Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

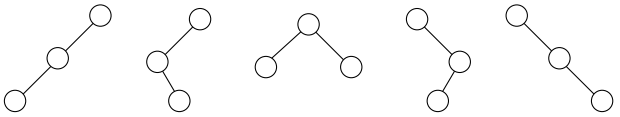
Institute of Theoretical Computer Science
Mohsen Ghaffari, Angelika Steger, Emo Welzl, Peter Widmayer



# Solution 1: Close Neighbours

(a) For $n = 2$, $Z_2^{(1)} = Z_2 = 1$ always; so $\mathbf{E}\left[Z_2^{(1)}\right] = \mathbf{E}[Z_2] = 1$.

For $n = 3$ we tabulate:

| Tree |  | | | | |
|---|---|---|---|---|---|
| Probability | 1/6 | 1/6 | 1/3 | 1/6 | 1/6 |
| $Z_3^{(1)}$ | 1 | 1 | 1 | 2 | 1 |
| $Z_3$ | 1 | 1.5 | 1 | 1.5 | 1 |

We find

$$\mathbf{E}\left[Z_3^{(1)}\right] = \frac{1}{6}(1+1+2+1) + \frac{1}{3}\cdot 1 = \frac{7}{6},$$
$$\mathbf{E}[Z_3] = \frac{1}{6}(1+1.5+1.5+1) + \frac{1}{3}\cdot 1 = \frac{7}{6}.$$

.

(b) Let $x_n := \mathbf{E}\left[Z_n^{(1)}\right]$. For $n \geq 2$ we have

$$\begin{aligned}
x_n &= \sum_{k=1}^{n} \Pr\left[\mathrm{rk}(\mathrm{root}) = k\right] \cdot \mathbf{E}\left[Z_n^{(1)} \mid \mathrm{rk}(\mathrm{root}) = k\right] \\
&= \frac{1}{n}\left(\mathbf{E}\left[Z_n^{(1)} \mid \mathrm{rk}(\mathrm{root}) = 1\right] + \mathbf{E}\left[Z_n^{(1)} \mid \mathrm{rk}(\mathrm{root}) = 2\right] + \sum_{k=3}^{n} \mathbf{E}\left[Z_n^{(1)} \mid \mathrm{rk}(\mathrm{root}) = k\right]\right) \\
&= \frac{1}{n}\left(1 + \mathbf{E}\left[D_{n-1}^{(1)}\right] + 1 + \sum_{k=3}^{n} x_{k-1}\right) \\
&= \frac{1}{n}\left(H_{n-1} + 1 + \sum_{k=2}^{n-1} x_k\right).
\end{aligned}$$

For $n \geq 3$ it follows that $nx_n - (n-1)x_{n-1} = \frac{1}{n-1} + x_{n-1}$ and hence $x_n = x_{n-1} + \frac{1}{n(n-1)}$. Expanding this formula we find

$$\begin{aligned}
x_n &= \frac{1}{n(n-1)} + x_{n-1} \\
&= \frac{1}{n(n-1)} + \frac{1}{(n-1)(n-2)} + \frac{1}{(n-2)(n-3)} + \frac{1}{(n-3)(n-4)} + \cdots + \frac{1}{3\cdot 2} + x_2
\end{aligned}$$

$$
\begin{aligned}
&= \frac{2}{n(n-2)} + \frac{1}{(n-2)(n-3)} + \frac{1}{(n-3)(n-4)} + \cdots + \frac{1}{3\cdot 2} + x_2 \\
&= \frac{3}{n(n-3)} + \frac{1}{(n-3)(n-4)} + \cdots + \frac{1}{3\cdot 2} + x_2 \\
&= \frac{n-2}{n\cdot 2} + x_2 \\
&= \frac{n-2}{n\cdot 2} + 1 \\
&= \frac{3}{2} - \frac{1}{n}.
\end{aligned}
$$

(c) First we look at $\sum_{k=1}^{n-1} Z_n^{(k)}$. This sum is basically counting the edges traversed during a post-order traversal of the tree, except that the traversal starts in node 1 (instead of the root) and ends in node $n$ (instead of the root). The post-order traversal goes by every edge exactly twice, and there are $n-1$ edges, which makes for $2(n-1)$ in total; we just have to subtract the lengths of the paths missing from the post-order traversal: the path from the root to node 1, and the path from node $n$ back to the root. Thus:

$$
\begin{aligned}
Z_n &= \frac{1}{n-1} \sum_{k=1}^{n-1} Z_n^{(k)} \\
&= \frac{1}{n-1} \left(2(n-1) - D_n^{(1)} - D_n^{(n)}\right) \\
&= 2 - \frac{D_n^{(1)} + D_n^{(n)}}{n-1}
\end{aligned}
$$

from which we conclude

$$
\mathbf{E}[Z_n] = 2 - \frac{\mathbf{E}\left[D_n^{(1)}\right] + \mathbf{E}\left[D_n^{(n)}\right]}{n-1} = 2 - \frac{2\,\mathbf{E}\left[D_n^{(1)}\right]}{n-1} = 2 - 2\frac{H_n - 1}{n-1}.
$$

(d) Given the formula for $Z_n$ obtained in (c), we can simply observe that for all $n \geq 2$, $Z_n < 2$ holds *always*; so $N = 3$ always. If we do not have that formula, we can still use the hint and proceed as follows.

