www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 5 May, 2014 Page No. 5575-5578

Calculating the Area of the Union of Iso-oriented Rectangles Using MapReduce

***Seyed Vahid Sanei Mehri*[1], *Ehsan Akhtarkavan*[2], *Saeed Erfanian*[3]**

[1] Department of Computer Engineering, Garmsar Branch,
Islamic Azad University, Garmsar, Iran
Vahid.sanei@gmail.com

[2] Department of Computer Engineering, Garmsar Branch,
Islamic Azad University, Garmsar, Iran
Akhtarkavan@iau-garmsar.ac.ir

[3] Department of Electrical Engineering, Garmsar Branch,
Islamic Azad University, Garmsar, Iran
serfanian@iau-garmsar.ac.ir

Abstract— *In this paper we aim to propose a faster algorithm for solving the problem of 'area of the union of iso-oriented rectangles'. For this purpose we use MapReduce which is a powerful tool in parallel data processing to divide the task among P separate processors. We utilize the Interval Tree data structure and Sweep Line technique to obtain a solution with* $\Omega(\frac{N}{P}\log\frac{N}{P})$ *time complexity.*

Keywords—*Union of Rectangles; MapReduce; Interval Tree; Line Sweep; Parallel Processing*

1. Introduction

A well-known two-dimensional Klee's measure problem [1] can be stated as calculating the area of the union of iso-oriented rectangles. The Klee's measure problem studied in [2-4]. The best known complexity time for calculating the area of the union of iso-oriented rectangles in plane is $O(N\log N)$ where N is number of the rectangles. In this paper we aim to present a method to solve this problem by parallel processing with the help of MapReduce as a programming model.

Nowadays parallel data processing is one of the most important methods in the field of data processing. MapReduce which is widely known for its use by Google [5] is a modern approach for processing data which has a high fault tolerance. In this approach we use low-end machines for data processing. MapReduce has valuable features such as scalability, simplicity, high fault tolerance which make it a special and useful tool in academic and industrial projects [6-9].

Programs which are written in MapReduce format are executed parallelly and automatically. The input data will be distributed among a number of machines and will be processed by those machines in a cluster. In fact MapReduce programs turn a list of input data to a list of output data.

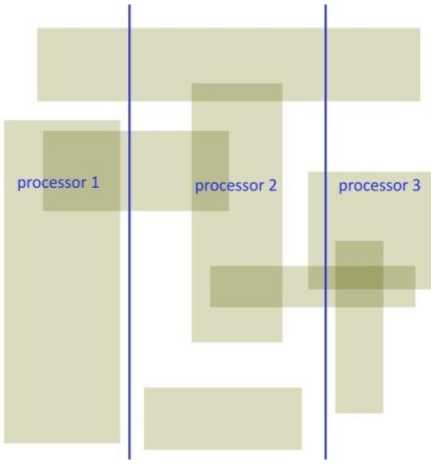


Figure 1: rectangles divided among three processors

This operation is done by Map and Reduce functions.

A. *Mapping input list*

The first phase of a MapReduce program is mapping. In this phase a list of input data is received and the Mapping function maps each input data to an output one.

B. *Reducing input list*

In this phase input data are gathered to generate an output data. In fact the Reduce function turns a high amount of input data to one or a few output data.

The goal of this paper is to calculate the area of a set of given iso-oriented rectangles which may overlap. Without loss of generality, throughout this we will assume that the rectangles are parallel to x and y axes. From this point on, we denote the set of rectangles by R, and the area of the union of these rectangles, by A. The goal of this paper is to calculate quantity A.

2. RESULTS AND DISCUSSION

Calculating the area of the union of Iso-oriented rectangles using MapReduce is done in Mapping and Reducing phases. Rectangles are split and distributed among processors in Mapping phase and then in Reducing phase the area of the union of emerged rectangles in each processor is calculated.

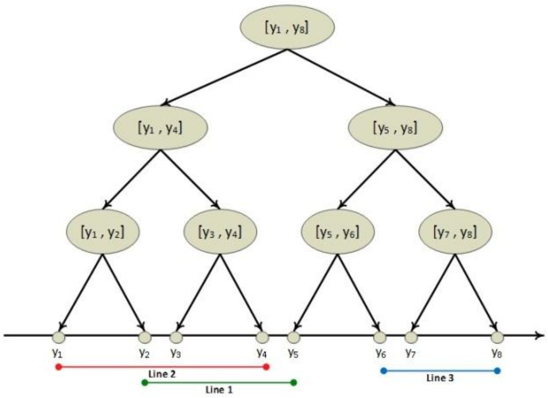


Figure 2: Structure of Interval Tree

2.1 SPLITTING RECTANGLES TO DISTRIBUTE AMONG PROCESSORS

We define the set of P processors as $\Pr = \{p_1, p_2, p_3, \dots p_P\}$.

Definition 1. We define a set of vertical lines as a set of real numbers $V = \{v_1, v_2, v_3, \dots v_P\}$ where equation of the $i_{th}\,(0 \le i \le P)$ line is $x = v_i$ and $v_i < v_j$ for every $i < j$.

