IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-4, Issue-11 E-ISSN: 2347-2693

Improvement of Time Complexity on External Sorting using Refined Approach and Data Preprocessing

S.Hrushikesava Raju[1*], M.Nagabhusana Rao[2]

[1]*Research Scholar, Regd.No:PP.CSE.0158,Rayalaseema University,Kurnool A.P.*
[2]*Professor, Department of CSE, K L University, Vijayawada, A.P.*
hkesavaraju@gmail.com, mnraosir@gmail.com

Available online at: www.ijcseonline.org

Received: Oct/09/2016 Revised: Oct/17/2016 Accepted: Nov/20/2016 Published: Nov/30/2016

Abstract- Generally, huge data of any organization possess data redundancy, noise and data inconsistency. To eliminate, Data preprocessing should be performed on raw data, then sorting technique is applied on it. Data preprocessing includes many methods such as data cleaning, data integration, data transformation and data reduction. Depending on the complexity of given data, these methods are taken and applied on raw data in order to produce quality of data. Then, external sorting is applied. The proposed external sorting now takes the number of passes less than actual passes $\log_B (N/M) + 1$ for the traditional B – way external merge sorting. Also, the number of Input / Outputs of proposed method is less than $2*N*(\log_B (N/M) + 1)$ of Input / Outputs than traditional method, and also proposed method consume least number of runs compared to actual basic external sorting.

Keywords— data preprocessing, external sorting, Data cleaning, passes, Inputs / Outputs, and runs.

I. Introduction

In the real world, most data collected should be huge and that consists of lot of irregularities in terms of missing data, noise, or even outliers. This data doesn't possess quality of information. A data mining technique called Data Preprocessing is required in order to get quality data by removing such irregularities. The data Preprocessing technique has four methods and these are used appropriately to eliminate particular complexities that each method can remove. Those methods are Data cleaning, Data Integration and Transformation, Data reduction, and Data Discretization and Summarization.

The first method, Data cleaning is used when incomplete, noise, or any outliers exist in the data that can be removed using any of binning, clustering, and regression techniques. Second is Data integration and transformation is used when the data set contains objects with many values or different objects with same value and the data is not in required interval or range and this can be eliminated by processing using care in case of integration and use any of smoothing, attribute/feature construction, normalization, or aggregation in case of transformation. The third, data reduction is when the data set is high dimensional or in large volume, and this can be avoided by using any of dimensionality reduction, numerosity reduction or data compression in order to output the reduced size of that data set which produce the same results. The data discretization and summarization is used when data is in continuous and that can be broken into intervals using either top down or bottom constructions. The following table shows when each stage required and what that method will do.

Method name	Irregularity	output
Data cleaning	Incomplete, noise, inconsistent, missing	Quality data Before integration
Data Integration and transformation	Object identity problem	Quality data with care taken
Data reduction	Data set is high dimensional	Reduced size
Data Dicretization and summarization	Data continuous	Simplified data sets

Table 1.1: Data preprocessing method's irregularities & their output

All these external sorting techniques don't minimize the disk Input / Output time efficiently. Each technique have their own drawbacks. Thus, these drawbacks resulted because of data redundancy and replication on each page or tape. This leads to redundancy after sorting runs in each phase. To avoid much time complexities or Minimize I/ O costs, data preprocessing is necessary before sorting on external storage devices such as tapes, pages or disks etc. The advantage of performing data preprocessing is redundant data is eliminated from tapes or pages, sorting data doesn't contain redundancy which also minimize the I/O costs in sorting.

II. Related Work

© 2016, IJCSE All Rights Reserved

According to [14],[15] and [1,2], the various external sorting techniques although performing external sorting that achieved with a variety of overheads. According to [9], certain types of lemmas are used to achieve efficient external sorting but it works on only one disk model although it takes less Input / Output operations than normal merge sort. To overcome the overheads of various external sorting techniques and also external sorting using 3 lemmas, data preprocessing [4,5] is proposed before external sorting is used. The external sorting are categorized into 2 types importantly. They are k-way external sorting and poly phase merge sorting for k-way merge sorting. According to Mark Allen Weiss, the external sorting applied on the data although that are of disk accesses or inputs / outputs costs are huge compared to disk accesses on data without redundancy.
The consume of time for k-way merging and poly phase merging on the data that involves redundancy is illustrated in the table 2.1. The variables here are k denotes k-way sorting, N denotes number of items on initial tape, and M is initial run size.

