Analysis of Link Graph Compression Techniques

David Hannah, Craig Macdonald, and Iadh Ounis

Department of Computing Science,
University of Glasgow, G12 8QQ, UK
{hannahd,craigm,ounis}@dcs.gla.ac.uk

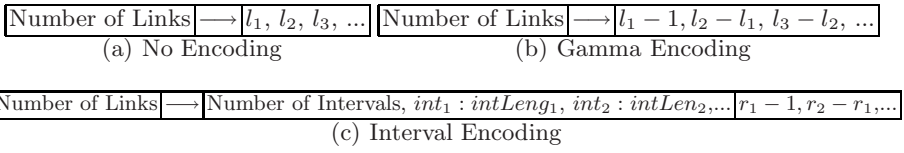
Abstract. Links between documents have been shown to be useful in various Information Retrieval (IR) tasks - for example, Google has been telling us for many years now that the PageRank authority measure is at the heart of its relevance calculations. To use such link analysis techniques in a search engine, special tools are required to store the link matrix of the collection of documents, due to the high number of links typically involved. This work is concerned with the application of compression to the link graph. We compare several techniques of compressing link graphs, and conclude on speed and space metrics, using various standard IR test collections.

1 Introduction

The retrieval performance of many Information Retrieval (IR) systems can be benefited by the application of techniques based on the structure of links between documents, such as PageRank [1] (famously applied by Google), and Hypertext Induced Topic Selection (HITS) [2]. To calculate such link analysis techniques, it is necessary to have a matrix of the links between all documents in the collection. However, due to the large nature of such a matrix, it would be impossible to work with it wholly stored in memory. As such, a disk-based structure is required, known as a link database. In particular, a link database stores, for each document, a list of the incoming (or outgoing) links to (from) the documents. In a typical collection of Web documents, there is typically an order of magnitude more links than documents, and the number of incoming links to a document follows a power-law distribution. To quickly and easily compute the various link graph-based query independent features, it is necessary that the Web search engine has timely and efficient access to the link graph.

This paper investigates the use of various state-of-the-art compression techniques, and how they can be applied to the compression of a link graph. We experiment with six recent large Web IR test collections, such as those used in TREC and the very large UK-2006 collection, and conclude on the most efficient representation for the link graph. While these compression techniques have been proposed in literature, they have never been studied extensively in terms of both time and space efficiency, and over as many test collections, of various age, size and domain, and using exactly the same experimental setting.

C. Macdonald et al. (Eds.): ECIR 2008, LNCS 4956, pp. 596–601, 2008.



(a) No Encoding (b) Gamma Encoding

(c) Interval Encoding

Fig. 1. Encoding techniques applied for compressing the link database

2 Link Graph Compression

For the state-of-the-art link graph compression techniques, we base our work on that of Boldi & Vigna [3], who detail various compression techniques for link graphs. Firstly, it is assumed that all documents in the collection have a numerical integer document identifier (document id). We then store for each document, the document ids of each of its incoming links, which we denote inlinks. Moreover, we can without loss, describe the transpose of the inlinks matrix, which we denote outlinks. We experiment with three techniques for compression:

No Encoding: (Fig. 1(a)) A vector that contains the number of links of each document together with pointers to the offset in a second file of the document ids for the links to (from) that document. Links are encoded using 32 bit integers.

Gamma Encoding: (Fig. 1(b)) Again, a vector containing the number of links for each document and a pointer into another file containing the links. However, the links are stored as the differences between the doc id of each link and the previous, written using 'Elias gamma encoding'. To gamma encode a number, it is first written in binary, then a number of zeros (the number of bits required to write the number minus one) is prepended to the number [5]. It follows that each link can take a variable number of bits to encode[1].

Interval Encoding: (Fig. 1(c)) This technique is similar to Gamma Encoding except that it encodes intervals of links. This is based on the intuition that if the documents ids in a collection are stored in lexicographical order of URL, then there will be common 'runs' of links to documents with adjacent document ids. For this compression style, the number of intervals are stored as a gamma encoded integer. Then each intervals of links is stored as the left extreme and the length of the run - again using gamma encoding. Finally, the extra links which are not consecutive are stored using gamma encoding as before. Intervals of less than L_{min} are not encoded.

It is of note that in [3], Boldi & Vigna describe a further compression technique (Reference Encoding), whereby the links for a document are encoded by stating how much it has in common with the links of the previous few documents, iteratively applied. For reasons of brevity, we leave this technique as future work.

[1] In contrast to [3], we encode the first link as document id + 1, to ensure uniformity with existing compression used Terrier, the platform on which this work is performed. However this change does not affect any experimental conclusions.

