Static Range Multiple Selection Algorithm for Peer-to-Peer System

Mohamed Othman, Kweh Yeah Lun, Fatimah bt. Dato Ahmad, Hamidah Ibrahim

***Abstract*—In this research, a new multiple selection algorithm, which is known as "static range statistical multiple selection algorithm" is proposed. This algorithm is developed based on the statistical knowledge about the uniform distribution nature of the data which has been arranged according to certain order in the file. A global file with *n* keys is distributed evenly among *p* peers in the peer-to-peer network. The selection algorithm can performs multiple selections concurrently to find multiple target keys with different predefined target ranks. The algorithm uses a fixed filter approach in which the algorithm is able to make sure that the target key is within certain filter range in each local file. The range is made smaller and smaller as the selection process iterates until all target keys are found. The algorithm is able to reduce the number of messages needed and increases the success rate of all multiple selections in the selection process compared to the previous multiple selection algorithms proposed by Loo in 2005.**

***Index Terms*—multiple selection, peer-to-peer system, static filter, filter range**

I. INTRODUCTION

Selection algorithm has been developed to ease the networking operation and peer-to-peer computing. It is an algorithm that deals with the problem of selecting k^{th} smallest key out of a group of keys, which are distributed almost evenly in the distributed environment or peer-to-peer environment. Selection operations are needed in some distributed sorting algorithms ([4], [8], [23], [10], [12], [13]). In these distributed sorting algorithms, it is necessary to find the n/i^{th} keys (where $i = 1, 2, \ldots, p - 1$; n is size and p is the number of computers involved). For a file, F with n records which distributed in a few sites and all this records are totally ordered, resolution algorithms [19] have been designed to minimize the amount of communication activity rather than the amount of processing activity.

Manuscript received January 12, 2010; revised Jan 2, 2011.

Mohamed. Othman is with the Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43300, UPM Serdang, Selangor, MALAYSIA. (Tel: +603-89466535, email: mothman@fsktm.upm.edu.my)

Kweh Yeah Lun is with the Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43300, UPM Serdang, Selangor, MALAYSIA. (email: kwehyl@fsktm.upm.edu.my)

Fatimah Dato Ahmad is with the Center for Research Management and Post Graduate Studies, National Defense University of Malaysia, Kem Sungai Besi, 57000 Kuala Lumpur, MALAYSIA. (email: fatimah@upnm.edu.my)

Hamidah Ibrahim is with the Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43300, UPM Serdang, Selangor, MALAYSIA. (email: hamidah@fsktm.upm.edu.my)

Different solutions and bounds exist for the distributed selection problem in the point-to-point network depending on the topology of the network ([5], [14], [22]).

The sampling techniques are used in designing the distributed algorithms [16]. Some distributed algorithms ([20], [18], [8]) were suitable for the intranet environment. Wei has developed an efficient selection and sorting schemes for processing large files distributed over a network [24].

Many other distributed selection algorithms have been developed for various purposes based on different topologies and assumption. Rodeh presented an algorithm for the case where two computers are connected together [17]. Shen presented his algorithm on hypercube [21] and Hao et al. presented an algorithm on mesh [7]. Aggrawal et al. presented a selection algorithm for Pyramid [1].

In 2003, Wu et al. [25] has designed fast and scalable parallel algorithms for selection and median filtering. It is the most time efficient algorithm, especially compared to the other algorithms mentioned in [15] and [6]. Alexandros et. al. [2] presented a randomized selection algorithm whose performance is analyzed in an architecture independent way on the bulk-synchronous parallel (BSP) model of computation. Bader presented an efficient randomized high-level parallel algorithm (Fast and UltraFast) and solves the general selection problem that requires the determination of the element of rank k [3].

Loo et al. [11] presented a statistical selection algorithm, which are designed to select k^{th} smallest key from a very large file distributed over many computers. This algorithm aims to minimize the number of communication messages necessary to the selection problem.

Loo [9] presents an efficient distributed multiple selection algorithm which is designed to select multiple keys simultaneously from different data sets which are distributed in a peer-to-peer system and aimed to reduce the number of communication messages compared to the single selection algorithm that he proposed previously [11] and other algorithms in the literature.

This research work is based on the work by Loo et al., [9] which is currently the best multiple selection algorithm in the peer-to-peer environment. A simulation is set up to re-implement the algorithm and the new algorithm to measure them based on the performance parameter selected.

