World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:4, No:2, 2010

Optimal External Merge Sorting Algorithm with Smart Block Merging

Mir Hadi Seyedafsari, and Iraj Hasanzadeh

***Abstract*—**Like other external sorting algorithms, the presented algorithm is a two step algorithm including internal and external steps. The first part of the algorithm is like the other similar algorithms but second part of that is including a new easy implementing method which has reduced the vast number of input-output operations saliently. As decreasing processor operating time does not have any effect on main algorithm speed, any improvement in it should be done through decreasing the number of input-output operations. This paper propose an easy algorithm for choose the correct record location of the final list. This decreases the time complexity and makes the algorithm faster.

***Keywords*—**External sorting algorithm, internal sorting algorithm, fast sorting, robust algorithm.

I. Introduction

As we know one of the main problem about information and data is the need for sorting it . This necessity has been lead to creation of many algorithms with different time and memory specifications. But when we are going to sort huge number of data traditional algorithms could not deal with these kinds of problems because the lack of enough memory in compare with data capacity which is being processed. In addition as these algorithms were designed just to operate on main memory an external memory was required to deal with these problems[2][6][7]. Magnetic disks were the best choice between all other external mem and they had some specifications in compare with internal mem including:

A. They had more capacity for saving data than internal ones.
B. Data was accessible more easily than it was on internal mem

3- Because of the disk driver mechanism, random accessing to bits on disk space was taking more time than that on internal mem.

But this time spent was shorter than the time was spent on transferring data to main mem so as the blocks on disk were made bigger the accessing time was made shorter and the operations were done faster.

So, as it was told, some new algorithms should have been created to match with internal and external mem.

Mir Hadi Seyedafsari Author is with the Computer Engineering Department, Islamic Azad University of KHOY Branch, Iran (e-mail:hadi.1357@hotmail.com).
Iraj Hasanzadeh Author is with the Computer Engineering Department, Tabriz University, Iran (e-mail: Izadeh@tabrizu.ac.ir).

II. Review of External Sorting Algorithms

Now, before going on more detailing, some definitions should be considered as follow:
N=number of records on disk that is related to file
B=number of records in each block of disk
M=number of records that could be stored on main mem or internal mem at a moment (M<N)

External sorting algorithm has two main parts:

1. Reading M elements of internal mem, sorting the data and then writing each sorted block of data on external mem.
2. Merging all the sorted blocks to create only a large block in external memory.

Before of going on more details, some notes are as follow:
N/M= the number of executions in the first step or internal phase
N/B=n= the number of existing blocks in an external memory
M/B=m=the number of blocks in main memory.

First stage

Until reading all the records from file, the following steps will be repeated:

A. reading the m records from disck(external memory) to main mem
B. sorting each record of main mem by using one of the internal sorting algorithm
C. writing the sorted files into a new file in the external mem

Second stage:

Until reading all the sorted files- mentioned in the last step of previous part (3) - the following steps will be repeated:

1. Reading executions from new file to main mem
2. Merging all the existing executions due to the smallest number of these sorted executions and producing larger executions until one last large sorted list which is our final list is produced[5][8][9].

In this algorithm there are two kinds of time cost unlike the other internal sorting algorithms in order to evaluate the time complexity:
-The time cost of sorting data of the internal mem by using of an internal sorting algorithm.

The required time cost of reading and writing the information from and onto disk –external mem-or equally the input –output cost which includes the number of reading and

writing operation from external mem to internal mem and vise versa.

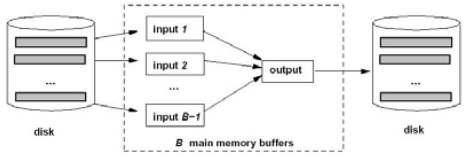

Fig. 1 General state of the external sorting

III. Review of Commonly used External Sorting Algorithms

One of the most commonly used algorithms, is the 2-way merge sort algorithm. After creating small sorted executions in the first step, the algorithm in the second step will merge these sorted executions two by two to prepare an exact list of sorted data. All the records of original file are fetched to mem and written to a new destination file so the number of the input – output operation will be T1 = 2n = 2N/B at the second stage after one pass of the main loop[1][10] , all the records of destination file are read and the same number of records written to original file so again the number of input-output operation (will be)is equal T1.Now it may be asked that how many iteration are there in the second stage?