3 Experimental Design

To analyse the effectiveness of the various compression techniques, we use six different Web IR test collections, related to different domains and timescales. In particular, the collections we experiment with are: two older TREC Web test collections WT2G and WT10G, which are small-scale general Web crawls from early 1997; .GOV and .GOV2 are more recent TREC Web test collections, both of which are crawls of the `.gov` domain from 2002 and 2003 respectively - .GOV2 being the largest TREC collection at 25M documents; the TREC CERC collection which is a crawl of the CSIRO website from early 2007; and finally the UK-2006 collection is a large crawl of the `.uk` domain from 2006[2].

For our experiments, we apply Web IR techniques deployed in the Terrier platform [6]. To assess the efficiency of the link database compression methods, we record various metrics for each collection: Firstly, we record the time taken to build the compressed copies of the incoming and outgoing link graphs; Secondly, we record the time taken to compute the PageRank prior using the link database. This is motivated by the fact that the PageRank computation is a realistic application for a link database. However as the PageRank calculation can take many iterations to converge, we normalise the times by the number of iterations to account for the different sizes of collections; Lastly, we record the space required to store the link graph, in terms of the mean number of bits of space required per link. From these metrics, we can conclude in terms of the time to write and read each of the link database compression techniques, as well as their space requirements. Note that we vary the parameter *Lmin* of interval encoding.

4 Results and Analysis

Table 1 presents the compression level achieved for the inlinks and outlinks tables of the link databases, in terms of mean number of bits per links - the lower the number of bits required per link, the better compression achieved. Observe that, as expected, the No Encoding technique has a stable usage of 32 bits per link.

Applying Gamma Encoding on the same link graphs produce a markedly better level of compression (as low as 4.07 bits per link for the inlinks of the UK-2006 collection, which is similar to that reported by Boldi & Vigna in [4] for their smaller sample of `.uk`). On the unsorted collections, Interval Encoding has comparable but not as good compression as Gamma Encoding. Increasing the *Lmin* parameter generally improves the compression of the Interval Encoding. Interestingly, similar to that reported by [3], inlinks compresses better than outlinks for most collections, except the domain specific .GOV and .GOV2. On comparing compression techniques across collections, we note that, for inlinks, the collections with the highest number of links per document (i.e. UK-2006 and CERC) exhibit the highest compression, while the older WT2G and WT10G are

[2] More information about obtaining the UK-2006 collection is found at `http://www.yr-bcn.es/webspam/datasets/`

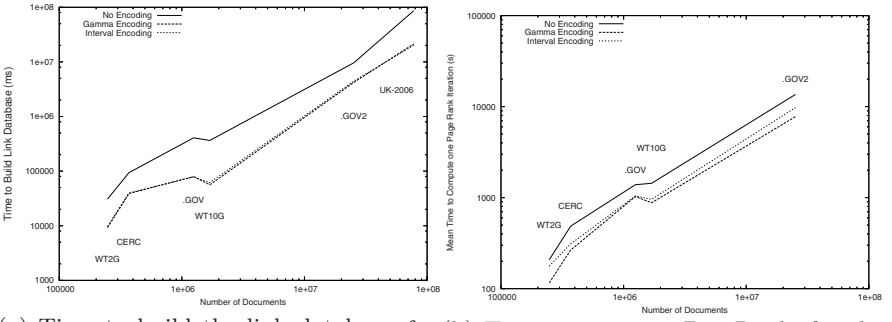
Table 1. Comparative compression (bits per link) on Web IR collections. Sorted denotes when the document ids are ordered lexicographically by URL. NB: UK-2006 and CERC collections are initially numbered this way.

