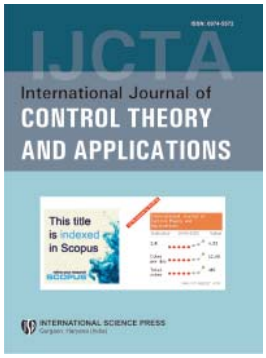

International Journal of Control Theory and Applications

ISSN : 0974–5572

Volume 9 • Number 40 • 2016

Performance Improvement of Geometric Decision Tree using Parallel Approach and XGBoost Algorithm

V. Ramya[a] G. Senthil kumar[b] S. Veena[b] A. Maria Nancy[b] and S. Saranya[b]

[a]*Asst.Proff, Department of Computer Science,Facuty of Science & Humanities, SRM University, Katankulathur, TN, India*

E-mail: ramya.v@ktr.srmuniv.ac.in, Corresponding Author

[b]*Asst.Proff, Department of Software Engineering, SRM University, Katankulathur, TN, India*

E-mail: chenthi2004@hotmail.com, veena.s@ktr.srmuniv.ac.in marianancy.a@ktr.srmuniv.ac.in, saranya.s@ktr.srmuniv.ac.in

Abstract: Decision trees are widely used in classification methods, specifically in the operation research, for decision analysis decision trees are used. It helps in taking the decision that meets organizations goal. Decision trees are also used in data mining. Geometric decision trees are used to come over the drawback of previous algorithms which are unable to capture geometric structure from the data. The idea of the algorithm is taken from the SVM machine, which will capture the geometric structure very well. So this algorithm gives better accuracy than previously proposed algorithms. In this paper, we analyzed the sequential performance of the geometric decision tree algorithm. Then we executed this code on multiple cores and analyzed its parallel performance. The system is divided into different sub modules which are independently executed on the multiple cores. Then we discussed performance comparison between parallel and sequential execution. From these results we concluded that the parallel execution gives faster solution. Apart from this we can implement XGBoost algorithm as a gradient boosted decision trees designed for performance and speed.

Keywords: *Decision Tree, SVM, XGBoost.*

1. INTRODUCTION

Decision trees are the most simple and easy way for decision making task. It came out as a better replacement for the different tools used in data mining. A decision tree is a decision support tool which uses a tree-like graph or model of various decisions and their possible consequences, which includes chance event outcomes, resource costs, and utility. A decision tree is one way to display an algorithm. Decision trees are commonly used in operations research, mainly in decision analysis, to identify a strategy most likely to reach a goal, and also a popular tool for machine learning. Now days the decision trees are used in every business applications. Many algorithms were proposed for generating the decision trees. These algorithms split sets of data into branches in which root node is kept at the top. This root node describes the main objective of analysis along with its contained values. Decision rules are discovered on the basis of which methods are used to capture

the relationship between the root node and the fields that are used as input for generating the branches of the decision tree. A general decision tree is as shown in the following fig 1.1. In the figure one branch is shown, when such branches are nested into each other form a decision tree. In which every branch is termed as the node and the bottom node as leaves.

