DYNAMIC HIERARCHICAL DICTIONARY DESIGN FOR MULTI-PAGE BINARY DOCUMENT IMAGE COMPRESSION**

Yandong Guo[⋆]*, Dejan Depalov*[†]*, Peter Bauer*[†]*, Brent Bradburn*[†]*, Jan P. Allebach*[⋆]*, and Charles A. Bouman*[⋆]

[⋆] School of Electrical and Computer Engineering, Purdue University,
West Lafayette, IN 47907-2035, U.S.A.;
[†] Hewlett-Packard Co., Boise, ID 83714, U.S.A.

ABSTRACT

The JBIG2 standard is widely used for binary document image compression primarily because it achieves much higher compression ratios than conventional facsimile encoding standards. In this paper, we propose a dynamic hierarchical dictionary design method (DH) for multi-page binary document image compression with JBIG2. Our DH method outperforms other methods for multi-page compression by utilizing the information redundancy among pages with the following technologies. First, we build a hierarchical dictionary to keep more information per page for future usage. Second, we dynamically update the dictionary in memory to keep as much information as possible subject to the memory constraint. Third, we incorporate our conditional entropy estimation algorithm to utilize the saved information more effectively. Our experimental results show that the compression ratio improvement by our DH method is about 15% compared to the best existing multi-page encoding method.

Index Terms— Binary document image compression, multi-page, JBIG2, dynamic hierarchical dictionary design, conditional entropy estimation

1. INTRODUCTION

Binary document image compression is widely used for document scanning, storage, and transmission. Very often, people compress multi-page binary document images. Since these images usually come from consecutive pages of the same document source, there is typically information redundancy among pages. In this paper, we focus on how to utilize this information redundancy to improve the compression ratio for multi-page binary document image compression.

The JBIG2 compression standard, developed by the Joint Bi-level Image Experts Group [1], is considered to be the best existing standard for binary image compression because it can achieve much higher compression ratios than the previous methods, such as T.4, T.6, and T.82 [2, 3, 4, 5, 6]. The high compression ratio of JBIG2 compression comes from its dictionary-symbol encoding procedure. A typical JBIG2 encoder works by first separating the document into connected components, or symbols. Next it creates a dictionary by encoding a subset of symbols from the image, and finally it encodes all the remaining symbols using the dictionary entries as a reference [7, 8, 9, 10, 11].

There are two straightforward methods to compress multi-page document images with the JBIG2 encoder. The first method creates an independent and static dictionary (IS) for each of the pages. Since the IS method does not utilize the information redundancy among pages, it generally results in the highest bit rate. Alternatively, the global-dictionary approach builds a single dictionary for the entire document, so it can generally achieve the lowest bit rate. However, the global-dictionary approach is not practical, since it requires an enormous memory to buffer the entire multi-page document before any compression can occur.

In fact, practical JBIG2 encoders usually load only one page (sometimes even part of one page) to compress, and do not load the next page until the compression is finished [12]. Therefore, the information redundancy among pages is utilized by sharing dictionaries among pages, which is supported by the standard [1]. The first and widely cited practical algorithm, called LDM, was proposed by Ye and Cosman in [13, 14]. The LDM algorithm forms a dictionary for each of the pages by starting with the dictionary from the previous page, adding new dictionary entries needed for the current page, and expunging the dictionary entries (from the previous page) not used for the current page. The LDM algorithm improves the compression ratio compared to the IS method, and is memory efficient. Figuera, Yi, and Bouman also proposed a multi-page encoding algorithm referred to as "dynamic symbol caching (DSC)" in [15, 16]. The DSC algorithm is claimed to have higher compression ratio compared to LDM, because DSC only discards dictionary entries when there is no space available in the memory. The discarded dictionary entries are the least recently used ones.