Links		Unsorted					Sorted				
			Interval					Interval			
Lmin =	No	Gamma	2	3	4	5	Gamma	2	3	4	5
WT2G (247,491 Docs, 1,166,146 Links)											
out	32	**8.95**	11.87	10.69	10.16	9.91	10.32	13.22	11.48	10.96	10.76
in	32	**8.75**	9.70	9.37	9.21	10.38	8.28	10.04	9.18	8.85	8.68
CERC (370,715 Docs, 4,577,312 Links)											
out	32	-	-	-	-	-	**11.67**	14.53	13.30	12.65	12.33
in	32	-	-	-	-	-	5.26	6.08	5.21	6.06	**4.99**
.GOV (1,247,753 Docs, 11,110,989 Links)											
out	32	**18.68**	19.97	19.19	18.97	18.91	7.87	9.76	8.96	8.65	8.45
in	32	**22.68**	22.85	22.89	22.89	22.89	11.78	15.67	13.32	12.61	12.34
WT10G (1,692,096 Docs, 8,063,026 Links)											
out	32	**14.28**	15.51	14.74	14.60	14.55	11.75	15.34	13.33	12.67	12.32
in	32	**13.86**	15.20	14.36	14.22	14.16	10.05	12.09	11.22	10.88	10.58
.GOV2 (25,205,179 Docs, 261,937,150 Links)											
out	32	**20.72**	21.51	20.09	20.83	20.81	21.16	23.51	22.04	21.64	21.47
in	32	**30.87**	31.09	30.90	30.94	30.93	35.14	35.38	35.24	35.22	35.22
UK-2006 (77,741,020 Docs, 2,951,370,103 Links)											
out	32	-	-	-	-	-	**9.36**	12.53	10.61	10.06	9.79
in	32	-	-	-	-	-	4.07	5.13	4.32	4.07	**3.92**

generally worse. It is also of note that in the UK-2006 and CERC collection, the document ids are sorted lexicographically by URL, as recommended in [3], and by increasing *Lmin* to 5 on these collections, Interval Encoding can achieve higher compression than Gamma encoding. To assess the best compression achievable for the other unsorted Web test collections, we renumber the documents to match the lexicographical order of the URLs, then rebuild the link databases.

On analysing the compression between the unsorted and sorted collections, we note that sorting increases the compression achieved for the WT10G and .GOV collection, as well as for the inlinks of the WT2G collection. For the .GOV2 collection, the compression level decreases, and for the inlinks, both Gamma Encoding and Interval Encoding result is less effective compression than the No Encoding technique. We suggest this is due to a combination of low overall linkage combined with high document ids.

Figure 2(a) plots the build time of the three forms of link database compression across the 6 collections applied. Moreover, Figure 2(b) plots the mean time to perform one iteration of PageRank calculation (Note that due to computational reasons, the PageRank for UK-2006 collection was not computed.). From the figures, we can see that the No Encoding technique takes the longest to write and read, even though this is a simpler technique compared to the Gamma and Interval encoding techniques. We believe that this is due to the markedly



(a) Time to build the link database, for the various tested collections.

(b) Time to compute PageRank, for the various tested collections.

Fig. 2. Timing plots for writing and reading the various link databases (natural collection ordering)

higher number of disk operations required by this technique. Noticeably, the speeds of the Gamma and Interval encoding techniques are equivalent for both read and write operations, reflecting the very similar compression they achieve. Overall, the Gamma and Interval encodings are approximately 3-4 times faster to compress, and 1.5 times faster to decompress.

5 Conclusions

In this work, we thoroughly analysed three techniques for compressing the link graph of six different samples of the Web, of various size. We found that the simple integer (No Encoding) technique suffered from excessive Input/Output overheads. The Gamma Encoding gave best overall compression, and was among the fastest at reading and writing. The Interval Encoding technique does exhibit the high compression promised in [4], but requires an appropriate setting of the *Lmin* parameter. Overall, we conclude that the Gamma Encoding technique, similar to that already used by Terrier for direct and inverted file compression [6], should be deployed for link graph compression.

The large-scale analysis in this work is important as while more effective compression of the link graph may be obtained by the suitable ordering of the document ids in a collection, this ordering may not be compatible with other document id orderings applied by the underlying search engine - for example, some search engine number documents ids by ascending PageRank, or by natural crawl order (which approximates high quality pages first). Another interest of this study is that some conclusions - for instance URL ordering improving compression - do not necessarily generalise to all collections. Moreover, the speed increases shown by applying the compression techniques would benefit a large commercial search engine by allowing less machines to be involved in the computation of PageRank, resulting in data centre power and equipment savings.

References

1. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the Web. Technical report, Stanford Digital Library Technologies Project (1998)
2. Kleinberg, J.: Authoritative sources in a hyperlinked environment. J. ACM 46(5), 604–632 (1999)
3. Boldi, P., Vigna, S.: The WebGraph Framework I: compression techniques. In: Proceedings of WWW 2004, pp. 595–602 (2004)
4. Boldi, P., Vigna, S.: The WebGraph Framework II: Codes for the WWW. Technical Report 294-03, Universit di Milano (2003)
5. Elias, P.: Universal codeword sets and representations of the integers. IEEE Transactions on Information Theory 21(2), 194–203 (1975)
6. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Lioma, C.: Terrier: A high performance and scalable IR platform. In: Proceedings of SIGIR OSIR Workshop, pp. 18–25 (2006)