The Web Changes Everything: Understanding the Dynamics of Web Content

Eytan Adar
University of Washington
Seattle, WA, USA
eadar@cs.washington.edu

Jaime Teevan, Susan T. Dumais
Microsoft Research
Redmond, WA, USA
{teevan, sdumais}@microsoft.com

Jonathan L. Elsas
Carnegie Mellon University
Pittsburgh, PA, USA
jelsas@cs.cmu.edu

ABSTRACT

The Web is a dynamic, ever changing collection of information. This paper explores changes in Web content by analyzing a crawl of 55,000 Web pages, selected to represent different user visitation patterns. Although change over long intervals has been explored on random (and potentially unvisited) samples of Web pages, little is known about the nature of finer grained changes to pages that are actively consumed by users, such as those in our sample. We describe algorithms, analyses, and models for characterizing changes in Web content, focusing on both time (by using hourly and sub-hourly crawls) and structure (by looking at page-, DOM-, and term-level changes). Change rates are higher in our behavior-based sample than found in previous work on randomly sampled pages, with a large portion of pages changing more than hourly. Detailed content and structure analyses identify stable and dynamic content within each page. The understanding of Web change we develop in this paper has implications for tools designed to help people interact with dynamic Web content, such as search engines, advertising, and Web browsers.

Categories and Subject Descriptors

H.5.4 [Information Systems]: Information Interfaces and Presentation (e.g., HCI) – Hypertext/Hypermedia: User issues.

General Terms: Human Factors, Measurement.

Keywords: Web page dynamics, change, re-finding.

1. INTRODUCTION

The content on the Web is different from most other types of content we normally interact with because it changes regularly. For example, while most documents on a person's desk remain constant, with perhaps only an annotation or two added over time, even Web pages we think of as relatively static, like the WSDM conference home page, change in subtle ways (see Figure 1).

In this paper, we characterize Web change by analyzing a multi-week Web crawl of 55,000 pages. The pages in the crawl were selected to represent a range of visitation patterns, and our dataset is unique in that it reflects how the *observed* Web is changing. Understanding this segment of the Web is important for crawlers and search engines that must keep pace with changing content, as well as applications intended to help people interact with dynamic Web content. Our research confirms a number of previously identified trends in Web evolution and highlights several important differences for pages that are visited by users compared with those selected at random. Through analysis of hourly, sub-hourly, and concurrent crawls we explore page change on a fine-grained time scale. We find that many pages change in a way that can be represented using a two-segment, piece-wise linear model that would be unobservable in a coarser crawl.

Figure 1. Web content changes regularly. Here changes to the WSDM 2009 homepage are highlighted.

We further extend previous work by characterizing the nature of the changes to Web page content and structure. The analysis of content change presented in this paper focuses on understanding which terms within a page appear consistently over time, and which come and go. We introduce the notion of the *staying power* of a term within a document over time. It appears there is a bi-modal distribution of terms, with most terms either being very stable over time or changing very rapidly. Stable terms reflect the ongoing central topic of a page as well as common function words or navigational elements. Our analyses can inform algorithms for improved search engine ranking and contextualized advertising.

We also explore the dynamics of Web page structure through analysis of DOM-level changes. In particular we concentrate on short-term survivability of various DOM elements, an important metric for Web clipping and template extraction applications which rely on the DOM structure to function [1][3][6]. The large amounts of data analyzed in our DOM analysis necessitated the creation of an efficient algorithm for tracking the motion of blocks of text, changes in the DOM structure, and identification of blocks of simultaneously changing content. The algorithm we develop can be used for both analysis and higher level tasks like predicting content flow in the page and block identification.

We begin this paper with a discussion of related work and a detailed description of our unique dataset. We then explore both content and structural change in greater detail. We conclude with a discussion of future work and potential applications.

WSDM'09, February 9-12, 2009, Barcelona, Spain.

2. RELATED WORK

Characterizing the amount of change on the Web has been of considerable interest to researchers [4], [8], [11], [12], [13], [14], [16]. For example, Cho and Garcia-Molina [4] crawled 720,000 pages once a day for a period of four months and looked at how the pages changed. Ntoulas, Cho, and Olston [14] studied page change through weekly snapshots of 150 websites collected over a year. They found that most pages did not change according to a bag-of-words measure of similarity. Even for pages that did change, the changes were minor. Frequency of change was not a great predictor of the degree of change, but the degree of change was a good predictor of the future degree of change.

Perhaps the largest scale study of Web page change was conducted by Fetterly et al. [8]. They crawled 150 million pages once a week for 11 weeks, and compared the change across pages. Like Ntoulas, Cho, and Olston [14], they found a relatively small amount of change, with 65% of all page pairs remaining exactly the same. The study additionally found that past change was a good predictor of future change, that page length was correlated with change, and that the top-level domain of a page was correlated with change (e.g., edu pages changed more slowly than com pages). The longest longitudinal study we are aware of is the 360 page sample studied between 1996 and 2001 by Koehler [12], finding that page change plateaus as a page ages.

More recently, Olston and Panday [15] crawled 10,000 random Web pages and 10,000 pages sampled from the Open Directory every two days for several months. Their analyses measured both change frequency and information longevity (the average lifetime of a shingle), and found only a moderate correlation between the two. They developed new crawl policies that are sensitive to information longevity. In a study of changes observed via a proxy, Douglis et al. [7] identified a relationship between revisitation rates and change. However, the study was limited to content visited by a restricted population (AT&T Labs), and pages were not actively crawled for changes between visits.

Researchers have also looked at how search results change over time [19], [20]. The focus in these studies was on understanding the dynamics of search engines and the consequences change has for searchers who want to return to previously viewed pages.

Our study differs from previous work in the pages we selected to crawl (sampled from pages that we know have been visited), the granularity of the crawl (hourly and sub-hourly crawls), and in the level of detail with which we analyze changes in the content and structure of Web pages. Through analysis of our unique dataset we are able to contribute to the general understanding of Web change as well as develop new algorithms for exploring and utilizing changes in a number of applications.

3. METHODOLOGY

In this paper we study how approximately 55,000 Web pages change. Here we described how the pages were selected, how they were crawled, and how change to the pages was measured.

3.1 How Pages Were Selected

Our goal in selecting Web pages for study was to sample pages with a diversity of revisitation patterns. This is a significant departure from other studies that have traced the evolution of subsets of Web pages (typically gathered by breadth-first crawls from seed pages) regardless of use. In our work, we have captured a sample representing those pages which are actually revisited and consumed by users in different ways.