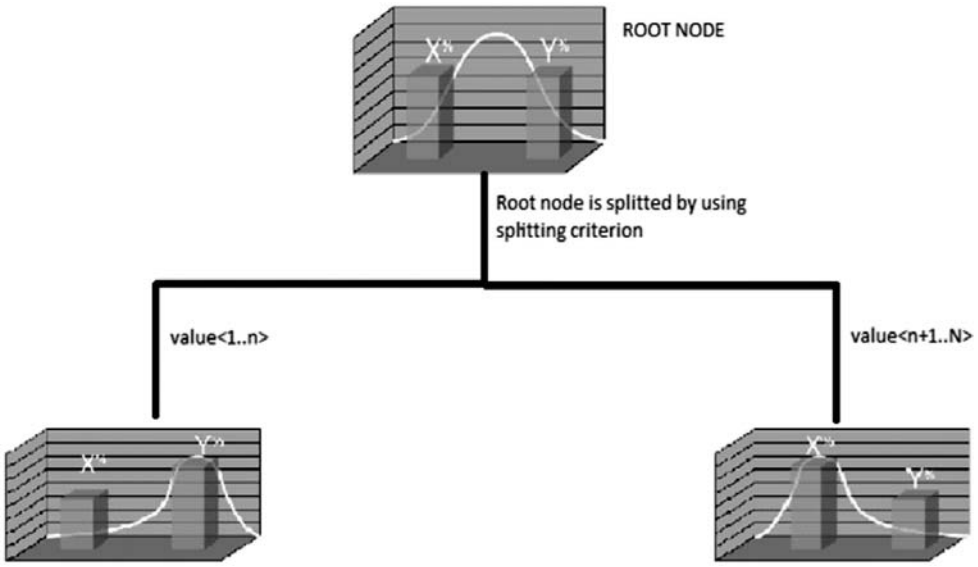

Figure 1: Decision tree illustration

The class label is attached to the every leaf node from the decision tree. Amongst all of the decision tree algorithm, the GDT (Geometric Decision Tree) is a more feasible algorithm which tries to construct the tree by using geometric structures [1].

There are two main types of Decision trees as axis parallel decision tree and oblique decision tree. In the axis parallel trees, the splitting rule used for splitting each node depends upon the single component of feature vector whereas in the other it depends on the combination of components. The GDT presents an algorithm for learning such oblique trees. For oblique trees also two approaches are described as in the first approach the structure of the tree is fixed. This kind of approach is used very rarely. In the second approach the tree is designed in the top down manner [4] [5]. GDT algorithm designed mainly for learning these oblique trees. Another important algorithm is the XGBoost , which is a library designed and optimized for boosting trees algorithms. The algorithm of XGBoost is a similar extension of the classic gbm algorithm. XGBoost is able to utilize the more computational power and get more accurate prediction by employing multi-threads and regularization.

2. LITERATURE SURVEY

As the data size is increasing, the trouble to handle the data and to retrieve information from data is also increasing .Hence the data warehouse and data mining came in to existence. So many algorithms are proposed for data warehouse and data mining, but we are mainly deal with data mining where we deal with classification means to find out which dataset will fall in which class.

In this paper, we are dealing with decision tree based classification. So many algorithms are proposed for decision tree based classification, we will discuss some of them here.

An oblique decision tree is also used for classification which combines deterministic hill climbing and two randomized forms for each node of the decision tree. This is mainly used for the numeric data type, but we can adopt for symbolic attributes also. [3]

Decision trees are the method for doing classification,this is proved by so many statistics, pattern matching and data mining tool, for proving this a unified algorithm was taken and various splitting criteria and pruning technique is implemented. [6]

In decision tree so many rules is generated ,unneccesary we have to check each rule individually, so to remove this problem researcher gave the GID algorithm which will convert the rules according to their semantics of the decision tree, as the algorithm is dealing with the semantics so cost of computation and construct the decision tree is negligible.

In traditional decision tree method, there are so many drawbacks which can be removed when we use multi class, so for multi class, multi stage decision tree algorithm is used which deal with highest cohesion and lowest coupling degree of clustering .the problem of multi class is diveded into two classes *i.e* highest cohesion degree and lowest coupling degree based on inner class and intra class margin.this way we can decrease the complexity and computation time of the decision tree which is lacking in the traditional decision tree algorithms.[8]

Parallel to construct the decision ,we also pruned some items with the help of split rules, which is generated at each node, Alopex algorithm is used for evaluating the split rules at each node,but this methodology also has some drawback, *i.e.*, it works for two class only.[7**]**

3. GEOMETRIC DECISION TREE ALGORITHM

As we already discussed in above discussion that the GDT uses top- down approach. Any decision tree based algorithm has two issues that are how to store training data on a node and how to do rating in between hyperplanes for classification. In every this kind of algorithm, the performance depends upon rating of hyperplanes at every node. For this algorithm, two cases are considered that is 'two class problem' and the 'Multiclass problem.'