The time complexities or overheads incurred for the same external sorting strategies that don't involve redundancy are found are less than actual complexities for the data that involve redundancy by some examples in Experimental results.

Method	**Tapes required**	**# of passes**
K-way external sorting	2 * k	Log_k N/M +1
Poly phase merging	k + 1	Depends on how large the data size on initial tape

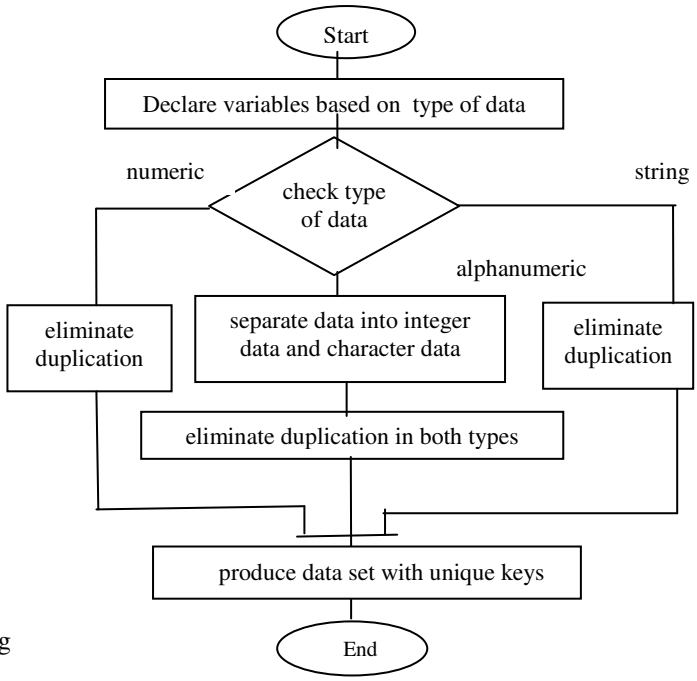
Table 2.1: Time Consumption of works for external sorting strategies

III. Proposed Work

In this, Data preprocessing is an important task that removes redundancy by using a variable length record. This record avoids loss of data and is used to represent the data only once although it maintains the number of occurrences of that item.
This data preprocessing also used to reduce the number of runs involved in the merge during each pass when the data possess redundancy hugely. A method of data preprocessing called Data Cleaning is used to eliminate data redundancy, noise, and inconsistency that exist in data available on initial tape or page. The advantages of data preprocessing is shown in table 3.1 as follows according to type of data.

In all above types, data preprocessing reduce the number of disk accesses or Input / output cost in terms of eliminating redundancy. Data Preprocessing supposed be applied on the data before sorting results many benefits greatly that leads to perform external sorting efficiently. Here, data preprocessing algorithm or pseudo code is given for each type of data and also algorithm is defined for external sorting.

FlowChart for Data Preprocessing: It shows the flow of actions in data preprocessing based on type of data. This module in flow chart takes the data first, then applies appropriate data preprocessing method depending on type of data provided to sort.
Moreover, It is a Graphical technique that clearly conveys information and its meaning to the end user who even doesn't know about programming. The following denotes flow chart for data Preprocessing module which can be applied prior to sorting.



Flow Graph 3.1: Data Preprocessing Steps

B. Algorithm for external sorting: It provides the pseudo code that accomplishes the external sorting for m-way merge sorting. This sort works same line K-way merge sort but it involves least number of runs for the data that initially have lot of redundancy.

