DIAPRISM Hardware Sorter
— Sort a Million Records Within a Second —

Shinsuke Azuma, Takao Sakuma, Tetsuya Takeo, Takaaki Ando, Kenji Shirai

Mitsubishi Electric Corp.

Mitsubishi Electric Corp. has released the DIAPRISM hardware sorter for OLAP (On-Line Analytical Processing) and data warehouse applications. It enables high-speed sorting on a commercial PC server. The sorter employs a pipeline merge sort algorithm. It consists of multiple sort processors connected linearly and necessary amount of memory for each processor. The architecture of the sort processor is capable of sorting 4GB of data at one time. The sorter has achieved Datamation benchmark record, sorting a million 100-byte records in 0.998 second.

1 Introduction

Sorting is one of the most fundamental operations in databases and data warehouses. High-speed sorting is especially indispensable in integrating data warehouses from legacy databases. Sorting consists of key comparisons and data movements. Data movements usually occupy most of the execution time. PC systems in general are inexpensive but very weak for large amount of data movement. Our approach is off-loading such memory intensive operation onto a special sorting board from a commodity PC. The hardware sorter employs multi-bank memory organization that considerably improves the memory throughput necessary for sort operation. By installing the hardware sorter in a PCI slot, high-speed sorting can be achieved on inexpensive systems.

2 Algorithm

Pipeline merge sort algorithm is capable of sorting in $O(N)$ time. That is, a hardware sorter can sort disk-resident data keeping up with the data stream from the disk. Figure 1 shows the configuration of the hardware sorter, where S_i and M_i indicate a sort processor and its local memory respectively. To generalize the explanation, a k-way merge sort is assumed below. S_1 inputs a series of records and outputs them in ascending order every k records. Partially sorted records are collectively called a "string". S_2 inputs S_1's output, merges k strings and generates a series of strings each of which consists of sorted k^2 records. In general, S_i inputs a series of strings composed of k^{i-1} records, and outputs a series of strings composed of k^i records. When merging k strings, S_i stores the first k-1 strings in M_i and merges them with the incoming kth string. This indicates that M_{i+1} needs k times the capacity of M_i. All processors behave in a pipeline fashion. Figure 2 illustrates this sorting process in 2-way case, where each number corresponds to a record and a rectangle indicates a string.

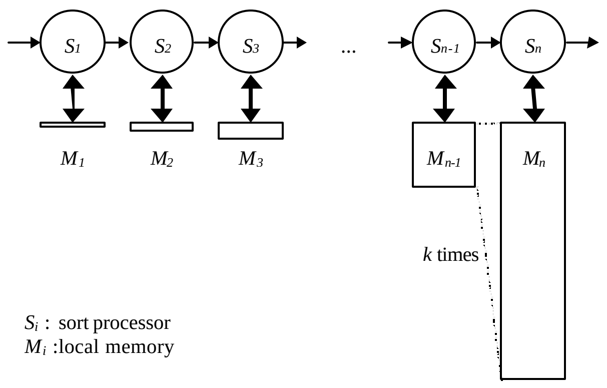

Figure 1: Configuration of k-way Hardware Sorter

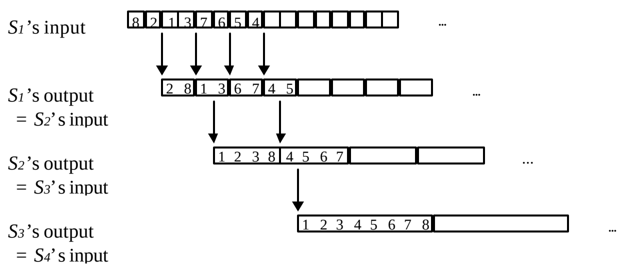
S_1's input
8 2 1 3 7 6 5 4
S_1's output = S_2's input
2 8 1 3 6 7 4 5
S_2's output = S_3's input
1 2 3 8 4 5 6 7
S_3's output = S_4's input
1 2 3 4 5 6 7 8

Figure 2: Process of 2-way Pipeline Merge Sort

3 Architecture of DIAPRISM Hardware Sorter

The sort processor[1] adopts an 8-way merge sort algorithm in consideration of its design complexity and the number of sort processors and DRAMs to build a sorter.

The memory utilization efficiency of the naive merge sort algorithm deteriorates if the length of the incoming records is different from the length presumed in hardware. We have developed the algorithm called String Length Tuning (SLT) to resolve this problem. Each sort processor dynamically chooses the number of merge ways from two to eight or bypasses a string in order to fully utilize its memory. Figure 3 shows the effectiveness of SLT. With SLT, the sort system achieves a flat memory utilization rate and consequently it can sort a constant amount of data regardless of the record length.

