Hypergraph-Theoretic Partitioning Models for Parallel Web Crawling

Ata Turk, B. Barla Cambazoglu and Cevdet Aykanat

Abstract Parallel web crawling is an important technique employed by large-scale search engines for content acquisition. A commonly used inter-processor coordination scheme in parallel crawling systems is the link exchange scheme, where discovered links are communicated between processors. This scheme can attain the coverage and quality level of a serial crawler while avoiding redundant crawling of pages by different processors. The main problem in the exchange scheme is the high inter-processor communication overhead. In this work, we propose a hypergraph model that reduces the communication overhead associated with link exchange operations in parallel web crawling systems by intelligent assignment of sites to processors. Our hypergraph model can correctly capture and minimize the number of network messages exchanged between crawlers. We evaluate the performance of our models on four benchmark datasets. Compared to the traditional hash-based assignment approach, significant performance improvements are observed in reducing the inter-processor communication overhead.

1 Introduction

In order to maintain the accuracy of the information presented to their users, search engines build very large document repositories, trying to cover the entire Web while maintaining the freshness of constructed repositories. Web crawlers are among the most important software employed in forming these repositories. A web crawler is mainly responsible for locating, fetching, and storing web pages by following the link structure between them.

A. Turk · C. Aykanat (✉)
Computer Engineering Department, Bilkent University, Ankara, Turkey
e-mail: aykanat@cs.bilkent.edu.tr

B. B. Cambazoglu
Yahoo! Research, Barcelona, Spain

E. Gelenbe et al. (eds.), *Computer and Information Sciences II*,
DOI: 10.1007/978-1-4471-2155-8_2,

There are many challenges in sequential web crawling [1]. Efficient web crawling requires the use of distributed systems having high network bandwidth, large processing power, memory, and disk capacities [2, 3]. Consequently, in state-of-the-art search engines, the web crawling problem is addressed by parallel crawlers running on distributed memory parallel architectures, where each processor hosts a separate crawler. In [4], a taxonomy of parallel crawlers is provided, based on the coordination among processors. As explained in [4], the exchange scheme is one of the best inter-processor coordination schemes for parallel crawling. In the exchange scheme, the retrieval task for a link is assigned to the processor that has the responsibility of storing the page pointed by the link.

The hash-based task assignment scheme [4–7] has been widely used in parallel web crawling systems that utilize the exchange coordination scheme. In [7], the hashes are computed over the whole URLs to assign pages to crawlers, whereas, in [4–6], the hashes are computed over the host component of URLs. Compared to the page-hash-based scheme, the site-hash-based scheme has the advantage of reducing the number of inter-processor links since pages within the same site are more likely to link each other.

The above-mentioned page-to-processor assignment schemes implicitly address the load balancing problem, but they do not capture the cost of inter-processor communication. In [8], a greedy constructive algorithm is proposed to integrate the cost of communication into the execution cost of a crawling task on a processor in a heterogeneous system. In [9], we proposed a graph-partitioning-based (GP-based) model for page-to-processor assignment. In this model, the partitioning constraint addresses the load balancing problem while the partitioning objective of minimizing the cutsize defined over the edges that span more than one parts corresponds to minimizing the total volume of inter-processor communication. In parallel crawling, the messages exchanged are relatively small in size (around 50 bytes), but many. Hence, the latency overhead, which can be expressed in terms of the number of messages exchanged between processors, is a better indicator of the communication overhead. In [10], a GP-based page-to-processor assignment scheme is proposed to improve the partitioning strategy of a geographically distributed crawler by utilizing geographical information (server location and geographic scope of web pages).

The state-of-the-art parallel crawling systems generally have a distributed architecture, and crawling agents are connected through a wide area network. Depending on the characteristics of the underlying communication network, the message latency overhead in link exchanges between processors may become a bottleneck. We try to minimize this latency overhead (number of network messages) in the exchange scheme via a hypergraph-partitioning-based assignment model that minimizes the total message count while enforcing two load balancing constraints. The first constraint enforces a balance on the retrieval and storage task loads of processors while the second constraint enforces a balance on the number of issued page download requests.