To construct our sample we made use of URLs collected from the logs of opt-in users of the Live Search toolbar. The toolbar provides augmented search features and reports anonymized usage behavior to a server. Pages were pulled from a sample of those visited by 612,000 people for a five week period starting August 1, 2006. We restricted our sample to English speaking users in the United States, and attempted to remove robots and other anomalous users (a complete description of how these users were selected and filtered is available elsewhere [2]).

As we were interested in pages with different popularities and different revisitation patterns, we defined several attributes for each page which were then used to systematically sample pages for further analysis. Specifically, we considered the number of unique visitors to a page (*unique-visitors*), the average inter-arrival (i.e., revisit) times for a page (*inter-arrival time*), and the average number of revisits per user for a page (*per-user revisits*). Most pages in the raw data were visited by very few people overall, with a small number of revisits per person and at low (fast) inter-arrival times. To sample from the space without over-selecting from the tail we applied a pseudo-exponential binning to find thresholds for the unique-visitor (minimum of two) and per-user revisits. For the inter-arrival time criteria, we opted to use thresholds that correspond to well understood time periods (e.g., 10 seconds, minute, hour, day, week).

In total, there were four bins for the unique-visitor criteria, five for the per-user revisit criteria, and six for the inter-arrival time criteria (for a combination of 4 x 5 x 6 = 120 possible bins). Each URL was annotated with three bin numbers, and we sampled from this space. Some oversampling of popular pages was added by explicitly including the top 5000 most visited pages. The Web pages were crawled to ensure that they were still publicly available (in conformance with the robots.txt), and those that were not were removed. The final sample included 54,788 URLs with an average of 468.3 (median 650) URLs in each of the 120 bins.

Each page was categorized into topical categories using standard text classification techniques [18]. The topical categories used are similar to those in the first two levels of the Open Directory (www.dmoz.org). A second classifier, used by Live Search, identified various genres (e.g., children, mail, pornography, etc.). Additionally, pages were grouped by the number of unique visitors (as described above), the URL depth (how deep the page is within the site), and top level domain.

3.2 How Pages Were Crawled

To understand how the content of these Web pages change over time, we crawled each URL hourly for 5 weeks, starting May 24, 2007. (Because of some randomization in the crawler, the time between subsequent re-crawls may be slightly more or less than an hour). For those pages that displayed a substantial number of hourly changes a secondary crawl was initiated over the course of four days to capture sub-hour changes, as described in Section 4.1.2. Finally, for pages that displayed many changes at the two-minute interval, a third crawl was constructed to collect the page twice at the same time from two synchronized machines. All three crawls capture changes at a much finer grain than previous work. The full HTML text of the page was collected and stored for each retrieved version.

3.3 How Change Was Measured

We explore how the Web changes by examining the crawled data in several ways. An extremely coarse measurement of change can make use of document checksums that detect any difference. In a

way this is unsatisfying as both large and small changes, as well as content and structural changes, are treated the same.

Previous work has focused on measuring textual differences by, for example, calculating the differences between blocks of text [8] or word frequencies [14]. We begin our study by utilizing a similar approach, based on the Dice coefficient, to represent the amount of textual change. The Dice coefficient, which measures the overlap in text between various document versions, allows us to develop a high level model of page change over time (i.e., $Dice(W_i, W_k)=2*|W_i \cap W_k|/(|W_i|+|W_k|)$, where W_i and W_k are sets of words for the document at time *i* and *k* respectively). A high Dice coefficient (i.e., 1) reflects high similarity, whereas a low Dice coefficient (i.e., 0) indicates no similarity. Having identified this model we further refine our metrics to explore term-level document change. By comparing term occurrence within a page with a background language model, we identify the specific terms that change or persist in a document.

We augment this analysis of content change with a study of structural change. We develop several algorithms to identify and track the movement of DOM elements within the documents and describe the persistence of structural blocks over time.

4. CONTENT CHANGE

4.1 Overview

We begin our analysis of content change by looking at the frequency and amount of change that occurred for each page individually in our sample. As an example, a page may not change at all for several days after it is first crawled, but change a lot when it finally does. The same page may then begin to experience small changes at frequent intervals. All of the changes that are made to a page over the period of our crawl can be represented as a scatter plot like the ones shown in Figure 2. Each point represents an instance where the page changes. The amount of time elapsed since the last change is represented along the *x*-axis, and the amount of change that occurs, measured using the Dice coefficient, is represented along the *y*-axis. For example, the plot for the New York Time's homepage in Figure 2a shows most changes occur with every crawl (i.e., every 60 minutes), although some happen after two. In contrast, Figure 2b represents a page that provides horoscopes and shows larger changes occurring on a daily basis.

In our collection, 18971 pages (34%), displayed no change during the studied interval. On average, documents that displayed some change (35997 pages) did so every 123 hours and the average Dice coefficient for the change was 0.794. Figure 3a shows the distribution of average inter-change times, and Figure 3b shows the average inter-version Dice coefficients.

4.1.1 Change by Page Type

The mean time between changes, and the mean Dice coefficient for the changes that did occur, are shown in Table 1, broken down along several different lines including: the number of visitors, top-level domain, URL depth, and 6 most prevalent categories. For each page, we calculate the average time between changes (i.e., the inter-version time) as well as the amount of change (as reflected by the Dice similarity). This roughly corresponds to the centroid of the points for each page in Figure 2. The mean inter-version time (in hours) and Dice coefficient are then calculated for pages of the same class (e.g., News pages or pages in the .edu domain). In this section we concentrate on the mean inter-arrival times and similarities, returning to the knot data in subsequent sections. All differences in the table are significant ($p<0.0001$) with the exception of differences in mean change interval that are less than 10 hours, and differences in the mean Dice coefficient that are less than 0.02. Additionally, the .org top-level domain Dice coefficient is only significantly different from .net.

Table 1. Pages change at different rates and change different amounts. This table shows the mean interval between changes, the mean amount of change to a page, and the mean knot point, broken down by page type.

		Inter-version means		Knot Point Location	
		Hours	*Dice*	*Hours*	*Dice*
Total		123	.7940	145	.7372
Visitors	2	138	.8022	.146*	.7594*
	3 - 6	125	.8268	143*	.7692*
	7 - 38	106	.8252	145*	.7458*
	39+	102	.8123	139*	.7621*
Domain	.gov	169	.8358	153	.8177
	.edu	161	.8753	200	.8109
	.com	126	.7882	145	.7408
	.net	125	.7642	132	.7195
	.org	95	.8518	129	.6743
URL depth	5+	199	.6782	173	.7150
	4	176	.7401	159	.7413
	3	167	.7363	160	.7378
	2	127	.7804	144	.7340
	1	104	.8200	137	.7432
	0	80	.8584	141	.7334
Category	Industry/trade	218	.6649	156	.6680
	Music	147	.8013	166	.7693
	Porn	137	.7649	140	.7365
	Personal pages	88	.8288	124	.7347
	Sports/recreation	66	.8975	137	.7138
	News/magazines	33	.8700	104	.6415

*Not significant

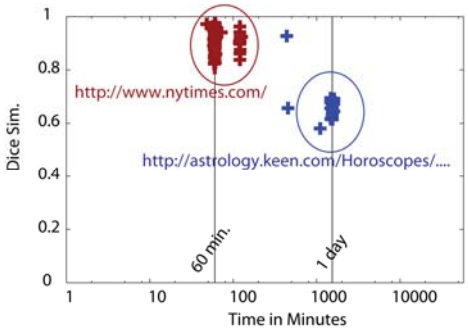

Figure 2. The interval between successive changes, plotted against the amount of change for two pages. Note that the NY Times page changes on every crawl (i.e., every 60 min.).