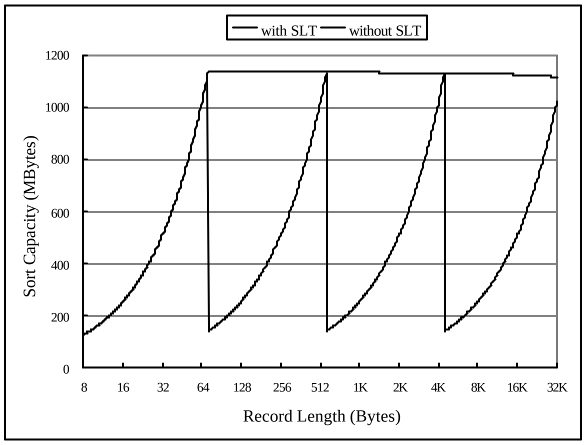

Figure 3: Effectiveness of SLT

In our former product[2], memory was accessed record-by-record by a sort processor. In the current product, in order to further improve the performance, a sort processor adopts page mode access to its local memory.

The sort processor can transfer 4-bytes of data at each clock cycle between adjacent processors and 1-page (256-bytes) of data in 32-clock cycles between a processor and its local memory. The processor runs at 70MHz. Figure 4 shows the sort processor's data path diagram.

The DIAPRISM hardware sorter board has eight sort processors and one 64-bit 33-MHz PCI interface. As 1GB memory is connected to the eighth sort processor, the sorter can handle up to 16 million (=8^8) records or 1.1GB (? 1GB / 7 ? 8) data.

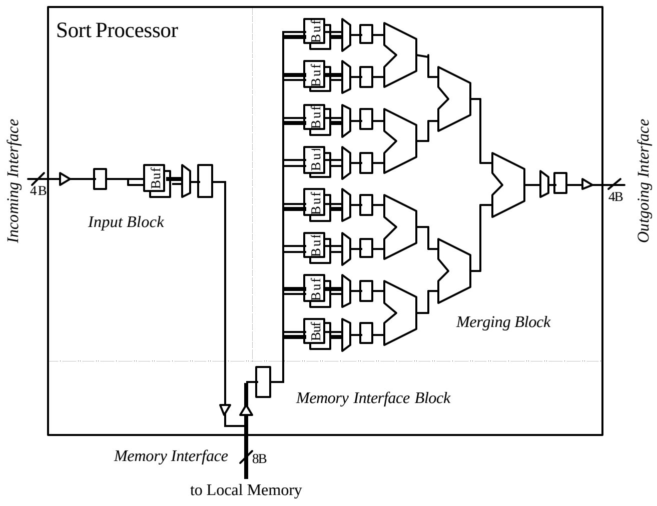

Figure 4: Sort Processor's Data Path Diagram

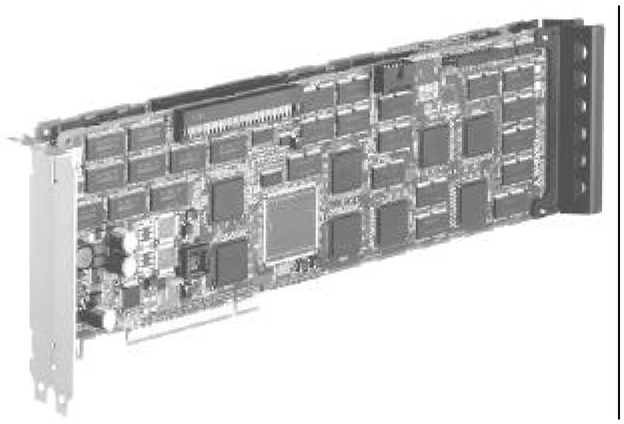

Figure 5: DIAPRISM Hardware Sorter

4 Performance Evaluation

Our machine is a PC server with the hardware sorter in a PCI slot. Table 1 shows the machine configuration and Table 2 shows the storage configuration. Table 1 also shows approximate price of each component. The machine has three PCIs, two of them are 32-bit and the other is 64-bit. We are using four Adaptec 2-channel Ultra2 SCSI controllers and four Seagate disks[1] on each channel for data (32 disks in total).

The operating system is Windows NT Server 4.0. We utilize the stripe set feature of ftdisk driver to enlarge disk I/O. The input file is laid out on a stripe set of 28 disks and the output file is laid out on a stripe set of 32 disks. The file system is NTFS.

