Detecting Phrase-Level Duplication on the World Wide Web

Dennis Fetterly
Microsoft Research
1065 La Avenida
Mountain View, CA, USA
+1.650.693.3814
fetterly@microsoft.com

Mark Manasse
Microsoft Research
1065 La Avenida
Mountain View, CA, USA
+1.650.693.2751
manasse@microsoft.com

Marc Najork
Microsoft Research
1065 La Avenida
Mountain View, CA, USA
+1.650.693.2928
najork@microsoft.com

ABSTRACT

Two years ago, we conducted a study on the evolution of web pages over time. In the course of that study, we discovered a large number of machine-generated “spam” web pages emanating from a handful of web servers in Germany. These spam web pages were dynamically assembled by stitching together grammatically well-formed German sentences drawn from a large collection of sentences. This discovery motivated us to develop techniques for finding other instances of such “slice and dice” generation of web pages, where pages are automatically generated by stitching together phrases drawn from a limited corpus. We applied these techniques to two data sets, a set of 151 million web pages collected in December 2002 and a set of 96 million web pages collected in June 2004. We found a number of other instances of large-scale phrase-level replication within the two data sets. This paper describes the algorithms we used to discover this type of replication, and highlights the results of our data mining.

Categories and Subject Descriptors

H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia; K.4.m [**Computers and Society**]: Miscellaneous; H.4.m [**Information Systems**]: Miscellaneous.

General Terms

Measurement, Algorithms, Experimentation.

Keywords

Web characterization, web pages, web spam, data mining, content duplication.

1. INTRODUCTION

Two years ago, we conducted a study [7] on the evolution of web pages. In the course of that study, we crawled a set of 151 million web pages once every week, for a total of 11 weeks, and measured how much each page changed from one week to the next. In so doing, we discovered that pages in Germany were about nine times as likely to change completely as pages on the Web at large. Upon further investigation of this anomaly, we discovered that our data set contained over 1 million URLs (out of about 6 million URLs we crawled in the `.de` domain) that were drawn from 116,654 hosts, but originated from only a single IP addresses, all operated by the same entity. Each of these pages changed completely on every download, irrespective of the time interval between downloads. The pages were generated “on the fly,” and the program or script generating them did not take the requested URL into consideration. More interestingly, the pages consisted of grammatically well-formed German sentences, stitched together at random.

From the content of these pages, it was evident that they were designed to “capture” search engine users. The web site operator dynamically generated pages so as to inject as many pages as possible into the indexes of the major search engines. The links in these pages pointed to other pages that appeared to be on other hosts; however, all the different host names resolved to just a single IP address. Apart from creating the illusion of non-nepotistic links (beneficial in the context of some link-based ranking algorithms), using many different host names also circumvents web crawler politeness policies aimed at not overloading any particular host, if those policies are predicated by the symbolic host name, not the IP address. Finally, by populating each dynamically generated page not with a random sequence of words, but with a succession of grammatically well-formed sentences, the web site operator prevented web searchers from spotting the deceit by simply looking at the brief synopsis that major search engines return with each result.

This discovery motivated us to investigate techniques for detecting *spam web pages*: web pages that hold no actual informational value, but are created to lure web searchers to sites that they would otherwise not visit [9]. We identified some suitable features for identifying spam web pages, including characteristics of the URL, the link structure, the page content, and the rate of change of the pages on a site. However, we realized that it was chance that led us to that particular German site. Had the operator chosen to always generate the same content for the same URL (for example, by using a hash of the URL to seed the random-number generator used to pick sentences from his corpus), we would not have discovered him.

This realization provided the motivation for the work described in this paper. We set out to develop techniques for finding instances of sentence-level synthesis of web pages, and more generally of web pages that consist of an unusually large number of popular phrases (where a popular phrase is a sequence of consecutive words that appears in at least five web pages). In the process of doing so, we discovered some of the most popular 5-word phrases on the web (none of which are very exciting; few of which are even interesting). But by looking for pages such that a large fraction of each page consists of popular phrases (excluding pages that are near-duplicates of other pages), and by averaging the fraction of a page caused by phrase-level replication

SIGIR'05, August 15–19, 2005, Salvador, Brazil.
Copyright 2005 ACM 1-59593-034-5/05/0008…$5.00.

over a web site, we were able to re-identify the aforementioned German site as well as a number of other sites using similar page-generation techniques.

We believe the algorithms described in this paper are capable of identifying one particular kind of web spam, and as such are of interest to web search engines and to authors whose works are being reused.

The remainder of the paper is structured as follows: Section 2 reviews related work; section 3 describes the algorithms we use to pinpoint phrase-level replication, and the discoveries we made by applying our algorithms to two substantial data sets. Finally, section 4 offers some concluding thoughts and avenues for future research.

2. RELATED WORK

In a seminal 1997 study, Broder *et al.* [4] investigated the fraction of web pages that are near-duplicates of other web pages. Their study employed a technique called *shingling*, whose basic idea is to select a small set of m k-word phrases (where k is typically in the range of 4 to 7) uniformly at random from each document. In that study, these phrases form a feature set representing the document, and comparing the feature sets of documents for overlap provides an approximation to similarity. That is, two documents that have substantial overlap in their feature sets are substantially similar. The algorithm described in the 1997 paper required $O((mn)^2$ feature set comparisons to cluster n documents into near-equivalence classes, but more efficient variants of the algorithm, such as the one described in [8], are able to form these near-equivalence classes in nearly $O(n)$ time. In that variant, a set of randomly-chosen selector functions is used, each of which identifies a single phrase from each document. The feature set instead becomes a feature vector, where intersection can be computed position-by-position; by combining multiple features into single features, most documents producing feature sets with large overlap can be quickly identified.

