Performance Analysis of Runtime Data Declustering over SAN-Connected PC Cluster

Masato Oguchi [1,2] and Masaru Kitsuregawa [2]

[1] Research and Development Initiative, Chuo University
42-8 Ichigaya Honmura-cho, Shinjuku-ku Tokyo 162-8473, Japan
Email: oguchi@computer.org
[2] Institute of Industrial Science, The University of Tokyo

ABSTRACT

Personal computer/workstation (PC/WS) clusters have come to be studied intensively in the field of parallel and distributed computing. They are considered to play an important role as a large scale computer system in the next generation, such as large server sites and/or high performance parallel computers, because of their good scalability and cost performance ratio. In the viewpoint of applications, data intensive applications including data mining and ad-hoc query processing in databases are considered very important for high performance computing, in addition to the conventional scientific calculation. Thus, investigating the feasibility of such applications on a PC cluster is meaningful.

In this paper, a PC cluster connected with Storage Area Network(SAN) is built and evaluated. In the case of SAN cluster, each node can access all shared disks directly without using LAN; thus, SAN clusters achieve much better performance than LAN clusters for disk access operations. However, if a lot of nodes access the same shared disk simultaneously, application performance degrades due to I/O-bottleneck. A runtime data declustering method, in which data is declustered to several other disks dynamically during the execution of application, is proposed to resolve this problem.

Parallel data mining is implemented and evaluated on the SAN-connected PC cluster. This application requires iterative scans of a shared disk, which degrade execution performance severely due to I/O-bottleneck. The runtime data declustering method is applied and characteristics of the system such as I/O and network operations are evaluated in detail. According to the results of experiments, the proposed method prevents performance degradation caused by shared disk bottleneck in SAN clusters.

KEY WORDS
Cluster Computing, Data Mining, Storage Area Network, Runtime Data Declustering

1 Introduction

Recently, high performance computer systems are using commodity parts as their components, including CPUs, disks, and memories, rather than proprietary parts. This is because technologies for such commodity parts have matured enough to be used for high-end computer systems. While an interconnection network between nodes has not yet been commoditized until now, some common-purpose networks, e.g. Fast/Gigabit Ethernet, are the strong candidates as a de facto standard of high speed communication networks. With the progress of technologies for such commodity high speed local area networks(LANs), future high performance computer systems will undoubtedly employ commodity networks as well. Thus, Personal computer/workstation (PC/WS) clusters using high speed commodity LANs have become an exciting research topic in the field of parallel and distributed computing. They are considered to be a promising platform for future high performance parallel computers, because of their good scalability and cost performance ratio.

Data intensive applications including data mining and data warehousing are extremely important for high performance computing in the near future. We previously built a PC cluster connected 100 Pentium Pro PCs with ATM-LAN, and implemented several database applications to evaluate their performance and the feasibility of such applications using PC clusters[1][2].

LAN-connected PC clusters are used as a system of large server site and/or a high performance parallel computer. In both cases, huge volume of data might be transferred frequently from one node's disk to another, for the execution of parallel computing, load distribution, maintenance of the system, and so on. In order to reduce LAN traffic and raise availability of nodes in the cluster, Storage Area Network(SAN), e.g. Fibre Channel, has come to be adopted[3]. SAN can link storage devices directly to all nodes of the cluster, therefore, SAN prevents the congestion of LAN traffic. In the case of SAN clusters, different from LAN clusters, each node does not have to communicate with each other through LAN for reading data

from other nodes' disks, because a pool of storage is shared among all nodes and can be accessed directly through SAN with no burden to the other nodes nor LAN.

In this paper, we have built a PC cluster which has a SAN-connection as well as a LAN-connection, and examined its performance features. Basic characteristics of data transfer on the cluster are evaluated. Performance of parallel data mining application on the SAN cluster is examined. In the case of SAN cluster, each node can access all shared disks directly. However, if a lot of nodes access the same shared disk simultaneously, performance of application must degrade due to I/O-bottleneck. A runtime data declustering method, in which data is declustered to several other disks through a SAN during the execution of application, is proposed and evaluated.

The rest of paper is organized as follows. In Section 2, one of data mining applications, called Hash Partitioned Apriori (HPA), and its parallelization are explained. In Section 3, an overview of our SAN-connected PC cluster is shown, and disk-to-disk copy is examined. HPA program is implemented and evaluated on SAN-connected PC cluster, using multiple shared Fibre Channel disks, in Section 4. A runtime data declustering method, which is expected to prevent I/O-bottleneck situation in use of shared disks, is proposed and evaluated in Section 5. Final remarks are made in Section 6.

