Improving Web Information Indexing and Retrieval based on Center Block Duplication Detection

Tyrone Cadenhead ¹, Jinlin Chen² and Terry Cook³

¹ Computer Science Department
Queens College, City University of New York
thcaden2003@yahoo.com

² Computer Science Department
Queens College, City University of New York
jchen@cs.qc.edu

³ Computer Science Department
Graduate Center, City University of New York
Terrycook1@aol.com

Corresponding author:

Jinlin Chen
Computer Science Department,
Queens College, City University of New York
65-30 Kissena Blvd,
Flushing, New York, 11367
USA

Email: jchen@cs.qc.edu
Tel: (001)-718-997-3497
Fax: (001)-718-997-3513
Abstract: Duplicated information in today’s Web has serious negative impact to Web search engines in that it increases the size of the index and results in low efficiency for Web information retrieval. One important fact is that a large amount of Web content duplication happens at block level in addition to site and page level due to various reasons. Besides, when searching through the Web, in most cases the desired information is located at the center block of a relevant page. Based on these two observations, we propose an efficient block level duplication detection algorithm based on resemblance transitivity, and index center blocks instead of entire Web pages for Web information retrieval. Experiments show that these strategies can effectively reduce index size and index construction time without sacrificing the effectiveness of Web information retrieval.

Keywords: duplication detection, inverted index, layout structure detection, information retrieval
Improving Web Information Indexing and Retrieval based on Center Block Duplication Detection

I. Introduction

A large amount of redundant information exists in today’s Web at different levels such as site, page, and content block level. One major problem of redundant Web information is that search engines have to crawl, store and process duplicated content, which greatly increases the size of the indices and results in low efficiency for Web information retrieval. Besides, users may feel frustrated with the same content repeated in the search results.

Many approaches [1]-[3][5]-[19] have been proposed to detect Web content duplication, based on which we can decrease the size of indices and achieve better performance for Web search engines. One major feature of most previous approaches is that they detect duplication at site or page level, which is not enough because quite a large amount of duplication happens at content block level inside Web pages for most of today’s websites. Fetterly et al [16] proposed phrase-level duplication detection, but that approach is more suitable for spam detection instead of improving index compression.

On the other hand, traditional IR systems search terms throughout a Web page to decide whether the Page is relevant or not. However, for normal Web pages, in most cases the desired information is located at the center block of the page (Based on our experiments, this observation is also true for some of the new developments of the Web 2.0 phenomena such as Weblog. This may not be true for some new media type which involves rich multimedia information. However, in our case we mainly consider text information retrieval and therefore we assume that the content a user search is mainly text based.). Other blocks, e.g. header, footer, navigation bars, etc., often “pollute” the search results with irrelevant information. These parts increase the size of indices, and also make the retrieval results less precise. Yu et al [20] performs Web information retrieval based on content segment instead of entire Web page and achieves promising result. However, the focus of that paper is information retrieval and does not deal with duplicated information detection and indexing.

Complementary to previous approaches on site and page level duplication detection and full page indexing and searching, in this paper we propose the following strategies on Web content duplication detection, indexing and searching, (1) detecting duplication at block level; (2) indexing only the unique center blocks of pages for Web information retrieval. We present an efficient duplication detection algorithm based on resemblance transitivity is developed, which can greatly decrease the complexity of duplication detection. Compared to page level indexing without duplication elimination, our strategies achieve a much smaller index size (65% of the original index) in our testing Web page collection without sacrificing the effectiveness of Web information retrieval. Furthermore, these strategies also reduce index construction time by 71% in our experiment.

The rest of the paper is organized as follows. Section II describes related work on duplication detection and Web page segmentation. Section III discusses the motivation of this paper. Section IV proposes detailed approach on block level duplication detection and indexing. Section V discusses experimental results and performance evaluation. Section VI summarizes the paper and discusses future work.

II. Related Work

In this section we first give a review on related work on duplication detection. Besides, in this paper we will make use of page segmentation technique to detect content block regions in a Web page, which is the basis for our proposed center block duplication detection approach. Because of this, we will also give a brief introduction on related work on page segmentation.

A. Duplication detection

Duplicated contents refer not only to completely identical contents but also to those near identical contents. Exactly duplicated content can be easily detected using a hash function based approach. Unfortunately, quite a large amount of duplicated contents are actually near-duplicated contents in which slight differences exist. Near-duplicated content detection requires more complicated algorithms at a higher computation cost. In this paper we focus on the near-duplicated content detection unless otherwise specified. Generally speaking, there are three major types of duplication detection methods, shingling based techniques, similarity measure calculations, and image processing based approaches.