2 Preliminaries

We represent the Web as a two-tuple $(\mathcal{P}, \mathcal{L})$, where $\mathcal{P}$ and $\mathcal{L}$ indicate the set of pages and the set of links between pages, respectively. Without loss of generality, multiple links from a page p_i to page p_j in $\mathcal{P}$ are assumed to be coalesced into a single link ℓ_{ij} in $\mathcal{L}$. For efficient crawling, our models utilize the web structure obtained in the previous crawling session to provide a better page-to-processor mapping for the following crawling session.

A hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{N})$ is defined as a set of vertices $\mathcal{V}$ and a set of nets (hyperedges) $\mathcal{N}$, where each net connects a number of distinct vertices. The vertices connected by a net n_i are said to be its pins and denoted as $Pins(n_i)$. A cost $c(n_j)$ is assigned as the cost of a net $n_j \in \mathcal{N}$. Multiple weights $w^1(v_i), w^2(v_i), \ldots, w^R(v_i)$ may be associated with a vertex $v_i \in \mathcal{V}$.

$\Pi = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_K\}$ is said to be a K-way partition of a given hypergraph $\mathcal{H}$ if vertex parts (i.e., every $\mathcal{V}_k$, for $k \in \{1, \ldots, K\}$) are mutually disjoint and their union is exhaustive. The K-way hypergraph partitioning (HP) problem can be defined as finding a K-way vertex partition Π that optimizes a partitioning objective defined over the nets that connect more than one part while satisfying a given partitioning constraint. The partitioning constraint is to satisfy multiple balance criteria on part weights, i.e.,

$$W^r(\mathcal{V}_k) \leq W^r_{\text{avg}}(1 + \epsilon^r), \text{for } k = 1 \text{ to } K, r = 1 \text{ to } R. \tag{1}$$

For the rth constraint, weight $W^r(\mathcal{V}_k)$ of a part $\mathcal{V}_k$ is defined as the sum of the weights $w^r(v_i)$ of vertices in $\mathcal{V}_k$, W^r_{avg} is the weight that each part should have in the case of perfect balancing, and ϵ^r is the maximum imbalance ratio allowed.

In a partition Π of hypergraph $\mathcal{H}$, a net is said to connect a part if it has at least one pin in that part. The connectivity set $\Lambda(n_j)$ of a net n_j is the set of parts connected by n_j. The connectivity $\lambda(n_j) = |\Lambda(n_j)|$ of a net n_j is the number of parts connected by n_j. A net n_j is said to be cut if it connects more than one part (i.e., $\lambda(n_j) > 1$) and uncut otherwise. In our particular problem, the partitioning objective is to minimize the connectivity-1 metric

$$Cutsize(\Pi) = \sum_{n_j \in \mathcal{N}_{\text{cut}}} c(n_j)(\lambda(n_j) - 1), \tag{2}$$

defined over the set $\mathcal{N}_{\text{cut}}$ of cut nets. The HP problem is known to be NP-hard [11, 12]. However, there are successful HP tools (e.g., hMeTiS [13] and PaToH [14]).

3 Site-Based Partitioning Model

In our site-based HP model, we represent the link structure $(\mathcal{P}, \mathcal{L})$ by a site hypergraph $\tilde{\mathcal{H}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{N}})$. We assume that a set $\mathcal{S} = \{S_1, S_2, \ldots\}$ of sites is given,

Table 1 Properties of the datasets

Dataset	Number of			% of intra-site	Site-based hypergraph Number of		
	Pages	Sites	Links	links	Vertices	Nets	Pins
google[a]	913,569	15,819	4,480,218	87.42	15,819	86,552	277,805
in-2004 [15, 16]	1,382,908	4,380	16,917,053	95.92	4,380	205,106	555,195
de-fr [17][b]	8,212,782	38,741	39,330,882	75.06	38,741	2,091,986	4,968,036
indochina [15, 17]	7,414,866	18,984	194,109,311	92.80	18,984	250,931	799,783

[a] Google contest, available at http://www.google.com/programming-contest.
[b] Crawled with Larbin: multi-purpose web crawler, available at http://larbin.sourceforge.net.

where sites are mutually disjoint and exhaustive page subsets of $\mathcal{P}$. All pages belonging to a site S_i are represented by a single vertex $v_i \in \tilde{\mathcal{V}}$. The weights $w^1(v_i)$ and $w^2(v_i)$ of each vertex v_i are set equal to the total size of pages (in bytes) and the number of pages hosted by site S_i, respectively.

