence

In case of archiving a Web site that provides precise time stamps of Web contents we apply our measurable coherence approach. In a first step, we compute the change probabilities λ_i of those pages p_i to be crawled. Pages are then sorted according to their change probability in descending order and sent into the crawling queue. Our aim is to find a schedule, which allows us to assign all slots (from small to large) according to the triangle-like shape in figure 5. Hence, we introduce a user definable (e.g. by the crawl engineer) threshold η, which allows to specify the readiness to assume risk encountering incoherence upon download. The threshold η is to this end evaluated against the conflict probability (cf. formula 2) of the next page scheduled for download p_{slot}, in order to discard it if a "less risky" page exists in the schedule. We start with assigning the uncritical slot (as we assume that changes of Web pages might occur only per time unit immediately before download) with length 0 at the first position ($p_{slot} = 1$) to the most critical content ("joker" position). Given t_1, formula 2 becomes zero regardless of λ_i. However, from now on t increases stepwise so that any downloaded content bears the risk of having changed since the start of our crawl. In case, the currently assumed conflict probability is less than the given threshold ($\kappa(p_{slot}) < \eta$), the page p_{slot} is downloaded and the crawl is continued. But, if this condition is not fulfilled, we skip the page at p_{slot} for the moment being. Crawling then continuous until the n^{th} position of our queue is reached. Then we continue downloading those pages that have been skipped in the previous stage. In order to keep up a (depending on the predefined threshold η) slight chance of still downloading coherent contents, we now proceed downloading pages in reversed order until all n pages have been downloaded. A pseudo code implementation of the strategy described is shown in figure 7.

```
input: p_1, ..., p_n - list of pages in descending order of λ_i,
       η - readiness to assume risk threshold
begin
    Start with: slot = 1
    while slot ≤ n
    do
        if κ(p_slot) < η then    /* no conflict expected */
            Download the page p_slot
        end
        Continue with next iteration: slot++
    end
    Download skipped pages in reversed order of their index
end
```

Figure 7: Measurable coherence crawling

4.2 Crawling for Inducible Coherence

Due to the unreliability or non-existence of last modified stamps in most real life crawls, there is a need to ensure coherence based on inducible coherence. As outlined in section 3.5 this method is based on self created "virtual time stamps" by comparing the page's etag or content hash with its previously downloaded version in a three-stage process.

Starting point is a list of pages p_i to be crawled sorted in descending order according to their change probabilities λ_i. Like before, the intention is to identify those pages that might overstep the readiness to assume risk threshold η. Since now all pages need to be scheduled according to the reference time point $t_{reference} = t_n$ being the last page to be crawled during the crawling phase, we need a different queuing strategy: We try to create a V-like access schedule having the (large) slots of stable pages on top and the (small) slots of instable ones at bottom (cf. figure 6). Again, we start with assigning the uncritical slot (as we assume that changes of Web pages might occur only per time unit immediately before download) with length 0 to the most critical content at the first position ($p_{slot} = 1$) of our queue. Since, initially, the length of the slot in the "joker" position (t_n) to be assigned is zero, the threshold condition does not hold. However, from now on

```
input: p_1, ..., p_n - list of pages in descending order of λ_i,
       η - readiness to assume risk threshold
begin
    Start with: slot = 1, last_promising = n
    while slot ≤ last_promising
    do
        if κ(p_slot) ≥ η then     /* conflict expected!  */
            Move p_slot to position last_promising
            Decrease promising boundary: last_promising − −
        end
        else
            Increase promising boundary: promising + +
        end
    end
    slot = n  while slot ≥ 1
    do          /* visit from hopeless to promising */
        Download page p_slot
        Decrease slot counter: slot − −
    end
    slot = 2  while slot ≤ n
    do        /* revisit from promising to hopeless */
        Revisit page p_slot
        Increase slot counter: slot + +
    end
end
```

