ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute of Theoretical Computer Science
Mohsen Ghaffari, Angelika Steger, Emo Welzl, Peter Widmayer

Algorithms, Probability, and Computing — Special Assignment 1 — HS16

- The solution is due on **Tuesday October 25** by **4 pm**. Please bring a print-out of your solution with you to the lecture. If you cannot attend (and please only then), you may alternatively send your solution as a PDF, likewise **until 4 pm**, to `mmilatz@inf.ethz.ch`. We will send out a confirmation that we have received your file. Make sure you receive this confirmation within the day of the due date, otherwise complain timely.
- Please solve the exercises carefully and then write a nice and complete exposition of your solution using a computer, where we strongly recommend to use LaTeX. A tutorial can be found at `http://www.cadmo.ethz.ch/education/thesis/latex`.
- For geometric drawings that can easily be integrated into LaTeX documents, we recommend the drawing editor IPE, retrievable at `http://ipe7.sourceforge.net/` in source code and as an executable for Windows.
- Keep in mind the following premises:
  - When writing in English, one prefers short and simple sentences.
  - When writing a proof, one prefers precise statements.

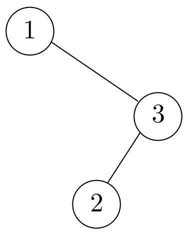
  The conclusion is, of course, that a special assignment should consist of sentences that are short, simple, and precise!
- This is a theory course, which means: if an exercise does not explicitly say "you do not need to prove your answer" or "justify intuitively", then a formal proof is **always** required.
- We would like to stress that the ETH Disciplinary Code applies to this special assignment as it constitutes part of your final grade. The only exception we make to the Code is that we encourage you to verbally discuss the tasks with your colleagues. It is strictly prohibited to share any (hand)written or electronic (partial) solutions with any of your colleagues. We are obligated to inform the Rector of any violations of the Code.
- There will be two special assignments this semester. Both of them will be graded and the average grade will contribute 20% to your final grade.
- As with all exercises, the material of the special assignments is relevant for the (midterm and final) exams.

Exercise 1 $4 + 6 + 5 + 5 = 20$ points
Close Neighbours

The *distance* of two nodes in a tree is the number of edges on the (unique) shortest path that connects them. For $n \geq 2$ and $1 \leq k \leq n-1$ let $Z_n^{(k)}$ denote the distance between the node of rank k and the node of rank $k+1$ in a random search tree on n keys. Furthermore let

$$Z_n = \frac{1}{n-1} \sum_{k=1}^{n-1} Z_n^{(k)}$$

denote the average of these numbers across the whole tree.



In this example $Z_3^{(1)}$ and Z_3 assume the values 2 and 1.5, respectively.

(a) Compute $\mathbf{E}\left[Z_n^{(1)}\right]$ and $\mathbf{E}[Z_n]$ for $n = 2, 3$.

(b) Find a closed formula for $\mathbf{E}\left[Z_n^{(1)}\right]$.

(c) Find a closed formula for $\mathbf{E}[Z_n]$.

HINT: Write Z_n differently, with the help of other random variables from the lecture. Starting with a recurrence for $\mathbf{E}[Z_n]$ may not be the best idea, because no points will be given for a recurrence that nobody can solve.

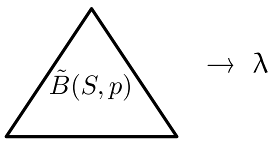
(d) Given three random search trees on n keys, show that the expected number of search trees that have $Z_n < 3$ is at least 1.

HINT: Even if you were not able to solve (c), you may use that $\mathbf{E}[Z_n] \leq 2$ holds.

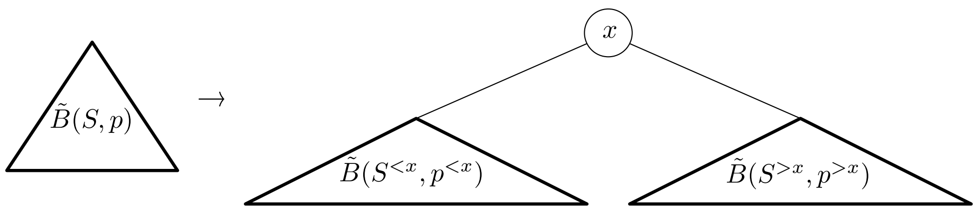
Exercise 2 $10 + 3 + 7 = 20$ points
Weighted Random Search Trees