2 Data mining application and its parallelization

2.1 Mining of association rules

In terms of applications in parallel and distributed computing, data mining has attracted a lot of attention from both research and commercial community. Data mining is a method for the efficient discovery of useful information, such as rules and previously unknown patterns existing among data items in large databases, thus allowing for more effective utilization of existing data. Large transaction processing system logs have been accumulated as a result of the progress of bar-code technology. Such data was just archived and not used efficiently until recently. The advance of microprocessor and secondary storage technologies allow us to analyze vast amount of transaction log data to extract interesting customer behaviors.

One of the best known problems in data mining is mining of association rules from a database, so called "basket analysis"[4][5][6]. Basket type transactions typically consist of a transaction identification and items bought per transaction. An example of an association rule is "if customers buy A and B, then 90% of them also buy C". The best known algorithm for association rule mining is the Apriori algorithm proposed by R. Agrawal of IBM Almaden Research[7]. Apriori first generates so-called candidate itemsets (groups consisting of one or more items), then scans the transaction database to determine whether the candidates have the user-specified minimum support.

In the first pass (pass 1), support for each item is counted by scanning the transaction database, and all items that satisfy the minimum support are picked out. These items are called large 1-itemsets. In the second pass (pass 2), 2-itemsets (pairs of two items) are generated using the large 1-itemsets, which are called candidate 2-itemsets. Support for the candidate 2-itemsets is then counted by scanning the transaction database. The large 2-itemsets that achieve the minimum support are determined. The algorithm goes on to find large 3-itemsets, large 4-itemsets, and so on. This iterative procedure terminates when a large itemset or a candidate itemset becomes empty. Association rules which satisfy user-specified minimum confidence can be derived from these large itemsets.

2.2 Parallelization of association rule mining

In order to improve the quality of the rule, very large amounts of transaction data must be analyzed and this requires considerable computation time. Several parallel algorithms for mining association rules have been previously studied[8], based on Apriori. One of these algorithms, called Hash Partitioned Apriori (HPA), is implemented and evaluated on the PC cluster.

HPA partitions the candidate itemsets among processors using a hash function, like the hash join in relational databases. HPA effectively utilizes the whole memory space of all the processors, and thus, it works well for large scale data mining. The steps of the algorithm are as follows.

1. Generate candidate k-itemsets:

   All processors have all the large $(k-1)$-itemsets in memory when pass k starts. Each processor generates candidate k-itemsets using large $(k-1)$-itemsets, applies a hash function, and determines a destination processor ID. If the ID is the processor's own, the itemset is inserted into the hash table, otherwise it is discarded.

2. Scan the transaction database and count the support value:

   Each processor reads the transaction database from its local disk. It generates k-itemsets from those transactions and applies the same hash function used in phase 1. The processor then determines the destination processor ID and sends the k-itemsets to it.

   When a processor receives these itemsets, it searches the hash table for a match, and increments the match count.

3. Determine large k-itemsets:

   Each processor checks all the itemsets it has and determines large itemsets locally, then broadcasts them to the other processors. When this phase is finished

Table 1. Each node of the PC cluster

CPU	Intel 800MHz Pentium III
Main memory	128Mbytes
IDE hard disk	Quantum Fireball 20Gbytes
SCSI hard disk	Seagate Cheetah 18Gbytes
OS	Solaris 8 for x86
Fast Ethernet NIC	Intel PRO/100+
Fibre Channel NIC	Emulex LP8000 Host Bus Adapter

at all processors, large itemsets are determined globally. The algorithm terminates if no large itemset is obtained.

3 SAN-connected PC cluster and its performance

3.1 Our SAN-connected PC cluster pilot system

Various research projects which develop and examine PC/WS clusters have been reported. Initially, the processing nodes and/or networks were built from customized designs, since it was difficult to achieve good performance using only off-the-shelf products[9][10]. Such systems are interesting as a research prototypes, but most of them failed to be accepted as a common platform. However, because of advances in workstation and network technologies, we can build reasonably high performance WS clusters using off-the-shelf workstations and high speed LANs[11][12].

Several projects on PC clusters were reported[13][14], in which some scientific calculation benchmarks were executed on the cluster. Because performance of PCs and networks used in those projects was not good enough, absolute performance of such clusters was not attractive compared with high-end massively parallel processors. However, preferably good cost/performance has been achieved in these PC clusters[14]. As the performance of PCs has increased dramatically afterward, variety of research projects on PC clusters have been reported until now[15][16][17][18][19]. Previously, we built a large scale ATM-connected PC cluster, and implemented and evaluated several database applications on it[1][2][20].

