Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute of Theoretical Computer Science
Bernd Gärtner, Rasmus Kyng, Angelika Steger, David Steurer, Emo Welzl

Algorithms, Probability, and Computing **Solutions KW41** **HS23**

Solution 1

Let N_{deep} denote the number of nodes of depth $n-1$. We observe that a binary search tree B for n vertices has one node of depth $n-1$ if and only if it is a path of length $n-1$. Let p_n denote the probability that there is a node of depth $n-1$. We have

$$\mathbf{E}\left[N_{\text{deep}}\right] = p_n \cdot 1 + (1-p_n) \cdot 0 = p_n.$$

So it remains to compute p_n. Clearly, $p_1 = 1$. For $n \geq 2$ we apply induction. Note that if a tree T is a path then its root is either the smallest or the largest key. Hence

$$\begin{aligned} p_n &= \underbrace{\Pr\left[\text{rk}(\text{root}) = 1\right]}_{\frac{1}{n}} \cdot \underbrace{\Pr\left[\text{one node has depth } n-1 | \text{rk}(\text{root}) = 1\right]}_{p_{n-1}} + \\ &\quad \underbrace{\Pr\left[\text{rk}(\text{root}) = n\right]}_{\frac{1}{n}} \cdot \underbrace{\Pr\left[\text{one node has depth } n-1 | \text{rk}(\text{root}) = n\right]}_{p_{n-1}} \\ &= \frac{2}{n} \cdot p_{n-1} \end{aligned}$$

Induction yields that $p_n = \frac{2^{n-1}}{n!} \cdot p_1 = \frac{2^{n-1}}{n!}$ and so we are done.

Remark. Note that this also provides us with the number of trees on n nodes that have height $n-1$: since each such tree is a path of length $n-1$, for each such tree there is exactly one ordering of n keys which produces this search tree (since a parent must always be inserted before its child). So each tree of height $n-1$ separately has a probability of $1/n!$. Since in total $p_n = 2^{n-1}/n!$, we conclude that there must be exactly 2^{n-1} trees of this type.

Of course, we could have found this number more easily: the number of trees on n nodes that are a single path is 2^{n-1} simply because for each edge (of which there are $n-1$ many) we can decide whether it should point to the left or to the right.

Solution 2

Variant 1: Computation via conditioning on the rank of the root. The usual way, we first obtain

$$\mathbf{E}[S_n] = \sum_{i=1}^{n} \underbrace{\mathbf{E}[S_n \mid \text{rk}(\text{root}) = i]}_{(\star)} \cdot \underbrace{\Pr[\text{rk}(\text{root}) = i]}_{\frac{1}{n}}$$

where

$$(\star) = \begin{cases} n, & \text{if } i = 1, \\ \mathbf{E}[S_{i-1}], & \text{otherwise} \end{cases}$$

Denote $s_n := \mathbf{E}[S_n]$. Then this yields a recurrence of the form.

$$s_n = \frac{1}{n}\left(n + \sum_{i=2}^{n} s_{i-1}\right),$$

holding for all $n \geq 1$. As we are, by now, proficient in solving recurrences, let us multiply by n and then instantiate the recurrence for both n and $n-1$ so that for $n \geq 1$ we have

$$ns_n = n + \sum_{i=1}^{n-1} s_i \tag{1}$$

and for $n \geq 2$,we get

$$(n-1)s_{n-1} = n - 1 + \sum_{i=1}^{n-2} s_i \tag{2}$$

Then subtracting (??) from (??), we obtain

$$ns_n - (n-1)s_{n-1} = 1 + s_{n-1}.$$

Rearranging and dividing by n, this yields

$$s_n = \frac{1}{n} + s_{n-1}.$$

We are familiar with this recursion and know that telescoping it out will produce $s_n = H_n$.

Variant 2: Computation via indicator variables. Alternatively, we can use the well-known indicator variables

$$A_i^j := [\text{node } j \text{ is an ancestor of node } i]$$

In that case we obviously have

$$S_n = \sum_{i=1}^{n} A_i^1$$

$$\Rightarrow s_n = \mathbf{E}[S_n] = \mathbf{E}\left[\sum_{i=1}^{n} A_i^1\right] = \sum_{i=1}^{n} \mathbf{E}\left[A_i^1\right].$$

Those expectations have been computed in the lecture notes where we have obtained that

$$\mathbf{E}\left[A_i^j\right] = \frac{1}{|i-j|+1}$$

and thus

$$\Rightarrow \mathbf{E}\left[A_i^1\right] = \frac{1}{i-1+1} = \frac{1}{i}.$$

Therefore, this variant, too, yields

$$s_n \;=\; \sum_{i=1}^{n} \mathbf{E}\left[A_i^1\right] \;=\; \sum_{i=1}^{n} \frac{1}{i} \;=\; H_n.$$