Each iteration in the second stage will reduce the number of execution fifty percent. After [N/M] number of executions there will be one last execution so for each input – output operation there will be Log 2(n/m) iteration and T2 could be written as follow:

$$T2= 2n \text{ Log } 2(n/m)$$

There is no difference between M -way and 2-way merge sort algorithm except that former method applies an m-way merge instead of 2-way in the second step of the algorithm and the first step of both of them remains unchanged so T2 is as it is comes bellow:

$$T2= 2n \text{ Log } M (n/m)$$

The third kind of algorithm, Multiway merge sort algorithm, is similar to the two above in its first stage, but in the main loop of its second stage all the [N/M] executions will be merged at once. This will be performed by using [N/M] buffer each with a single element then the following is the time cost of algorithm [1] [4] [14]:

$$T2=2n$$

Its main drawback is that it needs a vast volume of main mem equal to whole input data so it couldn't be realized (implemented)[11][13].some methods which could be applied to improve the algorithm are being listed as follow:

1. to increase the main mem buffers
2. sequential access instead of random access
3. to decrease the time delay of input-output requests
4. to create the larger initial execution in the first step (replacement sort algorithm)

Comparing to other algorithms, a new method has been developed for external phase of new algorithm so it reduces the total number of input-output operations to 2N/B meaning that each of reading and writing operations will be include of 2N/B numbers of operations separately.

Like the other external sorting algorithm, in the first step of new algorithm a stable internal sorting algorithm has been used to sort the existing records in mem and producing executions. In the second step a priority queue has been implemented aimed to decrease the input-output operation in an order that the executed information in the first step will be buffered in this queue and used to do the best merging operation.

Generally this algorithm will read each record for twice, the once for creating executions and the other once for merging them. Also each record will be written for twice which will reduce the input – output operation complexity. An important issue of this algorithm is to select part of the input file appropriately to place it in the main mem. Records will be stayed in the main mem until the right output place is determined. In traditional algorithms some executions of the input file were being read and merged to produce output executions but merging in the new algorithm will be done with the least input-output operation by using the queue priority defined in the exterior phase. The m-way sort algorithm could be similar to the new one in aspect of input-output operation as long as m is equal to N/M but it will cause the m-way sort algorithm to be impractical.

IV. Proposed Algorithm

The new algorithm has two internal and external phases.

- Internal phase

In this part the input file will be divided into sub files which they could be stored in the main mem and then sorted by using an internal sorting algorithm.

Each sorted sub file is called an execution and the total number of executions will equal to N/M in this step. After producing i'th executions, the smallest element of the each block along with its pointer will be inserted in a priority queue which is built in this period. The number of input –output operation could be reduced by priority queue

-external phase

Due to the queue priority data, the algorithm determines the order of blocks which are read from input file. The data block with least priority is fetched to main mem Then the algorithm will consider the least record of the block as a reference point to compare it with other records of mem . mem records which are smaller than the reference ,could be written on the external mem before the reference record.

After selection of each block of queue priority, the associated pointer will be deleted from the queue until removing all the queue elements. In order to keeping and merging the records, the buffers B1 and B2 will be used after steps mentioned above.

At first the least block of queue priority is fetched to B2 while the B1 is empty.

Then B1 and B2 will be merged together and written to B1.After that, the second block will be fetched to B2 and the least record of B2 is considered as a reference point. If the number of elements in B1 which are smaller than the

reference point is equal or bigger than the number of records in each block, they will be written to external mem which will result in reducing of buffer (B1) length and the other elements of B1 will merge with B2 and be written to B1.

This process will continue until all the blocks of priority queue is being fetched and at the end if the B1 has records it will be added to the end of output file in the external mem.

New Algorithm (x,y) routine

/Internal phase /

X and Y are input and output files respectively.