Recently, we have built a SAN-connected PC cluster pilot system. 32 nodes of 800MHz Pentium III PCs are connected with Fast Ethernet as well as Fibre Channel. Each node consists of the components shown in Table 1.

All 32 PCs of the cluster and 32 SCSI hard disks are connected with a Fibre Channel. Seagate Cheetah 18Gbytes is used as SCSI hard disks, and Brocade SilkWorm 2800 is employed as a Fibre Channel switch. Switching ability of this device is 200MB/s per port. Hitachi Black Diamond 6800, which has 64Gbps switching ability, is used as a Fast Ethernet Switch. This switch has more than enough capacity to connect 32 nodes through Fast Ethernet with non-blocking. An overview of the PC cluster is shown in Figure 1.

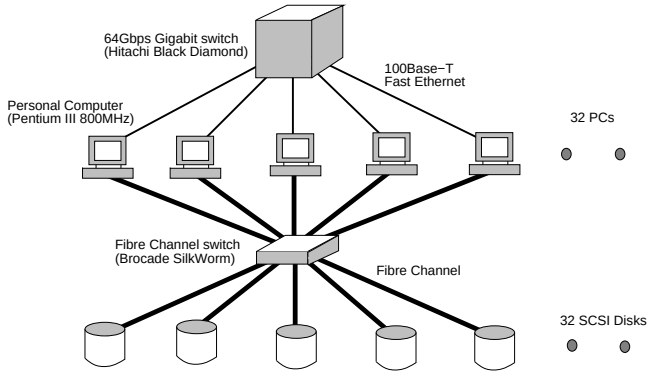

Figure 1. An overview of SAN-connected PC cluster pilot system

3.2 Disk-to-disk copy performance of the system

Disk-to-disk copy performance is measured on the PC cluster pilot system described in the previous subsection. The following two cases of disk copies are compared in this experiment:

In the first case, data is copied from one disk to the other just like LAN cluster. That is to say, the source node reads data from a hard disk, then sends it through Fast Ethernet LAN to the destination node, which receives the data and writes it to its own hard disk. In this case, although the hard disks are accessed through Fibre Channel, each node uses one of shared disks exclusively just like its local disk.

In the second case, one node accesses both the source disk and the destination disk through Fibre Channel, and copies the data directly by oneself. (This is so-called "LAN-free" copy.)

The result of performance at the source and the destination nodes are shown in Table 2. The volume of copied data is 100Mbytes, and the block size of LAN transfer is 8Kbytes. As shown in Table 2, Fast Ethernet LAN is occupied by the transferred data in the first case. Moreover, CPU usages are relatively high in this experiment, especially at the destination node. These features are not preferable for a PC cluster used as a large scale server system. In the second case, on the other hand, the data is copied only through the source node, so that neither other node nor LAN is bothered by the copy traffic. This mechanism is suitable for a PC cluster, which saves bandwidth of LANs and CPU power for other purposes.

Table 2. Disk-to-disk copy performance of the PC cluster

Case1: Copy through Fast Ethernet LAN

Node	Source	Destination
CPU(Sys)	20%	40%
LAN(Send)	90Mbps	10Mbps
LAN(Receive)	10Mbps	90Mbps
I/O(Read)	10MB/sec	0
I/O(Write)	0	10MB/sec

Case2: Copy through Fibre Channel

Node	Source	Destination
CPU(Sys)	20%	–
LAN(Send)	0	–
LAN(Receive)	0	–
I/O(Read)	10MB/sec	–
I/O(Write)	10MB/sec	–

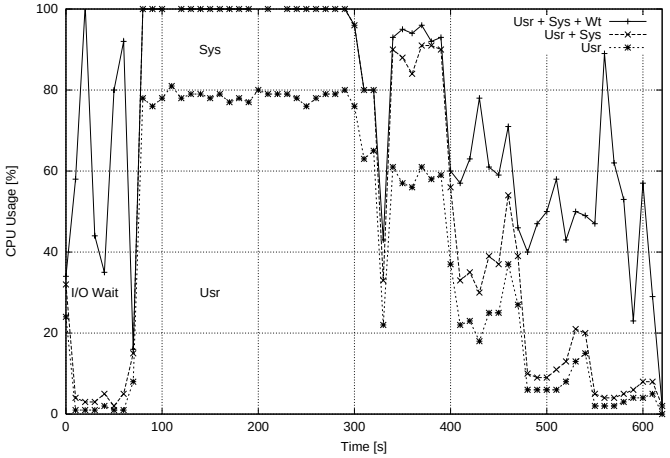

Figure 2. CPU usage (using 1 Disk)

4 Execution of data mining application on SAN-connected PC cluster

4.1 Implementation and execution of HPA program

The HPA program explained in Section 2 is implemented on our SAN-connected PC cluster pilot system. Transaction data is produced using data generation program developed by Agrawal[7], designating some parameters, such as the number of transaction, the number of different items, and so on. The produced data is stored at one of SCSI FC hard disks, which is shared by all PCs through Fibre Channel. During execution of the application, this data is accessed by all processes concurrently. Each node of the cluster accesses its own portion of the data, which is assigned to each node dividing the data by the number of application execution nodes almost equally.

