AnD Kahoot Quiz - Explanations

Julius (julius.jung@inf.ethz.ch),
Victoria (victoria.miebach@inf.ethz.ch),
Haaroon (hmohammed@ethz.ch),
Hannah Lee (hannahlee.pfennig@inf.ethz.ch),
Michelle (mhonegger@inf.ethz.ch) ,
Rui (rui.zhang@inf.ethz.ch)

December 2024

Thank you very much for participating in the Kahoot Quiz, or just checking our Kahoot in general. In this document, you will find some explanation for the solutions. If anything is unclear, feel free to reach out to any of the TAs above.

1 Mathematical Foundations

Question 1: Yes / No?

$$n \leq O(n^2)$$

Answer 1: Yes
Explanation 1: n^2 grows asymptotically faster than n. You may also calculate $lim_{n\to\infty}(n/n^2)$ to see this.

Question 2: Yes / No?

$$n^2 + 14n + 1 \leq O\left(n^2\right)$$

Answer 2: Yes
Explanation 2: n^2 grows at an equal asymptotic rate as $n^2 + 14n + 1$. To see this, notice that $lim_{n\to\infty}\frac{n^2+14n+1}{n^2} = 1$

Question 3: Yes / No?

$$n^2 + 14n^{15} + n^5 - 48732 \cdot n^7 \cdot 95273 \cdot n^{10} + e \cdot n^8 + 37281937 \leq O\left(n^{15}\right)$$

Answer 3: No

Explanation 3: On an intuitive level, we have the summand $48732 \cdot n^7 \cdot 95273 \cdot n^{10} = 48732 \cdot 95273 \cdot n^{17}$ which has a higher summand than n^{15} so this cannot hold.

Question 4: Select the correct definition. For $f : \mathbb{N} \to \mathbb{R}^+$, we have $O(f) =$
Answer 4:

$$\left\{g : \mathbb{N} \to \mathbb{R}^+ : \exists C > 0, \forall n \in \mathbb{N}, g(n) \leq Cf(n)\right\}$$

and

$$\left\{g : \mathbb{N} \to \mathbb{R}^+ : \exists C > 0, \forall n \in \mathbb{N}, Cg(n) \leq f(n)\right\}$$

Explanation 4: The first definition can be found in exercise sheet 2. The second definition is equivalent because

$$\begin{aligned} & g(n) \leq C \cdot f(n) \\ \Leftrightarrow \ & \frac{1}{C} \cdot g(n) \leq f(n) \end{aligned}$$

And if we choose $C' := \frac{1}{C}$, then $C' \geq 0$ and the second definition follows.

Question 5: Put the steps of induction in the correct order.
Answer 5: Base Case, Induction Hypothesis, Induction Step (in this order)
Explanation 5: See first few lecture notes.

Question 6: True / False: $\lim_{n\to\infty} \frac{n^2}{n^2+4n+1} = 0$
Answer 6: False
Explanation 6:

$$\begin{aligned} & \lim_{n\to\infty} \frac{n^2}{n^2+4n+1} \\ =& \lim_{n\to\infty} \frac{1}{1+\frac{4}{n}+\frac{1}{n^2}} \\ =& 1 \neq 0 \end{aligned}$$

Question 7: $\lim_{n\to\infty} \frac{n}{n^n} = 0$
Answer 7: True
Explanation 7:

$$\begin{aligned} & \lim_{n\to\infty} \frac{n}{n^n} \\ =& \lim_{n\to\infty} \frac{1}{n^{n-1}} \\ =& 0 \end{aligned}$$

Question 8: $\lim_{n\to\infty} \frac{n}{1.0000001^n} = 0$
Answer 8: Yes
Explanation 8: Intuitively, exponential growth is always asymptotically faster than linear growth. Formally:

$$\begin{aligned}
&\lim_{n\to\infty} \frac{n}{1.0000001^n} \\
=&\lim_{n\to\infty} \frac{e^{\ln(n)}}{e^{\ln(1.0000001)\cdot n}} \\
=&\lim_{n\to\infty} e^{\ln(n)-\ln(1.0000001)\cdot n}
\end{aligned}$$

And since $\lim_{n\to\infty} \ln(n) - \ln(1.0000001)\cdot n = -\infty$ we have that the above limit evaluates to 0.

