ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

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

Solution 1

(a) Let C be an almost minimum cut of size K. First, we prove that the probability that C survives (none of its edges are contracted) until the remaining graph has 4 nodes is bounded by $\Omega(1/n^4)$.

Notice that every node of G has degree at least $\mu(G)$, and thus $|E(G)| \geq \frac{n\mu(G)}{2}$, where $\mu(G)$ stands for the minimum size of a cut in G. The probability that C survives the first round is at least

$$1 - \frac{K}{|E(G)|} \geq 1 - \frac{K}{n\mu(G)/2} \geq 1 - \frac{4}{n}$$

where we used $K \leq 2\mu(G)$.

Recall, $\mu(G/e) \geq \mu(G)$ for any edge e in G, i.e., contracting an edge does not decrease the minimum size of a cut. Therefore with a similar argument as above, running the algorithm BasicMinCut(G) until we have at most 4 nodes, the probability that C survives is at least

$$(1-\frac{4}{n})(1-\frac{4}{n-1})(1-\frac{4}{n-2})(1-\frac{4}{n-3})(1-\frac{4}{n-4})\cdots(1-\frac{4}{5}) =$$
$$\frac{n-4}{n}\cdot\frac{n-5}{n-1}\cdot\frac{n-6}{n-2}\cdot\frac{n-7}{n-3}\cdot\frac{n-8}{n-4}\cdots\frac{1}{5} = \Omega(\frac{1}{n^4}).$$

Now, we have at most 4 nodes and a multi-graph on 4 nodes has constantly many cuts. Therefore, the probability that the algorithm picks C is lower-bounded by a constant. This implies that the algorithm returns K with probability $\Omega(1/n^4)$.

(b) Consider the following multigraph, $G = (V, E)$. Let $V = \{s, t, v_1, \cdots, v_n\}$ and $E = E_S \cup E_T$ where $E_S = \{(s, v_i) : i = 1, \cdots, n\}$ and $E_T = \{(v_i, t) : i = 1, \cdots, n\}$ and the edges in E_T occur twice. The minimum s-t cut consists of the node s and has value n. So there is one edge from s to each node v_i for all i, and two edges from v_i to t for each i. Then, in order for the algorithm to terminate successfully, only edges from E_T can be contracted. If an edge from E_S is contracted, the cut value returned will be at least $n+1$, as neither of the two edges from the contracted node can be chosen (as the contracted node is now the s-node). By this same argument, every time an edge is contracted, three edges (including the one contracted) are no longer possible choices as they are either contracted or edges connecting the s-node to the t-node. Thus, at every iteration the probability of choosing a contraction preserving the possibility of success is $\frac{2}{3}$, and there are a total of n iterations. Then the probability of success is $(\frac{2}{3})^n$.

Solution 2

(a) We give a proof by induction of the following (stronger) statement, which then immediately implies the desired result.

Claim 1. *Let $S = \{l, l+1, \dots, r\}$ be any set of keys consisting of consecutive natural numbers, and let $j < i$ be two specific keys from that set. Consider the corresponding random search tree $\tilde{B}_S$ (as defined on the exercise sheet) and the random variable A_i^j. Then,*

$$\Pr\left[A_i^j = 1\right] = \frac{2j}{(i-j+1)(i+j)}.$$

Proof. The induction is over the size $n := r - l + 1$ of the set $S = \{l, l+1, \dots, r\}$. We consider the following two cases, one being straight-forward and the other inductive.

- $i - j + 1 = n$ (in words, $j = l$ is the smallest key in S and $i = r$ is the largest). Clearly, in this case j is an ancestor of i if and only if j is the root of the whole tree $\tilde{B}_S$. By definition, this happens with the following probabilty.

$$\Pr[A_i^j = 1] = \Pr[\text{“}j \text{ is root in } \tilde{B}_S\text{”}] = \frac{j}{\sum_{y \in S} y} = \frac{j}{\binom{i+1}{2} - \binom{j}{2}} = \frac{2j}{(i-j+1)(i+j)}$$

- $i - j + 1 < n$ (in words, there are keys in S that are smaller than j or bigger than i). In this second case, we use the law of total probability and condition on the event E that the root of $\tilde{B}_S$ is in the range $j, j+1, \dots, i$.