II. STATIC RANGE MULTIPLE SELECTION ALGORITHM

Static range selection algorithm is an algorithm that makes use of the hashing approach and predefined filter range to locate the target key with certain target rank. It is a multiple selection algorithm in which multiple target keys

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

are to be found with each selection will occupied a fixed filter range but with different upper filter and lower filter.

A total number of keys, *n* will be distributed evenly to *p* participants in the system. In this research, the total number of keys in the global file is 100 millions. Preprocessing stage include the hashing stage, the determination of lower filter and upper filter for each selection. After that, the first pivot is identified and the selection round starts. The upper filter and the lower filter will be adjusted based on the actual global rank of the pivot. The details of the algorithm are as follows.

A. Hashing Stage

Based on the statistical knowledge of the uniform distribution of all keys which are randomly generated by each node in the local file, it is possible to hash the keys in to hash values, which can be used as indicators to the actual global rank of the keys in the global file. By gathering all local maximum and the minimum keys from all nodes, the coordinator determines the global maximum and the global minimum keys and broadcast them to all nodes involved.

The equation of the hash value generator is as follow:

$$Hash\_value = \text{floor}\left\lfloor \frac{key - \min}{range} \times n \right\rfloor \quad [1]$$

where min: minimum global key
range = maximum global key – minimum global key
n : number of keys in the global file

B. Identification of Lower and Upper Filter

Based on the predefined filter range, two filters will be selected for each of the selection, namely upper filter and lower filter.

C. Determination of Global Rank for Filter

After the identification of both upper and lower filter keys for each selection by each node, these filter keys will be sent to other nodes. The nodes will determine the local rank for the filter keys. After that the participants will send the local ranks for all the filter keys back to the sender. Upon receiving the local ranks from each participant, actual global rank for each filter keys can be calculated based on the following equation:

$$G = \sum_{r=1}^{p} R[r] \quad [2]$$

where *R[r]* : local rank from node *r*

After the pre-selection process, the global ranks for each filter keys (lower and upper) will be determined.

D. Selection Process

The actual round of the selection process started after the determination of the global rank of each filter keys of each selection.

The first pivot and the subsequent pivot for each selection are calculated based on the predicted value from the following equation:

$$Predicted\_value_i = A_i + \frac{k_i - AR1_i}{AR2_i - AR1_i}(B_i - A_i) \quad [3]$$

where $AR1_i$: actual global rank for lower filter A_i
$AR2_i$: actual global rank for upper filter B_i
k_i: *target rank i*

The pivots are selected by finding the greatest value that is smaller or equal to the predicted value.

As shown in figure 1, the receiver nodes received the pivots from the sender and the calculation for the local ranks of the pivots begun. Upon receiving the local ranks from receiver nodes, the sender calculates the global rank for each of its pivots.

If the global rank of the pivot is equals to the target rank for that particular selection, then the searching process for that selection will be terminated.

If the global rank of the pivot is not equal to the target rank, then a new predicted value will be calculated again based on the equation mentioned above. The local file will be divided into two sub-files. The first sub-file contains all keys that are smaller than the pivot and vice versa. If the global rank of the pivot is greater than the target rank, then the first sub-file is used. Upper filter and lower filter will be adjusted to the smallest and the largest key of the sub-file. If the global rank of the pivot is smaller than the target rank, the second sub-file is used with the upper and lower filter will be adjusted as mentioned.

New pivot values are calculated. These pivots will be sent to all nodes. This process will be iterated until all target keys have been found.