Question 9: $\lim_{n\to\infty} \frac{n^2}{n\cdot\ln(n)} = 0$
Answer 9: False
Explanation 9: $\lim_{n\to\infty} \frac{n^2}{n\cdot\ln(n)} = \lim_{n\to\infty} \frac{n}{\ln(n)}$ and linear growth is asymptotically faster than logarithmic growth so this limit evaluates to ∞.

Question 10: $\lim_{n\to\infty} \frac{n^{1.01}}{n\cdot\ln(n)} = 0$
Answer 10: no
Explanation 10:

$$\begin{aligned}
&\lim_{n\to\infty} \frac{n^{1.01}}{n\cdot\ln(n)} \\
=&\lim_{n\to\infty} \frac{n^{0.01}}{\ln(n)} \\
=&\lim_{n\to\infty} \frac{0.01\cdot n^{-0.99}}{\frac{1}{n}} \\
=&\lim_{n\to\infty} 0.01\cdot n^{-0.99}\cdot n \\
=&\lim_{n\to\infty} 0.01\cdot n^{0.01} \\
=&\infty
\end{aligned}$$

Where we used L'hospitals rule.

Question 11: Select base case (B.C.), induction hypothesis (I.H.) and induction step (I.S.) for proving $n \leq 3^n$ for all $n \in \mathbb{N}$

Answer 11: B.C.: $1 \leq 3^1$ I.H.: $n \leq 3^n$ I.S.: $n+1 \leq 3^{(n+1)}$
Explanation 11: Since we want to prove this statement for all $n \in \mathbb{N}$, which does not include 0, we start with $1 \leq 3^1$ as the base case. This is also the only options where induction hypothesis and induction step where the respective statements we want to prove for n and $n+1$.
Note that the option "B.C.: $0 \leq 3^0$" I.H.: $n \leq 3^n$ I.S.: $n+1 \leq 3^{(n+1)}$ would have also been technically correct, since then we would prove this statement, for all $n \in \mathbb{N}_0$ and $\mathbb{N} \subset \mathbb{N}_0$

Question 12: Select the correct options. For all $n \in \mathbb{N}_0$ we have
Answer 12: $\sum_{i=1}^{n} i^k = \Theta\left(n^{k+1}\right)$ and $\sum_{i=1}^{n} n^3 \leq n^4$
Explanation 12: See exercise 1.2, Definition of Θ-Notation.

Question 13: $\lim_{n\to\infty} \frac{n}{e^n} = 0$
Answer 13: True
Explanation 13: Again, because exponential growth is asymptotically faster than linear growth. To prove this formally, you can apply L'hospital's rule.

Question 14: If f grows asymp. slower than g, and h grows asymp. faster than 1, then f grows asymp. slower than $g \circ h$
Answer 14: False
Explanation 14: Exercise 1.3 f)

Question 15: $\sum_{i=1}^{n} \log\left(i^i\right) \geq \Omega\left(n^2 \log\left(n\right)\right)$
Answer 15: true
Explanation 15: Exercise 3.1(b)(2)

2 Searching & Sorting

Question 16: We can always use binary search to check whether an element is in an array or not
Answer 16: False
Explanation 16: This does not work for arrays that are not sorted.

Question 17: What is the runtime of selection sort in an already sorted array?
Answer 17: $O\left(n^2\right)$
Explanation 17: For each iteration, we always select the largest element in $O(n)$.

Question 18: How many swaps does Bubble Sort perform in an array like this:

$$[\, n, n-1, n-2, ..., 1]$$

Answer 18: $\frac{(n-1)\cdot n}{2}$
Explanation 18: the first iteration sets n in the correct place (index $n-1$) with $n-1$ swaps, the second iteration sets $n-1$ in the correct place in $n-2$ swaps,...

$$\sum_{i=1}^{n-1}(n-i) = (n-1) + (n-2) + \cdots + 1 = \frac{(n-1)\cdot n}{2}$$

Question 19: Quicksort is guaranteed to sort an array faster than insertion sort
Answer 19: false
Explanation 19: For an already sorted array, quicksort takes $O(n^2)$ if the rightmost element is selected as pivot, since $O(n)$ iterations with $O(n)$ comparisons are needed. Insertion sort with binary comparison only takes $O(nlog(n))$ on the already sorted array.
The runtime of Quicksort depends on the selection of the pivot. In practice, randomized selection is used. (Further explanation in AnW.)

