

# Advanced Data Structures (ADS-MIRI): Assignment; Empirical Study of the Average-Case Cost of Partial Match Queries in Relaxed $kd$ trees

Amalia Duch

In this assignment you should implement random relaxed  $kd$  trees as well as the algorithm to answer partial match (pm) queries on them. Then, you will have to conduct an experimental study on the average-case cost of your implementation of this algorithm and compare it with known theoretical results. For your experiments you will need also a way to build random  $kd$  trees of different sizes from scratch.

You can use, for instance, the following structure defining a node of a relaxed  $kd$  tree (or any similar of your choice):

```
...
template <typename T>
class kdtree {
    struct node {
        T key;
        int discr;
        node* left;
        node* right;
        int sz; // size, might not be required
        node(const T& k, int d) :
            key(k), discr(d), left(NULL), right(NULL), sz(1) {}
        ~node() {
            delete left; delete right;
        }
    };
    node* root;
    int dim;

    kdtree(const kdtree& t) {};
    kdtree<T>& operator=(const kdtree& t) {
        return *this;
    }
    node* insert(node* p, const T& key) {
        ...
    }

    int pm(node* p, int j, double z) const {
        ...
    }
}
```

```

    }
public:
    kdtree(int K) :
        root(NULL), dim(K) {
    }
    ~kdtree() {
        delete root;
    }
    void insert(const T& key) {
        root = insert(root, key);
    }
    int pm(int j, double z) const {
        ...
        return pm(root, j, z);
    }
};
...

```

In order to produce a random relaxed *kd* tree of size  $n$  we propose the following procedure (other possibilities are also acceptable, as for instance random permutations of  $k$ -dimensional keys).

First generate  $n$  random points in the unit interval  $[0, 1]^k$  (that is, the  $x_0, x_1, \dots, x_{k-1}$  coordinates of each point are independently drawn from the uniform distribution in  $[0, 1]$ ). Then insert each point into your tree, assigning uniformly at random a discriminant to every new node of the tree.

For each tree of size  $n$  generate  $q$  partial match queries in the same way you generate the random points. For  $s$  out of  $k$  specified coordinates, count the number of visited nodes by your *pm* algorithm.

Notice that because of the way in which the trees are generated you can use the first  $s$  coordinates as the specified ones without loss of generality (is this the case for standard *kd* trees? why? and for quad trees?).

Optional: (Almost for free) Proceed similarly with standard *kd* trees and/or with two-dimensional quad trees.

Run your partial match algorithm with trees of several different (large) sizes and with different values of  $s$  by varying also the parameter  $k$ . For every dimension, size and value of  $s$  generate  $q$  queries and run the algorithm of each to get averages, variance, etc.

Once the full suite of experiments has been executed and data has been gathered, you have to prepare a report.

1. Describe briefly the partial match algorithm in the implemented trees, set a recurrence for the cost of partial match in random relaxed *kd* trees and solve it using the Continuous Master Theorem.
2. Describe briefly your implementation of *kd* trees and the program to execute the experiments. Give full listings of the code as an appendix of your report.
3. Describe briefly the experimental setup, how many different combinations of the parameters have you studied, how many runs have you performed, what are you looking for, how, etc.

4. Provide tables and plots summarising the results of the experiments. In particular, you should give plots showing how the average cost of the partial match algorithm evolve with  $n$ , and how it varies with  $k$  and  $s$ .
5. Compare the experimental results with the theoretical predictions, in particular, the average cost of a partial match query with  $s$  specified coordinates, in a random relaxed  $k$ -dimensional tree of  $n$  nodes is  $O(n^\alpha)$ . where  $\alpha = \alpha(s/K) = 1 - \frac{s}{K} + \phi(s/K)$  with  $\phi(x) = \sqrt{9 - 8x}/2 + x - 3/2$ .  
Plots combining the theoretical values and the experimental results are useful, but it is also important to quantify the difference between the theoretically predicted values and the empirical values.
6. Write down your conclusions.

We encourage you to use L<sup>A</sup>T<sub>E</sub>X to prepare your report. For the plots you can use any of the multiple packages that L<sup>A</sup>T<sub>E</sub>X has (in particular, the bundle TikZ+PGF) or use independent software such as gnuplot and then include the images/PDF plots thus generated into your document.

Submit your reports in PDF format using the *Racó*.