When broken down by unique visitors, we observe that pages that are more popular tend to change more frequently than less popular pages. This most likely explains why the percentage of pages in our sample that displayed no change was nearly half of what has been observed in prior studies ([4], [8]). Pages that are revisited (as all pages in our sample are) change more often than pages collected using other sampling techniques. However, while the

Table 2. Change properties for sub-hour crawls.

Interval	Docs. with change (% of 54816)		Mean / Median Dice sim. (given change)
	Change in any sample (upper bound)	*Change in all samples (lower bound)*	
Instant (0 min.)	6303 (11.5%)	3549 (6.5%)	.937 / .985
2 minutes	10616 (19.3%)	4952 (9%)	.937 / .982
16 minutes	12503 (22.8%)	6206 (11.3%)	.927 / .975
32 minutes	13126 (23.9%)	6445 (11.8%)	.920 / .969
60 minutes	35997 (65.7%)	22818 (41.6%)	.867 / .950

number of unique visitors is related to the rate of change, it does not appear to correspond to the amount of change.

Looking at change by top-level domain, it appears that educational and government domain addresses do not change as frequently or as much as pages in other domains do. This is consistent with prior research ([8], [11]), and may reflect the fact that sites in these domains typically provide richer and less transient content that only requires small, infrequent updates.

Pages with deep URL structure change less frequently than pages at the root, but change more when they do. This may be because top level pages are used as jumping points into the site, with a high navigational to content ratio. Thus while new ads or short teaser links to new deep content are often added, these are fairly minor changes. Internal pages, which may have a low navigation to content ratio, will see large changes when they do change.

The final breakdown of change shows that pages from different topical categories change at different rates and by different amounts. Pages from categories like News, Sports and Personal pages change the most frequently as would be expected. The amount of change, however, is not generally as large as that seen for categories that change less frequently.

4.1.2 Change over Short Intervals

Because 22818 of the pages in our sample changed nearly every hour, we initiated two sub-hour crawls (*fast-change* and *instant-change*) to gain a better understanding of how these pages changes at intervals of less than an hour. Fast-change pages were collected by taking a single snapshot of the page, a second one at 2 minutes, a third at 16, and a fourth at 32. This was done over a period of three days, with each sample shifted by 4 hours in order to capture different conditions (e.g., during the day versus at night), for a total of 8 four-sample rounds. For those pages that displayed changes during the initial 2 minute delay (10616 pages), instant-change data was collected by simultaneously requesting the page on two synchronized machines. Although this data does not necessarily inform us about rates of change, it does tell us if pages change through some automated process on every request rather than changing as a result of some human-driven process.

Table 2 presents high level statistics for this analysis. We calculate the number of documents that changed *at least once* during sampling as well as the number of pages that change in *all* samples. The range between the two approximates the lower and upper bound on the true number of pages that change at a given rate. In all cases, the Dice similarity is high, reflecting small amounts of change in the short time-scales. As the change interval gets longer, the amount of change a page undergoes increases. Manual analysis of documents displaying instant change reveals many instances of randomly changing advertisements. Additionally, details about how fast a page was served (e.g., "Page served in 0.00252 secs,") to whom (e.g., "IP address is.."), and to how many (e.g., "There are 665 users online," or, "Visitor number 81434,") resulted in changes in the concurrent crawl. Because the content of these pages is driven by the visiting client, it is important to recognize that many of the changes detected by a crawler may be insubstantial as they are driven by the crawler itself. We return later to a discussion of how we might detect interesting versus non-interesting changes, though we note that a simple pattern for these automated changes could be easily learned by automated analysis of the textual differences between two instantly-changing documents.

4.2 Summarizing a Page's Change over Time

Thus far we have only discussed changes between two successive versions of a page. Such analysis does not capture a page's change trajectory over a period a time or the more detailed content of that change. A page that has one restricted area of change that updates every time it is crawled (e.g., an advertising block) will show very similar amounts of change between successive versions as a blog that adds a new entry every crawl. But, over time, the content of the page with the advertising block changes very little, while the content of the blog changes a lot.

4.2.1 Change Curve

To quantify the change of each Web page over time we define the notion of a *change curve*. A change curve represents the amount of textual change (as measured by the Dice coefficient) from a

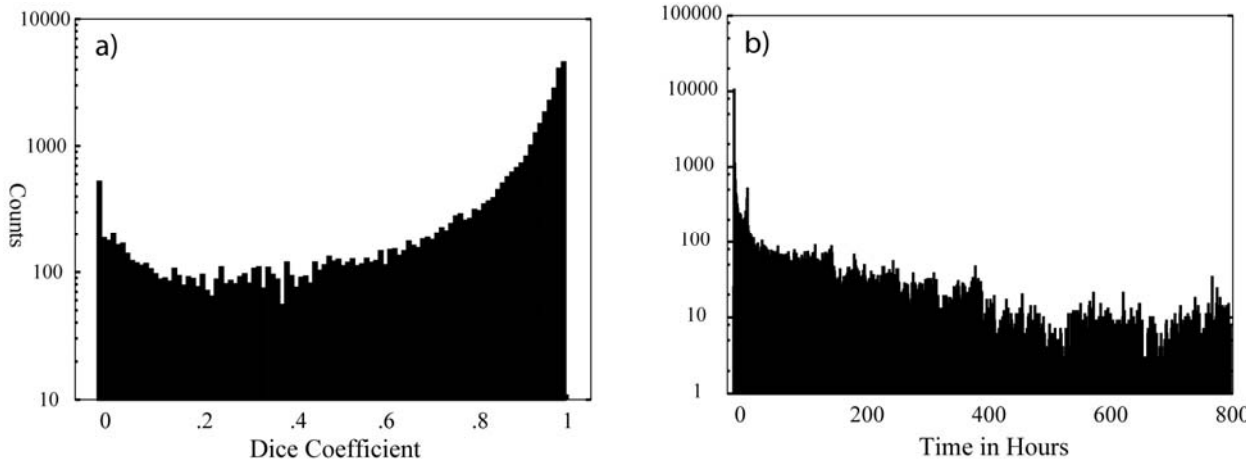

Figure 3. Histograms of a) the average time between changes in our dataset for all sequential pairs of Web pages downloaded from the same URL, and b) the average amount the content changed, as measured by the Dice Coefficient.

fixed point in the document's history. For each page we select, at random (biased to the first week of samples), up to *n* starting points (*n*=5 in the analyses reported below). We define D_t to represent the Web page content at time *t*, and D_{r1} to be the content at the first randomly selected time. Content, for this analysis, is defined to be the page stripped of markup. The value of the change curve at each time point, *t*, is calculated as the average Dice coefficient from each of the randomly selected starting points to the Web page content *t* time steps in future.