In our paper, we introduce a dynamic hierarchical (DH) dictionary design method for efficient compression of multi-page documents. The encoding efficiency improvement of the

**Research supported by the Hewlett-Packard Company.

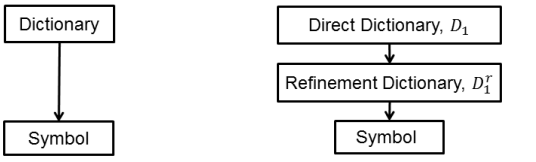
document symbols requires better utilization of the information redundancy, which can be accomplished by constructing and sharing large dictionaries among pages. However, the large dictionaries also require a large number of bits to be encoded, and a great deal of memory to retain. Our DH method addresses these competing demands by employing both a hierarchical structure that efficiently encodes large dictionaries, and a dynamic strategy that retains the more information subject to the memory constraint. In addition, we use the conditional entropy estimation technique of [17, 18] to measure the information redundancy between two dictionary entries. This results in further improvements to the encoding efficiency of the dictionary design and symbol encoding processes. Our experimental results show that the DH method improves the compression ratio by approximately 15% relative to the best existing method.

The rest of the paper is organized as follows. We present the dynamic hierarchical dictionary design method in details in Sec. 2. The experimental results are shown in Sec. 3 and the conclusion is in Sec. 4.

2. DYNAMIC HIERARCHICAL DESIGN

In this section, we introduce the dynamic update approach for encoding of hierarchical dictionaries. Sec. 2.1 explains how the first page is encoded. Sec. 2.2 and Sec. 2.3 then explain the dynamic updating strategy for successive pages.

2.1. Encoding of First Page



(a) Single dictionary (b) Hierarchical dictionary

Fig. 1. Single dictionary and hierarchical dictionary structure for the first page. The hierarchical dictionary structure efficiently encodes a large refinement dictionary, which more efficiently encodes the document symbols.

The Figs. 1 (b) illustrates how the hierarchical dictionary structure is used to encode the first page of the document. First, we encode a smaller dictionary, called the direct dictionary, denoted as D_1. This direct dictionary is encoded using the direct coding mode of the JBIG2 standard [1]. Next, we use the refinement coding mode of the JBIG standard [1] to encode a much larger refinement dictionary, denoted by D_1^r. The refinement dictionary is compressed very efficiently because refinement coding uses a reference symbol from the direct dictionary to encode each new symbol in the refinement dictionary. Finally, we encode all the symbols in the first document page by using the refinement dictionary as a reference.

In order to construct the hierarchical dictionary of Figs. 1b), we use a bottom-up procedure. First, we extract all the distinct symbols on the first page. Then, for each of the distinct symbol, we construct one refinement dictionary entry. Next, we group the similar refinement dictionary entries into clusters, and create one representative for each of the clusters. These representatives are the dictionary entries which form the direct dictionary. In order to perform the clustering, we use the conditional entropy estimation-based dictionary indexing and design algorithm (CEE-DI) of [17].

2.2. Encoding of Successive Pages

The Fig. 2 illustrates the dynamic hierarchical dictionary construction of successive pages of the document. For successive pages, we introduce the stored dictionary, denoted as D_k^s, for the k^{th} page ($k \neq 1$). When there is no memory constraint, the stored dictionary D_k^s is the union of all dictionaries from the previous pages (of which indices $< k$). The case with memory constraint will be discussed in the next subsection.

Once again, the refinement dictionary D_k^r is formed by every unique symbols in the k^{th} page. For each of the given refinement dictionary entries, we try to find a good match in the stored dictionary, D_k^s, to encode it efficiently. Typically, most of the entries in the refinement dictionary will have a good match in the stored dictionary. Thus, in this case, the refinement dictionary is encoded very efficiently.

However, there will typically be some refinement dictionary entries, that do not have a good match in the stored dictionary. In order to encode these unmatched refinement dictionary entrees, we form a new direct dictionary D_k. We build this direct dictionary using the conditional entropy estimation-based dictionary indexing algorithm (CEE-I) [17].