Assumption 1. Processor $p_i\,(0 \le i \le P)$ calculates the area between v_{i-1} and v_i which is covered by rectangles. So we only need to determine to which processors rectangle $r\,(r \in R)$ should be assigned.

Assumption 2. The processor $p_i\,(0 \le i \le P)$ includes a point with coordinate (x, y) iff $v_{p-1} \le x \le v_p$.

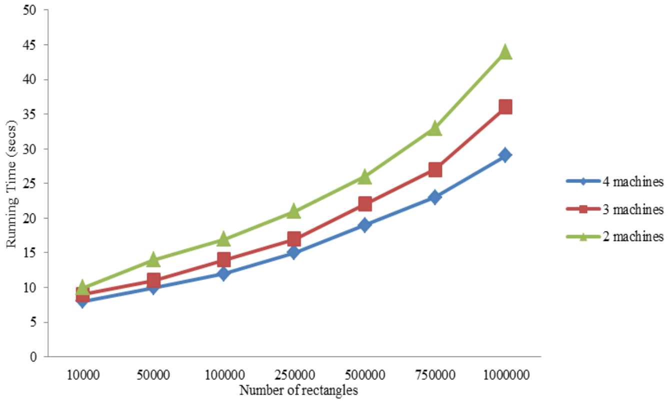
Now consider rectangle r, we assume that extension of its left vertical side intersects with x axis at the point $(x_1, 0)$ and similarly the right vertical side at $(x_2, 0)$, where x_1 and x_2 are both real numbers. By assumption 1 we name the processors which include these points respectively p_a and p_b, where $1 \le a \le b \le P$. Since V is sorted we can find a and b in $O(\log N)$. After finding a and b we are able to divide the task of computing the area of this rectangle among processors p_a through p_b.

Figure 1 illustrates an example of how rectangles divided among three processors. In the next section we will show how to calculate the area of the rectangles assigned to a processor.

2.2 THE AREA OF THE UNION OF ISO-ORIENTED RECTANGLES IN EACH PROCESSOR

Now we aim to calculate the area of iso-oriented rectangles assigned to a single processor by presented algorithm in [2].

We define set $Q = \{q_1, q_2, q_3, \dots q_N\}$ where q_i is the number of rectangles assigned to processor p_i. Sub-problem of a specific processor can be solved by the use of sweep line technique [10, 11]. Conceptually the sweep line is a vertical line swept across the plane. We can assume that the sweep line scans from left to right. Actually the answer of the problem is only related to the points placed in at-least one rectangle and the sweep line has swept them.



We consider reaching the sweep line to the right and left sides of a rectangle as events. We sort all of events in increasing order according to their *x*-value because the events are segments parallel with *y* axis. As a result, we obtain a set of segments, that each one of them indicates the left or the right side of some rectangle. We can say each rectangle has emerged twice in the set.

When the sweep line scans the plane, if it detects an event which is the left side of some rectangle we insert that rectangle into the active set. If the event is the right side of a rectangle we remove that rectangle from the active set. Therefore in two successive events we can determine the rectangles scanned by sweep line. Let Δy be the length of the sweep line scanned the rectangles of active set and Δx be the distance between current event and the previous one, the product of Δy and Δx is the area of union of the rectangles between the two events.

Simply Δx is the difference between x-value in the current and previous event. For calculating Δy we can use the data structure of Interval Tree [12]. Using Interval Tree we can obtain Δy in $O(\log n)$. Δy is the length of parts of sweep line that lie on rectangles in the active set. Since we want to calculate Δy, only the *y*-value of vertical sides matters. We suppose that list $E = \{y_1, y_2, y_3, ... y_M\}$ consists of the *y*-value of all end points of the vertical sides where $0 \leq M \leq 2 \times N$, $y_1 \leq y_2 \leq y_3 \leq ... \leq y_M$. As shown in Figure 2 Interval Tree is based on the idea of representing set of intervals using balanced binary tree [13]. Each internal node has two child nodes and also maintains an interval.

Assumption 3. If the left and right children of an internal node maintain $[y_A, y_B]$ and $[y_B, y_C]$ respectively, the parent node will maintain $[y_A, y_C]$ $(y_A \leq y_B \leq y_C)$.

Assumption 4. Each leaf maintains one point in list E.

Figure 3: Comparision of running time of 2, 3 and 4 machines in hadoop cluster to calculate the area of the union of iso-oriented rectanlges

We can conclude from Assumption 3 and 4 that the nodes which are one level higher than leaves maintain the interval with two successive points in list E and the nodes which are two levels higher maintain four successive points and so on. Therefore we can assume that *M* which denotes the size of list E is a power of 2 $(M = 2^K)$. It is easy to verify that nodes in level T represent an interval with length of $2^{\log M - T} (0 \leq T \leq \log M)$. An Interval Tree can be constructed in the recursive manner.

Lemma 1. The Interval Tree for M points can be built in $O(M \log M)$.