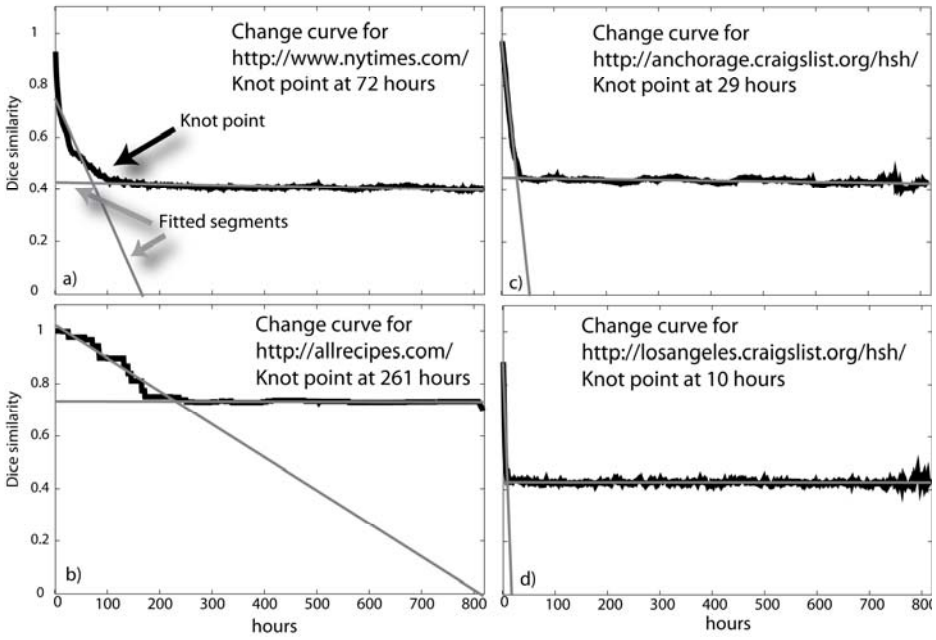

Figure 4. Several example change curves depicting Dice coefficients over time.

Change curves allow us to summarize a Web page's evolution over time. Several example curves can be seen in Figure 4. We observe that most documents change rapidly from the initial starting point as content shifts off the page or is changed during the initial hours. For example, in a news homepage, specific news stories may move off the page at a certain rate as new articles are added. This causes a rapid falloff in the Dice coefficient. At some point, the curve levels off, as the similarity of all subsequent versions to the initial version is approximately equal. This creates a change curve with a "hockey stick" form (see Figure 4).

Note that not all versions of the page after the curve levels off are the same. It is rather their similarity to the original starting point that is the same. In the news example, at the inflection point or *knot*, where the curve levels off, all stories that were present in the initial sample have moved off the page. Every page following the inflection point has different stories from the starting point, and this represents the constant Dice coefficient in the level portion. The part of the site that remains similar to the initial sample is the text that always resides on the page, including navigational content and terms that appear regularly in all stories. In general, the textual content past the inflection point that is similar to the starting point is template information and a rough representation of the underlying language model of the page.

Manual analysis of the change curves identified three main types: *knotted* (made up of two piecewise linear segments, the "hockey stick" shape), *flat* (one unchanging, zero slope line), and *sloped* (one sloped line with no obvious knot).

4.2.2 Identifying the Knot

To compare the change curves of different Web pages to each other, we model the curves using the characteristic hockey stick shape. We do this by identifying the curve's knot and fitting two linear regressions to the curve, one up to the knot, and the other following it. Piecewise linear regression is a well known problem which requires either knowing the knot location a priori or determining it through a grid search with a well defined objective function. To find the knot point efficiently, we have defined a heuristic scheme that works well in practice.

The algorithm works by first fitting one linear segment to the full change curve. The leftmost intercept of the fit curve to the change curve becomes our initial guess for the knot location. A segment is then fit to the portion of the change curve occurring before the knot. The knot point is incremented as long as the mean-squared error of the first segment does not increase. Figure 4 shows the fitted segments for different pages. In situations when no knot point appears within the sampled time period (i.e., a knot point at 0 hours or the two segments have nearly identical slopes), we fit one segment which is either continuously sloped or flat (an unchanging page). Of course, sloped pages may eventually level off in a hockey stick pattern, but we do not have sufficient evidence to conclude when, or even if, this might happen.

The combination of knot point and characteristic regression information (slopes and intercepts) provides a representation of the data which can be used to automatically classify the curves as knotted, flat, or sloped. Using 200 random, manually labeled curves as a training/test example, we are able to achieve over 93% accuracy with a classifier based on Additive Logistic Regression [9]. We find that the classifier is somewhat conservative in deciding on the knotted category, resulting in very few false positives but a number of false negatives which are categorized as sloped. In the end, our algorithm classified 819 pages (of 36767) as flat (2%), 10462 as sloped (28%), and 25486 as knotted (70%). For all pages with knot points, the mean time is 145 hours (median time 92 hours), and the mean Dice coefficient is 0.74 (median Dice 0.80).

Table 1 shows the average knot point for pages broken down in a number of different ways, including by top-level domain, URL depth, and category. The change time (in hours) reflects how long it takes to reach steady state content, and the Dice coefficient represents the amount of overlap at that time (the x/y coordinates of the knot point). All of the differences in change intervals that are greater than four hours are significant, except that .gov domain is only significantly different from .edu. All differences in Dice coefficient greater than 0.01 are significant.

We note a few interesting points about these change curves.

Knot location: Different pages (e.g., Figure 4b and 4d) level off at different Dice values. This level represents the eventual

(dis)similarity of the page once content has "decayed" off. Allrecipes, with much more navigation and structure retains more content over time than the bare Craigslist page that primarily consists of new ads. The time of the knot point (i.e., its *x*-coordinate) reflects the rate at which the most quickly changing data, does change. Note the difference between Figures 4c and 4d that have the same eventual Dice value, but different knot locations in time. The Los Angeles Craigslist page (Figure 4d), with many more new home products being sold, has a knot point at 10 hours versus Anchorage's 29 (4c).