Algorithm **B-way Externalsortnig**(dataset)
Input: put the items of dataset on first tape or page of inputs. 2 * B tapes or pages are used.
output: Sorted data placed on single tape or page

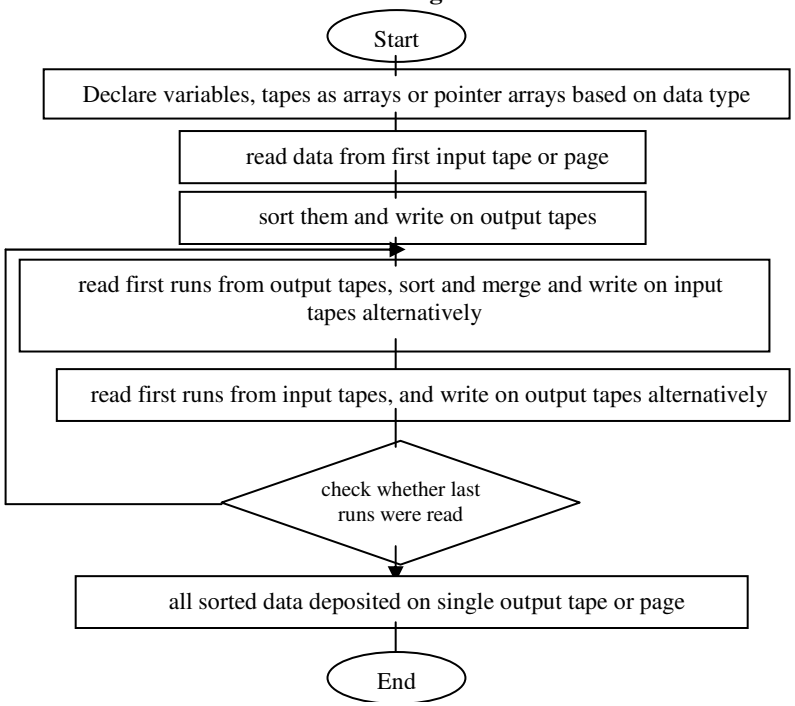
1. Take data on first tape or page of size M which is main memory size successively till data size is reached. Make main memory size is double the page size in order to reduce the no. of accesses or input / output costs.
2. Sort them internally using either Quick sort or Merge sort and Write them on output tapes alternatively.

© 2016, IJCSE All Rights Reserved

3. Read first runs from all output tapes or pages, merge and sort them internally, and write it on input tape. If all the data deposited is on one input tape. This step makes the transfer of entire data onto the output tape automatically. This increases the performance of the external sorting. Similarly, Read second runs on all output tapes and write them on input tape alternatively on input tapes till last runs are read from all output tapes.
4. Read first runs from input tapes, merge and sort them using sorting algorithm and write on output tapes alternatively from second run to last runs that are read from input tapes.
5. Repeat 2 to 4 steps till total data comes on single output tape.

Advantages: Although it yields same number of passes in some cases, the number of disk accesses or input and output costs are reduced greatly in terms of runs.

Flow chart for efficient sorting:



Flow Graph 3.2 : External Sorting Steps

IV. Experimental results with Examples

This presents a table which shows the number of runs incurred before data preprocessing and after data preprocessing. From this, how the numbers of disk accesses are involved or the complexity involved is affected in terms of input and output costs and runs.

Example1: Consider the data set 10 3 3 7 1 1 1 78 2 2 2 2 3 3 3.

The purpose of taking higher way of merge sorting is to reduce number of passes. Examples include 2-way merging takes 4 passes, 3-way merging takes 3-passes, and so on. Consider T_i and T_o are input and output tapes from which both read and write are possible.

a. With Redundancy:

Step1: Initial Run construction pass – assume main memory size is 3, this can be changed to double of it i.e. 6.

$T_{in}1$ 10 3 3 7 1 1 1 78 2 2 2 2 3 3 3
$T_{in}2$
$T_{in}3$
$T_{op}1$
$T_{op}2$
$T_{op}3$

Step2: read data of run size 3, sort and write on output tapes alternatively.