In two class problem, the geometric decision tree algorithm, in which the node splits into a left child and right child based on the split rule. The author considers the hyper plane as the split rule at the node. Firstly find the two clustering hyper planes at any node having set of patterns and then one of the hyperplanes among two hyper planes used as angle bisector. The gini index approach gives the better hyperplane that is used as bisector. Each hyperplane is such that it is having the maximum number of points to one class only among two classes that are each plane nearest to the one plane among two sets of data and farther from another set. Generalization errors are minimized for finding the optimal tree. The impurity at each node gets reduced using this GDT algorithm.

The difference between two class and multiclass is, in multiclass for deciding the node is a leaf or not, take some points from the class having maximum points.

In the multiclass problem, the input is training data and output is a class label.

Algorithm for multiclass GDT [1]:

1. Let S be our training data then divide S into two parts *i.e.* S_+ and S_- .
2. S_+ is majority class and S_- contains the remaining data.
3. Find two matrices for both classes and also compute two hyperplanes.
4. Find angle bisector using Gini index.
5. If (Depth of tree = Maximum depth) then

   Node $\rightarrow$ Leaf Node

   Node $\rightarrow$ Class label
6. Else Repeat steps 3 to 5.

The previous section describes the working of GDT. Now in further work, we execute this code parallely on the multiple cores. Then we check the performance gap between the serial execution of code and parallel execution.

4. XGBOOST

Xgboost to build a model and make predictions. It is an efficient and scalable implementation of gradient boosting framework

linear model, *tree learning* algorithm. It supports various objective functions, Including *regression*, *classification* and *ranking*. The package is made to be extendible, so that users are also allowed to define their own objective functions easily.

It has several features:

1. **Speed:** It can automatically do parallel computation on *Windows* and *Linux*, with *OpenMP*. It is generally over 10 times faster than the classical gbm.
2. **Input Type:** it takes several types of input data:
   - **a) Dense Matrix:** R‘s dense matrix, *i.e*. matrix ;
   - **b) Sparse Matrix:** R‘s sparse matrix, *i.e*. Matrix::dgCMatrix ;
   - c) Data File: local data files ;
   - **d) xgb.DMatrix:** its own class (recommended).
3. **Sparsity:** It accepts sparse input for both tree booster and linear booster, and is optimized for sparse input.
4. **Customization:** It supports customized objective functions and evaluation functions.

5. PERFORMANCE ANALYSIS

Here we implemented Geometric decision Tree through Java. So, we have used Java Open Multi-Programming [9] (JOMP) for parallelization, to increase the performance. Java has no specific compiler directive, so we embedded FORTRAN as a comment in our code to parallelize the serial code.

JOMP architecture is based on Fork Join model, and it uses work sharing construct (used to share the work among threads), Data Environment constructs (used to define the scope of the data) and extensive API for control the program.

In this paper we consider the multicores to see the performance of GDT algorithm across them. For performance measure, we checked execution time of program for different table sizes. Following table illustrate the difference between serial execution and parallel execution.

Table 1
Execution Time

Table Size(in Tuple)	*Serial Execution Time*	*Parallel Execution Time (for 2 cores)*	*Parallel Execution Time (for 4 cores)*
500	1.4703	0.8914	0.3172
1000	1.9784	1.3424	0.6847
1500	2.7205	1.7872	0.8723
2000	3.1238	2.1623	1.0623

The performance can be illustrated using graphically as shown below.