Unique visitors: Unlike inter-version means, there is no statistical difference in where the knot point falls as a function of unique visitors. This is consistent with the fact that while popular pages change more often, they change less when they do, and thus require the same amount of time to "stabilize" as less popular pages.

URL Depth: The deeper the page is in the page hierarchy the further the knot point, potentially indicating that content on pages deep within a site "decay" at a slower rate.

Category: Perhaps unsurprisingly, News and Sports pages have an earlier knot point as content in these pages is likely to be replaced quickly. Industry/trade pages, including corporate home pages, display a much more gradual rate of content decay before reaching the knot point.

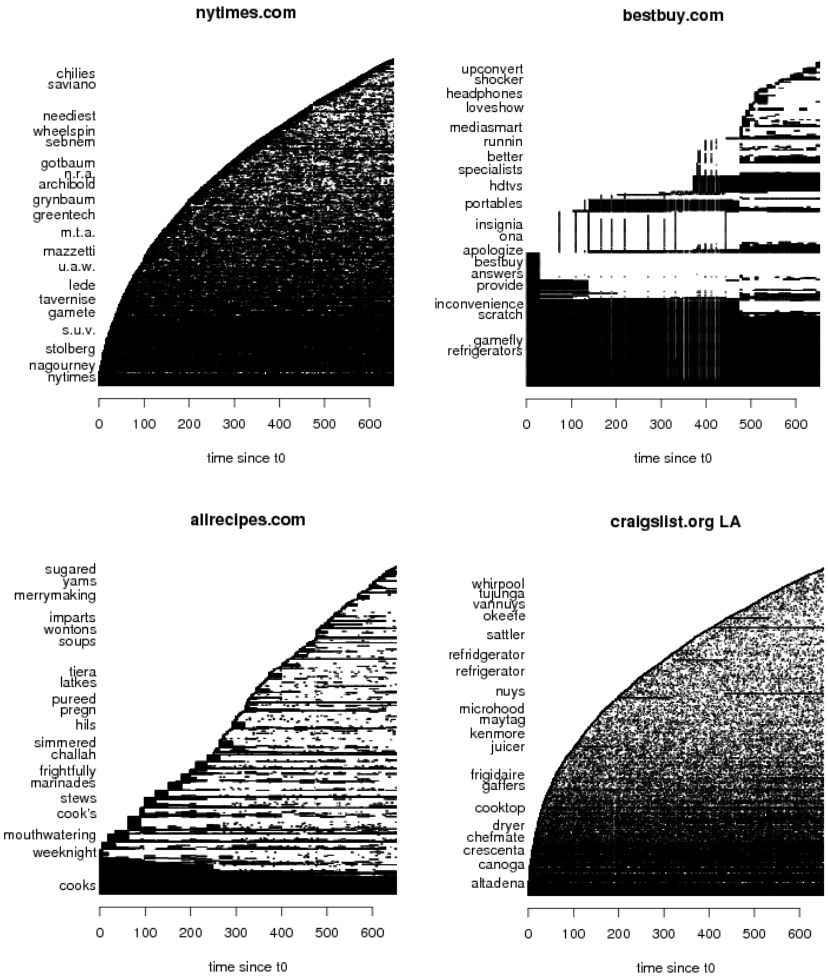

Figure 5. Term lifespan plots for several pages. The lower right hand plot has been replaced with the BestBuy homepage. Time (in days) is shown along the *x*-axis, and terms are along the y-axis, ordered by their first occurrence in the documents. Representative terms at various levels are shown along both sides.

4.3 Term-Level Change

The above analysis explores how page content changes across an entire Web page. This type of page evolution analysis tells us about the rate at which the page changes as a whole, but not specifically how the textual content is evolving. In this section we present several ways to characterize the content change at the term level. Doing this allows us to explore the types of changes that occur in different parts of the change curve. After identifying the common hockey-stick change pattern, we continued crawling (over a six month period from June to December 2007) to study text content changes in greater detail. Previous attempts to study page content change at the sub-document level have focused on chunk-level measurements, such as page tiling and hashing techniques [8][15]. While these techniques give some sense of the volume and frequency of page content that changes over time, they do not shed any insight into the nature of those changes.

Most basically, we are interested in how page vocabulary changes as a page updates. Figure 5 shows several *term lifespan plots* which visualize the dynamics of vocabulary change over time. Time is shown along the *x*-axis, and terms are shown as horizontal lines on the *y*-axis. When a term occurs in the document at a given time, a point is drawn, and when the term is absent no point is drawn. Terms are ordered by their first occurrence in the page, and then by their longevity. The change curves in Figure 4 are essentially a summary of the term lifespan plots in Figure 5. Change curves can be generated by selecting a reference vertical slice in a lifespan plot and measuring the overlap of terms in that slice with those in subsequent slices.

The longevity plots are denser at the bottom indicating that the most constant terms tend to appear early in the observation period, and those that are more ephemeral tend to appear later. We observed there is a qualitative difference between the bottom and top terms; intuitively, the terms near the bottom characterize aspects such as the central topic of the page, as well as navigational elements, while those towards the top may be seasonal or related to a small niche of the pages central topic.

To characterize how likely a term is to appear over time, we defined a measure of the *staying power* of a term in a Web page:

$$\sigma(w, D) = \sum_{t=0}^{T-1} \sum_{\alpha=1}^{T-t} \frac{1}{T(T-\alpha)} I(w \in D_t \ \wedge \in D_{t+\alpha})$$

where w is a word; $t \in [0 \ldots T]$ indexes timestamps; D_t is a document at time t; α is the interval between time stamps and $I(.)$ is an indicator function. Intuitively, the measure is roughly equivalent to the following probabilistic interpretation:

$$\sigma(w, D) \approx P(t)P(\alpha)P(w \mid D_t, D_{t+\alpha})$$

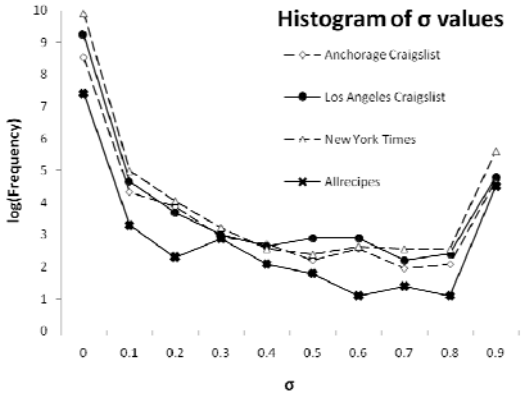

Figure 6. Histogram of staying power (σ) values for four Web pages.