Begin

1. $For(i = 1 to\ N)$

Put *M* number of the next records of file *x* into the B_1 buffer.

2. $Disk\ \mathrm{Re}ad(B_1, X, M)$
3. $Internal\ Sort(B_1)\ to\ produce\ run(i) \Rightarrow$
4. $Insert\ the\ smallest\ \mathrm{Re}cord\ of\ M / B\ blocks\ in\ run(i)\ to\ Qi\ (priority \Rightarrow Queue)$
5. $Disk\ write(B_1, y) \Rightarrow$
6. $End\ \ for$
7. $Clear\ (B_1)$

/Termination of internal phase/

/External phase/

8. $\mathrm{Re}\ peat$
9. $k_{\min} = \min_i \{ front(Q_i)\} \Rightarrow$
10. $B_{\min} = Block\ of\ k_{\min} \Rightarrow$
11. $Disk\ \mathrm{Re}ad(B_2, Y, B_{\min}) \Rightarrow$
12. $Del\ (Q_i, k_{\min}) \Rightarrow$
13. $Pivot = \min(B_2)$
14. $S = \{k \mid k \le pivot\ and\ k \in B_1\}$
15. $While \mid S \mid \ge B$
16. $P = B$
17. $Disk\ Write(B_1, X, P)$
18. $S = S - P$
19. $End\ While$
20. $Merge(B_1, B_1, B_2) \Rightarrow$
21. $Until\ all\ Q_i\ are\ empty \Rightarrow$
22. $Disk\ write(B_1, X)$

/Termination of external phase/

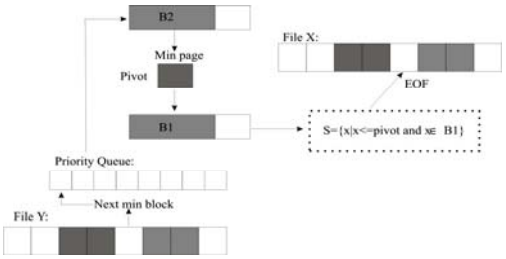

Fig. 2 New algorithm procedure

V. Analyze

We will study the algorithm in detail including "algorithm validity", "computation of input-output complexity" and the "amount of mem required in term of the number of input records".

Theorem 1:

The algorithm has the same number of input and output operation as 2N/B with the o (n) as its complexity.

Proof: In order to produce each execution and because of existing of M records in the main mem per execution, the M/B number of writing and reading operation ,separately, will be needed in the internal phase. Also the total number of executions in the internal phase are equal to N/M so we have the equation of N/M*M/B =N/B for writing and reading operation. In the external phase, one block will be added to main mem in each loop and when all the records of that block found appropriate place they will be written to output file for once. Furthermore as there is N/B blocks, so there is N/B reading and writing operation with the o (n) as following:

2N/B+2N/B=4/N

Theorem 2:

After external phase, all the elements will be placed precisely in its location ascending.

Proof: In the internal phase each execution will have records sorted ascending after internal sorting. In the external phase and according to the queue priority the algorithm will place the least block in the mem and all the B1 records which are smaller or equal than the reference element will be written to the external mem.

In addition, the records of B1 that is bigger than the reference element will be merged with B2 records and replace in the B1.

As we know the nth reference element is smaller than the n+1th reference (as the blocks are placing in the mem according to the queue priority) so the records of the $n+1^{th}$ step is greater than the nth step's record which is written to the external mem and the location of records in the output file is precise.

Theorem 3:

The minimum required mem of algorithm is N/B+ (NB) 1/2

Proof: because of the total number of blocks in the internal phase is equal to N/B so the queue priority need to N/B mem and consequently M≥N/B (M is equal to the number of main mem required).

