19th International Computer Science and Engineering Conference (ICSEC)
Chiang Mai, Thailand, 23-26 November, 2015

An Algorithm to Improve MPI-PageRank Performance by Reducing Synchronization Time.

Sumalee Sangamuang
Department of Computer Engineering
Chiang Mai University
Chiang Mai, Thailand
Email: sumalee.sa@cmu.ac.th

Pruet Boonma
Department of Computer Engineering
Chiang Mai University
Chiang Mai, Thailand
Email: pruet@eng.cmu.ac.th

Lai Lai Win Kyii
Department of Information Technology
Mandalay Technological University
Mandalay, Myanmarand
Email: laelae83@gmail.com

***Abstract*—Ranking is an important operation in web searching. Among many ranking algorithms, PageRank is a most notable one. However, sequential PageRank computing on a large web-link graph is not efficient. T o a ddress s uch l imitation, parallel PageRank implemented on Message Passing Interface (MPI) is a viable choice. Generally speaking, MPI-PageRank will be implemented using a root node and many computing, i.e., child, nodes. In each PageRank iteration, root node will partition web-link graph and distribute to child nodes. Then, each child node will perform PageRank on its partial web-link graph. Next, child nodes will send the result back to be combined at the root node. This operation will be performed iteratively before the ranking is converged. From the observation that when the number of nodes increase, the time to communicate between root and child nodes, i.e., synchronization time, increases rapidly such that it overcomes the benefit of parallel computing. This paper proposed an algorithm to reduce such time with a trade-off on ranking accuracy. The evaluation result show that the proposed algorithm can improve performance in term of the execution time with a bargain of accuracy.**

I. Introduction

Web-pages ranking is an important process for web search engine. There are several efficient m ethod f or r anking web-pages, e.g., Hub and Authority [1] which determines an important of a web-page by Hub score and Authority score, PageRank [2] which considers an important of a web-page by incoming links and outgoing links.

PageRank considers importance of web-pages on a web link graph, a structure of web-pages and their web-links. Web-pages are represented by nodes and their web-links are represented by edges. Importance of web-pages on a web link graph are determined by their incoming and their outgoing links. Each web-pages considers the summation of PageRank values from incoming links as its PageRank value while distributes it's PageRank value to the other pages through the outgoing links. PageRank computes iteratively until the PageRank value of each web-page is not charged. An execution time of it, knows as convergence time, depend on largeness of a web link graph. Because a web link graph can be generally large. PageRank computing in such structure is not effective.

Parallel processing techniques for PageRank computing are widely adopted to improve efficiency o f P ageRank. Many research proposed methods for parallel PageRank computing, e.g., [3], [4], [5], [6] , [7] and [8]; these methods using MPI framework for PageRank computing. In MPI-based PageRank computing, a web link graph will be divided into many small parts which are distributed to child processors by the root processor (Processor ID/PID 0). Each child processor locally computes and send it back to the root processor. The root processor will combine the results of the others processors for updating the global PageRank values. Figure 1 shows an example of MPI-based PageRank computing. The root processor divide a web link graph G into n small parts and distributes them into $n-1$ child processors (PID 1 - $n-1$). Each processor locally computes and send it back to the root processor. Before executing next iteration of PageRank computing, the root processor has to wait every processors finish, this is called a synchronization period. This synchronization period can slow down the whole operation especially when the number of processors increased.

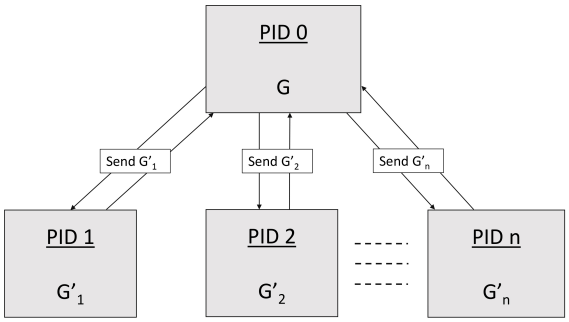

Fig. 1: An example of MPI-based PageRank computing.