Shingling based techniques [1][2][5][11][12][19][21][22] divide each document into multiple shingles. Each shingle is a set of contiguous terms and assigned a hash value for direct similarity comparison. The resemblance of two documents can be calculated based on the number of matching shingles. Fetterly et al [16] uses similar approach for phrase level duplication detection. Due to the large number of documents to be examined in today’s Web information retrieval systems, many approaches have been proposed to reduce the number of shingle comparisons. Broder et al [2] selects shingles with the lowest N hash values and removes shingles with high frequencies. This reduction, however, may lead to less accurate results. In [22] exact copies are removed in advance and then every two or four lines of document are made as a shingle. Fetterly et al. [12] use five-gram as a shingle and apply a 64-bit hash to get fingerprints of shingles, and then employ 84 different hash functions to
construct a feature vector for each document. One problem with these approaches is that they did not address the impact of the sampling strategies to the accuracy of their duplication detection algorithms.

Approaches that compute document-to-document similarity measures \([3][9][15][25][26][27]\) use similarity computations to group potentially duplicate documents. All pairs of documents are compared, which results in an \(O(N^2)\) complexity. Information filtering approach is used to reduce the computational complexity. The decision about whether a document contains new information (therefore not a duplicated document) depends on whether the relevant information in the document is covered by information in documents previously examined. Therefore, decisions about duplication depend on where in the collection stream a document occurs \([10]\).

Research for image based duplication detection makes use of image processing techniques to detect duplication rather than document processing. These types of approach can be found in \([7][23][24]\).

Most previous approaches focus on detecting duplication at site or page level. Unfortunately, for today’s commercial websites, quite a large amount of duplication actually happens at block level inside Web pages.

B. Web page segmentation

Web page segmentation is important for many applications such as adaptive Web content presentation \([4][30]\), Web information retrieval \([20]\), etc.

Many approaches have been proposed for Web page segmentation. In our previous work \([4][32]\) visual clues as well as heuristic based approaches were used for detecting Web page layout structure. Kovacevic et al \([28]\) detects common page areas such as header, footer, left and right menu, and center of a page based on visual information. Cai et al \([31]\) use a vision based page segmentation algorithm to detect hierarchical structure of Web content block. Romero et al \([34]\) partition pages into content blocks using clustering method directly based on Document Object Model (DOM) [DOM] tree of Web pages. Vitali et al \([33]\) use a heuristic approach based on DOM tree for analyzing structure of Web pages. Chen et al \([30]\) use a rule based approach based on both visual and DOM information to detect Web page structure. Generally speaking, approaches based on visual information are more robust and extensible comparing to DOM based approaches.

III. Motivation

A. Block level duplication

Web duplication happens at different levels out of different reasons.

At site level, one major reason of site duplication is website mirroring (usually to increase accessibility and reliability of a website). At page level, duplication may be the result of legal or illegal copy. At block level, duplication may happen across pages inside the same website, in the form of shared fragments such as navigation bars, legal information, etc. This is often the result of using the same template to generate a set of pages in the same website. Fig. 1 shows two example pages from the same website. From the two pages we can find that except for the center blocks as labeled in the pages, all other blocks have exactly the same information. Actually this is quite normal for most of today’s websites.
Our experiments show that around 7.8% of center blocks in pages across different sites are duplicated (See Table 3 in Section V). It means that eliminating duplicated center blocks alone can effectively decrease the index size by 7.8% if we index center blocks of Web pages. A reduced index will greatly improve the performance of Web search engines. Therefore, we will detect block level duplication as a complementary to page and site level duplication detection.

To detect block level duplication, we need first correctly decide block boundaries and then classify the major block regions. Besides, we need find an efficient approach to compare pair-wise resemblance of blocks for duplication detection. We will address these issues in detail in Section IV.

### B. Center block indexing

Most previous information retrieval systems index whole Web pages despite the fact that in most cases user desired information is located at the center blocks of retrieved pages, which can be easily verified in the two example Web pages in Fig. 1. Other parts of a page often “pollute” the search result with unrelated information. These parts increase the size of index and the time for ranking, and lower the number of valid responses provided to the user. For example, the two pages in Fig. 1 may both be returned as relevant if we search with the keywords *Queens College* and *Academic program review*. However, examining the pages in detail, we can decide that only page 1 is relevant because page 2 contains the keywords in the header, left navigation bar, and footer blocks, which are common areas from templates. The major content block of page 2 bears no relevance to the search.

Table 1 shows experiment results on the location of desired information on normal Web pages. The experiment was done by sending queries to Google (www.google.com), and then checking the first 50 relevant results (first 5 returned pages from Google if each page is configured to return 10 links) for each query to see how many of the first 50 links include desired information in the center region (Note: Here we only check the first 50 links because in most cases users stop at the first several returned pages from Web search engines. If a link points to a non-html document such as a pdf document, we simply discard that link and retrieve an additional html link.)