$$\Pr[A_i^j = 1] = \underbrace{\Pr[A_i^j = 1 \mid E]}_{\overset{(1)}{=} \frac{2j}{(i-j+1)(i+j)}} \cdot \Pr[E] + \underbrace{\Pr[A_i^j = 1 \mid \bar{E}]}_{\overset{(2)}{=} \frac{2j}{(i-j+1)(i+j)}} \cdot \Pr[\bar{E}] = \frac{2j}{(i-j+1)(i+j)}$$

Equality (1) follows for the same reasons as in the first case; conditioned on E, j is an ancestor of i if and only if j is the root of $\tilde{B}_S$ and thus we get the same probability. Equality (2) is where we use Claim 1 inductively; for example, if the root of $\tilde{B}_S$ is equal to some key $k < j$, then we apply Claim 1 to the right subtree of $\tilde{B}_S$, which is defined over the (strictly smaller) set of keys $S^{>k} = \{k+1, k+2, \dots, r\}$; analogously, for $k > i$ we apply Claim 1 to left subtree of $\tilde{B}_S$, which is defined over the (strictly smaller) set of keys $S^{<k} = \{l, l+1, \dots, k-1\}$. □

Alternative solution. Let $S = \{l, l+1, \dots, r\}$ be any set of consecutive natural numbers. We define a new probability distribution over binary search trees with key set S in the following iterative manner. We choose keys randomly from S, one after the other, and then insert them in sequence into an initially empty search tree. In the k-th iteration, every potential key $x \in S_k$ is chosen with probability $\frac{x}{\sum_{y \in S_k} y}$, where $S_k \subseteq S$ denotes the subset of remaining keys that have not yet been inserted in the previous $k-1$ iterations. In particular, this means that in the first iteration, every potential key $x \in S$ is chosen with probability $\frac{x}{\sum_{y \in S} y}$, thus becoming the root of the tree.

We use the symbol $\tilde{\Pr}$ when referring to this new probability distribution. In particular, for any fixed binary search tree T with keys S, we let $\Pr[T]$ denote the probability of T according to the recursive definition given on the exercise sheet, and we let $\tilde{\Pr}[T]$ denote the probability of T according to the iterative definition given in the preceding paragraph.

Claim 2. *For any set of keys of the form $S = \{l, l+1, \dots, r\}$, the two defined probability distributions are identical. That is, for every tree T with keys S, we have $\Pr[T] = \tilde{\Pr}[T]$.*

Proof. The proof is by induction over the size $n := r - l + 1$ of the set S. The induction base is $n = 0$, where the statement is trivially true since there is only one tree, namely the empty tree. In every other case, let $x \in S$ denote the root of T, and let T_L and T_R denote the left and right subtrees of T with key sets $S^{<x}$ and $S^{>x}$, respectively. We then have

$$\Pr[T] \stackrel{(1)}{=} \frac{x}{\sum_{y \in S} y} \cdot \Pr[T_L] \cdot \Pr[T_R] \stackrel{(2)}{=} \frac{x}{\sum_{y \in S} y} \cdot \tilde{\Pr}[T_L] \cdot \tilde{\Pr}[T_R] \stackrel{(3)}{=} \tilde{\Pr}[T].$$

Equality (1) is what we get from the recursive definition given on the exercise sheet, equality (2) is where we use Claim 2 inductively, and equality (3) needs some further explanation.

Let A denote the set $\{<, >\}^{n-1}$. Intuitively, an element $\alpha \in A$ is a string of length $n - 1$ over the symbols $<$ and $>$, where the entries indicate whether, in any particular iteration except the first, the selected key was inserted into the left ($<$) or right ($>$) subtree of the root. Formally, we treat α as the corresponding event. Moreover, we denote by A_x the subset of A which contains all strings α with precisely $|S^{<x}|$ occurrences of the symbol $<$ and, hence, precisely $|S^{>x}|$ occurrences of the symbol $>$. Intuitively, the subset A_x contains all events from A that are compatible with x being the root of the tree, thus explaining our choice of notation.