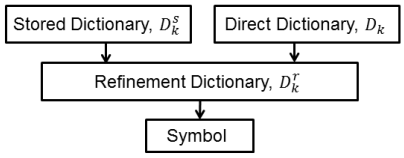


Fig. 2. Hierarchical dictionary structure for the k^{th} page. The stored dictionary D_k^s is the pruned union of all the dictionaries from previous pages. Some entries in the refinement dictionary are encoded using the stored dictionary, while the rest are encoded using the direct dictionary.

The criteria to determine whether we can find a good match for the given refinement dictionary entry in the stored dictionary is based on the conditional entropy estimation (CEE) in [17]. Let $\boldsymbol{d}_{k,j}^r$ denote the j^{th} entry in D_k^r, and $\boldsymbol{d}_{k,i}^s$ denote the i^{th} entry in D_k^s. The best match for $\boldsymbol{d}_{k,j}^r$ in D_k^s is

found by

$$\boldsymbol{d}^s_{k,\hat{i}(j)} = \underset{\boldsymbol{d}^s_{k,i} \in D^s_k}{\operatorname{argmin}} \hat{H}(\boldsymbol{d}^r_{k,j} | \boldsymbol{d}^s_{k,i}) \,, \tag{1}$$

where $\hat{H}(\boldsymbol{d}^r_{k,j}|\boldsymbol{d}^s_{k,i})$ is the estimation of the conditional entropy of $\boldsymbol{d}^r_{k,j}$ given $\boldsymbol{d}^s_{k,i}$. If the conditional entropy of $\boldsymbol{d}^r_{k,j}$ given $\boldsymbol{d}^s_{k,\hat{i}(j)}$ is smaller than the predefined threshold T_R,

$$\hat{H}(\boldsymbol{d}^r_{k,j} | \boldsymbol{d}^s_{k,\hat{i}(j)}) \leq T_R \,, \tag{2}$$

we encode $\boldsymbol{d}^r_{k,j}$ using the stored dictionary entry $\boldsymbol{d}^s_{k,\hat{i}(j)}$ as a reference. Otherwise, we do not encode $\boldsymbol{d}^r_{k,j}$ using the stored dictionary.

2.3. Dynamic updating strategy

In order to make our algorithm practical, we must ensure that the size of the dictionaries retained in the memory will not grow beyond our available memory size.

The method we used is to discard some of the stored dictionary entries whenever the memory size for all the dictionaries for the k^{th} page is larger than 1M bytes. We choose the threshold value to be 1M because the standard requires a decoder to have at least a 1M byte of storage for the dictionary [12].

The memory size for the dictionaries for the k^{th} page is the summation of the memory size for D_k, D^r_k, and D^s_k, which can be calculated using the formula in [12]. The entry to be discarded is the stored dictionary entry $\boldsymbol{d}^s_{k,\hat{m}}$ satisfying both of the following two conditions.

1. The entry $\boldsymbol{d}^s_{k,\hat{m}}$ is not referred by any entry in D^r_k.
2. The entry $\boldsymbol{d}^s_{k,\hat{m}}$ is least distinct, defined as

$$\hat{m} = \underset{m}{\operatorname{argmin}}\, d_{XOR}(\boldsymbol{d}^s_{k,m}, \boldsymbol{d}_{k,n}) \,, \tag{3}$$

where $\boldsymbol{d}_{k,n}$ is any dictionary entry different from $\boldsymbol{d}^s_{k,m}$, that belongs to D_k, D^r_k, or D^s_k. The function d_{XOR} calculates the Hamming distance between two dictionary entries.

Similar dictionary entries typically have more mutual information. Therefore, by the above strategy, we maintain as much total information as possible in the memory under the memory size constraint.