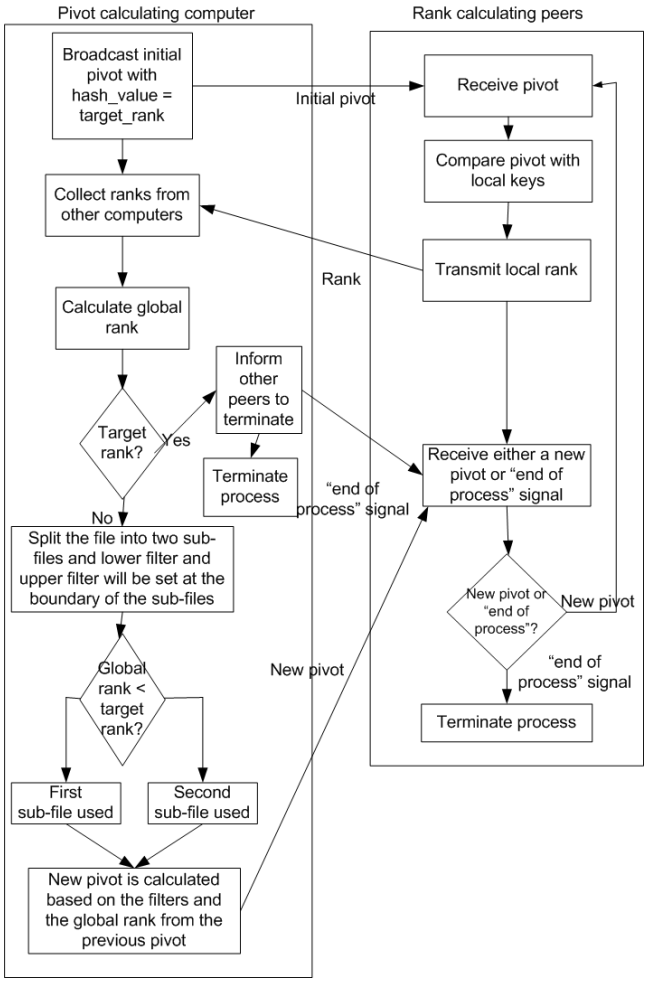

Fig. 1: Selection Process

III. NUMBER OF MESSAGES NEEDED

According to figure 2, the number of messages needed by the Loo's algorithm [9] is 61.78 for 10 nodes and 215.38 for 40 nodes. As the number of nodes increases, the number of

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

messages increases. Loo's algorithm [9] has the highest number of messages needed to complete the selection process compared to static range multiple selection algorithms with different filter ranges (25000, 20000 and 15000).

As the result shown, filter range 25000 selection algorithm will be able to finish the whole selection process with lower number of messages required compared to [9]. The number of messages required is within the range from 53 to 183. For the case of 10 nodes, the improvement is 13.05%. As the number of nodes increases, the improvement becomes more significant. For the case of 40 nodes, the improvement is 15.41%.

For filter range 20000 selection algorithm which occupied a narrower range, the number of messages needed to complete the selection process has becomes smaller. The improvement for the case 10 nodes and 40 nodes are 13.48% and 15.71% respectively.

The number of messages required for filter range 15000 selection algorithm has a range from 52 to 179. The improvement shown by this approach for the case of 10 nodes and 40 nodes are 14.41% and 17.62% respectively.

As the number of nodes increases, the number of messages required increases. With the increment of the number of nodes involved, number of target keys that need to be searched increases. For each node added to the system, the number of target keys that need to be found is increases by 1. In the experiment provided above, the number of nodes increases 5 between each cases, this means that the number of selections (target keys) that need to be found is increases by 5 too. The number of messages required is increases with greater number of target keys that need to be found.

By comparing to the original algorithm, filter range 15000 selection algorithm shows the best improvement in reducing the number of messages needed in the case of 40 nodes with 17.62% compared to filter range 25000 with 15.41% and filter range 20000 with 15.71%.

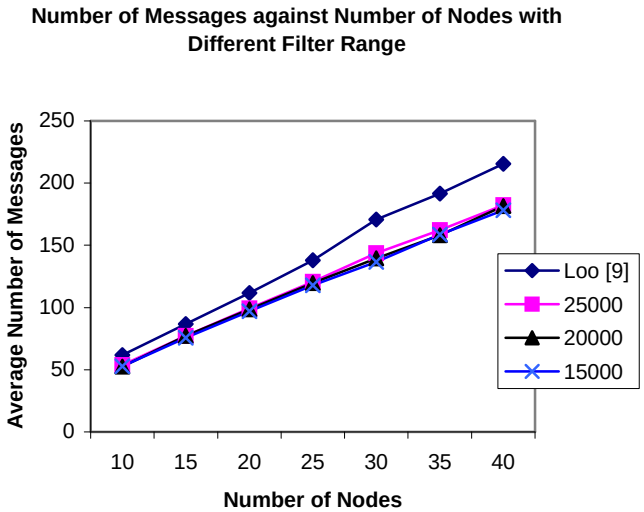

Fig. 2: Number of messages against number of nodes

IV. SUCCESS RATE

According to figure 3, the success rate of the original algorithm is within 0.53 to 0.6. Success rate is a measurement of how many selections that has been succeed in finding the target keys in the multiple selection process. With more nodes are involved, the success rates of the selections increase. The best performance of Loo's algorithm [9] occurs in the case of 40 nodes, where the success rate is 0.601.