The crucial step now is to observe that $\tilde{\Pr}[T \mid \alpha] = \tilde{\Pr}[T_L] \cdot \tilde{\Pr}[T_R]$ holds for every $\alpha \in A_x$, whereas we have $\tilde{\Pr}[T \mid \alpha] = 0$ otherwise. The second equation is easy to see, since conditioning on an event $\alpha \in A$ determines the key that is inserted in the first iteration. As for the first equation, we similarly see that conditioning on any event $\alpha \in A_x$ determines x to be the key first inserted. In addition, conditioned on such α, obtaining the tree T is equivalent to observing an insertion order of the keys $S^{<x}$ that produces the tree T_L during the iterations that correspond to the symbols $<$ in α and, likewise, observing an insertion order of the keys $S^{>x}$ that produces T_R during the iterations that correspond to the symbols $>$ in α. Due to the condition α, the individual probabilities that govern the respective insertion orders of the keys $S^{<x}$ and $S^{>x}$ are the same as if we were to construct iteratively one tree with key set $S^{<x}$ and a separate tree with key set $S^{<x}$, and the two constructions are also independent; hence, the desired equation follows. Starting with the law of total probability, we can thus conclude the proof by deriving

$$\tilde{\Pr}[T] = \sum_{\alpha \in A} \tilde{\Pr}[T \mid \alpha] \cdot \tilde{\Pr}[\alpha] = \tilde{\Pr}[T_L] \cdot \tilde{\Pr}[T_R] \cdot \underbrace{\sum_{\alpha \in A_x} \tilde{\Pr}[\alpha]}_{= \tilde{\Pr}[\text{“}x \text{ is inserted first”}]} = \tilde{\Pr}[T_L] \cdot \tilde{\Pr}[T_R] \cdot \frac{x}{\sum_{y \in S} y}.$$

□

Because of Claim 2, we are now in a position of being able to prove the desired statement for our new, iteratively defined, probability distribution. Since we have seen that it does not make a difference, we will simply write $\Pr$ instead of $\tilde{\Pr}$ from now on.

Claim 3. *Let $S = [n]$, and let $j < i$ be two specific keys from that set. Consider the corresponding random search tree $\tilde{B}_S$ (as defined on the exercise sheet or, equivalently, as defined above) and the random variable A_i^j. Then,*

$$\Pr\left[A_i^j = 1\right] = \frac{2j}{(i - j + 1)(i + j)}.$$

Proof. We start by making the following crucial observation. Let $I := \{j, j + 1, \ldots, i\}$; then, we have $A_i^j = 1$ if and only if j is the first key in the set I that is inserted into the tree during its iterative construction. The “if” holds because when the key i is finally inserted, it will follow the exact same path from the root to j, thus becoming its descendent. The “only if” holds because if some other key $x \in I \setminus \{j\}$ is inserted first, then both j and i (assuming $x \neq i$) will follow the exact same path from the root to x, thus ending up in its left and right subtree, respectively.

Recall that our random tree is constructed by n insertions. We now denote by E_k the event that the k-th insertion is the first where a key from the set I is selected. By the law of total probability, we then have

$$\Pr[A_i^j = 1] = \sum_{k=1}^{n-i+j} \underbrace{\Pr[A_i^j = 1 \mid E_k]}_{\overset{(1)}{=} \frac{2j}{(i-j+1)(i+j)}} \cdot \Pr[E_k] = \frac{2j}{(i-j+1)(i+j)}.$$

To understand equality (1) we note that conditioned on E_k, the statement $A_i^j = 1$ is equivalent to j being inserted at iteration k because of our observation from the beginning. Therefore, this conditional probability must be equal to $j/\sum_{y\in I} y$, and the numerical value follows as in the first solution. □

(b) We can write the depth of the largest key as $D_n^{(n)} = \sum_{j=1}^{n-1} A_n^j$. By using linearity of expectation and part (a), we then have

$$\mathbf{E}\left[D_n^{(n)}\right] = \sum_{j=1}^{n-1} \mathbf{E}\left[A_n^j\right] = \sum_{j=1}^{n-1} \Pr[A_n^j = 1] = \sum_{j=1}^{n-1} \frac{2j}{(n-j+1)(j+n)}$$
$$= \sum_{j=1}^{n-1} \left(\frac{1}{n-j+1} - \frac{1}{j+n} + \frac{1}{(n-j+1)(j+n)} \right).$$