From Table 1 we can find that in most cases (more than 97%) the desired information is located at the center block of a page. This observation is further confirmed by our experiments in Experiment 4 of Section V.

<table>
<thead>
<tr>
<th>Query</th>
<th>Center block</th>
<th>Other blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network security</td>
<td>49</td>
<td>1</td>
</tr>
<tr>
<td>Red Hat Linux</td>
<td>49</td>
<td>1</td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web Indexing</td>
<td>48</td>
<td>2</td>
</tr>
</tbody>
</table>

*Table 1. Block regions where queries are satisfied in normal Web pages*

Table 1 indicates that the desired information is generally located at the center block of normal Web pages. To study the effectiveness of this observation on new developments in the Web, in particular, the Web 2.0 phenomenon, we have also performed similar queries on Google Blog Search ([http://blogsearch.google.com/](http://blogsearch.google.com/)). Blog is an important new way of using the Web and a typical Web 2.0 phenomenon.

Table 2 summarizes the experiment results on the location of desired information on blog pages. The experiment was done by sending queries to Google Blog Search, and then checking the first 50 relevant results for each query to see how many of the first 50 links include desired information in the center region.

From Table 2 we can find that in most cases (more than 96%) the desired information is located at the center block of a page for blog search.

<table>
<thead>
<tr>
<th>Query</th>
<th>Center block</th>
<th>Other blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network security</td>
<td>48</td>
<td>2</td>
</tr>
<tr>
<td>Documentation</td>
<td>49</td>
<td>1</td>
</tr>
<tr>
<td>Web Indexing</td>
<td>48</td>
<td>2</td>
</tr>
</tbody>
</table>

*Table 2. Block regions where queries are satisfied in blog pages*

Based on the above experiments and observations, we can perform indexing and searching on center blocks of Web pages without sacrificing the performance of Web search engines while at the same time improve the response efficiency.

### Center Block Duplication Detection and Indexing

In this section we discuss our approach for center block duplication detection and indexing in detail. First we define the major block regions in a Web page. We then detect block regions inside a Web page based on visual information. Further on we present a novel approach for efficient content block duplication detection. Finally we index the unique center blocks of a Web page collection for information retrieval.

### C. Block region definition

As discussed in [28], a Web page can be divided into five major block regions, header, footer, left, right, and center. Header represents the headline or abstract of a Web page or gives highlight information. Footer adds some annotations such as copyright or contact information to a Web page. Left and Right are generally sidebars providing navigational guide. Center is the major part of a Web page, which presents the major content to users. Generally speaking, a Web page consists of one or more of these five regions. Fig. 2 shows an example page with the five block regions.
D. Block detection

To detect content block regions in a Web page we make use of Function based Object Model (FOM) [4] invented by one of the authors which detects the layout structure as well as functions of block Objects in a Web page. Here Object is the basic building element of a hypermedia system. It is a piece of Web content with some certain functions. Object can be classified into Basic Object and Composite Object. A Basic Object is the smallest information body that cannot be further divided. A Composite Object recursively consists of sub-Objects (Basic and Composite Objects) integrated under some clustering rules to perform certain functions. A Web page is a root level Composite Object. The five content block regions as defined in the sub-section above are sub-Objects inside a page Object. Recursively unfolding all the children Objects inside a page until all the leaves are Basic Objects, we have a tree-like hierarchical structure of the page, which reveals the layout structure of the page. For example, for the Web page shown in Fig. 2, the layout structure is shown in Fig. 3(a), which can be further abstracted using a hierarchical structure as shown in Fig. 3(b).
be considered as separator between blocks. Separators along X and Y axis are combined to detect content blocks inside a Web page. Starting from the lowest level (basic HTML elements), we perform projection recursively to detect the hierarchical layout structure of the whole page. Blocks are merged if they are visually similar to avoid breaking logical chunks. Comparing to approaches that detect page structures directly based on HTML DOM tree [33][34], our approach is independent of physical realization, more robust, and can be applied to contents with various physical implementation formats. Based on the detected hierarchical structure of the page, we further perform functional analysis [4] to all the Objects detected by FOM, we further make use of heuristics proposed by Chen et al [30] for detecting the five regions in a Web page. The heuristics make use of a combination of both shape and size to decide the position of different content block regions and is effective for both single column and multiple column blocks.

After detecting the five possible regions of the page, we further decide the content of a region by extracting content information of all the sub-Objects that belong to the region in the DOM tree (currently we focus only on text information). This will be used for similarity comparison in between different content block regions.

E. Block duplication detection

1) Block resemblance definition
In this paper we make use of block resemblance to decide whether two blocks are duplicated. Blocks with resemblance higher than a predefined threshold are considered duplicated blocks. The resemblance \( r(A, B) \) of two blocks \( A \) and \( B \) is defined as follows.