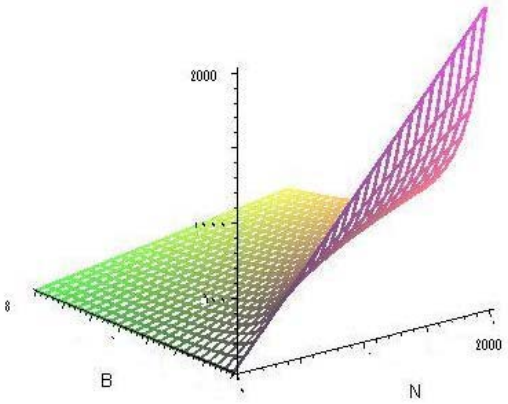

Fig. 3 Depict of equation 5

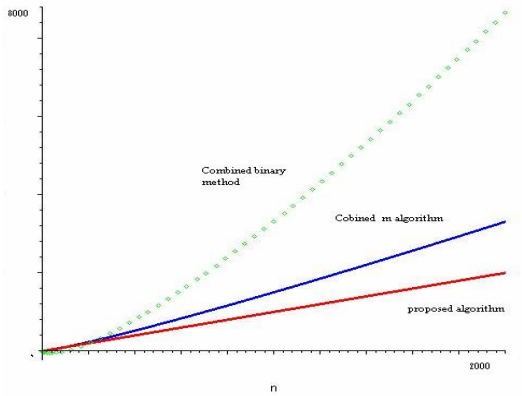

Fig. 4 Comparing the three algorithms

In the external phase of the algorithm the N/M blocks could be placed in the mem therefore M≥(N/M)*B and so M2 ≥N*B and by adding the required amount of mem in the first and second step we get the following statement:

$$M \geq N/B + (NB)\ 1/2$$

Above the diagram shows the results of three algorithms including m-way and 2-way merge sort in comparison with new algorithm. As it is pictured, the numbers of blocks in the mem is considered to be 100 and from up to down diagrams are belonging to 2-way, m-way merge sort and new algorithm. It is obvious that the new algorithm with a little bit analogy to m-way sort merge, is take the least time in comparison with the other two algorithms to get the answer except that the former is less practical.

VI. Conclusion

In this paper a new external algorithm was proposed. It is focused on the second phase of the other similar algorithms but second part of that is including a new easy implementing method which has reduced the vast number of input-output operations saliently. This paper propose an easy algorithm for choose the correct record location of the final list. This decreases the time complexity and makes the algorithm faster. Simulation results show the efficiency of the proposed method.

References

[1] Aggarwal. A. & vitter. J. S, "Input / out put Complexity Of Sorting and related problem", Communications of the ACM, 1988
[2] Arge. L. & Knudsen. M. & Larsen. K, "A general lower bound on the I/O complexity of comparision-based algorithms", LNCS 709, 1993, pages 83-94.
[3] Baker. M, "Discussion of Sorting Algorithms", Mark chris Soft Home, 1999.
[4] Chen. S.Y, "Complexity Estimation", York University, 1999.
[5] Carlson. D, "External Sorting", Saint Vincent College, 2004.
[6] Floros. I, "Analysis of Internal Computer Sorting", Journal of ACM (JACM), 1999.
[7] Neapolitan. R. E. & Naimipour. K, "Foundations of Algorithms", Jones and Barlett publishers, 1998.
[8] Pang. H. & Carey. M. J. & Livny. M, "Memory-Adaptive External Sorting", Wisconsin-madison University pages 618-628, 1993.
[9] Saltenis. S, "External-Memory Sorting", ACM, 2002.
[10] wang. P. Y, "A symptotic Complexity", George Mason University, 2003.
[11] Rafiqul. I, Nasim. A, Nur. I, Shohorab. H, "A new external sorting algorithm with no additional disk space", Information Processing Letters, v.86 n.5, pages.229-233, 15 June 2003.
[12] Martin. G, Christoph. K, Nicole. S, "Tight lower bounds for query processing on streaming and external memory data", Theoretical Computer Science, v.380 n.1-2, pages.199-217, June, 2007.
[13] Goetz. G, "Implementing sorting in database systems", ACM, September 2006.
[14] John. Y, Justin. Z, "Compression techniques for fast external sorting", Springer-Verlag New York, Pages: 269 - 291, April 2007.