3. EXPERIMENTAL RESULTS

In this section, we compare the DH method with the best existing method DSC of [15, 16]. The DSC method in our experiments is based on the widely used weighted Hamming distance (WXOR) [19, 20] for the dissimilarity measurement between symbols and dictionary entries. For the DH method, we investigated two versions. The first one is as described in the Sec. 2, called DH-CEE, since it uses the conditional entropy estimation (CEE) as the dissimilarity measure. For the second one, we substitute the CEE dissimilarity measure with the WXOR dissimilarity measure, in order to see the benefit due only to the dynamic hierarchical dictionary design. We call this method DH-WXOR.

The test images consists of three sets of document images, *EEPaper*, *Vita*, and *I9intro*. Each set was scanned from consecutive pages of the same document at 300 dpi. The set *EEPaper* contains 9 images with 3275×2525 pixels; *Vita* contains 10 images with 3300×2550, while *I9intro* contains 6 images with 3300×2550. These test images contain mainly text, but some of them also contain line art, tables, and graphical elements, but no halftones. The JBIG2 lossless text mode was used for all experiments.

We limited the memory usage for dictionaries to be less than 1MB, as described in 2.3. Unless otherwise stated, we adjusted the parameters of all the methods so that each of the methods achieved its optimal compression ratio.

3.1. Compression ratio improvement

In this subsection, the compression ratio is investigated by compressing the three documents. The following plot in Fig. 3 shows a comparison of the DH-CEE method to the alternative methods in encoding *EEPaper*. In the plot, everything is relative to the independent and static dictionary constructed with the WXOR dissimilarity measure (IS-WXOR). Notice that, our method DH-CEE has the highest compression ratio for all the pages. For the entire document *EEPaper*, the DH-CEE improved the compression ratio by 14% compared to DSC, while for *Vita* by 15% and for *I9intro* by 17% .

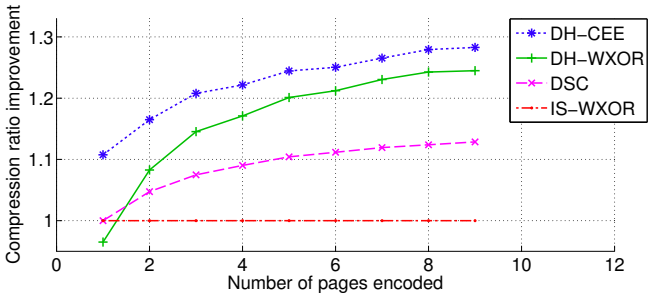
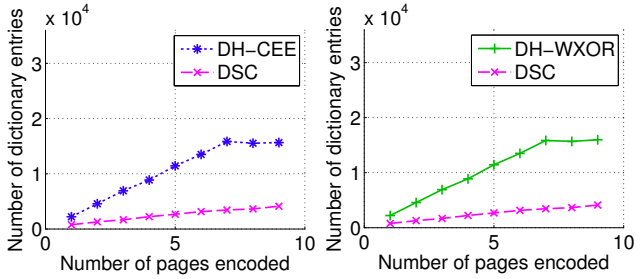


Fig. 3. Compression ratio improvements by different dictionary design methods (relative to IS-WXOR) for *EEPaper*.

The main reason for the compression ratio improvement by DH-CEE over DSC is that DH-CEE produced a much larger dictionary for each of the pages, shown in Figs. 4 (a), and the larger dictionaries can more efficiently encode the documents. Meanwhile, using DH-CEE, only a small overhead is needed to encode the large dictionary. The efficient encoding of large dictionaries is demonstrated with an example in the next subsection.



(a) DH-CEE vs. DSC (b) DH-WXOR vs. DSC

Fig. 4. The number of the dictionary entries obtained by using different dictionary design methods. The large dictionaries produced by the DH methods encode the documents efficiently. For DH-CEE and DH-WXOR, the dynamic updating controls the size of their dictionaries after the 7^{th} page due to the memory constraint.

3.2. Efficient encoding of large dictionaries

In this subsection, we show that the DH method encodes a large dictionary with a relatively little overhead. Different methods were used to create large dictionaries for the first page in *EEPaper*.