The algorithms described in this paper also use shingling as the basic mechanism for extracting a uniform random sample of features from documents. But while Broder *et al.*'s study was concerned with quantifying the amount of replication at the level of entire web pages, we are interested in the amount of replication at the phrase-level (where a phrase is a sequence of k consecutive words, as opposed to a sentence in the linguistic sense).

As we will show, the algorithms described in this paper are well-suited for detecting a certain class of spam web pages. We first became aware of the prevalence of web spam in the course of the aforementioned study of the evolution of web pages [7] which led us to discover a particularly prolific spam web site in Germany inspiring us to investigate various statistical techniques for identifying spam web pages [9]. The motivation for the work described in this paper stemmed from the realization that we discovered the German web site because it exhibited a statistical anomaly that could easily have been avoided by the web site operator, and that most of our statistical techniques for detecting spam web pages would not have flagged this site.

Several other papers have touched on the phenomenon of web spam. Henzinger *et al.* [10] identified web spam as one of the most important challenges to web search engines. Davison [6] investigated techniques for discovering nepotistic links, *i.e.* link spam. Broder *et al.* [5] investigated the link structure of the web graph. They observed that the in-degree and the out-degree distributions are Zipfian, and mentioned that outliers in the distribution were attributable to web spam. Bharat *et al.* [2] have expanded on this work by examining not only the link structure between individual pages, but also the higher-level connectivity between sites and between top-level domains. More recently, Amitay *et al.* [1] identified feature-space based techniques for identifying link spam.

3. FINDING PHRASE REPLICATION

In this section, we describe the algorithms we used for detecting phrase-level replication, and our observations after applying our algorithms to two data sets.

3.1 The Data Sets

The first data set we considered (DS1) was collected as part of the PageTurner study on the evolution of web pages [7]. It consists of 151 million HTML pages, collected between 26 Nov. 2002 and 5 Dec. 2002 by performing a breadth-first search crawl starting at `http://www.yahoo.com/`. The second data set (DS2) consists of 96 million HTML pages chosen at random from a larger crawl conducted by MSN Search.

3.2 Sampling

In order to determine whether documents are likely to contain a significant number of replicated phrases, we first reduce each document to a feature vector. For this, we employ a variant of the shingling algorithm of Broder *et al.* This reduces the data volume to be considered roughly by an order of magnitude.

We begin by replacing all HTML markup by white-space. We define the *words* of a document to be its maximal sequences of alphanumeric characters, and define the *k-phrases* of a document to be all sequences of k consecutive words in the document. We treat the document as a circle: its last word is followed by its first word. Consequently, an n-word document has exactly n phrases, independent of k. The single phrase for a one-word document will be k repetitions of that one word.

Intuitively, each of the aforementioned feature vectors is a sample of the phrases in the underlying document, chosen so that small changes to the document result in small changes to the feature vector.

Our sampling method exploits several properties of Rabin fingerprints [3],[11]. A Rabin fingerprinting function treats the bits of an input string as the coefficients of a Boolean polynomial. There are many different Rabin fingerprinting functions, each of which is parametrized by a *primitive polynomial* over the ring of Booleans. A primitive polynomial p of degree d has the property that x^i modulo p is not equal to x^j modulo p for all $0 \leq i < j < 2^d$. For degree 64, there are 143,890,337,947,975,680 different primitive polynomials — more than enough for our purposes.

In order to compute the Rabin fingerprint (with respect to a primitive polynomial p of degree d) of a bit-pattern b, we first prepend a one to the bit-pattern and append d zeros, resulting in a bit-pattern b'. The fingerprint of b is (the polynomial interpretation of) b' modulo p. Rabin fingerprints support efficient extension (that is, given the fingerprint of a string a along with a string b, we can compute the fingerprint of ab) and prefix deletion (that is, given the fingerprints of strings a and ab along with the length of b, we can compute the fingerprint of b). Moreover, the fingerprints of any two distinct bit-patterns of length d bits are distinct; in other words, fingerprinting functions are one-to-one when applied to fingerprints. Finally, Rabin fingerprints have good spectral properties.

Having established this background, we can now precisely specify our algorithm for computing feature vectors: First, we fingerprint each word in the document (using a primitive polynomial p_A), thereby reducing an n-word document to n tokens. Next, we compute the fingerprint of each k-token phrase (using a primitive polynomial p_B), utilizing the prefix deletion and extension operations. This leaves us with n phrase fingerprints. Third, we apply m different fingerprinting functions (with polynomials p_1 to p_m) to each phrase fingerprint, retaining the smallest of the n resulting values for each of the m fingerprinting functions. This process leaves us with a vector of m fingerprints representative of the document. We refer to the i^{th} element of this vector as the i^{th} *shingle* of the document, and the elements of the vector collectively as the *shingles*.

In the experiments described below, we set d to 64, k to 5, and m to 84. In other words, we used Rabin fingerprinting functions using degree 64 primitive polynomials[1], five-word phrases, and feature vectors of length 84. These choices reflect experience gained in the course of prior work.