Table 1: Machine Configuration

Machine	HP NetServer LXr8000	$25,000
CPU	Pentium III Xeon 550MHz ? 4	
Main Memory	1GB	
OS	Windows NT Server 4.0	
Storage	Eight Ultra2 SCSI channels Four disks on each SCSI channel for data (32 disks for data in total) One disk on another SCSI channel for OS	$27,000
DIAPRISM Hardware Sorter		$28,000
Total Price		$80,000

Table 2: Storage Configuration

PCI	SCSI Controller	Channel	Disk	Remarks
#0 (32-bit)	LSI Logic SYM53C896	–	Quantum AtlasII ? 1	for OS
	Adaptec 3950U2	A	Seagate Cheetah 9LP ? 4	
		B	Seagate Cheetah 9LP ? 4	
#1 (32-bit)	Adaptec 3950U2	A	Seagate Cheetah 9LP ? 4	
		B	Seagate Cheetah 9LP ? 4	
#2 (64-bit)	Adaptec 3950U2	A	Seagate Cheetah 18LP ? 4	3 for input
		B	Seagate Cheetah 18XL ? 4	3 for input
	Adaptec 3950U2	A	Seagate Cheetah 9LP ? 4	3 for input
		B	Seagate Cheetah 18XL ? 4	3 for input

We have our hardware sorter[2] on PCI #2 (64-bit). There are two steps to sort on our system:

Step-1: Read the input file, construct the records to key-and-pointer format, and put the data to the sorter

Step-2: Get the sorted data from the sorter, reform the records, and write the output file

The Datamation benchmark[3] defines the sort performance of a certain system as elapsed time to sort a million 100-byte records each of which has a 10-byte key. The input records are in random order and output records must be in ascending order. The elapsed time includes the time create the sort process, to read the input from disk and to create the output file and write the output to disk. The 1998's world record was 2.41 seconds by NOW-Sort[4] with a 32-node cluster of UltraSPARCs and the 1999's world record was 1.18 seconds by Millennium Sort[5] with a 16-node cluster of PentiumII PCs.

[1] Some of the disks (Cheetah 18XL) support Ultra160 SCSI but they function as Ultra2 because the SCSI controllers do not support Ultra160.

[2] We are using a prototype of hardware sorter, not a product.

Table 3 shows the elapsed time to sort 100-byte records. The elapse time is measured by means of high-resolution performance counter provided by Win32 API. It is possible to time at CPU clock frequency (550MHz) by the counter. The difference of the counter values before a sort process starts and after the process ends is used to calculate the elapsed time. In the case of a million records, the elapsed time is 0.998 second, which is the record of the Datamation benchmark. It also shows the elapsed time of each step and the CPU time. Only sort keys are copied in main memory to construct the records to key-and-pointer format in Step-1, while entire records are copied to reform the records in Step-2. Table 4 shows that our system is superior also from the price-performance point of view.

Table 3: Elapsed Time and CPU Time to Sort 100-byte Records

Record Count	Step-1 (sec)	Step-2 (sec)	Total Elapse (sec)	CPU Time (sec)
1 million	0.378	0.620	0.998	2.296
2 million	0.766	1.232	1.998	4.937
5 million	1.844	3.124	4.968	12.828
10 million	3.670	6.321	9.991	25.281

Table 4: Datamation Results

Year	System	Datamation (sec)	Price ($)
1998	NOW-Sort (32-node UltraSPARCs)	2.41	576,000 (= 18,000 ? 32 nodes)
1999	Millennium Sort (16-node PentiumII PCs)	1.18	92,800 (= 5,800 ? 16 nodes)
2000	DIAPRISM Hardware Sorter with PentiumIII Xeon Server	0.998	80,000

References

[1] Shinsuke Azuma, Takao Sakuma, Takashi Nakano, Takaaki Ando, Kenji Shirai, "High-Performance Sort Chip", Hot Chips 11, August, 1999.

[2] Shinya Fushimi, Masaru Kitsuregawa, "GREO: A Commercial Database Processor Based on A Pipelined Hardware Sorter", ACM SIGMOD '93, 1993.

[3] Anon. et al. "A Measure of Transaction Processing Power", Datamation, Vol. 31(7), pp.112-118, 1985.

[4] Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau, David E. Culler, Joseph M. Hellerstein, David A. Patterson, "High-Performance Sorting on Networks of Workstations", ACM SIGMOD '97, May, 1997.

[5] Philip Buonadonna, Joshua Coates, Spencer Low, David E. Culler, "Millennium Sort: A Cluster-Based Application for Windows NT using DCOM, River Primitives and the Virtual Interface Architecture", Proceedings of the 3rd USENIX Windows NT Symposium, July, 1999.