Question 20: What does the following array look like after the first merge of merge sort:

$$[5, 3, 7, 2, 4, 9, 8, 6]$$

Answer 20: $[3, 5, 2, 7, 4, 9, 6, 8]$
Explanation 20: After splitting the array into four pairs of two elements, the first merge compares within each pair and sets the larger element to the right. Thus 5 and 3, 7 and 2, 8 and 6 but not 4 and 9 are swapped.

Question 21: Suppose an array of n bits (each either 0 or 1) is given. There exists an algorithm that sorts this array in $O(n)$.
Answer 21: true
Explanation 21: The idea is that we use a non-comparison based sorting algorithm, which for example puts all 0's and all 1's in a bucket respectively and then writes these back into the array. This uses $O(n)$ space and $O(n)$ time.
This only works because we are given that there is a finite number of values (here 2: 0 and 1) that the elements in the array can take. The lower bound of $O(nlog(n))$ is only for comparison based sorting algorithms.

Question 22: Why is selection sort not frequently used in practice?
Answer 22: high time complexity
Explanation 22: selection sort is in-place, so it uses no significant extra space. Selection sort takes $O(n^2)$ time, while faster algorithms run in $O(nlog(n))$.

Question 23: For an 8 element array, how many comparisons does merge sort need in the worst case?

Answer 23: 24
Explanation 23: $8 \cdot log_2(8) = 24$.

Question 24: Which sorting algorithm is implemented recursively?
Answer 24: Quicksort, Mergesort (not: Selection Sort, Insertion Sort)
Explanation 24: In the lecture Selection and Insertion sort were implemented iteratively, while Quicksort and Mergesort can only be implemented recursively to achieve the desired runtime.

3 Datastructures

Question: Let $v \in V$ be a vertex of an undirected graph with adjacency matrix A. It takes time $O(1 + \deg(v))$ to compute $\deg(v)$ from A.

Answer: False. The degree of a vertex requires summing the entries in the corresponding row or column in the adjacency matrix, taking $O(|V|)$ time.

Question: Is the tree below an AVL tree?

Answer: False. All AVL trees are also BSTs (binary search trees), so the tree must also satisfy the conditions of a BST.

Question: How many swaps does calling heapify() perform after inserting node 23 in the Max-Heap below?

Answer: 2. After inserting 23, it swaps up twice to restore the max-heap property.

Question: What is the asymptotic runtime of searching an entry in a sorted linked list of size n^2?

Answer: $O(n^2)$. To search in a sorted linked list, we need to go through the entries linearly.

Question: In a binary search tree with n nodes, what is the minimal height it could be?

Answer: $\lfloor \log(n) \rfloor$. The minimal height is achieved when the tree is balanced.

Question: What is the runtime of BFS using an adjacency matrix for the graph $G(V, E)$?

Answer: $O(|V|^2)$. BFS checks all $|V|^2$ entries in the adjacency matrix to explore neighbors.

Question: Is the queue a first-in-first-out (FIFO) data structure?

Answer: True.

4 Dynamic Programming

- The B.C . of any 2D DP problem of starts at...

  Answer: "None of these". Although the B.C. often starts at these entries, this is not always the case (with for instance longest palindromic subsequence)

- For any array of size n, Subset Sum always runs ni O(n*c) for some fixed constant c

  Answer: False, as we cannot find a fixed C for all subset sum problems, as it depends on the sum we are trying to find and the values in the array.