Figure 2 shows the PageRank computing time and synchronization time for MPI-based PageRank computing. When the number of processors is increasing, of course, the computing time will be reduced because each processor has to process a smaller graph. On the other hand, the synchronization time is increased relatively to the number of processors. Totally, the execution time of MPI-PageRank will be reduced when the number of processors increased from one processor. However, then the system grows to a certain point, i.e., about 45 processor in 1, the execution time is increasing. This can be

978-1-4673-7825-3/15/$31.00 ©2015 IEEE

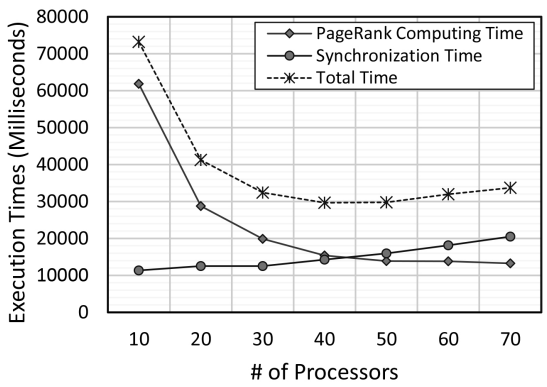

Fig. 2: Impact of number of processors on execution times of MPI-based PageRank.

observed in many MPI-PageRank implementations [4].

In this paper, the problem of excessive execution time from synchronization time is investigated. MPI-based PageRank can reduce execution time of PageRank computing but if the number of processors is increasing over a certain point, a synchronization time becomes overly large and override the benefit of parallel computing. To address this problem, a method to reducing the synchronization time is proposed in this paper. The evaluation results shows the impact of reducing the synchronization time on performance in term of MPI-based PageRank computing's execution time and performance in term of accuracy. Reducing the synchronization time can improve performance in term of the execution time. Although it can reduce the accuracy, appropriate reducing the synchronization time still acceptable.

II. Synchronization PageRank Values

Execution times of MPI-based PageRank depend on two parts of PageRank computing, i.e., a convergence time of PageRank computing and a synchronization time of updating PageRank values in a global list. Reducing the synchronization time can decrease overall execution times of MPI-based PageRank computing.

Base on [4], the method which can reduce the convergence time of PageRank values is proposed by clustering web-pages in a web link graph. Such method can improve performance in term of an execution time but excessive processors make it worse.

This topic, a problem of improving performance in term of MPI-based 's execution time is considered and a method for reducing synchronization time of MPI-based PageRank is proposed.

III. Proposed Algorithm

This section discuses the proposed algorithm, called PR-MPI2, which can improve the execution time of MPI-PageRank by reduce the synchronization cost. There are three parts of the algorithm; first part is to initialize PageRank computing, second part is to locally compute PageRank in each processor and final part is to synchronize PageRank value of all web-pages in web-link graph. Algorithm 1 shows the pseudo code of PR-MPI2.

Given a web link graph G, the web pages in G is iterated and included in the $globalList$ by the root processor (line 5). All of web-pages in $globalList$ must have initial PageRank value before starting the PageRank computing (line 6). Such value is ordinarily defined as $\frac{1}{SizeOf(G)}$ for every nodes. Then, the root processor will broadcast the $globalList$ to all of child processors (line 9). Then, in each child processor, the $localList$ will be generated by the $Partition$ function which divides the $globalList$ into equal parts and the processor will process only a part of it. The constant β is a percentage of PageRank iteration that need synchronnization. In particular, this value determines the ratio between the number of synchronization (line 11- 15) and the number of PageRank iterations. If β is 100 (percent), it means the algorithm will always synchronize PageRank values with the values in the $globalList$. On the other hand, the algorithm will never synchronize PageRank values when β is 0. For example, If β is 80 and the max number of iteration (μ) is 160, it means the algorithm will synchronize PageRank values 128 times (i.e., 80% of 160) from 160 iterations.