3.3 Duplicate Suppression & Popular Phrases

Previous studies ([4],[8]) have shown that replication of documents (or near-identical versions of a document) is rampant on the web; about one third of all web pages are near-duplicates of a page in the remaining two thirds. Since this study was aimed at investigating phrase-level replication, we did not want to be blinded by those cases of phrase-level duplication that are due to the entire document being replicated. We therefore clustered all the pages in each of our data sets into equivalence classes, where each class contains all pages that are exact or near-duplicates of one another. Two documents are near-duplicates if their shingles agree in two out of six of the non-overlapping runs of 14 shingles, which is unlikely (<5%) to happen when the documents have less than 82.5% commonality in their phrase-sets, and very unlikely (<1%) below 77% commonality. It is likely (>95%) to happen for documents with at least 96.5% commonality of phrase-sets. The equivalence classes are created by taking the transitive closure of this near-duplicate relationship. The reader is referred to an earlier paper [8] for details on the clustering algorithm we used. We then reduced each data set by eliminating all but one element from each equivalence class.

In this paper, we are investigating the prevalence of *popular phrases*. By this, we mean phrases which occur in more documents than would be expected by chance. As we will show, the popular phrases are, by themselves, typically not interesting, consisting of snippets of legal warnings, menus, JavaScript, and similar things. We also study the phenomenon of documents almost completely composed from popular phrases.

Our underlying assumption is that "normal" web pages are characterized by a generative model. In one model, we assume that text is generated using a Markov model: in each context, words are randomly drawn from a context-sensitive distribution. In a dramatically simpler model, we assume that words are added to phrases independently of each other. Given the size of our corpus and the word frequency distribution of its lexicon, we would expect most phrases to occur only once if all pages (minus the aforementioned near-duplicates) were created according to this generative model. It is worth noting that this model is an oversimplification; it ignores grammar, semantics, and idiomatic turns of phrase of human languages. Using a Markov model would better account for these phenomena; we observed that many "innocent" phrases occurred twice or thrice in our corpus.

In contrast, the pages we seek are generated by a copying model. In this model, we assume that a generator has sampled a collection of existing pages, and generates new pages by dividing existing text into sentence-length phrases, and produces new pages from random sequences of sampled phrases. In this case, we need to consider the number of phrases in the sampled collection, the length of typical generated pages, and the number of generated pages; random phrases typically occur at most thrice, occurring more often when copied or idiomatic.

Table 1. Some of the most-popular sampled five-word phrases in data set DS2.

Rank	Number	Phrase (five words long)
1	2266046	4 5 6 7 8
2	1490904	j k l m n
3	1430903	11 12 13 14 15
4	1184293	15 16 17 18 19
5	1133160	t u v w x
6	1069175	n o p q r
7	911276	18 19 20 21 22
8	841245	2 3 4 5 6
9	746061	powered by phpbb 2 0
10	743210	copyright 2000 2004 jelsoft enterprises
12-24	591434-328015	*More alphabetic and numeric sequences, and a jelsoft copyright notice*
25	317287	prev date next thread prev
27	295583	all rights reserved privacy policy
29	262044	user name remember me password
32	240301	vbulletin go to http www
34	224688	today s posts mark forums
35	221541	may not post attachments you
36	220530	notorious celebrity videos you name
37	216834	f***ing free paris Hilton celebrity
38	214960	celebrity videos you name it
41	205478	minnesota mississippi missouri montana nebraska
46	199403	forum powered by phpbb 2
47	194739	send a private message to
56	171197	profile log in to check
60	159460	product you are looking for

[1] For the size of our data sets, 64-bit fingerprinting results in a near-vanishing probability of collision.

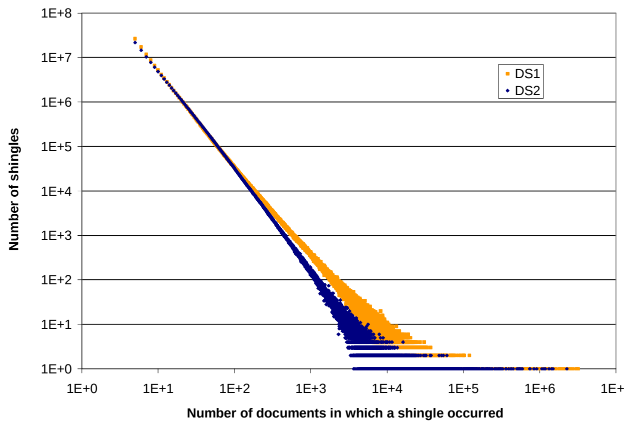

Figure 1. Histogram of occurrences of popular shingles

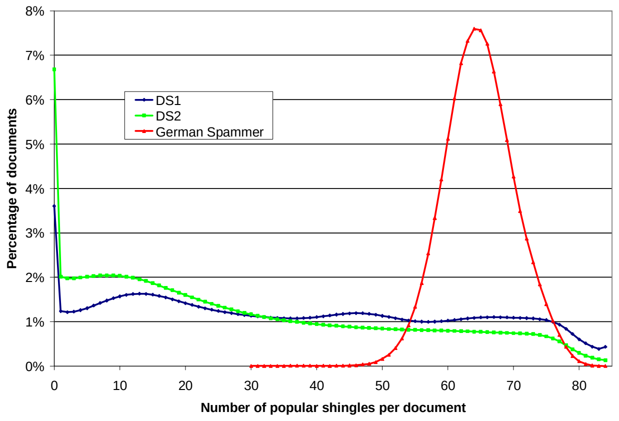

Figure 2. Histogram of popular shingles per document

We limit our attention to those phrases with fingerprints chosen to be shingles by the 84 sampling functions described above. In this way, we can, with reasonable probability, detect the common occurrence of a phrase in two documents, just by comparing shingles. Some phrases are unlikely ever to be selected by this technique (if, for example, the phrase has a large hash-value under all 84 hash functions), but we are more interested in identifying systematic copying than in the individual phrases. We could have attempted this problem without probabilistic sampling, but our data sets contain on the order of 100 billion five-word phrases, which would have made handling the data sets somewhat daunting. We therefore define a phrase to be *popular* if it is selected as a shingle in sufficiently many documents; for collections of the size we were examining, we chose five as an appropriate value for “sufficiently”, largely eliminating selection of independently generated, non-idiomatic random phrases. In a larger collection, we would undoubtedly need to increase this.