A K-way partition is applied on $\tilde{\mathcal{H}}$ for a parallel system that contains K crawlers/processors. In a K-way partition $\tilde{\Pi} = (\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_2, \ldots, \tilde{\mathcal{V}}_K)$ of $\tilde{\mathcal{H}}$, each vertex part $\tilde{\mathcal{V}}_k$ corresponds to a subset $\mathcal{S}_k$ of the set $\mathcal{S}$ of sites, where pages of each site $S_i \in \mathcal{S}_k$ are to be retrieved and stored by processor P_k.

In the site-based hypergraph $\tilde{\mathcal{H}}$, vertices are in site granularity, whereas nets are in page granularity. For each page p_j having at least one outgoing inter-site link, there exists a net n_j with cost $c(n_j) = 1$. Vertex v_k is a pin of net n_j if and only if site S_k contains page p_j or a page p_i pointed by link ℓ_{ji}. That is,

$$Pins(n_j) = \{v_k : p_j \in S_k\} \cup \{v_k : \ell_{ji} \in \mathcal{L} \wedge p_i \in S_k\}. \tag{3}$$

Pages of a site that does not have outgoing links to page(s) of any other site do not incur nets in $\tilde{\mathcal{H}}$.

4 Experimental Results

To validate the applicability of the proposed model, we run simulations over a set of precrawled page collections. These collections are converted into hypergraphs as described in Sect. 3. Properties of the test datasets and respective site-based hypergraphs are displayed in Table 1. Since the page size information is not available for the in-2004, indochina, and de-fr datasets, unit page sizes are assumed for vertex weighting in hypergraphs corresponding to these datasets. As seen in the table, in all datasets, more than 75% of the links remain among the pages belonging to the same site.

The state-of-the-art HP tool PaToH [14] is used to partition constructed hypergraphs. The imbalance tolerance is set to 5% for both weight constraints. Due to the randomized nature of PaToH, experiments are repeated 8 times, and average values are reported.

Table 2 Load imbalance values and message counts

Dataset	(K)	Load imbalance (%)		Message count (10^3)	
		Hash-based	HP-based	Hash-based	HP-based
google	4	14.57	4.99	24.0	8.4
	8	20.79	4.99	40.0	9.3
	16	22.34	4.99	56.4	10.9
	32	70.18	4.99	71.7	12.3
	64	131.46	4.79	84.7	13.7
in-2004	4	11.69	4.84	5.3	1.4
	8	25.00	4.29	8.9	1.9
	16	40.58	8.20	12.8	2.5
	32	73.47	9.54	16.6	3.1
	64	142.86	10.80	19.9	3.6
de-fr	4	12.75	5.00	52.6	9.5
	8	34.51	5.00	86.8	10.9
	16	58.62	5.00	128.8	10.7
	32	81.95	10.33	177.8	12.4
	64	203.39	91.61	231.9	14.1
indochina	4	3.71	5.00	18.4	4.1
	8	9.91	5.00	33.7	5.2
	16	21.77	5.00	54.5	6.4
	32	41.97	5.00	80.0	7.6
	64	51.91	8.62	109.8	8.1

Table 2 displays the performance of the proposed site-based HP model against the site-hash-based model in load balancing and reducing the number of messages. In terms of load balance, only computational load imbalance values related to the page download request counts of processors are reported. Storage imbalance values are not reported since actual storage requirements are not used due to their unavailability in three out of four datasets. The communication overhead due to the link exchange operation is reported in terms of the number of messages exchanged between crawlers. We use K values of 4, 8, 16, 32, and 64. As seen in Table 2, the HP model performs significantly better than the site-hash-based model in balancing the page request loads of processors. We observe that, the load balancing performance of the site-hash-based model drastically deteriorates with increasing K values. This is basically due to the high variation in the sizes of the sites in the datasets used. This experimental observation shows the need for intelligent algorithms instead of hash-based algorithms even solely for load balancing purposes in site-based assignment, especially for large K values.