All processors will start computing PageRank value in the $localList$ simultaneously. PageRank values are locally computed until the value of syn variable is $true$, then each child processor will start synchronize PageRank value with root node to updating PageRank value of all web-pages in the $globalList$ (line 23). Then, the $globalList$ will be broadcast from the root processor (19-29). Although, the PageRank computation will be performed until the values of PageRank are converted, this paper interrupts the PageRank computation with threshold μ which is a max number of iterations (line 16-17). The threshold μ should be large to increasing accuracy of final result.

IV. Experiment Results

This section shows the experiment results to evaluate PR-MPI2. In this experiment, PR-MPI2 is compared with a baseline algorithm in [4], labeled PR-MPI. PR-MPI and PR-MPI2 are similar however, PR-MPI2 reduces the number of synchronization. The experiment shows impact of the reduced synchronization time, with the varying of β, on execution time of MPI web-pages ranking. Moreover, the performance in term of accuracy is showed in also this section. This paper shows a trade-off between an execution time and accuracy when β are charged. However, the optimum β are not still determined.

A. Simulation Setup

The data set used in the experiments was generated from GraphStream, a dynamic graph library base on Java. The number of web-links used in the experiments is in the range of 200,000 to 500,000 web-links. The parallel applications are executed by MPJ Express that number of processes are in range between 10 to 60 processors. The results are average of 5 experiments.

Algorithm 1 PR-MPI2

Input: Graph $G = (V, E)$, Iteration threshold μ, Percentage of synchronization β.
Output: The web-pages ranking results on MPI.

```
1: MPI.Init
2: rank ← MPI.COMM_WORLD.Rank
3: size ← MPI.COMM_WORLD.Size
4: if root processor then
5:    globalList ← Read(G)
6:    Initialization(globalList)
7: end if
8: MPI.COMM_WORLD.Barrier
9: MPI.COMM_WORLD.Bcast
10: localList ← Partition(globalList, size)
11: syn ← true
12: iteration ← 1
13: synCount ← 1
14: synMax ← (β/100) * μ
15: count ← β/synMax
16: for each iteration ≤ μ do
17:    PageRank(localList)
18:    MPI.COMM_WORLD.Barrier
19:    if syn then
20:       synCount++
21:       if root processor then
22:          MPI.COMM_WORLD.Recv
23:          Update(globalList)
24:       else
25:          MPI.COMM_WORLD.Send
26:       end if
27:       MPI.COMM_WORLD.Barrier
28:       MPI.COMM_WORLD.Bcast
29:    end if
30:    if iteration == count * synCount||iteration == β
       then
31:       syn ← true
32:    else
33:       syn ← false
34:    end if
35: end for
36: Ranking(globalList)
37: return globalList
```

B. Results

Figure 3 shows impact of percentage of synchronization on execution time of MPI web-pages ranking. The X-axis is percentage of synchronization which is in rages between 20 to 100 and Y-axis is the execution times of the PR-MPI algorithm. A graph with 500,000 web-links is used in this experiment. This figure compares execution times of PR-MPI2 algorithm that uses 10, 20 and 30 processors. The result shows that when the percentage of synchronization is increased, the execute time of synchronization PageRank value is reduced regardless of the number of processors.

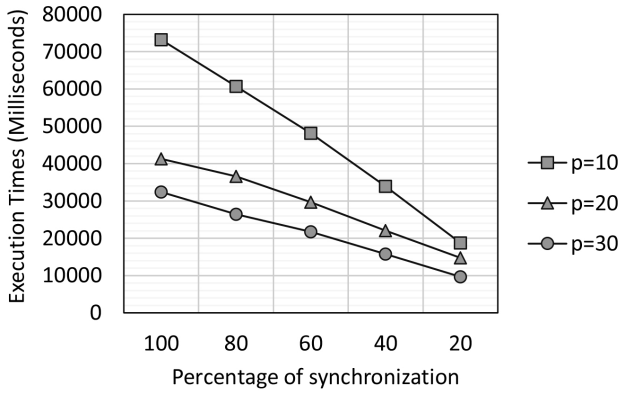

Fig. 3: Impact of percentage of synchronization on execution times of MPI web-pages ranking.