The refinement dictionary produced by the DH method is naturally large in size, because DH method creates one refinement dictionary entry for each of the distinct symbols in the page. For comparison, in the experiment in this subsection, we adjusted the parameters of DSC to enforce its single dictionary to be as large as the refinement dictionary with DH-CEE. Please notice that in the experiment in the last subsection, the dictionary size of DSC was optimized to achieve its highest compression ratio, and as shown in Fig. 4, was much smaller than that obtained by using the DH method.

As shown in Fig. 5, the bitstream filesize obtained by using DH-CEE are significantly smaller than that obtained with DSC. This is due to the hierarchical structure of DH-CEE: DH-CEE builds up the direct dictionary using CEE-ID to encode the refinement dictionary efficiently.

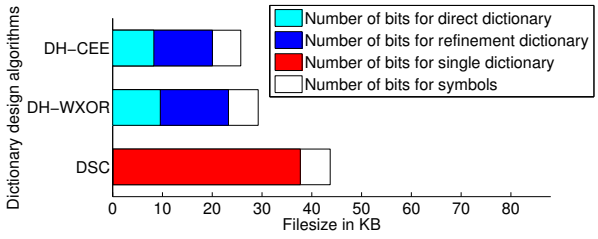


Fig. 5. Comparison of bit rate using three methods for dictionary compression. Different from the last subsection, the dictionary with DSC was enforced to be identical with the dictionary with DH-CEE. Note that DH-CEE results in the lowest bit rate in encoding the dictionary.

3.3. Conditional entropy estimation

The compression ratio improvement by DH-CEE also comes from the conditional entropy estimation (CEE). For comparison, we investigate the DH-WXOR method, which substitutes CEE with WXOR.

First, we repeated the single page experiment in Sec. 3.2 with the DH-WXOR method. The refinement dictionaries obtained by using DH-WXOR and DH-CEE are identical since they used the same method to create their refinement dictionaries. As shown in Fig. 5, the bit rate obtained by using DH-WXOR is smaller than that with DSC due to the hierarchical dictionary design. One the other hand, the bit rate with DH-WXOR is larger than that of DH-CEE. This is because CEE used in DH-CEE provides better measurement for the information redundancy between dictionary entries than WXOR in DH-WXOR [17]. And thus DH-CEE creates a better direct dictionary to encode the refinement dictionary.

Then, we repeated the multi-page experiment in Sec. 3.1 with the DH-WXOR method. As shown in Figs. 3, DH-WXOR improved the compression ratio by 11% compared to DSC. This improvement purely comes from the large dictionaries produced by the dynamic hierarchical design of DH-WXOR, shown as Figs. 4 (b). On the other hand, The compression ratio with DH-WXOR is about 4% less than that with DH-CEE. This is because, based on CEE, DH-CEE creates better direct dictionaries and selects better stored dictionary entries to encode the refinement dictionarythan DH-WXOR.

Please note that all the above experiments were conducted subject to the 1MB memory constraint. As shown in Figs. 4 (a) and (b), The dynamic updating has kept discarding stored dictionary entries since the 7^{th} page was encoded. If we release the memory constraint, no dictionary entries are discarded and extra memory is consumed. However, according to our experimental results, the extra memory usage only improved the compression ratio by less than 1%. This is because the dynamic updating minimizes the information loss caused by discarded dictionary entries by selecting the least distinct stored dictionary entries.

4. CONCLUSION

In this paper, we proposed the dynamic hierarchical (DH) dictionary design method for the multi-page binary document image compression. A typical dictionary design method for multi-page image improves the encoding efficiency by maintaining and utilizing the information from previous pages to encode the successive pages. The DH method outperforms the previously existing methods using the following technologies. First, hierarchical design allows us to maintain more information per page. Second, the dynamic updating helps us to maintain as much information as possible subject to the memory constraint. Third, the conditional entropy estimation helps us to utilize the maintained information more efficiently.