Or the likelihood of observing a word in document *D* at two different timestamps, t and $t + \alpha$, where $P(t)$ and $P(\alpha)$ are sampled uniformly.

In Figure 6, we show how the staying power is distributed across terms for several sample pages. There is a clear bi-modal distribution, with most of the vocabulary clustered at the ends. Terms with high staying power are likely to be in multiple crawls of the page, and those with low staying power are unlikely to occur in multiple crawls of the page.

As we saw with the longevity plots, terms with high staying power are either descriptive of the document's ongoing central topic, represent common words, or are navigation elements. In an attempt to distinguish the set of vocabulary that is potentially more informative of the document's central topic as well as having a strong staying power, we looked at the *divergence* or *clarity* of these terms with regard to the collection as a whole.

Previously, divergence has been used to measure how strongly a subset of documents is distinguished from a collection, and has been applied to the task of query difficulty prediction [5]. In that setting, the K-L divergence between a retrieved set of documents and the collection as a whole is used as a means to predict query difficulty—the more divergent the two sets are, the more likely the retrieved documents are to reflect a coherent (and relevant) subset of Web content. Here, we're applying a similar measure to identify which terms distinguish the language of a single document from that of the collection.

Our term divergence measure is a term's contribution to the K-L divergence of the document from the collection language model.

$$Div(w, D) = P(w|D) log \frac{P(w|D)}{P(w|C)}$$

The probabilities are computed as un-smoothed maximum likelihood estimates:

$$P(w|D) = \frac{1}{T}\sum_{t=0}^{T} \frac{tf_{w;Dt}}{|Dt|} \qquad P(w|C) = \frac{1}{T}\sum_{t=0}^{T} \frac{ctf_{w;Ct}}{|Ct|}$$

Where *tf* and *ctf* refer to document and collection term frequency respectively, |D| and |C| refer to the length of the document and collection respectively. This divergence measure can be thought of as a probabilistic form of TF-IDF weighting, favoring terms with a high likelihood of occurrence in the document and a low overall likelihood of occurrence in the collection. The divergence can be measured for the document over all crawled instances as shown in the equation above (*divergence*), or for a single crawled instance (*divergence at t*) by just taking a single term of the summations above.

Table 3. Terms from two websites. (left columns) Terms with a high *divergence at t* value, with t=0, the first day of our crawl. Terms with low staying power($\sigma < 0.2$) in bold. (right columns) Terms with a high staying power and high/low *divergence* values.

Allrecipes.com		**Craigslist.org (Los Angeles)**	
Div. @ *t=0*	$\sigma > 0.8$	**Div. @ *t=0***	$\sigma > 0.8$
recipes	High divergence:	pic	High divergence:
cooking	cooks	**porter**	refrigerator
recipe	cookbooks	**ranch**	dryer
advice	ingredient	nuys	kenmore
more	desserts	kenmore	southbay
salads	rachael	refrigerator	washer
cookbook	digest	angeles	whirlpool
tips	newsroom	dryer	antelope
sandwiches	trusted	west	microwave
cooks		hollywood	
easy	Low divergence:	washer	Low divergence:
pork	home	los	pic
survey	you	stove	new
bbq	search	santa	all
menus	free	vacuum	los
salad	your	sfv	beach
widgets	com	**koreatown**	sale
cheese	with	conditioner	with
cool	this	refrigerator	west
delivered	all	microwave	items

Table 3 shows several sets of terms for two sample pages. In the left columns, terms with a high *divergence at t* value are shown, for *t*=0, the first crawl in our collection. In bold are terms with a low staying power ($\sigma < 0.2$). These terms represent ephemeral vocabulary which are descriptive of the page at the time (here *t*=0) but are not descriptive on an ongoing basis. In the right columns, we show terms with high staying power in the documents ($\sigma > 0.8$) broken down by whether they have high or low divergence from the collection model. In this example, we see how the divergence measure can separate terms such as function words which are always present in the documents (low divergence) from those that are more descriptive of the document's content (high divergence). The analyses shown in Table 3 suggest that measures such as *staying power* and *divergence* are useful in identifying terms that are both characteristic and discriminating for the document as it evolves over time. In Section 6 we describe how such differences might be used to improve summarization and ranking.

In this section, we have explored how Web page content changes over time. We confirmed previous findings, and built on those findings to better understand how different types of pages change and what the nature of those changes is. We next describe how page structure changes over time and develop methods to efficiently analyze such changes.

5. STRUCTURAL CHANGE

In previous sections we have ignored the structure of a Web page. Nonetheless, structural changes represent an important way a page can change. A page with exactly the same textual content as a previous version can still be very different because that content is organized and presented in a different way.

In considering structural change we would like to compare the tree structure of each version of the Web page to one or many

Table 4: Percent of persistent DOM elements

	2 hour	1 day	1 week	2 weeks	4 weeks	5 weeks
Mean	99.3%	97.4%	91.7%	87.9%	86.4%	84.3%

other instances. Various efforts have attempted to address this difficult problem (extensively surveyed by Grandi [10]). The tree alignment problem is complex to implement, expensive to calculate, and usually only treats pairs of trees. Ideally, we would like to analyze a thousand or more instances of the same page tree rapidly and use the same algorithm to solve multiple problems. In particular we would like to test for the presence or absence of various DOM elements (described by XPath) and potential changes within them. Additionally, we would like to identify blocks of content that are changing synchronously.

To do so we develop a simple algorithm that serializes an HTML document in such a way that different forms of lexicographic sorting and aggregation can answer these questions. A limitation of this approach is that it is unable to detect the amount of change in content directly. We return to a possible solution below. But despite its limitations, it has the significant advantage of being implementable as a basic map-reduce style problem and scaled.

5.1 Serializing DOM Structures

In order to analyze many versions of a page simultaneously we convert the HTML pages into a form suitable for processing. We construct a serialized form that minimally consists of a version ID, the XPath, and a hash of the textual content of each DOM node. By sorting on different fields, we group elements that we would like to track. For example, by sorting on the XPath and version, we can identify when the content within the path changes or if the node vanishes at a particular point.