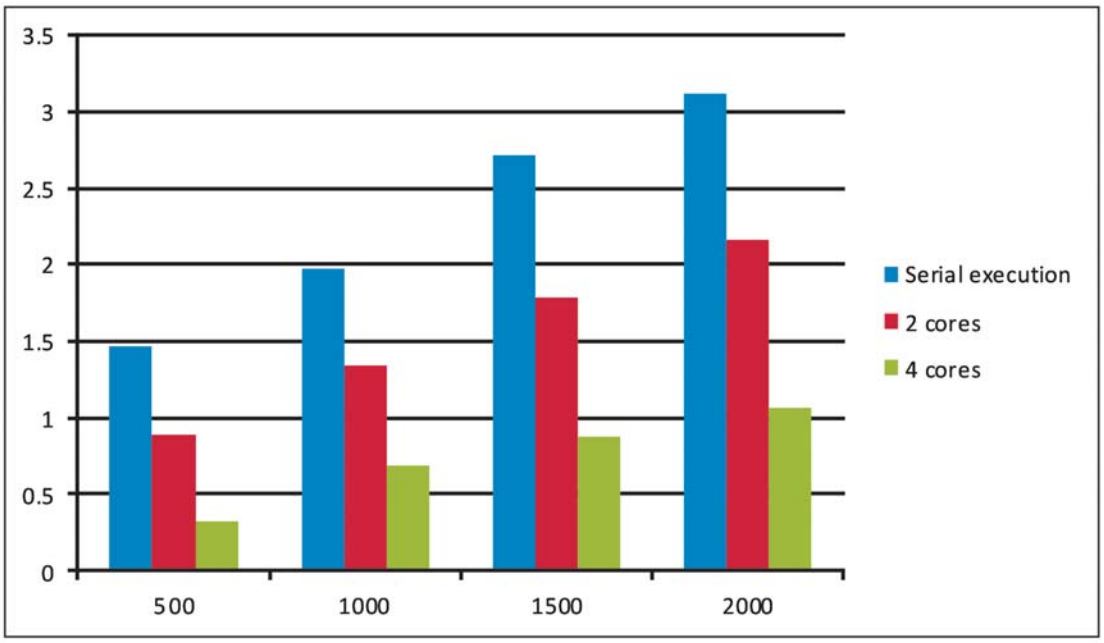

Figure 2: Histogram Representation of Execution Time

6. CONCLUSION

In this paper, we have analyzed the GDT algorithm for two class and multiclass problems. We have implemented GDT algorithm using Java. Firstly, we determine the serial execution of code on multicores. By using the JOMP we parallelized the code to improve the performance. Parallel implementation of code is checked on 2 cores and 4 cores and determined the difference in execution time. Serial implementation takes more time as compared to the parallel implementation. Using the multicores, the performance can be improved.

REFERENCES

[1] Manwani, N.; Sastry, P.S.,"Geometric Decision Tree," IEEE Trans. Syst., Man, and Cybernetics, Part B: Cybernetcs, vol. 42, no. 1, pp. 181-192, Feb. 2012.

[2] B. Chandra and P.P. Varghese, "Fuzzy SLIQ decision tree algorithm," IEEE Trans. Syst., Man, Cybern. B, Cyberen., vol. 38, no. 5, pp. 1294-1301, Oct. 2008.

[3] S.K. Murthy, S. Kasif, and S. Salzberg,"A system for induction of oblique decision trees," J. Artif. Intell. Res., vol. 2, no. 1, pp. 1-32, Aug. 1994.

[4] R. O. Duda and H. Fossum,"Pattern classification by iteratively determined linear and piecewise linear discriminant functions," IEEE Trans. Electron. Comput., vol. EC-15. No.2, pp.220-232, Apr. 1966

[5] A. Suarez and J.F. Lutsko,"Globally optimal fuzzy decision trees for classification and regression," IEEE Trans. Pattern Anal. Mach. Intell., vol.21, no. 12, pp. 476-487, Nov.2005

[6] L. Rolakh and O. maimon. "Top-down induction of decision trees classifiers- A survey," IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 35, no.4,pp 476-487,Nov.2005.

[7] C. Ferri, P.Flach,and j. Hernandez-orallo," Learning decision trees using the area under the ROC curve,"in proc. 19th ICML, San Francisco, CA, Jul. 2002, pp. 139-146.

[8] M. Lu, C. L. P.Chen, J.Huo, and X. Wang,"Multistage decision tree based on interclass and inner cflass margin of SVM," in Proc. IEEE Int. Conf. Syst., Man, Cybern, San Antonio, TX, 2009, pp.1875-1880

[9] Mark Kamnites,"Java OpenMP," http://www2.epcc.ed.ac.uk/computing/reaserchactivites/jomp/papers/sss0599.pdf.