When we defined $\tilde{B}_S$ in the lecture, we treated every key equally, so to speak. In practice, however, we might prefer some keys over others; e. g., we might know the access frequencies in advance. In that case, those keys that will be accessed more often should somehow end up higher in the tree. This leads to the notion of *weighted random search trees*:

Let S be a set of n keys, and let $p : S \to (0, 1]$ be a probability distribution that associates to every key a non-zero "weight". We define a distribution $\tilde{B}(S, p)$ on B_S, similar to the distribution from the lecture, by the following rules: If $S = \emptyset$ then



and if $S \neq \emptyset$ then



where x is drawn from S according to the distribution p. In the rule given above, $p^{<x}$ is the probability distribution on $S^{<x}$ defined by

$$\begin{aligned} p^{<x} \quad : \quad S^{<x} &\longrightarrow (0, 1] \\ k &\longmapsto \frac{p(k)}{\sum_{\ell \in S^{<x}} p(\ell)} \end{aligned}$$

and $p^{>x}$ is defined similarly.

Now let $S = [n]$. As in the lecture, we define

$$\begin{aligned} A_i^j &= [\text{node } j \text{ is an ancestor of node } i], \\ D_n^{(i)} &= \text{ depth of node } i. \end{aligned}$$

The expectations in the tasks below are meant with respect to the distribution $\tilde{B}(S, p)$.

(a) Show that for all $i, j \in [n]$ we have

$$\mathbf{E}\left[A_i^j\right] = \frac{p(j)}{\sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell)}.$$

(b) Show that we have $\frac{x}{y} \leq \ln y - \ln(y - x)$ for all real numbers $0 \leq x < y$.

(c) Show that we have $\mathbf{E}\left[D_n^{(i)}\right] \leq 2 \ln \frac{1}{p(i)}$ for all $i \in [n]$.

Hint: Find a formula for $\mathbf{E}\left[D_n^{(i)}\right]$, apply (b) to every term in it, then identify terms that cancel each other out.

Exercise 3 $2 + 6 + 2 + 10 = 20$ points
Finding the Median Slope

Recall that an *inversion* of a permutation π is a pair of indices $i < j$ such that $\pi(i) > \pi(j)$. For this exercise we admit the following two facts as a black box.

Fact A. There is an algorithm which, given a permutation π, computes the number of inversions of π in $O(n \log n)$ time.

Fact B. There is a randomized algorithm which, given a permutation π, generates an element (i, j) drawn uniformly at random from the set of inversions of π, in $O(n \log n)$ time.[1]

The permutation is given to these algorithms as an array, $[\pi(1), \pi(2), \dots, \pi(n)]$.
With these two algorithms you are well-armed for the following tasks.

(a) What is the maximum number of inversions that a permutation of n elements may have?

(b) Describe a deterministic algorithm for the following problem that runs in $O(n \log n)$ time.

Input: A set L of n non-vertical lines in the plane, and a number $a \in \mathbf{R}$.

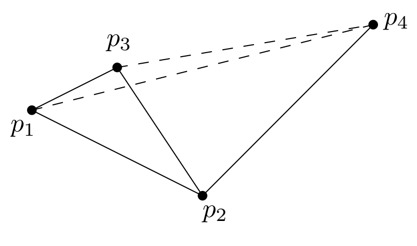
Output: The number of line intersections lying to the left of the vertical line $x = a$.

(c) Explain very briefly (without any proofs) why you could easily extend your algorithm from (b) so that it can also generate a pair of lines $\{\ell, \ell'\} \subseteq L$ drawn uniformly at random from those pairs that intersect in between two given vertical lines $x = b$ and $x = c$.

(d) For any two points p, q in the plane whose x-coordinates are not the same, we define the *slope of the pair* $\{p, q\}$ as the slope of the line that passes through p and q. Build on tasks (b) and (c) and describe a randomized algorithm for the following problem that runs in $O(n(\log n)^2)$ expected time.

Input: A set S of $n \geq 2$ points in the plane, no two of which have the same x-coordinate.

Output: A pair $\{p, q\} \subseteq S$ of points with the median slope.



You may wonder what we mean by the median slope when the number of pairs is even. The answer is that your algorithm is allowed to return either of the two values 'in the middle'. So in the example that you see on the left, the pair $\{p_3, p_4\}$ would be a correct output for your algorithm, and the pair $\{p_1, p_4\}$ would do the job, too.

[1]In case you are wondering, the $O(n \log n)$ is not meant as the expected time, but the algorithm really always finishes in $O(n \log n)$ time. The algorithm is randomized only insofar as it must at some point draw some random number that determines which inversion it will return.