Let $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$ denote the events that the first (second, third) random search tree has $Z_n < 3$, and let $N = [\mathcal{E}_1] + [\mathcal{E}_2] + [\mathcal{E}_3]$. We are asked to show that $\mathbf{E}[N] \geq 1$.

Indeed,

$$
\begin{aligned}
\mathbf{E}[N] &= \mathbf{E}\big[[\mathcal{E}_1]\big] + \mathbf{E}\big[[\mathcal{E}_2]\big] + \mathbf{E}\big[[\mathcal{E}_3]\big] \\
&= \Pr[\text{the first tree has } Z_n < 3] + \Pr[\text{the second tree has } Z_n < 3] \\
&\quad + \Pr[\text{the third tree has } Z_n < 3] \\
&= 3\Pr[Z_n < 3] \\
&= 3\left(1 - \Pr[Z_n \geq 3]\right) \\
&\geq 3\left(1 - \frac{\mathbf{E}[Z_n]}{3}\right) \qquad \text{(Markov's inequality)} \\
&\geq 3\left(1 - \frac{2}{3}\right) = 1.
\end{aligned}
$$

## Solution 2: Weighted Random Search Trees

(a) We use the idea of the proof of lemma 1.5 in the lecture notes. Abbreviate

$$M := \{\min\{i,j\} \,..\, \max\{i,j\}\}.$$

**First Observation:** If the the root is contained in the set $M$, then the only way that $j$ can be an ancestor of $i$ is to have $\mathrm{rk}(\mathrm{root}) = j$. Formally,

$$\begin{aligned}\mathbf{E}\Big[A_i^j \;\Big|\; \mathrm{rk}(\mathrm{root}) \in M\Big] &= \Pr\Big[j \text{ ancestor of } i \;\Big|\; \mathrm{rk}(\mathrm{root}) \in M\Big] \\ &= \Pr\Big[\mathrm{rk}(\mathrm{root}) = j \;\Big|\; \mathrm{rk}(\mathrm{root}) \in M\Big] \\ &= \frac{p(j)}{\sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell)},\end{aligned}$$

where the last step was just applying the definition of conditional expectation.

**Second Observation:** If $i = j$ then $A_i^j = 1$ always, and in this case the claim holds vacuously.

We now prove the formula

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

for the case $i \neq j$, by induction on $n \geq |j-i|+1$.

*Induction base case:* $n = |j-i|+1$. Then we must have $\min\{i,j\} = 1$ and $\max\{i,j\} = n$. But in this case the claim follows immediately from the 'first observation', because the condition $\mathrm{rk}(\mathrm{root}) \in M$ is now always satisfied.

*Induction step:* $n > |j-i|+1$. Let $x$ denote the root (considered fixed for the moment). If $x$ is not an element of $M$, then $i$ and $j$ are located in the same (left or right) subtree of $x$. Let us assume w.l.o.g. that they are located in the left subtree of $x > \max\{i,j\}$. The left subtree of $x$ is again a random search tree, but its set of keys is strictly smaller ($|S^{<x}| < n$); so we can apply the induction hypothesis to find

$$\begin{aligned}&\mathbf{E}\Big[A_i^j \;\Big|\; \mathrm{rk}(\mathrm{root}) = x,\ x \notin M\Big] \\ &= \mathbf{E}\Big[A_i^j \text{ interpreted as a random variable on trees on the set } S^{<x} \;\Big|\; \mathrm{rk}(\mathrm{root}) = x\Big] \\ &= \frac{p^{<x}(j)}{\sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p^{<x}(\ell)} \\ &= \frac{p(j)}{\sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell)}.\end{aligned}$$

Since the calculation was valid for arbitrary fixed $x$, it remains valid if we do not fix $x$ anymore:

$$\mathbf{E}\Big[A_i^j \;\Big|\; \mathrm{rk}(\mathrm{root}) \notin M\Big] = \frac{p(j)}{\sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell)}.$$

On the other hand, we have already seen in the 'first observation' that we have

$$\mathbf{E}\Big[A_i^j \;\Big|\; \mathrm{rk}(\mathrm{root}) \in M\Big] = \frac{p(j)}{\sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell)},$$

so the claim follows by total expectation.