Figure 8: Inducible coherence crawling

t (and thus the size of slots) increases stepwise so that any download bears the risk of being incoherent. To this end, we evaluate the current page's conflict probability (cf. formula 4) against the user defined threshold ($\kappa(p_{slot}) \geq \eta$). As it is rarely possible to include all pages in this V-like structure, we split the download schedule into a promising section and a hopeless section. In case, the given threshold is exceeded we move the page at p_{slot} to the $last_{promising}$ position, which is the (at this point in time) the first position after those pages not exceeding the conflict threshold η. Otherwise, the page will be scheduled for download at p_{slot}. This process is continued until all pages p_i have been scheduled either in the promising section or the hopeless section. In the next stage, the crawl itself starts. During the crawling phase, we begin with the most hopeless ones first until we continue with those pages that have been allocated in the promising section. After completion, we directly initiate the revisit phase in the reverse order. We begin with the first element after the "joker" position ($p_{slot} = 2$) until the revisit of the remaining pages has been completed. A pseudo code implementation of the strategy described is shown in figure 8.

5. EXPERIMENTAL RESULTS

We now compare our approach toward coherence improved crawling with related strategies. Experiments were run on synthetic data in order to investigate the performance of versatile crawling strategies within a controlled test environment. In order to resemble real life conditions, we simulated small to medium size crawls of Web sites consisting of 10.000 – 50.000 contents. In addition, we simulated the sites' change behavior to vary from nearly static to almost unstable.

All experiments followed the same procedure, but varied in size of Web contents and change rate. Each page of the data set has a change probability λ_i in the interval $[0; 1]$. Within the simulation environment a change history was generated, which registered every change per time unit. The probability that page p_i changed at t_j is $P(\mu_i) = P[\chi(t_j) \leq \lambda_i]$ where $\chi(t_j)$ is a function that generates per time unit a uniformly distributed random number in $[0; 1]$.

As mentioned before, conventional implementations of archiving crawlers are based on a breadth-first-search (BFS) crawling strategy and do not incorporate revisits. However, "virtual time stamping" is unavoidable in order to determine coherence under real life crawling conditions. Therefore, we compare our coherence improved crawling strategy based on inducible coherence with crawl revisit pairs based on BFS-LIFO (last in, first out) as well as BFS-FIFO (first in, first out). In addition, we indicate baselines for optimal and worst case crawling strategies, which are obtained from full knowledge about changes within all pages p_i during the entire crawl-revisit interval. Hence, these baselines are only considerable as theoretical achievable limits of coherence.

Figure 9 depicts the results of our improved inducible crawling strategy compared with its "competitors" BFS-LIFO and BFS-FIFO. Our improved crawling strategy always performs better than the best possible conventional crawling strategy. Experiments are based on a Web site containing 10.000 contents and different readiness to assume risk thresholds η ranging from $[0.45; 0.7]$. In addition, our strategy performs about 10% better given non-pathological Web site behaviour (neither completely static nor almost unstable). Values of η between $[0; 0.45)$ or $(0.7; 1]$ perform less effective. They induce an either too "risk-avoidant" ($\eta \in [0; 0.45)$) or too "risk-ignorant" ($\eta \in (0.7; 1]$) scheduling with minor (or even zero) performance gain, e.g. when acting "risk-ignorant" in heavily changing sites or "risk-avoidant" in mostly static sites.

Comparable results have also been produced given larger (and smaller) Web sites having similar change distributions in numerous experiments. In addition, our strategy introduced to improve measurable coherence shows similar performance, considering a by factor 2 decreased exponent of $\kappa(p_i)$.