To determine the popular phrases, we considered triples (i, s, d), where s is the i^{th} shingle of document d. We extracted all of the triples corresponding to our document collection. We then sorted these triples lexicographically, and looked for sufficiently long runs of triples with matching i and s values.

Table 1 shows some of the most popular sampled phrases in our data set DS2, along with the number of times these phrases were sampled by the shingling algorithm. The first 24 phrases are not very interesting, being largely sequences of consecutive numbers or letters (and further down, alphabetized ranges of U.S. states, names of consecutive months, and the like). We also find boilerplate navigational text, legal disclaimers and copyright notices, and software branding. A common property of all of these phrase replications is that they are not due to collusion: pull-down menus in web pages that allow an online shopper to select the state in which she lives will look similar no matter what; the web page designers came up with similar words independently of each other because there are only a limited set of design choices to be made.

However, starting with the 36th most popular phrase, we start to discover phrases that were caused by machine-generated content. The web pages from which the phrase emanates turn out to have a very templatic form: the pages contain some amount of text that is common to them, some amount that is optional, and lots of “fill in the blank” slots that are populated by two- to four-word units, where the units are chosen by picking words or two-word phrases from a limited dictionary and concatenating them. In the pages we have examined, each unit appeared numerous times on a page[2]. The next two most-popular phrases (and many others not much later) lead to pages similar in spirit (and indeed to some extent to the same pages).

The 60th most-popular phrase, on the other hand, is the first instance of a phrase that occurs in a multitude of pages not because it is templatic (in the innocent or not-so-innocent senses we have encountered before), but because it is idiomatic. In fact, examining pages on which this phrase occurs shows that there is no shared larger context; larger contexts include “Can't find the product you are looking for?”, “Click on a category above to find the product you are looking for”, “If you know the name of the product you are looking for”, “If you cannot find the number for the product you are looking for”, etc.

Figure 1 shows histograms of the number of occurrences of popular shingles in our data sets DS1 and DS2. Not too surprisingly, the histograms exhibit Zipfian distributions.[3] The curves do not show any popular shingles with fewer than five occurrences, since we defined a shingle to be popular if and only if it occurs at least five times in our data sets. Note that the two curves have very nearly (but not quite exactly) equal slopes. In both data sets, there are tens of millions of shingles that each occur in only five documents. At the other extreme (as described above), both data sets contain shingles which appear in millions or hundereds of thousands of documents.

Figure 2 shows a histogram of the number of popular shingles per document. The horizontal axis indicates the number of shingles in a given document that were popular, *i.e.* occurred in at least four other documents; the vertical axis indicates what fraction of documents had a particular number of popular shingles. Two of the curves show the distribution of popular shingles per document of our data sets DS1 and DS2; the third curve shows the distribution for just the 1.07 million pages that

[2] We deemed it inappropriate to include any of these pages. The interested reader is encouraged to locate the pages using her favorite search engine, preferably in the privacy of her own home, and with scripting turned off, pop-up blocking enabled, page redirection disabled, and children at a safe remove. Viewer discretion is advised....

[3] No paper on statistics of web pages is complete without a graph showing a power-law distribution.

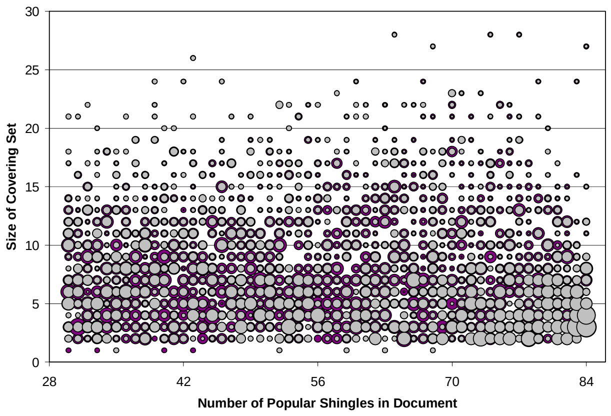

Figure 3. Covering set distribution for subset of DS2, sampling 50 pages per count of popular shingles

belonged to the German "spam" site that provided the initial motivation for this work.

The DS1 and DS2 curves differ significantly: DS2 contains about twice as many documents which contain no popular shingles at all. One possible source of this difference is that the crawling policies that gave rise to each data set were very different; the DS2 crawl considered page quality as an important factor in which pages to select; the DS1 crawl was a simpler breadth-first-search crawl with politeness.

As can be seen, the German "spam" site exhibits a very different distribution than our data sets at large: the vast majority of the pages emanating from this site have 50 to 80 shingles in common with other pages in DS1. Putting it differently, half to all of the phrases on these pages occur on other web pages as well. In other words, this distribution characterizes the "slice and dice" phrase replication that we observed. These pages were the HTML equivalent of Frankenstein's monster: they were stitched together from parts of other web pages. But unlike in Mary Shelley's novel, each part, once collected, was incorporated into thousands of pages. These two observations motivated us to devise a metric for such a phenomenon: a "covering set" of a web page, that is, a set of donor web pages that supplied the constituent parts.