5. REFERENCES

[1] "JBIG2 Final Draft International Standard," *ISO/IEC JTC1/SC29/WG1N1545*, Dec. 1999.

[2] "Standardization of Group 3 Facsimile Apparatus for Document Transmission," *CCITT Recommend. T.4*, 1980.

[3] "Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus," *CCITT Recommend. T.6*, 1984.

[4] "Progressive Bi-level Image Compression," *CCITT Recommend. T.82*, 1993.

[5] Roy Hunter and Harry Robinson, "International digital facsimile coding standards," *Proc. of the IEEE*, vol. 68, pp. 854–867, July 1980.

[6] Ronald B. Arps and Thomas K. Truong, "Comparison of international standards for lossless still image compression," *Proc. of the IEEE*, vol. 82, pp. 889–899, 1994.

[7] Ono Fumitaka, Rucklidge William, Arps Ronald, and Constantinescu Corneliu, "JBIG2 - the ultimate bi-level image coding standard," in *Proc. of IEEE Int'l Conf. on Image Proc.*, 2000, pp. 140–143.

[8] Paul G. Howard, Faouzi Kossentini, Bo Martins, Søren Forchhammer, and William J. Rucklidge, "The emerging JBIG2 standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 8, pp. 838–848, 1998.

[9] Comeliu Constantinescu and Ronald Arps, "Fast residue coding for lossless textual image compression," in *1997 IEEE Data Compression Conf.(DCC)*, 1997, DCC '97, pp. 397–406.

[10] Paul G. Howard, "Lossless and lossy compression of text images by soft pattern matching," in *1996 IEEE Data Compression Conf.(DCC)*, 1996, pp. 210–219.

[11] Yan Ye and Pamela C. Cosman, "Dictionary design for text image compression with JBIG2," *IEEE Trans. on Image Processing*, vol. 10, pp. 818–828, 2001.

[12] "Application Profiles for Recommendation T.88: – Lossy/lossless Coding of Bi-level Images (JBIG2) for Facsimile," *ITU-T recommendation T.89*, 2001.

[13] Yan Ye and Pamela C. Cosman, "Fast and memory efficient text image compression with JBIG2," *IEEE Trans. on Image Processing*, vol. 12, pp. 944–956, 2003.

[14] Yan Ye and Pamela C. Cosman, "Fast and memory efficient JBIG2 encoder," in *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, 2001, vol. 3, pp. 1753–1756.

[15] Maribel Figuera, Jonghyon Yi, and Charles A. Bouman, "A new approach to JBIG2 binary image compression," in *Proc. SPIE 6493, Color Imaging XII: Processing, Hardcopy, and Applications*, 2007, pp. 649305–1–649305–12.

[16] Maribel Figuera, *Memory-efficient algorithms for raster document image compression*, Ph.D. thesis, Purdue University, West Lafayette, IN, USA, 2008.

[17] Yandong Guo, Dejan Depalov, Peter Bauer, Brent Bradburn, Jan P. Allebach, and Charles A. Bouman, "Binary image compression using conditional entropy-based dictionary design and indexing," in *Proc. SPIE 8652, Color Imaging XIII: Displaying, Processing, Hardcopy, and Applications*. 2013, to be published.

[18] Guangzhi Cao, Yandong Guo, and Charles A. Bouman, "High dimensional regression using the sparse matrix transform (SMT)," in *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, 2010, pp. 1870–1873.

[19] William K. Pratt, Patrice J. Capitant, Wen-Hsiung Chen, Eric R. Hamilton, and Robert H. Wallis, "Combined symbol matching facsimile data compression system," *Proc. of the IEEE*, vol. 68, pp. 786–796, 1980.

[20] Stuart J. Inglis, *Lossless Document Image Compression*, Ph.D. thesis, University of Waikato, New Zealand, 1999.