More concretely, to generate the serialized version, the algorithm proceeds by progressing in a depth first parse of the HTML tree of the document through a standard SAX parser (HTML documents are "cleaned" with the *tidy* tool for XML validity). When the system reaches the start of a new HTML element (e.g., " " or "<h1>") the algorithm notes this in a stack. Textual content that is encountered is also pushed onto the stack. When an ending element tag is reached (e.g., "</h1>" or " ") the system emits a hash value of the text within the tags (as well as a hash of the children of that node). The system will also output the XPath used to arrive at the nodes in both set and direct access notation. For example, given "<a>foo bar" at time *k*, the system will output a *stream* of two lines:

```
/a[0]/b[0]   (/a/b)[0]   hash("bar")   hash("bar")       k
/a[0]        (/a)[0]     hash("foo")   hash("foo bar")   k
```

Each line corresponds to a temporally-annotated tuple: {*full_path*, *type_path*, *node_hash*, *subtree_hash*, *version*}. The type path reflects the full typing of the node based on its position in the hierarchy. Although the two are equivalent, we find that having two versions of the path, and in particular the type path, useful for detecting motion of elements between tree siblings. Similarly, having both a hash of the content of the node as well as the subtree allows us to isolate the cause of a change. We create this output for every version of the document. For example, if the *k*+1 version of the document contained the text "<a>foo baz" our stream would be:

```
/a[0]/b[0]   (/a/b)[0]   hash("baz")   hash("baz")       k+1
/a[0]        (/a)[0]     hash("foo")   hash("foo baz")   k+1
/a[0]/b[0]   (/a/b)[0]   hash("bar")   hash("bar")       K
/a[0]        (/a)[0]     hash("foo")   hash("foo bar")   K
```

We may additionally keep track of other features for later filtering (e.g., the length of the text, hash of tag attributes, etc.).

We define two operations on this dataset: *sort(sorting variables)*, which outputs a sorted stream, and *reduce(reduction variables)*, which outputs a set of sets. The sort function corresponds to "map" but for our purposes simply lexicographically sorts the serialized stream in the order of the sorting variables (e.g., *sort(node_hash,version)* would first sort by the *node_hash* and then by *version*). The reduce operation groups sorted data based on some matching parameters (e.g., *reduce(version)* would generate a set of sets where each subset was one version: $\{\{k\},\{k+1\},\ldots\{k+n\}\}$).

By sorting on different columns in different orders and linearly processing the files, we are able to track the movement and lifespan of a specific piece of text in the hierarchy, track the number and lifespan of pieces of text at a specific location, and combine these operations in various ways. The use of shingles as a hash value [17] is worth exploring in the future as we can take advantage of our algorithm to rapidly eliminate exact matches, and testing approximate ones when necessary. The benefit of our approach is that it is a) linear in the number of tree nodes and versions, and b) can be parallelized by mapping different sub-trees to different processors.

5.2 Tracking DOM Element Lifespan

Using the mechanism described above we are able to calculate the lifespan of various DOM elements within our crawled pages over time. We achieve this by applying:

> *sort(full_path,version)*
>
> *S = reduce(full_path)*
>
> *foreach s in S: calculate the difference between the minimum version id and last reported id*

Letting *S* be the set of sets, *s* a subset within *S*. This algorithm is somewhat of a simplification as, in reality, we only test for the vanishing point for those paths present in the original crawl. Additionally, we ignore nodes that have not survived the first hour since these tend to be unrelated to document structure and frequently contain simple formatting.

Table 4 presents the survival rates for DOM elements in pages that were successfully parsed (37,160 pages). The survivability of elements is quite high, even after 5 weeks (median survival of 99.8%). The large amount of survivable structure suggests that XPath-based extractions (e.g., [1]) over short periods would continue to function. The algorithm may also identify unstable content and indicate where XPath extractions are inappropriate.

5.3 Analyzing Blocks

Extending the ideas above, we can further calculate the persistence of any content anywhere within the document. This is particularly interesting for situations where content may move (as in the case of blogs or news sites) but also informs us about how long we might expect certain content to last on the page before vanishing. To calculate this we can execute the following:

> *sort(subtree_hash, full_path, version)*
>
> *S = reduce(subtree_hash)*
>
> *for s in S: find the minimum and maximum version for each tuple, t, in s at the given location(as specified by the path). Calculate the difference between the minimum and maximum (the lifespan), and recalculate the average lifespan for that path.*

Again the algorithm is a slight simplification for readability as we only consider the initial time that the content entered the node. The output of this analysis is a measure of how long we can expect a piece of text that starts at a given location to survive.

Figure 7 shows a simple visual rendering of give sample pages from our crawl processed using this approach. The opaque (red) regions are those that change the most rapidly, and the white regions are the most stable. Navigation and other persistent elements (e.g., search bars) tend to have higher survival rates, whereas text ads change rapidly. Notice, for example, the navigation bar on the left side of the amazon.com bestsellers page in Figure 7c contains highly survivable content as well as the top item which persisted much longer than other elements in the list.

This algorithm can be adapted to group different nodes with similar average life spans (e.g., when a new article is posted on the top of a blog, generally all posts below move, thus the average change rate of all posts is matched). Considering the rendered location of DOM elements, may allow us to identify *blocks* of changing content.

Metrics for structural changes are frequently task specific. By creating a flexible serialized processing scheme we are able to rapidly test and measure structural change in different ways. We are presently continuing to expand this technique, tying it more closely in with notions of content change.

Having explored Web change on both content and structure, we turn to a discussion of ways our models and algorithms can be used to help people better interact with dynamic Web content.

6. USING CHANGE ANALYSES

Previous studies of Web change have typically been used to inform search engine crawler policies, the idea being that pages

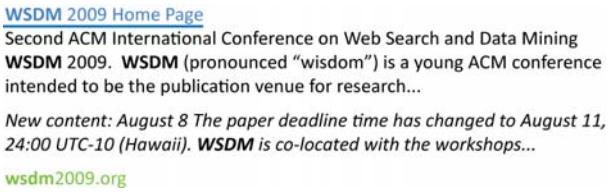

Figure 8. Summarizing change in the result snippet.

that change a lot are more likely to get stale and should be crawled more often. In this section we explore some of the ways our findings may be used. Like earlier work, our results can be used to refine crawler policies. In addition, because the sample of Web pages we studied was selected based on how people use the Web pages, we believe our results also have implications for tools that support human interaction with dynamic Web content, including Web services like search engines and client-side applications like Web browsers or toolbars.

Crawling. One important observation for Web crawlers in our analysis is that some types of change are more meaningful. By looking at *what* type of content is changing, crawlers can focus on meaningful change. For example, the fast crawl highlights the importance of ignoring change to advertising content or change that occurs as a result of touching a page (e.g., changes to visit counters or information about page load time).

