Advanced Data Structures (ADS-MIRI): List of Proposed Assignments

Amalia Duch

1 Assignment: Average-Case Cost of Partial Match Queries in Random Relaxed kd trees: empirical study

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. For your experiments you will need also a way to build random relaxed 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 $x_0, x_1, \ldots, x_{k-1}$ 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?).

Bonus: Proceed similarly with standard kd trees and 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 your implementation of kd trees and the program to execute the experiments. Give full listings of the code as an appendix of your report.

2. Describe briefly the experimental setup, how many different combinations of the parameters have you studied, how many runs have you performed, etc.

3. Provide tables and plots sumarising 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. Avoid 3-D plots.

4. 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.

5. Write down your conclusions.

We encourage you to use LaTeX to prepare your report. For the plots you can use any of the multiple packages that LaTeX 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*.

2 Assignment: Obtention of the coefficient of the leading term of the Average-Case Cost of Partial Match Queries in Random Relaxed kd trees

In this assignment you should use the Continous Master Theorem in order to obtain the value of the coefficient *alpha* of the leading term (which is of the form $\Theta(n^{\alpha})$) of the average-case cost of performing partial match (pm) queries on random relaxed kd trees. You can use literature review (correctly cited) in order to obtain the required results. Then, you will have to report in a detailed way the whole followed process.

We encourage you to use LaTeX to prepare your report.

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

3 Assignment: Empirical Obtention of the IPL of Random Binary Search trees

In this assignment you should implement random binary search trees. Then, you will have to conduct an experimental study on the average value of their internal path length (IPL) and try to show experimentally that this value is of the form $an \log n + bn + l.o.t$ for random binary search trees of size n. For your experiments you will need also a way to build random binary search trees of different sizes from scratch (See Assignment 1 for a description on how to generate them). You are asked to report in a detailed way the whole followed process.

We encourage you to use LaTeX to prepare your report.

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

4 Assignment: Free Implementation of Interval Trees

In this assignment you should implement an Interval Tree as well as the algorithm to solve the linear stabbing problem on it. Then, you will have to show its behaviour using some representative examples that you should also propose.

1. Describe briefly your implementation of interval trees and the program to execute the examples. Give full listings of the code as an appendix of your report.
2. Write down your observations.

We encourage you to use LaTeX to prepare your report. For the plots you can use any of the multiple packages that LaTeX 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*.

5 Assignment: Free Implementation of Sweep Lines

In this assignment you should implement a Sweep Line solution to the Segment Intersection Problem. Then, you will have to show its behaviour using some representative examples that you should also propose. In order to avoid precision problems, propose examples in which the segment intersections occur at points with integer coordinates.

1. Introduce the Segment Intersection Problem.
2. Describe briefly your implementation of sweep lines and the program to execute the examples. Give full listings of the code as an appendix of your report.

3. Write down your observations.

We encourage you to use LaTeX to prepare your report. For the plots you can use any of the multiple packages that LaTeX 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*.

6 Assignment: External Memory: B-Trees

In this assignment you should implement B-Trees. Then, you will have to show experimentally their behaviour on a particular feature of your choice (as for instance the relation between B and te height of the tree, the cost of long sequences of updates, among other possibilities).

1. Introduce the data structure.
2. Describe briefly your implementation of it. Give full listings of the code as an appendix of your report.
3. Describe what is what you want to study and the set of experiments that you have designed to prove or disprove your hypothesis.
4. Gather relevant data and show it using informative plots.
5. Write down your observations.

We encourage you to use LaTeX to prepare your report. For the plots you can use any of the multiple packages that LaTeX 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*.

7 Assignment: Cache-Oblivious: VanEmde-Boas-Trees

In this assignment you should implement Van Emde-Boas-Trees and show that they reduce the number of cache memory misses during a search compared to plane binary search trees.

1. Introduce the data structure.
2. Describe briefly your implementation of it. Give full listings of the code as an appendix of your report.
3. Implement plane binary search trees.
4. Look for a tool that allows you to measure somehow cache-misses or cache performance.
5. Compare the data structures for several choices of B.
6. Write down your observations.

We encourage you to use LaTeX to prepare your report. For the plots you can use any of the multiple packages that LaTeX 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*.