6. CONCLUSIONS AND OUTLOOK

Data quality in Web archiving was and – with the advent of Web 2.0 technologies – is an important issue in order to preserve our digital culture. As we have figured out in this paper, temporal coherence in Web archiving is a key issue in order to capture digital contents in a reproducible and, thus, later on interpretable manner. To this end, we have given an overview on strategies that help to overcome (or at least identify) the temporal diffusion of Web crawls that last from a view hours only up to several days. Based on the coherence framework introduced, we have shown how to schedule crawls in order to reduce the risk of crawling contents being incoherent. Even more, our experimental results have shown that we are able to improve the data quality in Web archiving by around 10% for non-pathological (neither completely static nor almost unstable) Web sites.

While the approach presented in this paper is capable of capturing Web sites as "authentic" as possible given a certain time point or interval, our long-term objective – of course – is to capture the Web for any given point in time. Hence, we currently pursue studies on extending our time point (or short interval) based coherence model to larger time-frames. By doing so, we try to increase the coverage of our archiving strategies from "isolated" time points toward a full coverage. In this aspect, we intend to incorporate partial revisit strategies that pay more attention to those contents, being more likely to change. We also plan to develop more sophisticated machine learning techniques that will help us to identify change probabilities of Web contents accurately.

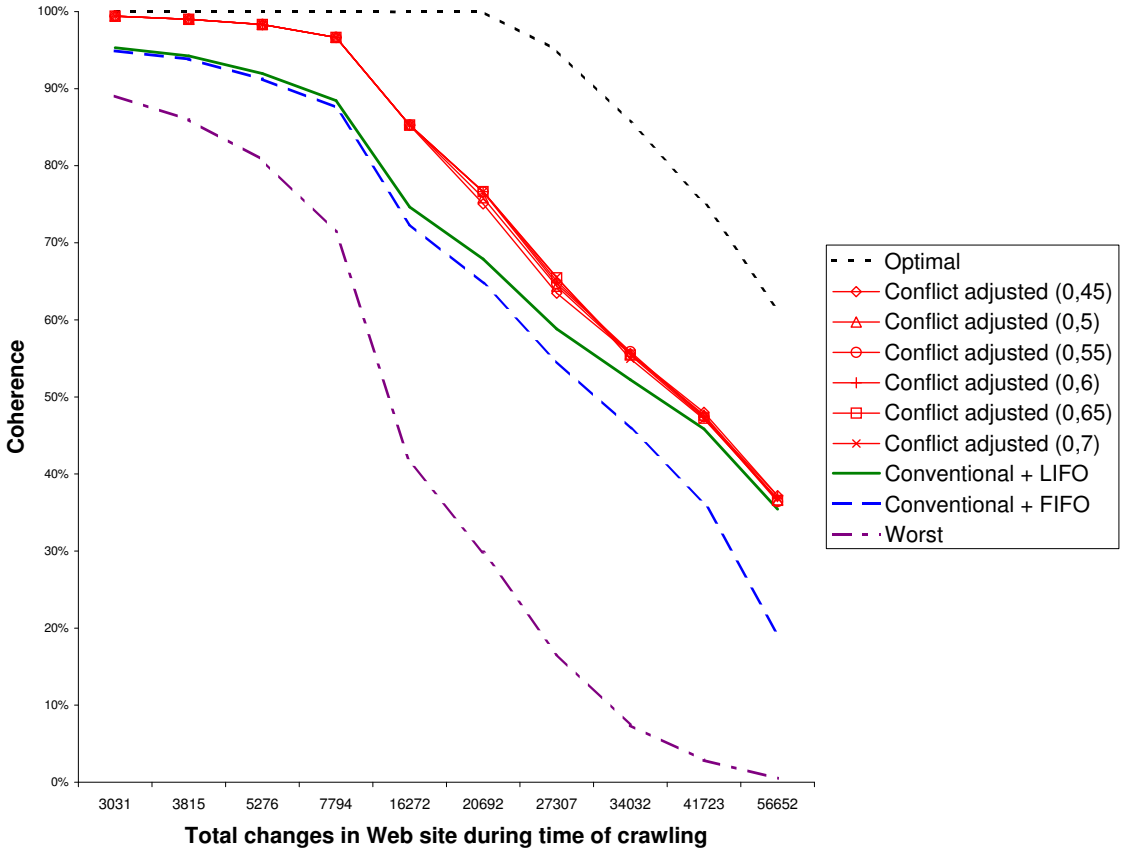

Figure 9: Comparison of inducible crawling strategies in a Web site with 10.000 contents

Acknowledgements

This work was supported by the 7^{th} Framework IST programme of the EC through the small or medium-scale focused research project (STREP) on Living Web Archives (LiWA) contract no. 216267. We thank our colleagues for the inspiring discussions.