3.4 Covering Sets

To look for evidence of pages constructed from snippets of other pages, we look at *covering sets* for the shingles of each page. We can only hope to cover the popular shingles for each document, so a covering set for document *f* is a set of documents *G* not including *f* such that every popular shingle of *f* is contained in the shingle set of some document $g \in G$. That is, if S_g is the shingle set of *g*, with elements $(i, s_{i,g}, g)$, and *I* is the set of *i* for which $(i, s_{i,f})$ is a popular shingle, then for all $i \in I$, there is some *g* in *G* such that $s_{i,f} = s_{i,g}$. Finding the minimum size of a covering set is NP-complete, in general,[4] so we don't try for this. Instead, we approximate a minimum covering set using a *greedy* heuristic: add to *G* a document with a shingle set with the largest non-empty intersection with the as-yet-uncovered portion of the shingle set for *f*. If there are multiple documents with the same size intersection, pick one with maximal intersection with S_f.

In some of the studies below, we place additional requirements on our choice of a document to add: we may add documents from the same domain as *f*, or from an IP address from which *f* might have been retrieved, only when no documents from other hosts have non-empty intersection with the set of as-yet-uncovered shingles. In this case, we define the *foreign contribution* of the covering set to be the number of shingles that are covered by documents from an IP address different than any of the ones that might have served *f*. In some of the studies below, we impose still further restrictions, by restricting the set of candidates to come from pages with other special properties (for DS1, for example, we only have complete pages for a subset, so we may prefer those pages to ones which contribute a greater number of matching shingles).

One could easily perform post-processing to discard sets added to the greedy cover which are rendered redundant by later sets; in the results reported below, we did not do this.

Computation of the greedy covering sets was performed by another merge-sort variant: initialize fingers into the sorted file of shingles at the first position at which the popular shingles occur. Take the smallest document referred to by any finger; if this document differs from the document previously under consideration, consider the previous document as a candidate for greedy selection based on the number of still-needed shingles it contains (and the IP address, and total shingle match, and any other criteria in play), and reset the counters associated with the current document. In any case, remember that the current document has coverage of the corresponding shingle, and advance that finger. The performance of this is the product of the number of overlapping shingles and the size of the covering set, which is expensive enough that we found covering sets only for selected samples of our documents.

Figure 3 shows the distribution of covering set sizes for a small sample of web pages from DS2. On the horizontal axis, we see the number of popular shingles for a given page. We sampled data set DS2 by choosing 50 documents for each unit on the horizontal axis, starting at documents with 30 popular phrases in their shingle sets. The vertical axis shows the size of the covering set chosen by the greedy algorithm for each document *d*; the algorithm favored foreign documents, that is, documents from IP addresses different than that of *d*. The area of the outer disc at each point corresponds to the number of documents contributing to this point; the inner disc is proportionally scaled to show the average foreign contribution; if all the documents corresponding to a disc can be assembled from foreign parts, the inner disc will completely cover the outer disc. From this graph, it appears that documents consisting mostly of popular phrases can be assembled from relatively few other documents, and that most popular phrases in documents with more than 30 such phrases are drawn from documents on other web servers.

[4] The reduction from 3-SAT is easy: given a formula to test for satisfiability, assign consecutive integers to the variables, and then to the clauses. The set we try to cover is all of the variables and all the clauses. The set of available sets consists, for each variable *i*, of a positive set and a negative set. Both contain *i*, and the positive set contains the clauses in which variable *i* occurs positively, and conversely for the negative set. The complete set forms a trivial cover; a cover of size equal to the number of variables corresponds to a satisfying assignment. To cover variable *i*, any cover must contain at least one of the corresponding positive and negative sets.

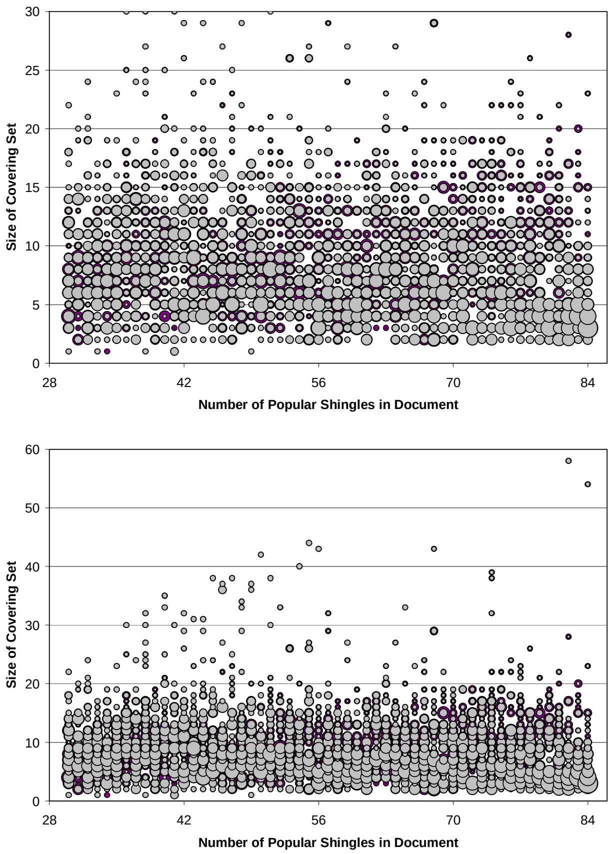

Figure 4 (A & B). Covering set distribution for subset of DS1, sampling 50 pages per count of popular shingles

Figure 4 A & B are similar to Figure 3, but show the distribution of covering sets in DS1 instead. The vertical axis of the upper figure tops out at 30, to make the comparison with Figure 3 easier; the vertical axis of the lower figure extends up to 60 to show all sampled cover sets. The principal difference we see is that some documents in DS1 have larger covering sets than we find in DS2; in other words, the replicated parts of these documents are assembled from a greater number of other documents. Also, less area of the outer discs is visible, which indicates that the foreign contribution to the documents sampled from DS1 is even greater than for those from DS2. Strikingly, this is particularly true for documents with relatively fewer (say, 30 to 50) popular phrases.