Solaris socket library is used for the inter-process communication. As a type of socket connection, SOCK_STREAM is used, which is two-way connection based byte stream. All processes are connected with each other, thus forming mesh topology.

In this experiment, the number of transaction is 10,000,000, the number of different items is 5,000, and the minimum support is 0.7%. The size of the transaction data is about 800Mbytes in total. The message block size is 8Kbytes, and the disk I/O block size is 64Kbytes in this experiment. The number of nodes used in this application is eight. The result of HPA is shown in Table 3.

Using above parameters, the execution of HPA program iterates until pass 6. It is known that the number of candidate itemsets in pass 2 is very much larger than that in other passes. This often happens in association rule mining.

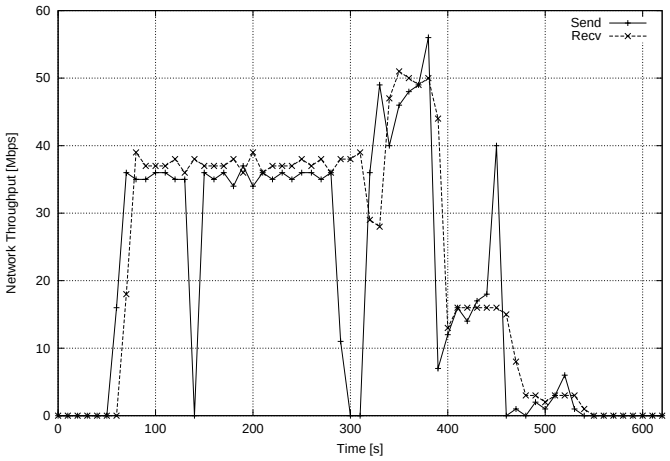

Figure 3. LAN Throughput (using 1 Disk)

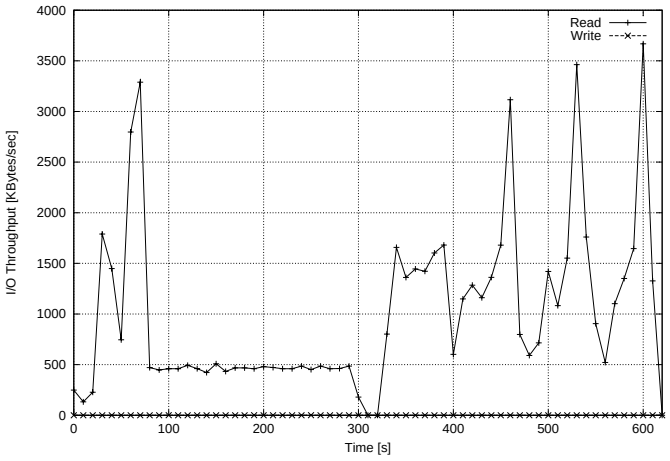

Figure 4. I/O Throughput (using 1 Disk)

Table 3. The number of itemsets and the execution time at each pass

C	Number of candidate itemsets
L	Number of large itemsets
T	Execution time of each pass [sec]

pass	C	L	T
pass 1	–	1043	64.8
pass 2	543405	504	253.1
pass 3	395	315	74.6
pass 4	172	109	73.3
pass 5	30	29	66.1
pass 6	4	2	66.2

The details of system performance are measured and analyzed to investigate the behavior of HPA program executed on the SAN-connected PC cluster. CPU usage, LAN throughput, and I/O throughput of one node of the cluster are shown in Figure $2-4$. These three figures are results measured on the same node. They indicate that performance is bounded by different factors from phase to phase during the execution of HPA program. According to Figure 2, Performance is bounded by CPU in pass 2. In pass 2, not only the user mode but also the system mode accounts a considerable part of CPU usage, which is supposed to be consumed for network operation, as shown in Figure 3. In other passes, on the other hand, performance is bounded mostly by I/O operation instead of CPU, as shown in Figure 2 and Figure 4. The characteristic of pass 3 is somewhat between CPU-bound and I/O-bound conditions.