Solution 3

(1) First we compute that $a_1 = 1$ and $a_2 = \frac{3}{2}$. Now for $n \geq 3$, we multiply the recurrence relation by n and write it once for n and once for $n-1$. This yields

$$n a_n = n + \sum_{i=1}^{n-1} a_i \tag{3}$$

and

$$(n-1) a_{n-1} = (n-1) + \sum_{i=1}^{n-2} a_i. \tag{4}$$

Now subtracting (??) from (??), we obtain

$$n a_n - (n-1) a_{n-1} = 1 + a_{n-1}$$

and thus

$$a_n = \frac{1}{n} + a_{n-1}.$$

This recursion can easily be telescoped from which we obtain

$$a_n = \frac{1}{n} + \frac{1}{n-1} + \ldots + \frac{1}{3} + \underbrace{a_2}_{=1/2+1} = H_n.$$

Therefore, $a_n = H_n$ for all $n \in \mathbf{N}$.

(2) We first compute that $b_1 = 1$ and $b_2 = 3$. Now for $n \geq 3$

$$b_n = 2 + \sum_{i=1}^{n-1} b_i \tag{5}$$

and

$$b_{n-1} = 2 + \sum_{i=1}^{n-2} b_i \tag{6}$$

Now subtracting (??) from (??), we obtain

$$b_n - b_{n-1} = b_{n-1},$$

therefore

$$b_n = 2\, b_{n-1}$$

and thus

$$b_n = 2^{n-2} b_2 = 3 \cdot 2^{n-2}.$$

Therefore, $b_1 = 1$ and $b_n = 3 \cdot 2^{n-2}$ for all $n \geq 2$.

(3) We first compute that $c_0 = 0$ and $c_1 = 0$. Then for $n \geq 2$, we first note that

$$\sum_{i=1}^{n} \frac{c_{i-1} + c_{n-i}}{2} = \sum_{i=1}^{n} c_{i-1}$$

which then allows us to write the simpler recurrences for c_n and c_{n-1}

$$c_n = n - 1 + \sum_{i=1}^{n} c_{i-1} \tag{7}$$

and

$$c_{n-1} = n - 2 + \sum_{i=1}^{n-1} c_{i-1} \tag{8}$$

If we now subtract (??) from (??), then

$$c_n - c_{n-1} = 1 + c_{n-1}$$

and thus

$$c_n = 1 + 2\, c_{n-1}.$$

For telescoping, it turns out to be convenient to divide the recurrence by 2^n, then we have

$$\frac{c_n}{2^n} = \frac{1}{2^n} + \frac{c_{n-1}}{2^{n-1}}$$

and we can telescope for $c_n/2^n$, yielding

$$\frac{c_n}{2^n} = \frac{1}{2^n} + \frac{1}{2^{n-1}} + \cdots \frac{1}{2^2} + \frac{c_1}{2^1} = \frac{1}{2} - \frac{1}{2^n}.$$

Therefore, $c_0 = 0$ and $c_n = 2^{n-1} - 1$ for $n \in \mathbf{N}$.

(4) We compute that $d_0 = 0$ and $d_1 = 1$. Then for $n \geq 2$, we may instantiate

$$d_n = 1 + 2 \sum_{i=0}^{n-1} (-1)^{n-i} d_i \tag{9}$$

and

$$d_{n-1} = 1 + 2 \sum_{i=0}^{n-2} (-1)^{n-1-i} d_i \tag{10}$$

This time, *adding* the recurrences (??) + (??) turns out to be more helpful as it yields

$$d_n + d_{n-1} = 2 - 2\, d_{n-1}$$

and thus

$$d_n = 2 - 3\, d_{n-1}.$$

To simplify telescoping, we rearrange this to

$$d_n - \frac{1}{2} = -3\,(d_{n-1} - \frac{1}{2})$$

and then use the substitution

$$f_n := d_n - \frac{1}{2}$$

from which

$$f_n = -3\, f_{n-1}.$$

Telescoping now immediately yields

$$f_n = f_1\,(-3)^{n-1},$$

thus

$$f_n = \frac{1}{2}\,(-3)^{n-1}$$

and so undoing the substitution we end up with

$$d_n = \frac{1}{2}\,(-3)^{n-1} + \frac{1}{2}.$$

In conclusion, $d_0 = 0$ and $d_n = \frac{1}{2}(1 + (-3)^{n-1})$ for $n \in \mathbf{N}$.