First each block is transformed into a set of \( k \)-grams (or shingles) denoted as \( S_i \), then resemblance is computed by:

\[
r(A, B) = \frac{|S(A) \cap S(B)|}{|S(A) \cup S(B)|},
\]

where \( |S| \) is the size of set \( S \).

2) Resemblance transitivity
To detect block duplication in a collection, one basic consideration is that we calculate the resemblance of each pair of blocks in the collection, if the resemblance is larger than a certain threshold (0.8 in our experiment), we will consider this pair of blocks duplicated. Otherwise the two blocks are not duplicated.

The major problem of this approach is that the computation complexity is \( O(N^2) \). This may be quite computational-intensive for a practical duplication detection system which handles millions or even billions of blocks.

To solve this problem we take a special pair reduction strategy based on the observation that resemblance in between blocks has transitivity. That is, if blocks \( A \) and \( B \) are similar to each other, and \( B \) and \( C \) are also similar to each other, then \( A \) and \( C \) are similar to some extent. On the other hand, if blocks \( A \) and \( B \) are similar to each other, and \( B \) and \( C \) are not similar to each other, then \( A \) and \( C \) are not similar to some extent.

Below we give a formal analysis to the transitivity of block resemblance.

**Theorem 1** For two content blocks \( A \) and \( B \), if \( r(A, B) \) is given, we have

\[
S(A) \cap S(B) = \frac{r(A, B) * (|S(A)| + |S(B)|)}{1 + r(A, B)}
\]

**Proof:**
Based on set theory, we have

\[
|S(A) \cup S(B)| = |S(A)| + |S(B)| - |S(A) \cap S(B)|
\]

Based on the definition of \( r(A, B) \), we have

\[
r(A, B) = \frac{|S(A) \cap S(B)|}{|S(A) \cup S(B)|} = \frac{|S(A) \cap S(B)|}{|S(A) + |S(B)| - |S(A) \cap S(B)|}
\]

Therefore,

\[
|S(A) \cap S(B)| = \frac{r(A, B) * (|S(A)| + |S(B)|)}{1 + r(A, B)}
\]

\[\Box\]

Theorem 2 gives the lower bound of \( r(A, C) \) if \( r(A, B) \) and \( r(B, C) \) are given.

**Theorem 2** For three content blocks \( A, B, C \), if \( r(A, B) \) and \( r(B, C) \) are given, we have

\[
r(AC) \geq \frac{|S(B) \cap S(C)| - |S(B) \setminus S(A)|}{|S(A)| + |S(C)| - |S(B) \cap S(C)| - |S(B) \setminus S(A)|}
\]

Note: Operation \( \setminus \) represents set difference operation.

**Proof:**
Based on set theory, we have

\[
S(A) = (S(A) \cap S(B)) \cup ((S(A) \setminus S(B))
\]

and

\[
(S(A) \cap S(B)) \cap ((S(A) \setminus S(B)) = \emptyset
\]

Therefore,

\[
|S(A) \cap S(C)| = |S(A) \cap S(B) \cap S(C)| + |(S(A) \setminus S(B)) \cap S(C)|
\]

Similarly, we have

\[
|S(B) \cap S(C)| = |S(B) \cap S(A) \cap S(C)| + |(S(B) \setminus S(A)) \cap S(C)|
\]

Based on (4) and (5), we have

\[
|S(A) \cap S(C)| = |S(B) \cap S(C)| - |(S(B) \cap S(C)) - (S(B) \setminus S(A))|
\]

Based on set theory, we have

\[
|S(A) \setminus S(C)| \leq |S(B) \setminus S(A)|
\]

and

\[
|S(B) \setminus S(C)| \leq |S(B) \setminus S(A)|
\]

Based on (6), (7), and (8), we have

\[
|S(A) \cap S(C)| \geq |S(B) \setminus S(C)| - |S(B) \setminus S(A)|
\]

Based on set theory, we have

\[
|S(A) \cup S(C)| = |S(A)| + |S(C)| - |S(A) \cap S(C)| + |S(A) \setminus S(C)| + |S(C) \setminus S(A)|
\]

Based on (4), (9), (10), we have

\[
r(A, C) = \frac{|S(A) \cap S(C)|}{|S(A) \cup S(C)|} \geq \frac{|S(B) \cap S(C)| - |S(B) \setminus S(A)|}{|S(A)| + |S(C)| - |(S(B) \cap S(C)) - (S(B) \setminus S(A))|}
\]
Theorem 3 gives the upper bound of $r(A, C)$ if $r(A, B)$ and $r(B, C)$ are given.