(b) We start from the well-known inequality $e^{\xi} \geq 1 + \xi$ $(\xi \in \mathbf{R})$. Plugging in $\xi = -\frac{x}{y}$ we find

$$e^{-x/y} \geq 1 - \frac{x}{y}.$$

From $x < y$ it follows that $1 - \frac{x}{y}$ is positive; so we can apply the monotonic function ln to both sides of the previous inequality and obtain

$$-\frac{x}{y} \geq \ln\left(1 - \frac{x}{y}\right) = \ln\frac{y-x}{y} = \ln(y-x) - \ln y$$

which yields the claim.

(c) From the lecture we know

$$D_n^{(i)} = \sum_{\substack{j \in [n] \\ j \neq i}} A_i^j.$$

Using linearity of expectation and task (a), we obtain

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

The hint gives

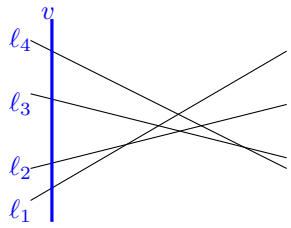
$$\begin{aligned}
\mathbf{E}\left[D_n^{(i)}\right] &\leq \sum_{\substack{j \in [n] \\ j \neq i}} \left( \ln\left( \sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell) \right) - \ln\left( \sum_{\ell=\min\{i,j\}}^{\max\{i,j\}} p(\ell) - p(j) \right) \right) \\
&= \underbrace{\sum_{j=1}^{i-1} \left( \ln\left( \sum_{\ell=j}^{i} p(\ell) \right) - \ln\left( \sum_{\ell=j+1}^{i} p(\ell) \right) \right)}_{\ln\left( \sum_{\ell=1}^{i} p(\ell) \right) - \ln\left( \sum_{\ell=i}^{i} p(\ell) \right)} + \underbrace{\sum_{j=i+1}^{n} \left( \ln\left( \sum_{\ell=i}^{j} p(\ell) \right) - \ln\left( \sum_{\ell=i}^{j-1} p(\ell) \right) \right)}_{\ln\left( \sum_{\ell=i}^{n} p(\ell) \right) - \ln\left( \sum_{\ell=i}^{i} p(\ell) \right)} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(Telescoping sums)} \\
&= \ln\left( \sum_{\ell=1}^{i} p(\ell) \right) - \ln p(i) + \ln\left( \sum_{\ell=i}^{n} p(\ell) \right) - \ln p(i) \\
&\leq \ln 1 - \ln p(i) + \ln 1 - \ln p(i) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{($p$ is a probability measure)} \\
&= 2 \ln \frac{1}{p(i)}.
\end{aligned}$$

## Solution 3: Finding the Median Slope

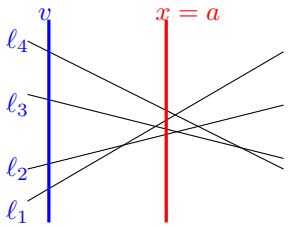
(a) An upper bound is $\binom{n}{2}$, because there are only so many pairs of indices. This bound is attained by the permutation $[n, n-1, \ldots, 1]$; so the answer is $\binom{n}{2}$.

(b) The algorithm first computes the slope of each line and then sorts the set of lines by slope. Let $\ell_1, \ldots, \ell_n$ denote the lines in sorted order, by *decreasing* slope and in the case of equal slopes by decreasing $y$-intercept. For the purpose of visualizing the situation, note that this

numbering is the same as the order in which you would encounter the lines if you took a vertical line $v$, positioned so far to the left that all L-intersections lie strictly to the right of $v$, and travelled along $v$ from bottom to top:




Second, the algorithm computes for each line $\ell_i$ the y-coordinate $y_i$ of its intersection with the vertical line $x = a$. Now the algorithm takes an array of length $n$, initialized with the values $[1, \ldots, n]$, and sorts it according to the values $y_i$. If it happens that $y_i = y_j$ for some $i \neq j$, then the entry that belongs to line with the *larger* slope is put before the other one in the sorted order.[1] Let the resulting array be called $\alpha$.




In this example, the array $\alpha$ will be $[2, 3, 1, 4]$.

Going back to our picture, the array $\alpha$ now contains the indices of our lines in the order in which you would encounter them if you travelled along the other vertical line, $x = a$, from bottom to top.

Finally the algorithm calls the algorithm from Fact A on the array $\alpha$ and returns the result.

*Correctness:* We note that two lines $\ell_i, \ell_j$ with $i < j$ have an intersection to the left of the vertical line $x = a$ if and only if $i$ occurs before $j$ within the array $\alpha$ – in other words, if and only if $\{i, j\}$ is an inversion of the permutation given by $\alpha$. The number of such pairs $\{i, j\}$ is exactly what our algorithm returns.