7. REFERENCES

[1] A.-L. Barabási. The Physics of the Web. *Physics World*, 14:33–38, 7 2001.

[2] U. Brandes and T. Erlebach (eds). *Network Analysis - Methodological Foundations*, *LNCS* vol. 3418. Springer, Berlin Heidelberg New York, 2005.

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

[4] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, August 2003.

[5] J. Cho and H. Garcia-Molina. The evolution of the web and implications for an incremental crawler. In *VLDB '00: Proc. of the 26th intl. conf. on Very Large Data Bases*, pages 200–209, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[6] J. Cho and H. Garcia-Molina. Effective Page Refresh Policies for Web Crawlers. *ACM Transactions on Database Systems*, 28(4), 2003.

[7] J. Cho and H. Garcia-Molina. Estimating Frequency of Change. *ACM Trans. Inter. Tech.*, 3(3):256–290, Aug. 2003.

[8] J. Cho, H. Garcia-Molina, and L. Page. Efficient Crawling through URL ordering. In *WWW7: Proc. of the 7th intl. conf. on World Wide Web 7*, pages 161–172, Amsterdam, The Netherlands, 1998. Elsevier Science Publishers B. V.

[9] J. Hendler, N. Shadbolt, W. Hall, T. Berners-Lee, and D. Weitzner. Web Science: An interdisciplinary approach to understanding the Web. *Commun. ACM*, 51(7):60–69, 2008.

[10] L. Jäger. Transkriptivität - zur medialen Logik der kulturellen Semantik (in German). In L. Jäger and G. Stanitzek (eds), *Transkribieren – Medien/Lektüre*, pages 19–41. Fink, München, 2002.

[11] R. Klamma and C. Haasler. Wikis as Social Networks: Evolution and Dynamics. In *2nd SNA-KDD Workshop Social Network Mining and Analysis.*, 2008.

[12] B. Latour. On recalling ANT. In J. Law and J. Hassard (eds), *Actor-Network Theory and After*, pages 15–25. Oxford, 1999.

[13] H.-T. Lee, D. Leonard, X. Wang, and D. Loguinov. IRLbot: Scaling to 6 Billion Pages and Beyond. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 427–436, New York, NY, USA, 2008. ACM.

[14] M. Levene and A. Poulovassilis (eds). *Web Dynamics – Adapting to Change in Content, Size, Topology and Use*. Springer, 2004.

[15] J. Masanès. *Web Archiving*. Springer, New York, Inc., Secaucus, NJ, 2006.

[16] G. Mohr, M. Kimpton, M. Stack, and I. Ranitovic. Introduction to Heritrix, an Archival Quality Web Crawler. In *4th intl. Web Archiving Workshop (IWAW'04)*, 2004.

[17] M. Najork and J. L. Wiener. Breadth-First Search Crawling Yields High-Quality Pages. In *In Proc. of the 10th intl. World Wide Web conf.*, pages 114–118, 2001.

[18] C. Olston and S. Pandey. Recrawl Scheduling based on Information Longevity. In *WWW '08: Proceeding of the 17th intl. conf. on World Wide Web*, pages 437–446. ACM, 2008.

[19] G. Pant, P. Srinivasan, and F. Menczer. *Web Dynamics – Adapting to Change in Content, Size, Topology and Use*, chapter Crawling the Web, pages 153–178. Springer, 2004.