4(a) The computational complexity of optimizing a tree of depth d in terms of m and n : O(mnlogn)

4(b) Applications with high dimensional sparse output requires most expensive computation in GBDT training. GBDT builds a regression tree that fits the residual from the previous tree. So at least O(MN) time and memory are required to build GBDT trees.

To solve this problem, we can solve a L_0 regularized optimization problem to enforce the prediction of each leaf node in each tree to have only a small number of non-zero elements/labels.

Problem 4(c)

We can train an individual decision tree in parallel in thread

Training a decision tree is an iterative process that requires looping over all possible ways of splitting the current leaf nodes into new leaf nodes. So each ~~del~~ decision tree takes a lot of time for computation. So these decision trees can be run in parallel threads.

Problem 4(d)

1) In gradient descent the goal of calculating gradients and updating θ accordingly is to optimize training loss. where in gradient boosting, one can fit a weak classifier to the data and a loss function with respect to classifier.

2) Gradient descent descends the gradient by introducing changes to parameters, whereas gradient boosting descends the gradient by introducing new models.