Having split the sum in three, we can now analyse all three parts in isolation. The first part is easy to express as a harmonic series, as follows.

$$\sum_{j=1}^{n-1} \frac{1}{n-j+1} = H_n - 1$$

For the second part, we make use of the fact $H_n = \ln(n) + \gamma + o(1)$ from page 24 of the lecture notes, where γ is Euler's constant.[1]

$$\sum_{j=1}^{n-1} \frac{1}{j+n} = H_{2n} - H_n - \frac{1}{2n} = (\underbrace{\ln(2n)}_{\ln(2)+\ln(n)} + \gamma + o(1)) - (\ln(n) + \gamma + o(1)) - \frac{1}{2n}$$
$$= \ln(2) + o(1) - o(1) - \frac{1}{2n} = \ln(2) + o(1)$$

For the third part, we will do another partial fraction decomposition.

$$\sum_{j=1}^{n-1} \frac{1}{(n-j+1)(j+n)} = \sum_{j=1}^{n-1} \left(\frac{1/(2n+1)}{n-j+1} + \frac{1/(2n+1)}{j+n} \right)$$
$$= \frac{1}{2n+1} \big((H_n - 1) + (H_{2n-1} - H_n)\big) = \frac{\Theta(\ln(n))}{2n+1} = o(1)$$

Putting everything together, we get the following estimate.

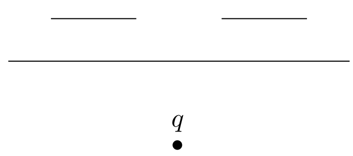
$$\mathbf{E}\left[D_n^{(n)}\right] = H_n - 1 - \ln(2) + o(1)$$

Up to the constant $\ln(2)$, this is the same as for usual search trees.

[1] For those unfamiliar with little-oh notation: Here, it suffices to know that $o(1)$ denotes a term that approaches zero as n tends to infinity.

Solution 3

(a) An example is given as follows:



Assume that we have $X_2 = 0$. Then s_1 must be the long segment, and $X_3 = 0$. In summary,

$$\Pr\left[X_3 = 0 \mid X_2 = 0\right] = 1.$$

However, $\Pr[X_3 = 0] < 1$; hence the variables X_2, X_3 are not independent.

(b) For each $i = 1, \dots, n$, let $S_i = \{s_1, \dots, s_i\}$ denote the arrangement of segments at time i. Furthermore, let $C_i \subseteq S_i$ denote the (random) set of segments that are incident to the trapezoid of S_i that contains q. Note that X_i is just the indicator variable for the event "$s_i \in C_i$".

As discussed in the lecture, we have $|C_i| \leq 4$. Now we choose sets $\hat{C}_i$ with the following properties:

- $C_i \subseteq \hat{C}_i \subseteq S_i$, and
- $|\hat{C}_i| = \min\left\{4,\ |S_i|\right\}$.

(We obtain these sets by simply filling up each C_i with arbitrary elements from S_i.) Now we define

$$Y_i = \begin{cases} 1, & \text{if } s_i \in \hat{C}_i, \\ 0, & \text{otherwise.} \end{cases}$$

We have $Y_i \geq X_i$ by construction, and with the abbreviation $p_i = \min\left\{1,\ \frac{4}{i}\right\}$ we find

$$\mathbf{E}[Y_i] = \Pr[Y_i = 1] = \Pr\left[s_i \in \hat{C}_i\right] = \frac{|\hat{C}_i|}{|S_i|} = p_i.$$

For mutual independence we need to argue that for all choices of $y_1, \dots, y_n \in \{0, 1\}$ we have

$$\Pr[Y_1 = y_1,\ \dots,\ Y_n = y_n] = \Big(\prod_{i\,:\,y_i=1} p_i\Big)\Big(\prod_{i\,:\,y_i=0} (1 - p_i)\Big). \tag{1}$$