**Theorem 3** For three content blocks $A$, $B$, $C$, if $r(A, B)$ and $r(B, C)$ are given, we have

$$r(A, C) \leq \frac{|S(B) \cap S(C)| + |S(A) \setminus S(B)|}{|S(A) \cup S(C)| - |S(B) \cap S(C)| + |S(A) \setminus S(B)|} \quad (11)$$

**Proof:**

Based on set theory, we have

$$|S(A) \cap S(C)| = |S(A) \cap S(B) \cap S(C)| + |S(A) \setminus S(B)| \cap S(C) |$$

$$\leq |S(B) \cap S(C)| + |S(A) \setminus S(B) \cap S(C)|$$

$$\leq |S(B) \cap S(C)| + |S(A) \setminus S(B)|$$

Therefore,

$$r(A, C) = \frac{|S(A) \cap S(C)|}{|S(A) \cup S(C)|} = \frac{|S(A) \cap S(C)|}{|S(A) \cup S(C)| - |S(B) \cap S(C)|} \leq \frac{|S(B) \cap S(C)| + |S(A) \setminus S(B)|}{|S(A)| + |S(C)| - |S(B) \cap S(C)| + |S(A) \setminus S(B)|}$$

Based on set theory, we have

$$|S(B) \setminus S(A)| = |S(B)| - |S(A) \cap S(B)| \quad (12)$$

Therefore, given three content blocks $A$, $B$, $C$, if we know the size of each content block, as well as $r(A, B)$ and $r(B, C)$, based on (2), (3), (11), and (12), we can calculate the lower bound and upper bound of $r(A, C)$. The lower bound can be used to decide whether $A$ and $C$ are duplicated, and the upper bound can be used to decide whether $A$ and $C$ not duplicated.

**Example 1** Given three Web content blocks $b_1$, $b_2$, and $b_3$, $|S(b_1)| = |S(b_2)| = |S(b_3)| = 800$, $r(b_1, b_2) = 0.88$, and $r(b_2, b_3) = 0.91$. Based on (2), we have,

$$|S(b_1) \cap S(b_2)| = \frac{0.88 \times (800 + 800)}{1 + 0.88} = 749$$

$$|S(b_2) \cap S(b_3)| = \frac{0.91 \times (800 + 800)}{1 + 0.91} = 762$$

and based on (12) we have

$$|S(b_2) \setminus S(b_1)| = 800 - 749 = 51.$$ 

According to (3) we have

$$r(b_1, b_3) \geq \frac{762 - 51}{800 + 800 - (762 - 51)} = 0.80.$$ 

Here we can find that the resemblance of $b_1$ and $b_3$ is pretty high.

From Example 1 we can find that if blocks $b_1$ and $b_2$ are similar to each other, and $b_2$ and $b_3$ are also similar to each other, then with certain confidence we can decide that $b_1$ and $b_3$ are also similar to each other. That is, resemblance in between different block pairs has transitivity.

To test resemblance transitivity, we have randomly selected 110 block triples $(b_1, b_2, b_3)$. For each triple we have $r(b_1, b_2) \geq 0.8$, and $r(b_2, b_3) \geq 0.8$. We then check the corresponding $r(b_1, b_3)$ in that triple, Table 3 shows the experiment result. From Table 3 we can find that more than 90% of the testing block triples have $r(b_1, b_3) \geq 0.8$. This further verifies the transitivity of block resemblance.

<table>
<thead>
<tr>
<th>Resemblance of $b_1$ and $b_3$</th>
<th>Number of triples</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\geq 0.9$</td>
<td>35</td>
</tr>
<tr>
<td>$\geq 0.8$</td>
<td>40</td>
</tr>
<tr>
<td>$&lt; 0.8$</td>
<td>10</td>
</tr>
</tbody>
</table>

*Table 3. Experiment result on block resemblance transitivity*

**Example 2** Given three Web content blocks $b_1$, $b_2$, and $b_3$, $|S(b_1)| = |S(b_2)| = |S(b_3)| = 800$, $r(b_1, b_2) = 0.6$, $r(b_2, b_3) = 0.20$. Based on (2), we have,

$$|S(b_1) \cap S(b_3)| = \frac{0.6 \times (800 + 800)}{1 + 0.6} = 600$$

$$|S(b_2) \cap S(b_3)| = \frac{0.2 \times (800 + 800)}{1 + 0.2} = 267$$

Based on (12) we have

$$|S(b_2) \setminus S(b_1)| = 800 - 600 = 200.$$ 

According to (11) we have

$$r(b_1, b_3) \leq \frac{267 + 200}{800 + 800 - (267 + 200)} = 0.41.$$ 

Here we can find that the resemblance of $b_1$ and $b_3$ is lower than the pre-defined threshold (0.8).