Filter range 25000 selection algorithm achieved the best result among all the other filter range approaches. The success rate for this algorithm is consistent which is about 0.99 from the case of 10 nodes to the case of 40 nodes.

Filter range 20000 selection algorithm achieved success rate that is quite consistent at about 0.98 for all cases from 10 nodes to 40 nodes. By comparing to filter range 25000 selection algorithm, the success rate has been reduced about 1.68% for the case of 10 nodes and 1.35% for the case of 40 nodes.

For filter range 15000 selection algorithm, the success rate is still more than 0.9. By comparing to filter range 20000 selection algorithm, the success rate has been reduced about 3.86% for the case of 10 nodes and 4.8% for the case of 40 nodes.

As the number of nodes increases, the number of selections that need to be performed in the multiple selection process will be increase as well. The success rates are increased because of the number of pivots that can be checked in a round had been increased. With more pivots that can be sent and checked in the whole process, the probability of finding the target key for a certain selection will be increased.

The success rate has been reduced as the filter range becomes narrower. With the narrower range, the probability of a target key that lies between the filters are lower. Thus, some selection might not succeed as the target key is falls out of the filter range. However, the reduction in success rates is less than 2% according to the result. This shows that in most of the cases, the target keys are still fall within the filter range although the ranges has become narrower.

By comparing to Loo's algorithm [9], filter range 25000 selection algorithm achieved the highest success rate among all the other filter range approaches. However, filter range 25000 has the greatest number of messages required compared to the other filter range approaches. Since filter range 15000 has the lowest number of messages needed to complete the whole multiple selection processes and at the same time has a success rate which is more than 0.9 that is still acceptable, so static range multiple selection algorithm with filter range 15000 can be considered as a substitution to Loo's algorithm [9].

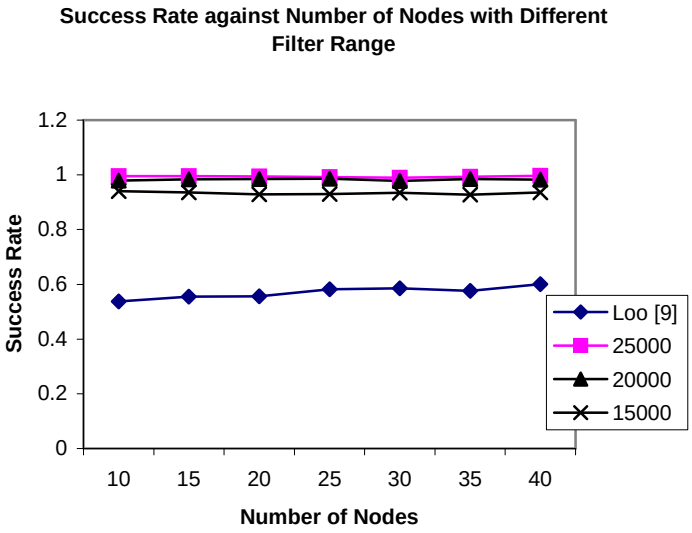

Fig. 3: Success rate against number of nodes

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. CONCLUSION

The static range multiple selection algorithm which utilized a fixed filter range with 15000 are able to reduce the number of messages needed to complete the multiple selection processes and achieving high success rate that is more than 0.9 compared to Loo's algorithm [9].

Future works might include other performance parameters like number of rounds needed and the execution time needed for the whole process.

REFERENCES

[1] Aggrawal, C., Jain, N. and Gupta, P., "An efficient selection algorithm on the pyramid", *Journal of Information Processing Letters*, 53, 1995, pp. 37-47.

[2] Alexandros, V., Gerbessiotis, Constantinos, and Siniolakis, J., "Architecture independent parallel selection with applications to parallel priority queues", Theoretical Computer Science, 301, 2003, pp. 119 – 142.

[3] Bader, D., "An Improved, randomized algorithm for parallel selection with an experimental study", Journal of Parallel and Distributed Computing, 64, 2004, pp. 1051-1059.

[4] Dechter, R. and Kleinrock, L., "Broad communications and distributed algorithms", *Journal of IEEE Transactions on Computers*, C-35(3), 1986, pp. 123-128.