4.2 Using multiple shared Fibre Channel disks

The influence of access conflicts at a shared disk is investigated in the next experiment. Transaction data is divided manually and copied to several disks before the execution of the program. This reduces the probability of read access conflicts at the shared disk.

The number of disks used in the experiment is two and four. The contents of data are divided by two/four almost equally, and copied to SCSI FC disks manually before the execution of HPA program. During the execution, each node reads its own portion from a designated disk. For example, four nodes access the same disk in the case of two FC disks being used.

In Figure 5, execution time of HPA program is shown, when one $-$ four disks are used. According to the result, the execution time becomes shorter when the number of disks changes from one to four. As the number of disks increases, disk accesses are distributed so that less read access conflicts could happen. As shown in this figure, the execution time becomes shorter in pass 1 and passes $3-6$. On the other hand, the execution time of pass 2 is almost equal in all cases. This is because pass 2 is a CPU-bound condition rather than a I/O-bound condition, as mentioned in the previous subsection.

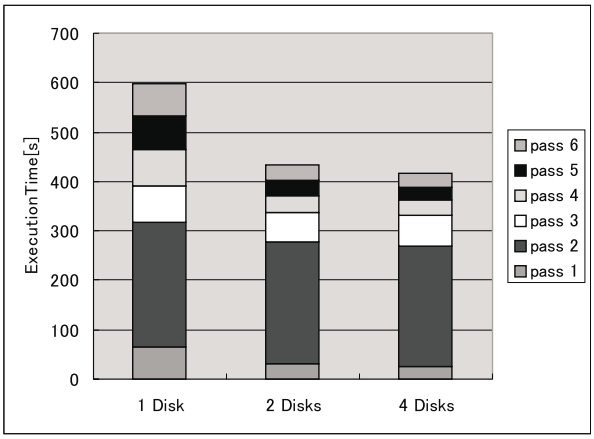

Figure 5. Execution time of HPA program using $1-4$ Disks

5 An evaluation of runtime data declustering on SAN-connected PC cluster

5.1 Runtime data declustering method

In SAN-connected PC cluster, each node can access all shared disks through Storage Area Network. Thus, an application on each node does not have to care where data is actually located; in a local disk or remote. However, when a considerable number of nodes access to the same shared disk simultaneously, application performance must degrade due to I/O-bottleneck.

If stored data is accessed repeatedly during the execution of application, the probability of access conflict on the shared disk increases. In such a case, it is preferable to decluster data from one disk to others during/after first access to the data. That is to say, each node copies portion of the data that will be accessed again afterward, from one shared disk to another which can be used exclusively. The copied portion of the data, instead of the original one, is accessed after the declustering is completed. We call this method runtime data declustering in the rest of this paper. In some applications, it is difficult to decluster the data before execution of program. Using this method, it is possible to decide dynamically which node needs which portion of the data.

Because SAN traffic does not interfere with each other when target disks are different, application performance should not degrade due to I/O-bottleneck after the data is declustered to several disks. Even in this method,

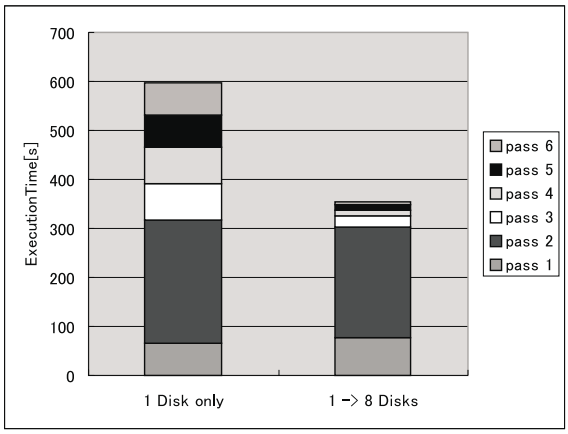

Figure 6. Execution time of HPA program (Runtime declustering from 1 to 8 disks)

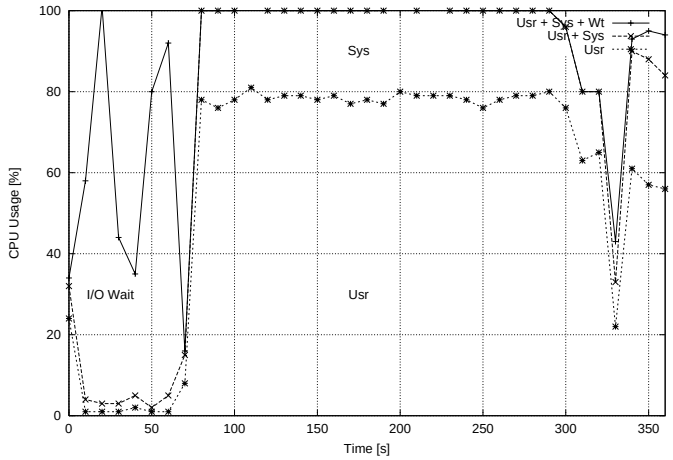

Figure 7. CPU usage (Runtime declustering from 1 to 8 disks)

first accesses to the shared disk may still conflict. In addition, copy operation to another disk is required. However, this seems to degrade performance little, because the data is already read to each node during the first access, and write operation does not conflict on SAN.