$T_{in}1$	
$T_{in}2$	
$T_{in}3$	
$T_{op}1$ 3 3 10	2 2 2
$T_{op}2$ 1 1 7	3 3 3
$T_{op}3$ 1 2 78	

Step3: Read first runs, merge and sort them, and write it on input tape. Next, read second runs and merge and sort them, write on input tapes alternatively till lazst runs from output tapes are read.

$T_{in}1$ 1 1 1 2 3 3 7 10 78
$T_{in}2$ 2 2 2 3 3 3
$T_{in}3$
$T_{op}1$
$T_{op}2$
$T_{op}3$

Step4: Read first runs from input tapes, merge and sort them, and write on output tape. Read second runs from input tapes, sort and merge them and write on output tapes alternatively till last runs were read from output tapes.

$T_{in}1$
$T_{in}2$
$T_{in}3$
$T_{op}1$ 1 1 1 2 2 2 2 3 3 3 3 3 7 10 78
$T_{op}2$
$T_{op}3$

There are four passes incurred for the above data set that involve redundancy.

b. Without Redundancy:

© 2016, IJCSE All Rights Reserved

First, data preprocessing applied on original data which is huge and involves inconsistency, and noise generally. So, data preprocessing is used to eliminate such deficiencies from the original data and produces data set of items: 10 3 7 1 78 2

Step1: Initial run construction pass – run size is 6

$T_{in}1$ 10 3 7 1 78 2
$T_{in}2$
$T_{in}3$
$T_{op}1$
$T_{op}2$
$T_{op}3$

Step2: read the data from input tape according to run size 3, sort and write on output tapes alternatively.

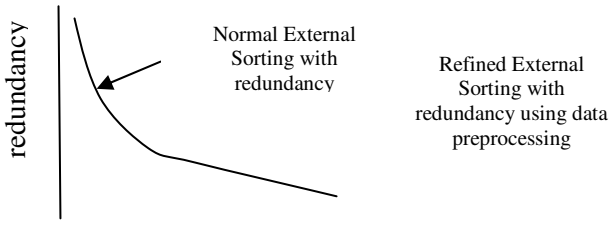
$T_{in}1$
$T_{in}2$
$T_{in}3$
$T_{op}1$ 1 2 3 7 10 78
$T_{op}2$
$T_{op}3$

Only two passes are required to sort data on external device for the data without redundancy.

The following table shows the significance difference between data with redundancy and data preprocessing whenever to sort data using B-way external sorting.

Factor	**Sorting on Data with redundancy**	**Sorting on Data after data preprocessing**
Input and output costs or number of disk accesses	(15 + 15+15 + 15) * 2 = 120 accesses = 2*N* (log $_B$ (N/M) + 1) =2 * 15 * 4	2 * 6 * 3 = 36 accesses
Number of runs	5+2+1 excluding initial pass	1 + 1 excluding initial pass
Number of passes	3 + initial run pass = 4	1 + initial pass =3

The following graph denotes that if the large data set contains lot of redundancy, then the number of disk accesses are reduced or minimized by applying data preprocessing. The data set that possesses lot of redundancy is inversely proportional to number of accesses or input and output costs, runs and even number of passes.



Number of disk accesses

Graph I : Redundancy versus Number of accesses relationship

V. Conclusion

This concludes that as data redundancy exists as coming data increased. Data preprocessing module helps to reduce the number of disk accesses or number of input and output costs, number of runs, and even number of passes for B-way external merge sort. The data preprocessing is flexible to work on the data of any data type. This data preprocessing module also helps to avoid loss of data by defining a record that contains item value and count of it for each duplicated element in the data set.

This work can be enhanced in future in such way that it can also be implemented on individual items when they are of records or some complex data types and can be allowed to sort them efficiently.

References:

[1] Mark Allen Weiss, "Data Structures and Algorithm Analysis in C++", Chapter7, Fourth Edition, Pearson, Florida International University, ISBN-13: 978-0-13-284737-7, ISBN-10: 0-13-284737-X.