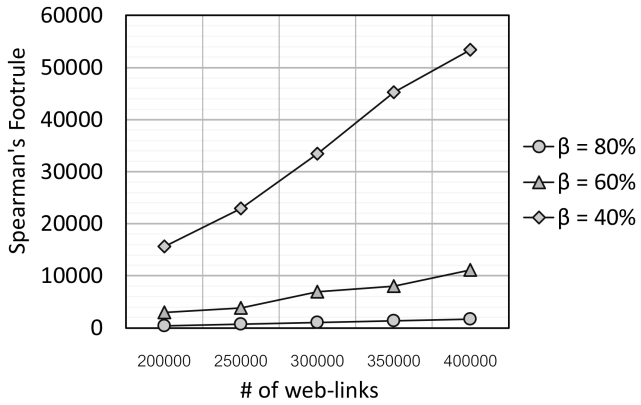

Fig. 4: Impact of the number of web-links on Spearmans Footrule.

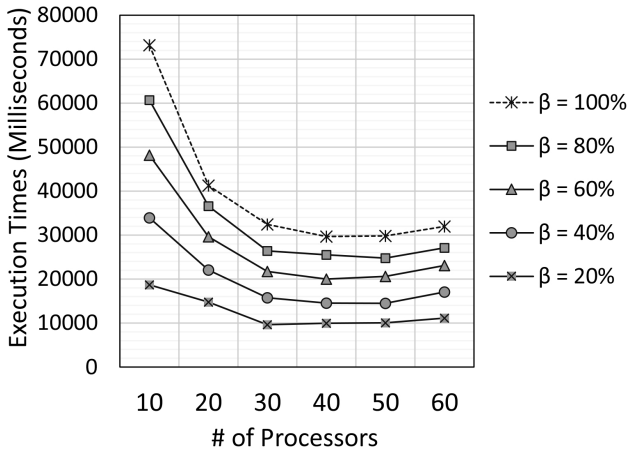

Fig. 5: Impact of the number of processors on execution times of MPI web-pages ranking.

However, there is a trade-off to the proposed approach; in particular, there is a accuracy decrement when the percentage of synchronization is reduced. Figure 4 shows impact of the number of web-links on Spearmans Footrule. The X-axis is

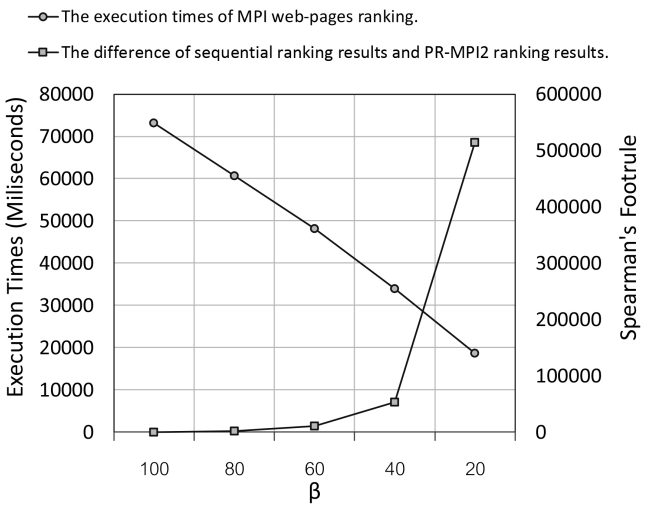

Fig. 6: Trade-off between execution times and accuracy of MPI web-pages ranking.

the number of web-links which are in range between 200,000 to 400,000 web-links. The Y-axis is Spearmans Footrule [9] which measures the difference of sequential ranking results and PR-MPI2 ranking results. The 20 processors are used in this experiment. The figure compares Spearmans Footrule of three PR-MPI2 ranking results that have the percentage of synchronizations are 40, 60 and 80, respectively. The result shows the growth of Spearmans Footrule depend on number of web-links. Moreover, decreasing the percentage of synchronizations can reduce the accuracy of PageRank.