Proof. Every time the recursive function to build the interval tree is called the length of E list is divided by 2. So the depth of tree is $O(\log M)$.On the other hand, the number of nodes in each level is $O(M)$.The reason is each node in the tree is visited exactly once. Hence building the entire tree takes $O(M \log M)$. ■

Lemma 2. Insertion and deletion of an event can be done in $O(\log M)$.

Proof. While searching for an event one of the following conditions might occur:

1. The left sub-tree lies entirely in the event.
2. The right sub-tree lies entirely in the event.
3. The event is found in both left and right sub-trees. ■

In the right path, when the third condition occurs, for all its the left sub-tree the first condition maintains. And for right sub-tree of the left path second condition occurs. Hence we only need to search the end points of an event which takes $O(\log M)$. time. Similarly, deletion of an event by the use of Lemma 2 will be done in $O(\log M)$.When the sweep line scans an event the Interval Tree will be updated for at most $O(M)$. events. Therefore the algorithm for calculating the area of union of iso-oriented rectangles on a single processor can be done in $O(M \log M)$.

2.3 EXPERIMENTS

To calculate the area of the union of iso-oriented rectangles using MapReduce we use cluster of PCs (Intel Core i5 processor, 2GB RAM) on hadoop-1.2.1. The input data contains *N* rectangles which each line of that represents coordinates of lower-left and upper-right of a rectangle. As shown in Figure 3 two, three and four machines as data nodes in hadoop cluster used to process the input data. The measured running time in Figure 3 is obtained by using the mean of 50 random input data.

The obtained results indicate increasing number of machines in cluster to process the input data is lead to decrease in running time. With the increase in the number of machines in a cluster, the probability of fewer rectangles being calculated in a processor, decreases and thus results in the reduction of running time. In the best case, all rectangles will be distributed among *P* processors. So the proposed method runs in $\Omega(\frac{N}{P} \log \frac{N}{P})$.

3. CONCLUSION

In this paper we provided a MapReduce algorithm to calculate the area of the union of iso-oriented rectangles. The described algorithm splits the rectangles as input among processors in Mapping phase then in Reducing phase the area of the union of assigned rectangles in each processor is calculated in $O(M \log M)$ where M is the number of assigned rectangles. The obtained results indicate increasing number of machines in cluster to process the input data is lead to decrease in running time. With the increase in the number of machines in a cluster, the probability of fewer rectangles being calculated in a processor, decreases and thus results in the reduction of running time. In the best case, all rectangles will be distributed among *P* processors. So the proposed method runs in $\Omega(\frac{N}{P} \log \frac{N}{P})$.

REFERENCES

[1] Klee, V., Can the Measure of∪ n 1 [ai, bi] be Computed in Less Than O (n log n) Steps? American Mathematical Monthly, 1977: p. 284-285.

[2] Bentley, J.L., Algorithms for Klee's rectangle problems, 1977, Technical Report, Computer.

[3] Vahrenhold, J., An in-place algorithm for Klee's measure problem in two dimensions. Information processing letters, 2007. **102**(4): p. 169-174.

[4] Sharma, G., et al., An Efficient Transformation for the Klee's Measure Problem in the Streaming Model, in CCCG2012. p. 83-88.

[5] Dean, J. and S. Ghemawat, MapReduce: simplified data processing on large clusters. Commun. ACM, 2008. **51**(1): p. 107-113.

[6] Deligiannis, P., H.-W. Loidl, and E. Kouidi, Improving the diagnosis of mild hypertrophic cardiomyopathy with MapReduce, in Proceedings of third international workshop on MapReduce and its Applications Date2012, ACM: Delft, The Netherlands. p. 41-48.

[7] Mantha, P.K., A. Luckow, and S. Jha, Pilot-MapReduce: an extensible and flexible MapReduce implementation for distributed data, in Proceedings of third international workshop on MapReduce and its Applications Date2012, ACM: Delft, The Netherlands. p. 17-24.

[8] Menon, R.K., G.P. Bhat, and M.C. Schatz, Rapid parallel genome indexing with MapReduce, in Proceedings of the second international workshop on MapReduce and its applications2011, ACM: San Jose, California, USA. p. 51-58.

[9] Qiu, J., Generalizing mapreduce as a unified cloud and HPC runtime, in Proceedings of the 2nd international workshop on Petascal data analytics: challenges and opportunities2011, ACM: Seattle, Washington, USA. p. 37-38.

[10] Chang, F., Y.-C. Lu, and T. Pavlidis, Feature Analysis Using Line Sweep Thinning Algorithm. IEEE Trans. Pattern Anal. Mach. Intell., 1999. **21**(2): p. 145-158.

[11] Merry, B. Line Sweep Algorithms. 2007; Available from:http://community.topcoder.com/tc?module=Static&d1=tutorials&d2=lineSweep.

[12] Cormen, T.H., et al., Introduction to algorithms. Vol. 2. 2001: MIT press Cambridge.

[13] daneilP, Range Minimum Query and Lowest Common Ancestor. 2009.