From example 2 we can find that even though blocks $b_1$ and $b_2$ are not similar enough (lower than the threshold 0.8) to be duplicated pair, as long as they bear some resemblance (0.6 in Example 2), and $b_2$ and $b_3$ do not bear enough resemblance (0.2 in Example 2) to each other, then we can conclude with certain confidence that $b_1$ and $b_3$ are not duplicated blocks. Note that here the threshold value plays an important role for the effectiveness of resemblance transitivity. In the future we expect to perform more study on this aspect.

3) Block Duplication Detection

Based on the above analysis, we provide a computationally-efficient strategy to calculate block-pair similarity in a block collection. The basic ideas are as follows.

(a) Given three blocks $A$, $B$, and $C$, if $A$ resembles $B$, and $B$ resembles $C$, according to the analysis in above section, we know that $A$ and $C$ also resemble each other to some extent. Therefore, to decide whether $A$, $B$, and $C$ resemble each other, instead of comparing all three possible pairs, i.e., $(A, B), (A, C),$ and $(B, C)$, we need only compare $(A, B)$ and $(B, C)$. The resemblance of $(A, C)$ can be inferred from the relationship of $(A, B)$ and $(B, C)$ to some extent. The more resemblance $(A, B)$ and $(B, C)$ have, the more possible $A$ and $C$ also resemble to each other.

Similarly, if we have detected that $A_1, A_2, \ldots, A_k$ all resemble $A_0$, then for a new block $B$, we need only compare it with $A_0$ and use its resemblance with $A_0$ to infer its resemblance with $A_1, A_2, \ldots, A_k$. In this way we can greatly decrease the total number of comparisons.

(b) On the other hand, given three blocks $A$, $B$, and $C$, if $A$ resembles $B$ to some extent, and $B$ does not resemble $C$, according to the analysis in above section, we know that $A$ and $C$ do not resemble each other. Here the non-resemblance of $A$ and $C$ can be inferred from the relationship of $(A, B)$ and $(B, C)$. The more resemblance $A$ and $B$ have, and the less
resemblance $B$ and $C$ have, the more possible $A$ and $C$ are not duplicated blocks.

Similarly, if we have detected that $A_1, A_2, \ldots, A_k$ all resemble $A_0$ to some extent, then for a new block $B$, if we detect that it is not duplicated with $A_0$, we can infer that it is also not duplicated with $A_1, A_2, \ldots, A_k$ without further comparison. In this way we can further decrease the total number of comparisons.

Based on the above analysis we can build our block duplication detection system as shown in Fig. 4. As discussed in section III, center block plays a key role for Web information retrieval. Therefore in our approach we focus on center block duplication detection. Duplication detection of other blocks is similar to center block duplication detection.

Below is a detailed description of our block duplication detection algorithm.

**Algorithm 1: center block duplication detection**

**Input:** Web page collection ($P$)

**Output:** Duplicated center block collection ($D$) and unique center block collection ($U$)

(a) We perform FOM based block analysis on each page in collection $P$ and generate center block collection ($C$) which contains the center blocks of the pages in $P$;

(c) We create two empty collections, Duplicated Block Collection ($D$) and Unique Block Collection ($U$). For each block $b$ in collection $U$, we maintain a list of other blocks in $U$ that bear partial resemblance with $b$ (i.e., their resemblances with $b$ are larger than a pre-defined threshold, 0.3, in our experiments);

(d) We sort the blocks in collection $U$ based on the descending order of the number of blocks in the partial resemblance list of each block and get a list of blocks, $u_1, u_2, \ldots, u_k$. For each center block $c$ in $C$, it is compared with $u_1, u_2, \ldots, u_k$ sequentially. If $r(c, u_j)$ is larger than a threshold (0.8 in our experiments), it is placed in $D$ without further comparison. If $r(c, u_j)$ is smaller than a threshold (0.1 in our experiments), it will not be compared with any block in the partial resemblance list of $b$. If the resemblances of all the blocks in $U$ with $c$ are smaller than 0.8, we put $c$ in $U$, and update the partial resemblance lists of all related blocks in $U$ accordingly.

(e) After all the blocks in $C$ are examined, we output $U$ as the collection of blocks without duplication, and $D$ as the duplicated collection.

4) **Center block based indexing and query**

After detecting all the unique center blocks of Web pages, we further make use of the inverted page index technology to build index for all the blocks in collection $U$. For each block stored in collection $U$, we also maintain a link to its corresponding Web page.

At query stage, a user query is compared with the center blocks instead of original documents in the index we have built. Ranking is performed based on the similarity of query and center blocks. Finally, we get an ordered list of relevant center blocks, and we will return the corresponding Web pages to the user in the same order. Fig. 5 shows the system framework for center block based indexing and query.