(5) For $n \geq 1$, we have the recurrence

$$e_n = 1 + n e_{n-1}.$$

It is convenient to divide this recurrence by $n!$ as then

$$\frac{e_n}{n!} = \frac{1}{n!} + \frac{e_{n-1}}{(n-1)!}$$

is a simple recurrence for the series $e_n/n!$. Telescoping it yields

$$\frac{e_n}{n!} = \frac{1}{n!} + \frac{1}{(n-1)!} + \ldots + \frac{1}{1!} + \frac{e_0}{0!} = \sum_{i=0}^{n} \frac{1}{i!}$$

for all $n \geq 1$. Therefore,

$$e_n = \left(\sum_{i=0}^{n} \frac{1}{i!} \right) n!$$

for all $n \in \mathbf{N}_0$ (note that by convention, $0! = 1$).

The above expression may be explicit but it still involves a sum, so we should routinely ask whether there is way to simplify it. The expression in the sum of course makes us think of the function $\exp(\cdot)$. In fact, we know that

$$\sum_{i=0}^{\infty} \frac{1}{i!} = e = 2.71...,$$

and thus in the case of our sequence, $e_n < e\,n!$. Due to the nature of the recursion, however, e_n is always an integer, thus we also have $e_n \leq \lfloor e\,n! \rfloor$. Let us compare how close this bound is to the truth.

n	0	1	2	3	4
$e\,n!$	2.71...	2.71...	5.44...	16.31...	65.24...
$\lfloor e\,n! \rfloor$	2	2	5	16	65
e_n	1	2	5	16	65

So it seems the bound is tight starting $n = 1$. And indeed, if we check,

$$e\,n! - e_n = n! \sum_{i=n+1}^{\infty} \frac{1}{i!} = \frac{1}{n+1} + \frac{1}{(n+1)(n+2)} + \dots$$

is decreasing in n. Since it is below 1 for $n = 1$, it stays below 1 for all n.

We have established

$$e_n = \begin{cases} 1, & \text{if } n = 0; \\ \lfloor e\,n! \rfloor, & \text{otherwise.} \end{cases}$$

Solution 4

Let L_n be the number of leaves and $l_n := \mathbf{E}[L_n]$.

Clearly, $l_0 = 0$ and $l_1 = 1$. Now for larger n, if the root has rank k, then the number of leaves is the sum of the number of leaves in the left subtree and the number of leaves in the right subtree. We thus get

$$\mathbf{E}[L_n] = \sum_{k=1}^{n} \underbrace{\mathbf{E}[L_n \mid \mathrm{rk}(\mathrm{root}) = k]}_{l_{k-1}+l_{n-k}} \cdot \underbrace{\Pr[\mathrm{rk}(\mathrm{root}) = k]}_{\frac{1}{n}}.$$

Therefore,

$$l_n = \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ \frac{2}{n} \sum_{k=0}^{n-1} l_k & \text{if } n \geq 2. \end{cases}$$

To solve the recurrence, first we compute that $l_2 = 1$. Now for $n \geq 3$, we multiply the recurrence relation by n and write it once for n and once for $n-1$. This yields

$$n l_n = 2 \sum_{i=1}^{n-1} l_i \tag{11}$$

and

$$(n-1) l_{n-1} = 2 \sum_{i=1}^{n-2} l_i. \tag{12}$$

Now subtracting (??) from (??), we obtain

$$n l_n - (n-1) l_{n-1} = 2 l_{n-1}$$

and thus

$$n l_n = (n+1) l_{n-1}.$$

Dividing by $n(n+1)$ yields

$$\frac{l_n}{n+1} = \frac{l_{n-1}}{n}.$$

Repeated application of this equality demonstrates that

$$\frac{l_n}{n+1} = \frac{l_2}{3} = \frac{1}{3}.$$

Therefore, $l_n = \frac{n+1}{3}$.

Solution 5

(a) We consider the event that the root of the treap changes after the insertion of key i is completed. Let X_i be the indicator variable for that event. According to how a treap works, we now have

$$X_i = 1 \;\Leftrightarrow\; \text{the node } i \text{ is rotated to the top when it is inserted,}$$

thus equivalently,

$$X_i = 1 \;\Leftrightarrow\; i \text{ receives minimum priority among the keys } \{1, \ldots, i\}$$

or equivalently

$$X_i = 1 \;\Leftrightarrow\; i \text{ appears first in the random sorting order of } \{1, \ldots, i\}.$$

From the last characterization it is evident that $\Pr[X_i = 1] = \mathbf{E}[X_i] = \frac{1}{i}$. Therefore the expected number of changes of the root of the treap is $\mathbf{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbf{E}[X_i] = \sum_{i=1}^{n} \frac{1}{i} = H_n$.

(b) For this part (and also then Part (c)) of the exercise, it is crucial to make use of the special (increasing) order of the keys in which the nodes are inserted into the treap. The fact that the key of every current node i is strictly larger than all keys already in the treap implies that temporarily —at the start of its insertion— it must always end up as the right-most leaf of the current treap. This means that after the necessary rotations for its insertion are completed, our node *must* end up somewhere on the *right spine* of the treap. Furthermore, all rotations that will ever be performed in the whole process are therefore "right-to-left"-rotations of some edge on the current right spine, in other words rotating the current node i upwards on the right spine.