Figure 5 also shows the distribution of covering set sizes for a sample of DS2, but unlike in Figure 3, the samples were drawn uniformly at random from those pages in DS2 that contained at least 30 popular phrases in their shingle sets. Similarly, Figure 6 bears the same relationship to Figure 4. The difference in the distributions that we first saw in Figure 2 is quite pronounced when comparing Figure 5 to Figure 6, with the former much more heavily weighted towards the left end.

In Figure 7, we again sampled pages uniformly, but restricted the sampling process to pages emanating from the IP address of the German spammer whom we discovered during the original collection of data set DS1. Within that set of pages, we again drew pages uniformly at random from those pages that contained at least 30 popular phrases in their shingle sets. The distribution looks markedly dissimilar from the previous graph: the number of popular phrases is concentrated between 55 and 77, and the covering sets are larger. The covering set sizes (depicted on the vertical axis) are also remarkably well concentrated, and the outer discs are almost completely obscured, *i.e.* the covering sets consist almost entirely of foreign contributions, suggesting that most phrases on each page are copied from pages on foreign servers.

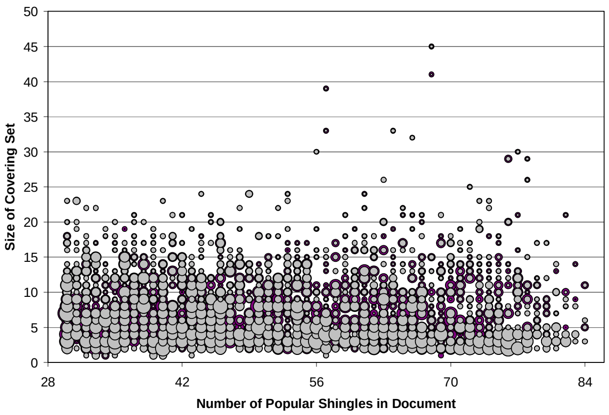

Figure 5. Covering set distribution for subset of DS2, uniformly sampling pages with at least 30 popular shingles

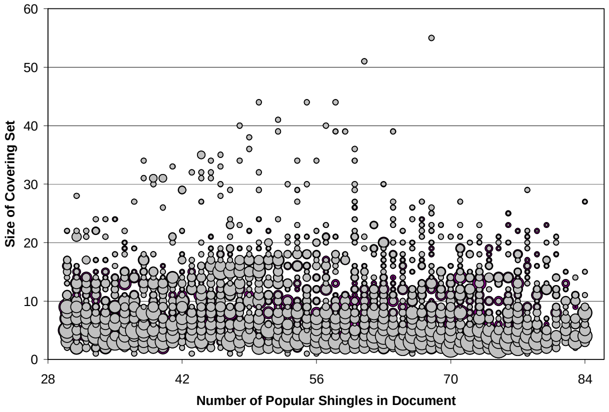

Figure 6. Covering set distribution for subset of DS1, uniformly sampling pages with at least 30 popular shingles

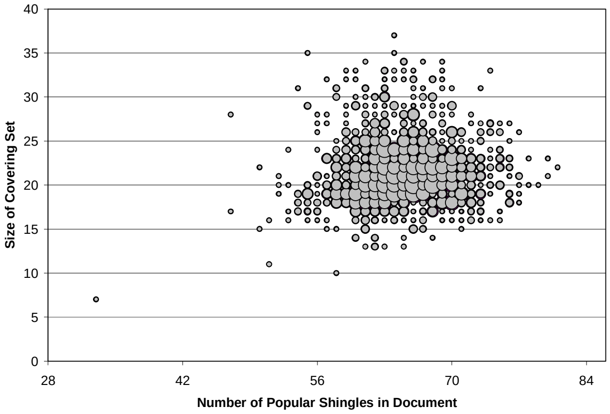

Figure 7. Covering set distribution for German spammer, uniformly sampling pages with at least 30 popular shingles

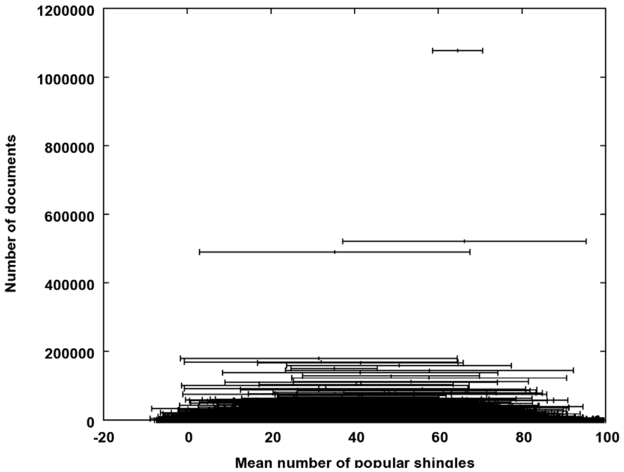

Figure 8. Mean and standard deviation of the number of popular phrases in each document from a given host in DS1.