**IV. Performance Evaluation**

All the experiments were performed in a Dell PowerEdge1800 server with two Xeon 3.0G Hz CPU and 4.0 GB main memory. To test and compare the duplication property of different websites, we collect Web pages via different portal websites such as Open Project Directory (http://dmoz.org/) and Yahoo! (www.yahoo.com). In this way we wish to get a miniature of the Web.

A. **Block level duplication detection**

First we make use of FOM technology as described in previous section to detect center block in a Web page. To search near duplicates among center blocks, we first convert each text node to lowercase, and remove the stop words. We then stem the words in the text nodes. Finally we detect center block duplication as described in previous section.

**Experiment 1** detects block level duplication in a collection of 10,026 html pages. These pages came from crawling the Open Project Directory (dmoz.org). Theoretically this includes more than 50 million pairs of center blocks. In our system only 1,191,385 pairs are compared (see Table 4). This is about 2.4% of the total possible block pairs, which is much smaller than the traditional $O(N^2)$ approach.
The result of experiment 1 is shown in Table 4. From Table 4 we can find that around 7.8% of the center blocks are duplicates. This means indexing at center block level without duplication alone will effectively reduce the index size by 7.8%, which is promising for improving Web search engine performance.

<table>
<thead>
<tr>
<th>Count</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total pages</td>
<td>10,026</td>
</tr>
<tr>
<td>Unique center blocks detected</td>
<td>9,245</td>
</tr>
<tr>
<td>Duplicated center blocks detected</td>
<td>781</td>
</tr>
<tr>
<td>Comparisons made</td>
<td>1,191,385</td>
</tr>
<tr>
<td></td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>92.2%</td>
</tr>
<tr>
<td></td>
<td>7.8%</td>
</tr>
<tr>
<td></td>
<td>2.4% (percentage to total possible comparisons)</td>
</tr>
</tbody>
</table>

Table 4 Block level duplication detection across different Sites

Experiment 2 detects block level duplication in a document collection of 3,554 pages from www.yahoo.com. Theoretically this includes about 6 million pairs of center blocks. In our system only 581,072 pairs are compared (see Table 5). This is about 9.2% of the total possible block pairs, which is still much smaller than the traditional O(N^2) approach.

<table>
<thead>
<tr>
<th>Count</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total pages</td>
<td>3,554</td>
</tr>
<tr>
<td>Unique center blocks detected</td>
<td>3,534</td>
</tr>
<tr>
<td>Duplicated center blocks detected</td>
<td>20</td>
</tr>
<tr>
<td>Comparisons made</td>
<td>581,072</td>
</tr>
<tr>
<td></td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>99.4%</td>
</tr>
<tr>
<td></td>
<td>0.6%</td>
</tr>
<tr>
<td></td>
<td>9.2% (percentage to total possible comparisons)</td>
</tr>
</tbody>
</table>

Table 5 Block level duplication detection inside www.yahoo.com website

One interesting phenomenon here is that of all these 3,554 pages we have tested, only 20 are duplicates of others at center block level. This is less than 1% of total pages, much less comparing to that of Experiment 1. The major reason is that here we perform a breadth-first search in www.yahoo.com. Since the collection size we tested is relatively small (only 3,554) we cannot go too deep in Yahoo’s website (at most 3 levels in our experiment). At the higher level, most of the links have been carefully scrutinized by human to avoid duplication. Therefore, it is quite normal that we do not find too many duplicates. Generally speaking, inside the same website, block level duplication may happen more in other regions such as Header, Footer, Left, and Right than in Center region. The reason is that those other regions are more likely machine generated based on a common template while Center region aims at providing unique content for users.

The relative percentage of comparisons to total possible comparisons made in Experiment 2 (9.2%) is much more than that of Experiment 1 (2.4%). The reason is that in Experiment 1 there are much more duplicates and near-duplicates. Therefore, for a testing block in experiment 1, when compared with blocks in collection U, duplication or near-duplication is more likely to happen (or happen earlier) and therefore stops the comparison of the block with the rest in collection U or eliminate a lot of unnecessary comparisons based on Theorem 3.

Experiment 3 extends block level duplication detection to page level by incorporating all five regions of a Web page. Experiments 3 use the same document collection as in Experiment 1. In our system only 1,231,238 pairs are compared (see Table 6). This is about 2.4% of the total possible block pairs, roughly the same as in Experiment 1.