Since the argument will be the same for every choice of the y_i's, and in order to reduce clutter in our notation, we only prove

$$\Pr[Y_1 = 1,\ \dots,\ Y_n = 1] = \prod_{i=1}^{n} p_i. \tag{2}$$

To this end we employ backwards analysis: Consider the backwards process that starts from the complete arrangement of segments S, then removes segments in the order $s_n, s_{n-1}, \dots, s_1$. At the first step of this process, we remove the element s_n which is chosen uniformly at random from S, and our success probability at this step is

$$\Pr[Y_n = 1] = p_n.$$

At the next step of the backwards process, s_{n-1} is chosen uniformly at random from the set $S \setminus \{s_n\}$. Now by definition, $\hat{C}_{n-1}$ is a 4-element subset of S_{n-1}, and we have $S_{n-1} = S \setminus \{s_n\}$. (Crucially, all of this holds independently of the choice of s_n.) Hence, our success probability at this step is

$$\frac{|\hat{C}_{n-1}|}{|S \setminus \{s_n\}|} = p_{n-1}.$$

Continuing this argument, we obtain an overall success probability of $\prod_{i=1}^{n} p_i$, which proves (2).

(c) We have $\mathbf{E}\left[e^{\ln(5/4)Y}\right] = \mathbf{E}\left[\left(\frac{5}{4}\right)^{Y}\right]$. By independence, this is $\prod_{i=1}^{n} \mathbf{E}\left[\left(\frac{5}{4}\right)^{Y_i}\right]$. We have

$$\mathbf{E}\left[\left(\frac{5}{4}\right)^{Y_i}\right] = 1 \Pr[Y_i = 0] + \frac{5}{4}\Pr[Y_i = 1] = 1 + \frac{1}{4}\Pr[Y_i = 1] \leq 1 + \frac{1}{4}\frac{4}{i} = \frac{i+1}{i}.$$

Hence

$$\prod_{i=1}^{n} \mathbf{E}\left[\left(\frac{5}{4}\right)^{Y_i}\right] \leq \prod_{i=1}^{n} \frac{i+1}{i} = n+1.$$

We have $\Pr\left[Y \geq \lambda \ln(n+1)\right] = \Pr\left[e^{\ln(5/4)Y} \geq e^{\ln(5/4)\lambda\ln(n+1)}\right]$.

Markov's inequality tells us that that $\Pr[W \geq \alpha] \leq \frac{\mathbf{E}[W]}{\alpha}$ for $\alpha > 0$ and W a nonnegative random variable. Setting $W = e^{\ln(5/4)Z}$ and $\alpha = e^{\ln(5/4)\lambda\ln(n+1)}$, the above probability is at most

$$e^{-\ln(5/4)\lambda\ln(n+1)}(n+1) = \frac{n+1}{(n+1)^{\ln(\frac{5}{4})\lambda}} = \left(\frac{1}{n+1}\right)^{\lambda\ln(5/4)-1}.$$

(d) There are $2n$ segment endpoints. Extending these endpoints by passing a vertical line through them partitions the plane in $2n+1$ regions. Intersecting these regions with all the segments gives us at most $(n+1)(2n+1) \leq 2(n+1)^2$ trapezoids (equivalence classes), such that the points inside each trapezoid behave identically in all possible history graphs. Let Q be a set of query points formed by taking one representative from each equivalence class, and for each $q \in Q$ define the random variable Y_q to be the query time for the query point q. Furthermore, let λ be such that $\lambda\ln(5/4) - 1 = 3$. Now we can compute that

$$\begin{aligned}
&\Pr[\text{query time is less than } \lambda\ln(n+1) \text{ in the worst case}] \\
&= \Pr[Y_q < \lambda\ln(n+1) \text{ for all } q \in Q] \\
&= 1 - \Pr[Y_q \geq \lambda\ln(n+1) \text{ for some } q \in Q] \\
&\overset{\text{U.B.}}{\geq} 1 - \sum_{q \in Q} \Pr[Y_q \geq \lambda\ln(n+1)] \\
&\overset{(c)}{\geq} 1 - |Q| \cdot \left(\frac{1}{n+1}\right)^{\lambda\ln(5/4)-1} \\
&\geq 1 - 2(n+1)^2 \cdot \left(\frac{1}{n+1}\right)^{3} = 1 - \frac{2}{n+1}.
\end{aligned}$$