Figure 5 shows impact of the number of processors on an execution time of MPI web-pages ranking. The X-axis is number of processors which are range between 10 to 60 processors and the Y-axis is the execution times of PR-MPI algorithm ($\beta = 100$) and the PR-MPI2 algorithm. A graph with 500,000 web-links is used in this experiment. The execution times of PR-MPI algorithm is always highest because it always synchronize PageRank values (shows in dash line). If the percentage of synchronization is decreased, an execution time of MPI web-pages ranking is also decreased.

From previous experiments, there is a trade-off between the execution times and the difference of sequential ranking results and PR-MPI2 ranking results, shows in Figure 6. The X-axis is percentage of synchronization which is in rages between 20 to 100 and Y-axis are the execution times of the PR-MPI2 algorithm (the left hand side) and the difference of sequential ranking results and PR-MPI2 (the right hand side). A graph with 500,000 web-links is used in this experiment which is processes on 20 processors. An execution time of MPI web-pages ranking is decreased when the percentage of synchronization is decreased. However,in term of accuracy will be increased. In this figure, the percentage of synchronization which less than 40 should not be used because term of accuracy will be decreased rapidly. Nevertheless, the percentage of synchronization are acceptable in range between 40 to 99 because term of accuracy is still acceptable region.

In conclusion, the result show that decreasing percentage of synchronization can improve execution times of MPI ranking with the cost of accuracy.

V. Conclusion and Future Work

In this paper, the problem of excessive execution time from synchronization time is observed. Thus, the method for reducing a synchronization time of MPI-based PageRank computing is proposed. Such method can reduce the synchronization time by decreasing percentage of synchronization. The experiment results show trade-off between the execution times and the difference of sequential ranking results and PR-MPI2 ranking results. Decreasing percentage of synchronization can improve performance in term of the execution time. Although it decreases performance in term of accuracy, appropriate reducing the synchronization time still acceptable.

References

[1] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, vol. 46, no. 5, pp. 604–632, 1999.

[2] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford University, Technical Report;, 1998.

[3] A. Rungsawang and B. Manaskasemsak, "Partition-based parallel pagerank algorithm," in *Information Technology and Applications, 2005. ICITA 2005. Third International Conference on*, vol. 2, July 2005, pp. 57–62.

[4] J. Arulraj, "Scalable link analysis algorithm for a distributed memory environment," in *Parallel Distributed and Grid Computing (PDGC), 2010 1st International Conference on*, Oct 2010, pp. 61–66.

[5] H. Yin, J. Li, and Y. Niu, "Ffsb: Fast fibonacci series-based personalized pagerank on mpi," in *Information, Communications and Signal Processing (ICICS) 2013 9th International Conference on*, Dec 2013, pp. 1–6.

[6] B. Manaskasemsak and A. Rungsawang, "Parallel pagerank computation on a gigabit pc cluster," in *Advanced Information Networking and Applications, 2004. AINA 2004. 18th International Conference on*, vol. 1, 2004, pp. 273–277 Vol.1.

[7] A. Selamat and F. Ahmadi-Abkenari, "Application of clickstream analysis as web page importance metric in parallel crawlers," in *Information Technology (ITSim), 2010 International Symposium in*, vol. 1, June 2010, pp. 1–6.

[8] A. Cevahir, C. Aykanat, A. Turk, and B. Cambazoglu, "Site-based partitioning and repartitioning techniques for parallel pagerank computation," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 5, pp. 786–802, May 2011.

[9] R. Fagin, R. Kumar, and D. Sivakumar, "Comparing top k lists," in *In Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2003, pp. 28–36.