<table>
<thead>
<tr>
<th>Count</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total pages</td>
<td>10,026</td>
</tr>
<tr>
<td>Unique center blocks detected</td>
<td>9,111</td>
</tr>
<tr>
<td>Duplicated center blocks detected</td>
<td>915</td>
</tr>
<tr>
<td>Comparisons made</td>
<td>1,231,238</td>
</tr>
<tr>
<td></td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>90.9%</td>
</tr>
<tr>
<td></td>
<td>9.1%</td>
</tr>
<tr>
<td></td>
<td>2.4% (percentage to total possible comparisons)</td>
</tr>
</tbody>
</table>

Table 6 Page level duplication detection across different Sites

The result of Experiment 3 is shown in Table 6. From Table 6 we find that around 9.1% of the pages are detected as duplicated pages. This is a little more than that of Experiment 1 (7.8%). After carefully examined the extra duplicated pages detected in this experiment, we discover that the extra pages actually have different main content at central blocks, i.e., their central blocks are unique. Unfortunately they do have high similarity in other regions with some other pages (e.g., common header, footer, navigation bar, etc), which leads to a higher page resemblance than center block resemblance. This also indicates that other blocks may “pollute” the search space and lead to irrelevant result for a user query in a search engine. Therefore, detecting duplication at center block leads to more accurate result.

The percentage of duplication in Experiment 3 (9.1%) is a little less than estimations or experiments made by previous researchers [11][12], we attribute this difference to the relatively smaller collection size and our breadth-first crawling strategy as explained in Experiment 2.

B. Center block based indexing performance

Experiment 4 After duplication detection in the document collection P as described in Experiment 1, we create a new collection U of center blocks without duplication. We then index each collection separately using inverted file index technology. Since our major purpose here is to compare the effect of center block based index with normal page based index, instead of using complicated bitwise inverted page indexing [29], we simply use integer data type to represent each posting in inverted list.

Table 7 shows the indexing time and index size of each collection. From Table 7 we can find that indexing on the new collection of center block without duplication decreases index construction time and index size by 71%, and 35%,
respectively. Therefore, indexing on center block without duplication can effectively improve the efficiency of indexer for Web search engines.

We have performed multiple queries based on the two indices we have built. And the returned results are roughly the same except that in the whole page collection we do encounter duplication while the other one has no duplication.

<table>
<thead>
<tr>
<th>Collection</th>
<th>Construction time (seconds)</th>
<th>Index size (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P (Page collection)</td>
<td>182</td>
<td>623270</td>
</tr>
<tr>
<td>U (Center block</td>
<td>52</td>
<td>402343</td>
</tr>
<tr>
<td>collection)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7 Indexing statistics

V. Conclusions and Future Work

In this paper we extend the concept of Web duplication from site and page level to block level. By making use of resemblance transitivity, we effectively reduce the computational complexity of block level duplication detection. By eliminating duplicated center blocks and indexing unique center blocks instead of whole pages, we can effectively reduce index size and improve the performance of Web search engines.

In our current experiments we use the value of 0.8 as a resemblance threshold to detect block duplication. Generally speaking, the smaller the threshold is, the better compression we can achieve for decreasing the index size, but at the cost of possible mis-detection of duplicated blocks. For the several datasets that we have tested, the value 0.8 works quite well in that it can effectively eliminate duplicated blocks without mistakenly remove non-duplicated ones. In the future we will perform further study on the impact of this threshold on the experiment results and we expect to design an adaptive algorithm for optimal threshold decision.

Besides, in our current approach we use block resemblance definition mainly based on shared terms to decide whether two blocks are duplicated. Similarity in between two objects is an important issue in many research areas such as data mining, geographic data matching, geometric pattern recognition, etc. In the future we expect to investigate the effectiveness of different resemblance definitions for duplication detection.

Furthermore, in the future we will apply our approach on block level duplication detection to other areas such as detecting spam pages (pages unrelated to user query that are ranked high by popular search engines), copyright infringement (using copyrightable work without the permission of the owner) and plagiarism (using the words or ideas of others and presents them as ones ideas).

Acknowledgment

This work was supported in part by a grant from The City University of New York PSC-CUNY Research Award Program (PSCREG-36-1054, PSCREG-38-892), and NSF fellowship grant (2005020870).

References


Author Biographies

Tyrone Cadenhead received his Master of Art degree (Computer Science) from Queens College in 2006. He has also received his Master of Science degree (Management Information Systems) in 1999 and Bachelor of Science degree (Computer Science) in 1995 from Univ. of West Indies, Jamaica. His research interests include Web information processing and data mining.

Jinlin Chen received his PhD (Automatic Control) in 1999, Bachelor of Engineering and Bachelor of Economics in 1994, all from Tsinghua University, Beijing, China. He is a faculty at Computer Science Dept, Queens College, the City Univ. of New York. His research interests include web information modeling and processing, information retrieval, and data mining. He is a member of IEEE.

Terry Cook is a Ph.D. student at the Graduate Center, City University of New York. He received a Masters of Science degree and a Bachelor of Science degree from Lehman College, City University of New York in 2005 and 2004 respectively. His field of research is information retrieval, in particular analyzing the effectiveness of various compression techniques for inverted files. He is also interested in determining optimal models for enhancing search relevancy. He was inducted into the International Golden Key society in 2003.