5.2 An evaluation of runtime data declustering on SAN cluster

According to Figure 2 – 4, we have found pass 2 of HPA program using one shared disk is a CPU-bound condition, but the CPU load is not high in other passes. In those passes, accesses to shared disks become bottleneck.

Runtime data declustering method proposed in previous subsection can be applied in this situation. Because each node can access all shared disks in the storage pool, it is possible to copy its own portion of the transaction data to another disk, which is used exclusively. In this method, portion of the data is copied to their own disks (eight disks in total) when the data is read in pass 1, then the copied data, instead of the original one, is accessed afterward.

In Figure 6, the execution time of the proposed runtime data declustering method are shown. The proposed method is compared with the original one in which data is read only from the same shared disk repeatedly.

As shown in the figure, the execution times in pass 1 and pass 2 are almost equal in both methods. In pass 1, this is because data must be read from a shared disk in both methods. After pass 1, the data is declustered to other disks which is accessed exclusively. However, because pass 2 is not a I/O-bound condition, the execution times are almost the same in both methods. In pass 3 – 6, on the other hand, the execution time of runtime data declustering method becomes extremely shorter than that of original method. This is because I/O-bottleneck is resolved by runtime data declustering.

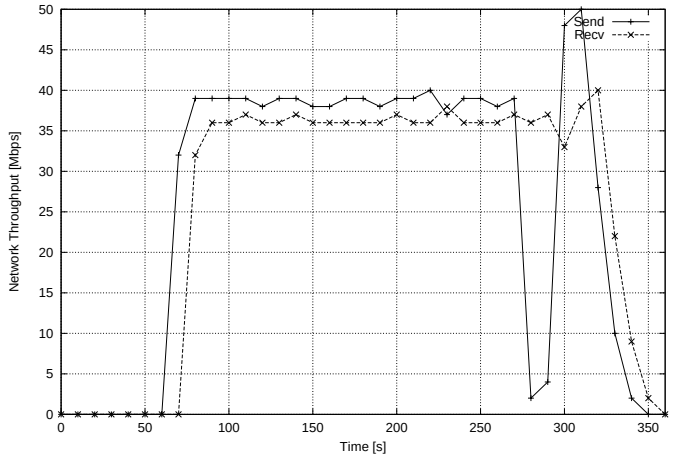

Figure 8. LAN Throughput (Runtime declustering from 1 to 8 disks)

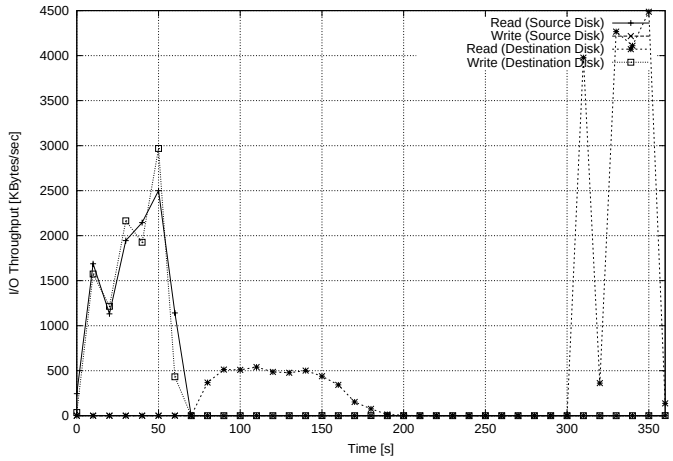

Figure 9. I/O Throughput (Runtime declustering from 1 to 8 disks)

The detailed behavior of the system is also investigated. CPU usage, LAN throughput, and I/O throughput of one node of the cluster are shown in Figure 7 – 9. These three figures are results measured on the same node.

As shown in Figure 9, the portion of transaction data is declustered from source disk to destination disk in pass 1. The data is declustered only through Fibre Channel, therefore almost no LAN traffic is observed in pass 1 as shown in Figure 8. According to Figure 7, CPU is mostly in the state of I/O wait in this pass.