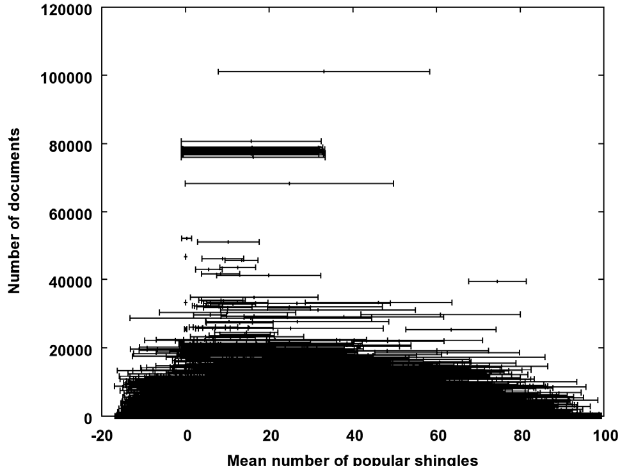

Figure 9. Mean and standard deviation of the number of popular phrases in each document from a given host in DS2.

3.5 Here There Be Monsters

To look for likely sources of phrase-duplication, we computed the mean and standard deviation for the number of popular phrases per document, averaged over complete host-name-determined web sites. Figure 8 shows all those web sites that have at least 10 pages in DS1 and for which the mean popular shingle count is at least 30. The horizontal axis displays the mean number of popular phrases per document, while the vertical axis displays the number of pages found at a web site. Each web site is depicted by a dot at the mean value for all pages on the site, with error bars stretching left and right with radius equal to the standard deviation. Our prototypical German adult spam site can be seen isolated near the upper right corner of this figure. In the corresponding graph for DS2, shown in Figure 9, there is no such dominant outlier.

We looked for other points in Figure 8 exhibiting similar characteristics. Our best current technique requires manual inspection of web-sites; this has limited the number of examples we have investigated to date. As candidates, we chose web sites with high page count and small (<15) standard deviation. Ranked by page count, our German pornographer sits in the first position. A weather site sits in second, with roughly half as many pages; these pages vary based on the location, a brief summary of the weather, and the weather outlook. These weather reports are drawn from a very limited vocabulary of terms, so the number of possible orderings of phrases describing the weather is small, but the pages differ enough to not cluster as nearly-equivalent.

Following that, in 3rd, 5th, and 6th places come database-driven information repositories (about.com, citysearch.com, bizrate.com); pages on these sites are dynamically generated from a small set of templates, and typically consist of brief on-topic content surrounded by a significant amount of boilerplate text.

In 4th, 7th, and 13th places come some pornographic websites, with roughly 100,000 pages each in DS1, but highly varying (35, 70, and 77) mean popular phrase count.

In 8th and 9th places, we strike paydirt: with mean count 75 and 94 thousand pages, a satellite TV spam site, slicing and dicing their own content, with follow-on links to no-prescription pharmaceutical sites and the like, and, with mean count 66, a website with a variety of spam, where over half of each page contains content lifted from the DMOZ open directory project. The remaining spots in the top fifteen are more database-driven sites.

Turning our attention to DS2, as depicted by Figure 9, we discovered that this data set, while containing fewer pages per site, is if anything more susceptible to pornographic web pages of the “fill in the blanks” variety, and has a bias towards French and Asian pornographic spam, whereas DS1 was biased towards German spam pages. Other than that, the top spots are occupied by other database-driven and conventional spam sites.

Table 2 shows two example covering sets from DS2. The bold URLs are the documents for which we computed the covering sets (the “monsters”), the lines below are the constituents of the covering sets (the “donors”). The first example is an instance of a web page borrowing content from the DMOZ open directory, while the second example is an instance of a catalog page where the individual product descriptions (on-line courses in this case) are replicated across many other web pages.

Table 2. Example covering sets from DS2.

http://www.searchingweb.com/cgi-bin/directory/searchingweb.cgi?/Business/Investing/

http://www.moneywebsearch.com/directory/Information_Technology/
http://newhoo.com/Business/Investing/
http://www.specialistweb.com/about.shtml
http://www.eclassifiedsweb.com/classifieds/help.shtml
http://www.omnicient.com/search/dmoz/index.asp?/Business/Investing/
http://www.dmoz.org/Arts/Literature/Authors/M/Munsey,_Terence/
http://www.nosachamos.com/cgi-bin/odp/index.cgi?/Business/Investing/
http://www.calculatorweb.com/calculators/transactioncalc.shtml
http://www.forwardingweb.com/forwarding/join.shtml
http://www.calculatorweb.com/cgi-bin/directory/click.cgi?account=ButtonAffiliate
http://www.registryweb.com/cgi-bin/whois/whois.cgi?show_global=1
http://www.specialistweb.com/contact.shtml
http://www.searchingweb.com/cgi-bin/directory/searchingweb.cgi?/Reference/
http://www.searchingweb.com/cgi-bin/directory/searchingweb.cgi?/Business/
…

http://www.accounting-courses.com/education/skill-development.htm

http://www.career-training-courses.com/education/TestPreparationStudies.htm
http://www.career-training-courses.com/education/careertraining.htm
http://experts.universalclass.com/dezra
http://www.writing-tools-courses.com/education/writinggenre.htm
http://www.business-software-courses.com/education/computers.htm
http://www.math-courses.com/course.htm
http://www.history-courses.com/education/counseling/selfimprovement.htm
http://www.self-awareness-courses.com/education/careertraining.htm
http://www.domestic-violence-courses.com/education/ParentingandFamily.htm
http://www.curiocityforkids.com/newthismonth.htm
http://www.ged-test-preparation.com/course.htm
http://www.counseling-courses.com/courses/satmath.htm
http://home.universalclass.com/i/crn/7514.htm

These examples illustrate the strengths and weaknesses of the covering set technique: While it finds most instances of “slice & dice” page generation, it does not distinguish legitimate from inappropriate uses.