As also seen in Table 2, the HP model performs significantly better than the hash-based model in reducing the message latency overhead as well. More specifically, on average, the HP model produces partitions with 4.9, 4.9, 7.3, and 8.7 times fewer number of messages than the partitions produced by the site-hash-based model in google, in-2004, de-fr, and indochina datasets, respectively. In general, the performance ratio between the HP and hash-based models increases

with increasing number of processors. For example, for the largest number of processors ($K = 64$), the site-based HP model produces partitions which incur 6.2, 5.5, 16.5, and 13.6 times fewer number of messages than the site-hash-based model in google, in-2004, de-fr, and indochina datasets, respectively.

These experimental findings confirm the need for intelligent algorithms such as HP models in order to maintain the message latency overhead at acceptable levels for large K values.

5 Conclusion

In this paper, we proposed a model for minimizing the communication overhead of parallel crawlers that work in the exchange mode. The model provides considerable improvement in terms of the number of network messages exchanged between the crawlers, relative to the hash-based assignment approach.

References

1. Lee, H.-T., Leonard, D., Wang, X., Loguinov, D.: IRLbot: scaling to 6 billion pages and beyond. In: Proceedings of the 17th International Conference on World Wide Web, pp. 427–436 (2008)
2. Baeza-Yates, R., Castillo, C., Junqueira, F., Plachouras, V., Silvestri, F.: Challenges in distributed information retrieval. In: International Conference on Data Engineering, pp. 6–20 (2007)
3. Cambazoglu, B.B., Plachouras, V., Junqueira, F., Telloli, L.: On the feasibility of geographically distributed web crawling. In: Proceedings of the 3rd International Conference on Scalable Information Systems, pp. 1–10 (2008)
4. Cho, J., Garcia-Molina, H.: Parallel crawlers. In: Proceedings of the 11th Int'l Conference on World Wide Web, pp. 124–135 (2002)
5. Edwards, J., McCurley, K., Tomlin, J.: An adaptive model for optimizing performance of an incremental web crawler. In: Proceedings of the 10th International Conference on World Wide Web, pp. 106–113 (2001)
6. Heydon, A., Najork, M.: Mercator: a scalable, extensible web crawler. World Wide Web **2**(4), 219–229 (1999)
7. Shkapenyuk, V., Suel, T.: Design and implementation of a high-performance distributed web crawler. In: Proceedings of the 18th International Conference on Data Engineering, pp. 357–368 (2002)
8. Teng, S.-H., Lu, Q., Eichstaedt, M., Ford, D., Lehman, T.: Collaborative web crawling: information gathering/processing over Internet. In: Proceedings of the 32nd Annual Hawaii International Conference on System Sciences (1999)
9. Cambazoglu B, B., Turk, A., Aykanat, C.: Data-parallel web crawling models. Lect. Notes. Comput. Sci. **3280**, 801–809 (2004)
10. Exposto, J., Macedo, J., Pina, A., Alves, A., Rufino, J.: Efficient partitioning strategies for distributed web crawling. Lect. Notes. Comput. Sci. **5200**, 544–553 (2008)
11. Berge, C.: Graphs and Hypergraphs. North-Holland Publishing Company, New York (1973)
12. Lengauer, T.: Combinatorial Algorithms for Integrated Circuit Layout. Wiley, UK (1990)

13. Karypis, G., Kumar, V.: Multilevel k-way hypergraph partitioning. In: Proceedings of the 36th annual ACM/IEEE Design Automation Conference, pp. 343–348 (1999)
14. Çatalyürek, U.V., Aykanat, C.: PaToH: a multilevel hypergraph partitioning tool, version 3.0. Technical report, Bilkent University. Department of Computer Engineering (1999)
15. Boldi, P., Codenotti, B., Santini, M., Vigna, S.: UbiCrawler: a scalable fully distributed web crawler. Softw. Pract. Experience **34**(8), 711–726 (2004)
16. Boldi, P., Vigna, S.: The WebGraph framework I: compression techniques. In: Proceedings of the 13th International Conference on World Wide Web, pp. 595–602 (2004)
17. Jean-Loup, G., Latapy, M., Viennot, L.: Efficient and simple encodings for the web graph. In: Proceedings of the 3rd International Conference on Advances in Web-Age Information Management, pp. 328–337 (2002)

http://www.springer.com/978-1-4471-2154-1

Computer and Information Sciences II
26th International Symposium on Computer and Information Sciences
Gelenbe, E.; Lent, R.; Sakellari, G. (Eds.)
2012, XIV, 586 p., Hardcover
ISBN: 978-1-4471-2154-1