*Running time:* The running time is dominated by the sorting and the call to the algorithm from Fact A. $\rightsquigarrow$ $O(n \log n)$.

(c) We could proceed as in (a), except that

- we would use Fact B instead of Fact A, and
- we would use different vertical lines: Instead of $v$ we would use the line $x = b$, and instead of $x = a$ we would use the line $x = c$.

(d) Let $S^* = \{p^* : p \in S\}$, where $p^*$ denotes the dual (line) of the point $p$. Since no two of our points have the same x-coordinate, every two points $p, q \in S$ determine a non-vertical line $\ell$. Since point-line duality is incidence-preserving, the lines $p^*$ and $q^*$ intersect in the point $\ell^*$. The x-coordinate of $\ell^*$ equals the slope of $\ell$, i. e., the slope of the pair $\{p, q\}$.

From this we see that, if $\{p, q\}$ is the pair of points from $S$ with the median slope, then $\{p^*, q^*\}$ is the pair of lines from $S^*$ whose intersection has the median x-coordinate.

[1] This is for counting the number of intersections *strictly* to the left. If we wanted instead to also count the intersections *on* the line, we could simply do the tie-breaking in the opposite way.

Since the set $S^*$ can simply be computed in linear time, it thus suffices to describe an algorithm which, given a set $L$ of $n$ lines no two of which have the same slope, computes the pair of lines whose intersection has the median $x$-coordinate, in $O(n \log^2 n)$ expected time.

We now describe an algorithm to do that. The idea is to do Quickselect on the set of line intersections, with the caveat that we cannot afford to compute all line intersections: Since no two lines are parallel, the number of line intersections is

$$s := \binom{n}{2}.$$

During the run of the algorithm we maintain an interval $[b, c]$ with the property that the median $x$-coordinate that we are looking for is contained in this interval.

- In the beginning we let $b = -\infty$ and $c = +\infty$.
- In a loop, we do the following:
  (a) If $b = c$ then we stop and return any pair of lines intersecting *on* the vertical line $x = b$. This we can compute e. g. by using algorithm (c) to compute a random pair that intersect *between or on* $b$ and $c$.
  (b) Otherwise ($b < c$), we call the algorithm from task (c) to obtain a random pair $\{\ell, \ell'\}$ of lines that intersect strictly between $b$ and $c$.[2] Now we compute the intersection $\ell \cap \ell'$; let $a$ denote its $x$-coordinate. Call the algorithm from (b) on $(L, a)$ to obtain $K :=$ the number of intersections that lie strictly to the left of the vertical line $x = a$.
    - If $K + 1 \geq \lceil s/2 \rceil$, we set $c := a$ and we continue with the loop.
    - If $K + 1 < \lceil s/2 \rceil$, we set $b := a$ and we continue with the loop.

*Correctness:* Follows from the loop invariant that the median $x$-coordinate is contained in the interval $[b, c]$.

*Running time:* Every iteration of the loop takes $O(n \log n)$ time, due to the calls to (b) and (c). To obtain the overall bound of $O(n \log^2 n)$ for the expected time, it remains to argue why the expected number of loop iterations is $O(\log n)$.

To this end, consider first the case that our set of lines is such that no two line intersections have the same $x$-coordinate. (In this case the argument becomes a bit simpler because there is now no tie-breaking to be considered in the ordering of the line intersections by $x$-coordinate.) The expected number of loop iterations is now exactly the same as the expected depth of the median key in a random search tree: The set of keys is the set of $x$-coordinates occuring, which are now exactly $s = \binom{n}{2}$; so with Theorem 1.6 we get the bound $2 \ln \binom{n}{2} = O(\log(n^2)) = O(\log n)$.

Now, what if some line intersections have the same $x$-coordinates? We can still use as a model a random search tree on the set $X$ of $x$-coordinates that occur, but we must take into account that the probability of a specific $x$-coordinate to be picked as the pivot $a$ is proportional to the number of line intersections on the vertical line $x = a$. Thus, in general, the expected number of loop iterations equals the expected depth of the median key in a *weighted* random search tree, where the set of keys is the set $X$, and every key $i$ has a weight $p(i) \in \{\frac{1}{|X|}, \frac{2}{|X|}, \ldots, \frac{|X|}{|X|}\}$. Using exercise 2c we get (again) the bound $2 \ln |X| \leq 2 \ln s = O(\log n)$.

[2] We can easily extend (c) to allow for infinite values, because a vertical line very far to the left meets the lines from $L$ in the order of decreasing slope, while a vertical line very far to the right meets the lines from $L$ in the order of increasing slope.