Knowing what all the rotations in the process look like, we can now think about the right child of the root, and conclude the following:

Observation 1. *Node i will only ever occur as the right child of the root if it does so directly after its own insertion.*

Proof. All rotations that occur in our process are rotations of an edge of the right spine: The currently inserted node i moves one step up on the right spine, and replaces some node there which will be moved to the left, away from the spine.

This also implies that the current root can never become the right child of the root anymore.

After the insertion of our node i is completed, and the later nodes are inserted, i will only potentially get rotated away from the right spine, and never upwards in the tree, which proves our claim. □

Now we want to compute the probability of the event

$$Y_i := i \text{ is the right child of root (after } i \text{ was inserted into the treap).}$$

We can compute $\Pr[Y_i]$ by the standard trick of conditioning on the root of the treap, for $1 \leq i \leq n$:

$$\Pr[Y_i] = \sum_{r=1}^{i} \underbrace{\Pr[Y_i \mid \text{root is } r]}_{*} \underbrace{\Pr[\text{root is } r]}_{**}$$

where

$$* = \begin{cases} \Pr[i \text{ gets minimum priority among } \{r+1, \ldots, i\}] = \frac{1}{i-r} & \text{if } r < i, \\ 0 & \text{if } r = i. \end{cases}$$

$$** = \Pr[r \text{ gets minimum priority among } \{1, \ldots, i\}] = \frac{1}{i}.$$

Therefore we obtain

$$\Pr[Y_i] = \frac{1}{i} \sum_{r=1}^{i-1} \frac{1}{i-r} = \frac{1}{i} \sum_{s=1}^{i-1} \frac{1}{s} = \frac{1}{i} H_{i-1}.$$

Alternative proof. If we consider the event

$$Y_i^k := i \text{ is the right child of root after } k \text{ nodes have been inserted into the treap,}$$

for $1 \leq i \leq k \leq n$, and use the standard approach of conditioning on the root, we obtain

$$\Pr\left[Y_i^k\right] = \frac{1}{k} \sum_{r=1}^{i-1} \frac{1}{k-r} = \frac{1}{k} \sum_{s=k-i+1}^{k-1} \frac{1}{s} = \frac{1}{k}(H_{k-1} - H_{k-i}),$$

where we defined $H_0 := 0$. Now we again have to use the crucial Observation 1 from above, that "i will only occur as the right child of the root if it does so directly after its own insertion". Formally, this means that the $\bigvee_{k=i}^{n} Y_i^k = Y_i^i$, therefore the desired probability of the union of the events is $\Pr\left[Y_i^i\right] = \frac{1}{i} H_{i-1}$.

(c) Using the same argument as in the beginning of the previous part (ii), we know: The only way that the left child of the root of the treap can change in the given process is by rotating the top-most edge of the right spine, meaning that the

rotation replaces the root with its former right child, and therefore the 'old' root now becomes the new left child of the new root. Therefore the indicator variable

$$X_i = [\text{the root changes after node } i \text{ is inserted}]$$

is equivalent to

$$Z_i = [\text{the left child of the root changes after node } i \text{ is inserted}]$$

except at the very first insertion, for $i = 1$, where the treap was still empty, so no rotation will be performed.

Therefore by using the result of (i) we have that $\Pr[Z_i = 1] = \Pr[X_i = 1] = \frac{1}{i}$ for $i \geq 2$, and $\Pr[Z_1 = 1] = 0$. So $\mathbf{E}\left[\sum_{i=1}^{n} Z_i\right] = \sum_{i=1}^{n} \mathbf{E}[Z_i] = H_n - 1$.

Solution 6

To begin with, note that when we talk about the 'number of comparisons', we mean the number of times one element is compared to another element of the sequence to sort. We do not count internal comparisons of the algorithm (like when the algorithm compares the index of an element to another index).

Let a, b be two distinct elements of rank i and j, respectively. In the script, a mapping is described from the possible runs of quicksort(S) to binary search trees (with the same distribution as our random search trees). According to that mapping, a, b are compared in quicksort(S) if and only if in the corresponding search tree, either b is the ancestor of a or a is the ancestor of b. We have denoted these events using indicator variables by $A_i^j = 1$ and $A_j^i = 1$, respectively.

$$\begin{aligned} \Pr[a, b \text{ are compared in quicksort}(S)] &= \Pr\left[A_i^j = 1 \vee A_j^i = 1\right] \\ &= \Pr\left[A_i^j = 1\right] + \Pr\left[A_j^i = 1\right] = \frac{2}{|i-j|+1}. \end{aligned}$$

The first equality follows from the fact that the two events A_i^j and A_j^i are disjoint (for $i \neq j$).