//2012/07/03 @SCUT

K-D tree is a natural extension of BST, it can store and search multidimensional point efficiently.

At every internal node it store a point that use ith(usually depth%k) dimension to separate the space into 2 part, it’s a hyperplane.

Search happen just like searching in the Binary Search Tree, but it use the ith dimension to compare, > node or < node ( = equal to the node must be dealt with carefully ).

Insert just like a failure of search. But removal is quite different and difficult.

k-d tree has drawback of the dimension. When in high dimension it because less efficient. Let N be the point of the dataset, and k be the dimension when N >> 2^k It works pretty well. But when k is quite big, e.g. hundred or something, we will not have enough point N to fulfill the condition.

When the K-D tree is constructed, it can be very useful for problem like nearest neighbor search and range search. When in range search, start at the rood, if the range covers(surpass) the hyperplane , both the branch must be search. If not, only one branch is need to be searched.

When is nearest neighbor search , it first search till the leaf, than goes back and see if the current min distance covers the hyperplane or not. If not, goes back one node, if yes, it recursively into the other branch.