4. CONCLUSIONS AND FUTURE WORK

This paper described a number of techniques for detecting phrase-level duplication on the web, and applied these techniques to two large collections of web pages. We have learned a number of things (and hope continued inspection will reveal more):

(1) Phrase frequency on the web, like word frequency, follows a power-law distribution. Most phrases occur in only a single document, but we discovered seven phrases in the shingle sets of all non-near-duplicate documents in our data set DS2 (which contains 96 million HTML pages before near-duplicate elimination and 77 million thereafter) that occurred in over a million documents each.

(2) The most popular phrases are not very interesting; they are often informed by limited design choices. For example, a web-based shopping site may have a pull-down menu for US state names in alphabetized order, making any *k*-word window into this list of states a highly popular phrase. Similarly, days of the week, months of the year, and numeric ranges for days of the months give rise to sets of highly popular phrases. This type of phrase-level replication occurs “in the wild”, *i.e.* on independent web sites without any collusion by the content providers.

(3) Legal disclaimers form a second class of popular phrases. Some of these legal disclaimers (*e.g.* “Copyright 2004 All rights reserved”) occur in the wild, others (*e.g.* “Copyright 2000-2004 Jelsoft Enterprises”) are specific to a particular site, but *really* popular on that site.

(4) Navigational phrases (such as “[Date Prev] [Date Next] [Thread Prev] [Thread Next]”, which appears in our list as “prev date next thread prev”) and boilerplate phrases (such as “Powered by phpBB 2.0”) are symptoms of many web sites using the same underlying infrastructure, such as mailing list archives and bulletin board systems.

(5) Starting with the 36th most-popular phrase (which is an order of magnitude less popular than the most popular phrase), we start to see phrases that are evidence of “fill in the blanks” machine-generated spam web pages. These phrases typically do not occur “in the wild”; they are either rampant on one particular site (modulo the fact that many web sites use DNS wildcarding to make one machine appear as millions of hosts), or on a network of sites operated by affiliated parties.

(6) 96% of all pages in DS1 and 93% of all pages in DS2 contain at least one phrase in their shingle set that is *popular*, *i.e.* shared with at least 4 other documents that are not near-duplicates of any other. For 44% of the pages in DS1 and 29% of the pages in DS2, at least half of the phrases included in the shingle set are popular. In other words, even after we discarded all those pages that are outright copies or near-duplicates of other pages about a third of the pages on the web consist of more replicated than original content!

(7) Sites with a high fraction of non-original phrases typically feature machine-generated content, and appear to be more likely to be “spam” sites than pages with predominantly original, non-replicated content.

We believe that the techniques described in this paper may be of interest to search engines, in that they provide another tool for spotting sites that publish large amounts of machine-generated spam web pages (of the “slice and dice” variety), and more generally in that these techniques provide a way to estimate how original the content of a page or an entire site is. A search engine might decide to underemphasize sites in its index that have high levels of phrase replication, or conceivably try to extract and index only the original (and thus presumably more salient) portions of each page.

5. REFERENCES

[1] Amitay, E., Carmel, D., Darlow, A., Lempel, R., and Soffer, A. The Connectivity Sonar: Detecting Site Functionality by Structural Patterns. In *14th ACM Conference on Hypertext and Hypermedia* (Aug. 2003), 38–47.

[2] Bharat, K., Chang, B., Henzinger, M., and Ruhl, M. Who Links to Whom: Mining Linkage between Web Sites. In *2001 IEEE International Conference on Data Mining* (Nov. 2001), 51–58.

[3] Broder, A. Some applications of Rabin's fingerprinting method. In Capocelli, R., De Santis, A., and Vaccaro, U., editors, *Sequences II: Methods in Communications, Security, and Computer Science*, 143–152, Springer Verlag, 1993.

[4] Broder, A., Glassman, S., Manasse, M., and Zweig, G. Syntactic Clustering of the Web. In *6th International World Wide Web Conference* (Apr. 1997), 393–404.

[5] Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., and Wiener, J. Graph Structure in the Web. In *9th International World Wide Web Conference* (May 2000), 309--320.

[6] Davison, B. Recognizing Nepotistic Links on the Web. In *AAAI-2000 Workshop on Artificial Intelligence for Web Search* (July 2000).

[7] Fetterly, D., Manasse, M., Najork, M., and Wiener, J. A large-scale study of the evolution of web pages. In *12th International World Wide Web Conference* (May 2003), 669–678.

[8] Fetterly, D., Manasse, M., and Najork, M. On the Evolution of Clusters of Near-Duplicate Web Pages. In *1st Latin American Web Congress* (Nov. 2003), 37–45.

[9] Fetterly, D., Manasse, M., and Najork, M. Spam, Damn Spam, and Statistics: Using statistical analysis to locate spam web pages. In *7th International Workshop on the Web and Databases* (June 2004), 37–45.

[10] M. Henzinger, R. Motwani, C. Silverstein. Challenges in Web Search Engines. *SIGIR Forum* 36(2), 2002.

[11] Rabin, M. Fingerprinting by random polynomials. Report TR-15-81, Center for Research in Computing Technology, Harvard University, 1981.