[2] Mark Allen Weiss, "Data Structures and Algorithm Analysis in Java ",Chapter7,Third Edition, Pearson, Florida International University ISBN-13: 978-0-13-257627-7,ISBN-10: 0-13-257627-9.

[3] Alfred V. Aho, John E. HopCroft and Jelfrey D. Ullman, "Data Structures and Algorithms", Chapter-Sorting,Addison –Wesley, 1983.

[4] Micheline Kamber and Jiawei Han,"Data Preprocessing, Data Mining Principles and Techniques".

[5] Margaret H Dunham, "Data Mining Introductory and Advanced Topics", Pearson Education, 2e, 2006.

[6] Sam Anahory and Dennis Murry,"Data warehousing in the Real World",Pearson Education,2003.

[7] D. E. Knuth (1985), *Sorting and Searching, The Art of Computer Programming, Vol. 3*, Addison –Wesley, Reading, MA, (1985).

[8]] Alok Aggarwal and Jeffrey Scott Vitter, Input and Output Complexity of Sorting and related problems, Algorithms and Data Structure, AV88.pdf.

[9] Leu, , Fang-Cheng; Tsai, Yin-Te; Tang, Chuan Yi,"An efficient External Sorting Algorithm", pp.159 – 163, Information Processing Letters 75 2000.

[10] Ian H. Witten, Eibe Frank, Morgan Kaufmann,"Data Mining: Practical Machine Learning Tools and Techniques", Second Edition (Morgan Kaufmann Series in Data Management Systems), 2005.

[11] Zhi – Hua Zhou, Dept. of CSE, Nanjing University,"Introduction to Data Mining", part3: Data Preprocessing, Pt03.pdf, Spring 2012.
[12] Chapter 3. Data Preprocessing, www.cs.uiuc.edu /homes/hanj/cs412/bk3.../ 03Preprocessing.ppt.
[13] Chapter 2. Data Preprocessing, ww.cs.gsu.edu/~cscyqz/ courses/dm/slides/ch02.ppt.
[14] R&G Chapter 13:External Sorting, inst.eecs.berkeley.edu /~cs186/fa06/lecs/05Sorting.ppt.
[15] Chapter11:External Sorting, www.cs.rutgers.edu /~muthu/lec9-04.ppt.
[16] DATAMINING/IT0467, http://www.srmuniv.ac.in/sites/ default/files/ files/Data%20Mining.pdf.
[17] Chiara Rebso, KDD- LAB, ISTI – CNR, Pisa, Italy ,http://www.techrepublic.com/resource-library /whitepapers/an-unique-data-mining-task-for-sorting-data-preprocessing-for-efficient-external-sorting/
[18] APPLICATION OF A DATA MINING TASK CALLEDDATA PREPROCESSING ON THE INPUT DATA AND EFFICIENT EXTERNAL SORTING USING REFINEMENT OF EXISTING ALGORITHM, http://esatjournals.net/ijret/2012v01/i03/IJRET 20120103022.pdf
[19] A Survey on Improved Time Complexities for the certain data structures using data preprocessing and refinement of existing algorithms used over them, http://ijarcet.org/wp-content/uploads/IJARCET-VOL-5-ISSUE-4-1147-1154.pdf
[20] Performance Analysis of Data Reduction Algorithms using Attribute Selection in NSL-KDD Dataset, http://ijesat.org/Volumes/2014_Vol_04_Iss_02/IJESAT _2014_04_02_16.pdf.

About Authors:

Mr. S. HrushiKesava Raju, working as a Professor in the Dept. of CSE, SIETK, Narayanavanam Road, Puttur. He is pursuing Ph.D from Rayalaseema University. His areas of interest are Data Mining, Data Structures, and Networks.

Dr. M.Nagabhushana Rao, working as Professor in the Dept. of CSE, K L University, Vijayawada, A.P. He had completed Ph.D from S.V. University in the area of Data mining. He is presently guiding many scholars in various disciplines.

© 2016, IJCSE All Rights Reserved