In pass 2, the profiles of the figures are almost the same with those of Figure 2 – 4. This means data declustering does not affect the system performance because pass 2 is bounded by CPU instead of I/O.

However, runtime data declustering resolves I/O-bottleneck in pass 3 – 6. CPU has almost no I/O wait time in these passes as shown in Figure 7. As a result, execution time of HPA program becomes extremely shorter in these passes.

5.3 Execution and evaluation of runtime data declustering using 16 nodes

Next, HPA program is executed using 16 nodes to compare with the cases using 8 nodes. In this experiment, the number of transaction data is 20,000,000. Other parameters are the same with those used in the previous experiments, i.e., the number of different items is 5,000, and the minimum support is 0.7%, and so on.

The transaction data is stored in one FC disk, which is shared by 16 nodes of the SAN-connected PC cluster. In the first experiment, the data is accessed by 16 nodes concurrently and repeatedly in all passes during the execution of HPA program. In the second experiment, runtime data declustering method is applied. That is to say, shared FC disk is accessed by all 16 nodes in pass 1, then the data is declustered to 16 FC disks in this pass, and the copied portion of data is accessed afterward. The two methods are compared in Figure 10.

In this case also, the execution times in pass 1 and pass 2 are almost equal in two methods. The reason should be the same; the shared disk must be accessed in pass 1 anyway, and pass 2 is not I/O-bound but CPU-bound. And in pass 3 – pass 6, the execution time of runtime data declustering method becomes shorter than that of original method, because I/O-bottleneck is resolved.

Total performance improvement by the proposed method is larger than that of the 8 nodes case shown in Figure 6. In 16 nodes experiment, the ratio of execution time in pass 2 against the total execution time is rather small compared with the 8 nodes case. As the number of application execution nodes increases, execution time in pass 2 becomes shorter because this pass is bounded by CPU, but performance does not improve in other passes because they are I/O-bound. According to this result, the proposed method becomes more important when the number of nodes increases in SAN cluster.

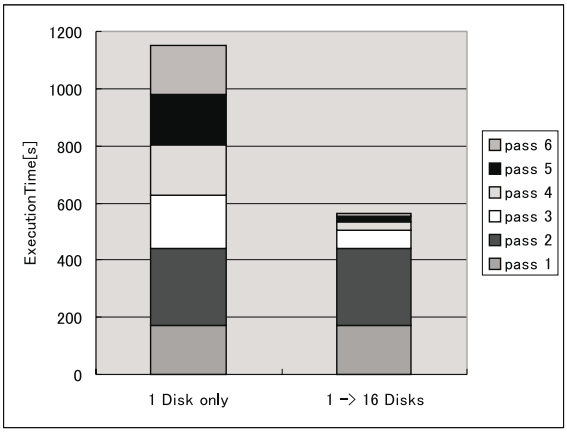

Figure 10. Execution time of HPA program (Runtime declustering from 1 to 16 disks)

6 Conclusion

In this paper, a PC cluster connected with Storage Area Network is built and evaluated. SAN-connected PC clusters are suitable for a large scale server site because data transfer between disks does not depend on a LAN, thus bandwidth of a network as well as CPU load can be saved. This feature is shown in this paper by a simple data transfer experiment.

We have implemented and evaluated a data mining application on our SAN-connected PC cluster pilot system. In this application, transaction data is scanned repeatedly in iterative passes. Therefore, a runtime data declustering method, in which data is declustered from a shared disk to other disks during the execution, is considered to be effective. As a result of the experiment evaluated on the SAN cluster, the execution time of each pass becomes shorter, after the declustering is completed in the first pass. The proposed runtime data declustering method achieves better performance especially when the number of nodes used in the cluster is large, because this method is effective for I/O-bottleneck problem.

While shared disks of a SAN cluster are quite useful for the parallel/distributed computing, disks might be scanned repeatedly in some data-intensive applications, which degrades execution performance severely. The runtime data declustering method achieves better performance in such cases.

Acknowledgment

This project is partly supported by the Japan Society for the Promotion of Science (JSPS) RFTF Program and New Energy and Industrial Technology Development Organization

(NEDO). We would like to thank Tokyo Electron Ltd. for technical help with Fibre Channel-related issues. Hitachi, Ltd. gave us extensive technical help with Gigabit switch.

References

[1] T. Tamura, M. Oguchi, and M. Kitsuregawa: "Parallel Database Processing on a 100 Node PC Cluster: Cases for Decision Support Query Processing and Data Mining", *Proceedings of SC97: High Performance Networking and Computing (SuperComputing '97)*, November 1997.