[5] Frederickson, G.N., "Tradeoffs for selection in distributed networks", *Proceedings of 2nd ACM Symposium Principles Distributed Computing*, Montreal, Quebec, Canada, 1983, pp. 154-160.

[6] Han, Y., Pan, Y., and Shen, H., "Sublogarithmic Deterministic Selection on Arrays with a Reconfigurable Optical Bus", *IEEE Transaction on Computers*, 51(6), 2002, pp. 702 – 707.

[7] Hao, M., MacKenzie, P. and Stout, Q., "Selection on the reconfigurable mesh", *Proceedings of 4th Symposium on the Frontiers of Massively Parallel Computation*, McLean, Virginia, USA, 1992, pp. 38-45.

[8] Huang, J.H. and Kleinrock, L., "Distributed selectsort sorting algorithms on broadcast communication networks", *Journal of Parallel Computing*, 16(2/3), 1990, pp. 183-190.

[9] Loo, A., "Distributed multiple selection algorithm for peer-to-peer systems", *The Journal of Systems and Software*, 78, 2005, pp. 234 – 248.

[10] Loo, A., Bloor, C. and Grey, D., "Complexity analysis of distributed database algorithm", *Proceedings of High Performance Computing in Engineering 97*, Gran Canaria, Spain, 1997.

[11] Loo, A. and Choi, Y.K., "A peer-to-peer distributed selection algorithm for the Internet", *Journal of Internet Research: Electronic Networking Applications and Policy*, 12 (1), 2002, pp. 16-30.

[12] Loo, A., Chung, C., Fu, R. and Lo, J., "Efficiency measurement of distributed statistical sorting algorithms", *Proceeding of Applications of High Performance Computing in Engineering*, Milan, Italy, 1995.

[13] Loo, A. and Ng, J., "Distributed statistical sorting algorithm", *Proceeding of IEEE Singapore International Conference on Networks (SICON)*, Singapore, 1991, pp. 222-225.

[14] Matsuhita, T.A., "Distributed algorithms for election", *Master Thesis*, Department of Computer Science, University of Illinois, Urbana, Illinois, USA, 1983.

[15] Pan, Y., "Order statistics on optically interconnected multiprocessor systems", *Proceeding of First International Workshop Massively Parallel Processing Using Optical Interconnections*, 1994, pp. 162 – 169.

[16] Rajasekaran, S. and Wei, D.S.L., "Designing efficient distributed algorithms using sampling techniques", *Proceedings of 11th International Parallel Processing Symposium (IPPS'97)*, Geneva, Switzerland, 1997, pp. 397-401.

[17] Rodeh, M., "Finding the median distributively", *Journal of Computer and System Science*, 24(2), 1982, pp. 162-167.

[18] Santoro, N., Sidney, J.B. and Sidney, S.J., "A distributed selection algorithm and its expected communication complexity", *Journal of Theoretical Computer Science*, 100, 1992, pp. 185-204.

[19] Santoro, N. and Suen, E., "Reduction techniques for selection in distributed files", *Journal of IEEE Transactions on Computers*, 38(6), 1989, pp. 891-896.

[20] Saukas, E. and Song, S., "Efficient selection algorithms on distributed memory computers", *Proceedings of High Performance Networking and Computing*, Orlando, Florida, USA, 1988.

[21] Shen, H., "A universal algorithm for parallel *k*-selection in hypercubes", *Journal of Parallel Computing*, 18, 1991, pp. 139-145.

[22] Shirira, L., Francez, N. and Rodeh, M., "Distributed *k*-selection: from a sequential to a distributed algorithm", *Proceedings of 2nd ACM Symposium Principles of Distributed Computing*, Montreal, Quebec, Canada, 1983, pp. 143-153.

[23] Wegner, L.M., "Sorting a distributed file in a network", *Journal of Computer Networks*, 8, 1984, pp. 451-461.

[24] Wei, D.S.L., Rajasekaran, S., Cheng, Z., and Naik, K., "Efficient Selection and Sorting Schemes Using Coteries for Processing Large Distributed Files", *Journal of Parallel and Distributed Computing*, 62, 2002, pp. 1295 – 1313.

[25] Wu, C.H., and Horng, S.J., "Fast and Scalable Selection Algorithms with Applications to Median Filtering", *IEEE Transaction on Parallel and Distributed Systems*, 14(10), 2003, pp. 983 – 992.

ISBN: 978-988-18210-3-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)