Our analysis of the terms that change in a page suggest some of the static content may be more valuable for characterizing a page, and thus more valuable to the search engine index. Crawlers could be updated to take into account several archived versions of a page to identify the characteristic static component, and then not worry about crawling the page to capture every new, relatively unimportant change. Further identifying those pages for which the

Figure 7. Renderings of the lifespan of elements on a number of pages (darker red blocks are shorter life spans) including a) boston.com, b) televisionwithoutpity.com (note the groups of similarly colored content), c) the DVD bestseller list on Amazon, d) gas prices in various cities on GasBuddy.com, and e) a list of earthquakes at the USGS. Not all blocks marked.

dynamic component is important (perhaps because the queries that lead to a page do so via matches with the dynamic component, or because the queries are similarly dynamic) would allow crawlers to focus on those pages in particular.

Ranking. A similar approach could be applied to improve Web search engines. Characteristics of Web pages, such as the knot point, that summarize page-level changes, over time could be used as features for improved ranking. In addition terms may warrant different weight for retrieval based on whether they appear in the static or dynamic portion of a Web page. The characterization of the page in the index could mirror the page's change patterns, and queries could be mapped to the appropriate aspect, either in general (e.g., static content is always weighted higher) or on a per-query basis (e.g., some queries bias towards dynamic content while others bias towards static content). Similar ideas could be used to improve selection of keywords for contextual ads.

User Interaction. A search engine with a rich understanding of Web dynamics could also benefit its end users in more direct, obvious ways by exposing page change in its user interface. Searchers who are revisiting previous information sources may be interested in re-finding [20] previously viewed content or in viewing newly available content. Both behaviors can better supported. For searchers interested in new content (identified either by the query alone or by the user's query history), the search result summaries could highlight the new content, such as seen in Figure 8. Additionally, many search engines have a "cached page" functionality. This is useful in instances when the page content has changed but the user is interested in the previous version (detectable by analysis of revisitation patterns [2]). This functionality could be further enhanced by comparing the cached version of the page with the live version, and exposing the differences, such as is shown in Figure 1.

The ability to expose and interact with change would be particularly useful within a client-side, Web browser context, where a user's history of interaction is known. The pages in a browser cache could be better exposed in ways similar to the above suggestions as to how a search engine might expose them, allowing the user a personal view into what's changed since their last visit. A browser could also act as a personal Web crawler for its user, and pre-fetch pages that are particularly likely to experience meaningful change. This would allow for a faster Web experience and give users the ability to access new content in offline environments. Combining pre-fetching with predictive revisitation patterns seems particularly valuable, and is an area of future work we're actively pursuing.

7. CONCLUSION AND FUTURE WORK

In this paper, we have described algorithms, analyses, and models for characterizing the evolution of Web content. Our analysis gives insight into how Web content changes on a finer grain than previous research, both in terms of the time intervals studied (we consider hourly and sub-hourly crawls instead of weekly or daily crawls) and the detail of change analyzed (in addition to looking at page-level changes, we explore DOM- and term-level changes). Change rates are higher in our behavior-based sample than has been found in previous work on randomly sampled pages, with a large portion of pages changing more than hourly. Detailed content and structure analyses identify stable and dynamic content within each page.

Looking forward, we hope to refine and test the designs described in Section 6, and to better characterize the impact of document change on revisitation. As observed in Table 1, the amount of change to a Web page's content appears to vary greatly as a function its visitor count (i.e., popularity). We believe that understanding revisitation patterns and intent can help us identify meaningful change and support interaction with cached content when appropriate.

ACKNOWLEDGEMENTS

We would like to thank Dan Liebling, Ronnie Chaiken, Bill Ramsey, and Dennis Fetterly for their help in obtaining and analyzing the data. We also appreciate helpful discussions with Sara Adar and Dan Weld.

REFERENCES

[1] Adar E., M. Dontcheva, J. Fogarty, D. S. Weld. Zoetrope: Interacting with the Ephemeral Web. UIST '08, 239-248, 2008.

[2] Adar, E., J. Teevan, and S. T. Dumais. Large scale analysis of Web revisitation Patterns. *CHI '08*, 1197-1206, 2008.

[3] Bolin, M., M. Webber, P. Rha, T. Wilson, and R. C. Miller, Automation and Customization of Rendered Web Pages, UIST '05, 163-172, 2005.

[4] Cho, J. and H. Garcia-Molina. The evolution of the Web and implications for an incremental crawler. *VLDB '00*, 200-209, 2000.

[5] Cronin-Townsend, S., Y. Zhou, W. B. Croft. Predicting query performance. *SIGIR'02*, 299-306, 2002.

[6] Dontcheva, M., S. Drucker, D. Salesin, and M. F. Cohen, Changes in Webpage Structure over Time, TR2007-04-02, UW, CSE, 2007.

[7] Douglis, F., A. Feldmann, B. Krishnamurthy, and J. Mogul. Rate of change and other metrics: A live study of the World Wide Web. *USENIX Symposium on Internet Technologies and Systems*, 1997.

[8] Fetterly, D., M. Manasse, M. Najork, and J. Wiener. A large-scale study of the evolution of Web pages. *WWW '03*, 669-678, 2003.

[9] Friedman, J., T. Hastie, and R. Tibshirani, Additive logistic regression: A statistical view of boosting. *Annals of Statistics, 28(20)*, 337-407, 2000.

[10] Grandi, F., Introducing an annotated bibliography on temporal and evolution aspects in the World Wide Web. *SIGMOD Records, 33(2), 84-86*, 2004.

[11] Kim, J. K., and S. H. Lee. An empirical study of the change of Web pages. *APWeb '05*, 632-642, 2005.

[12] Koehler, W. Web page change and persistence: A four-year longitudinal study. JASIST, 53(2), 162-171, 2002.

[13] Kwon, S. H., S. H. Lee, and S. J. Kim. Effective criteria for Web page changes. In Proceedings of APWeb '06, 837-842, 2006.

[14] Ntoulas, A., Cho, J., and Olston, C. What's new on the Web? The evolution of the Web from a search engine perspective. *WWW '04* , 1-12, 2004.

[15] Olston, C. and Pandey, S. Recrawl scheduling based on information longevity. *WWW '08*, 437-446, 2008.

[16] Pitkow, J. and Pirolli, P. Life, death, and lawfulness on the electronic frontier. *CHI '97*, 383-390, 1997.

[17] Ramaswamy, L., A. Iyengar, L. Liu, and F. Douglis, Automatic Detection of Fragments in Dynamically Generated Web Pages, WWW'04, 443-454.

[18] Sebastiani, F. Machine learning in automated text categorization. *ACM Computing Surveys 34(1)*, 1-47, 2002.

[19] Selberg, E. and Etzioni, O. On the instability of Web search engines. In *Proceedings of RIAO '00*, 2000.

[20] Teevan, J., E. Adar, R. Jones, and M. A. Potts. Information re-retrieval: repeat queries in Yahoo's logs. *SIGIR '07*, 151-158, 2007.