[2] M. Oguchi, T. Shintani, T. Tamura, and M. Kitsuregawa: "Optimizing Protocol Parameters to Large Scale PC Cluster and Evaluation of its Effectiveness with Parallel Data Mining", *Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing*, pp.34-41, July 1998.

[3] B. Phillips: "Have Storage Area Networks Come of Age?", *IEEE Computer*, Vol.31, No.7, pp.10-12, July 1998.

[4] U. M. Fayyad, G. P. Shapiro, P. Smyth, and R. Uthurusamy: "Advances in Knowledge Discovery and Data Mining", *The MIT Press*, 1996.

[5] V. Ganti, J. Gehrke, and R. Ramakrishnan: "Mining Very Large Databases", *IEEE Computer*, Vol.32, No.8, pp.38-45, August 1999.

[6] M. J. Zaki: "Parallel and Distributed Association Mining: A Survey", *IEEE Concurrency*, Vol.7, No.4, pp.14-25, 1999.

[7] R. Agrawal and R. Srikant: "Fast Algorithms for Mining Association Rules", *Proceedings of the Twentieth International Conference on Very Large Data Bases*, pp.487-499, September 1994.

[8] T. Shintani and M. Kitsuregawa: "Hash Based Parallel Algorithms for Mining Association Rules", *Proceedings of the Fourth IEEE International Conference on Parallel and Distributed Information Systems*, pp.19-30, December 1996.

[9] R. S. Nikhil, G. M. Papadopoulos, and Arvind: "*T: A Multithreaded Massively Parallel Architecture", *Nineteenth International Symposium on Computer Architecture*, pp.156-167, May 1992.

[10] M. Blumrich, K. Li, R. Alpert, C. Dubnicki, E. Felten, and J. Sandberg: "Virtual Memory Mapped Network Interface for the SHRIMP Multicomputer", *Proceedings of the Twenty-First International Symposium on Computer Architecture*, pp.142-153, April 1994.

[11] C. Huang and P. K. McKinley: "Communication Issues in Parallel Computing Across ATM Networks", *IEEE Parallel and Distributed Technology*, Vol.2, No.4, pp.73-86, Winter 1994.

[12] D. E. Culler, A. A. Dusseau, R. A. Dusseau, B. Chun, S. Lumetta, A. Mainwaring, R. Martin, C. Yoshikawa, and F. Wong: "Parallel Computing on the Berkeley NOW", *Proceedings of the 1997 Joint Symposium on Parallel Processing(JSPP '97)*, pp.237-247, May 1997.

[13] T. Sterling, D. Saverese, D. J. Becker, B. Fryxell, and K. Olson: "Communication Overhead for Space Science Applications on the Beowulf Parallel Workstation", *Proceedings of the Fourth IEEE International Symposium on High Performance Distributed Computing*, pp.23-30, August 1995.

[14] R. Carter and J. Laroco: "Commodity Clusters: Performance Comparison Between PC's and Workstations", *Proceedings of the Fifth IEEE International Symposium on High Performance Distributed Computing*, pp.292-304, August 1996.

[15] A. Barak and O. La'adan: "Performance of the MOSIX Parallel System for a Cluster of PC's", *Proceedings of the HPCN Europe 1997*, pp.624-635, April 1997.

[16] H. Tezuka, A. Hori, Y. Ishikawa, and M. Sato: "PM: An Operating System Coordinated High Performance Communication Library", *Proceedings of the HPCN Europe 1997*, pp.708-717, April 1997.

[17] M. Oguchi, T. Shintani, T. Tamura, and Masaru Kitsuregawa: "Characteristics of a Parallel Data Mining Application Implemented on an ATM Connected PC Cluster", *Proceedings of the HPCN Europe 1997*, pp.303-317, April 1997.

[18] Y. Ishikawa, A. Hori, H. Tezuka, S. Sumimoto, T. Takahashi, F. O'Carroll, and H. Harada: "RWC PC Cluster II and SCore Cluster System Software – High Performance Linux Cluster", *Proceedings of the Fifth Annual Linux Expo*, pp.55-62, 1999.

[19] M. Banikazemi, V. Moorthy, L. Herger, D. K. Panda, and B. Abali: "Efficient Virtual Interface Architecture (VIA) Support for the IBM SP Switch-Connected NT Clusters", *Proceedings of the International Parallel and Distributed Processing Symposium*, pp.33-42, May 2000.

[20] M. Oguchi and M. Kitsuregawa: "Dynamic Remote Memory Acquisition for Parallel Data Mining on ATM-Connected PC Cluster", *Proceedings of the Thirteenth ACM International Conference on Supercomputing*, pp.246-252, June 1999.