- Which of these problems are pseudo polynomial

  Answer: Knapsack

- The Bottom Up approach is more efficient than the Top Down approach for DP problems

  Answer: False, both are the same efficiency-wise

5 Graphs

- Sort the following Shortest-Paths-algorithms on unweighted graphs by runtime (fastest first): Dijkstra, Floyd-Warshall, Bellman-Ford, BFS

  Answer: BFS - Dijkstra - Bellman-Ford - Floyd-Warshall

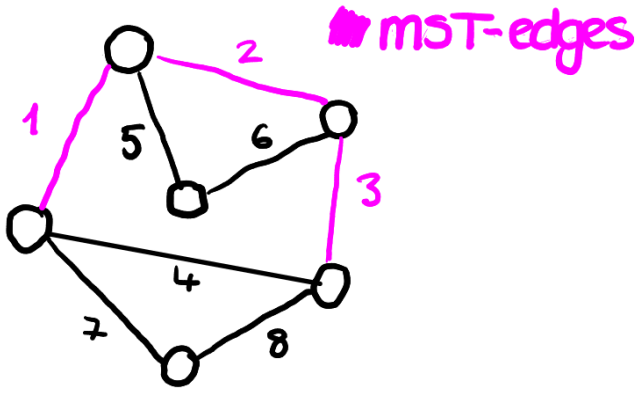
- Every graph with an Eulerian walk has a Hamiltonian path

  Answer: False, since we could have isolated vertices, i.e. vertices that are not connected to any other vertex.

- How long does deleting an edge in an adjacency matrix & adjacency list take?

  Answer: O(1) & O(n)

- What is the weight of the next added edge using Prim?



Answer: Prim always takes the cheapest edge in the cut, in this case the edge with weight 5. It is not the edge with weight 4 because this would create a cycle.

- If G contains an u-v-walk, then it also contains an u-v-path

  Answer: True, because we can always shorten an walk to a path by deleting all contained cycles.

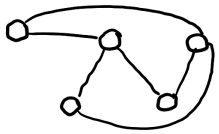
- How can we detect cycles in an undirected connected graph?

  Answer: We can detect cycles using DFS/BFS or by counting the edges/degrees of the graph. Since the graph is connected, we just need to test if it is a tree, i.e. has $|V| - 1$ edges or the sum of degrees equals $2|V| - 2$. If that is not the case, the graph has at least one cycle.

- How can we detect negative cycles in an undirected connected graph?

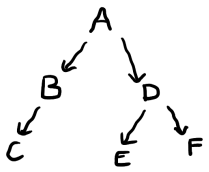
  Answer: We can use Bellmann-Ford. If we do one more iteration of the outer Loop, so n instead of (n-1) iterations, values will change in that iteration if and only if the graph contains a negative cycle.

- Does the following graph contain an Eulerian walk?



  Answer: False, the graph contains an Eulerian walk but not a cycle.

- Let the following be a DFS tree of G. Assume G has an edge (C,A), what is its type?



Answer: It is a back edge.

- A MST of a graph is always unique

  Answer: False, a MST of a graph G is unique if and only if G has pairwise distinct edge weights.

- If every vertex in G has an even degree, then G contains an Hamiltonian path

  Answer: False, since there could be a vertex v with deg(v)=0.

- Let C be a cycle in G and e be an edge on that cycle. Assuming G is connected there exists a spanning tree without e.

  Answer: Since e is on a cycle we can remove it and G will still be connected. Then we can find a spanning tree which will not contain e.

- Let G be a forest consisting of 4 trees. How many edges does G have?

  Answer: A tree on n vertices has n-1 edges. Since G has 4 trees and every tree contains one edge less than its number of vertices, the overall number of edges is 4 smaller than the number of vertices, so n-4.

- G contains an closed Eulerian walk. How fast can an algorithm find one? Choose the most precise answer.

  Answer: There exists an Algorithm on O(m + 1). The rest is not precise enough. Especially O(n+m) is not needed since the Algorithm indeed only needs to cover the edges, not all vertices, so it is only O(m + 1). The +1 is needed in case the graph contains no edges.

- All problems that can be solved with undirected graphs can also be solved with directed graphs.

  Answer: We can model each undirected Graph as a directed Graph with edges in both directions. Then the (almost) same algorithms can be used, so the asymptotic runtime stays the same.

- Edges with weights 5, 6 and 100 are connected to vertex A. Which of these are in the shortest path tree starting from B?

  Answer: If A and B are not connected, none would be in the tree. Also for example if the vertices at the other end of the edges are of degree one and A is reachable, all will be contained.

- G = (V, E) is a tree $\iff$ ...

  Answer: The first three are all equivalent conditions as one can acknowledge since it makes sense or one can look forward to Algorithmen und Wahrscheinlichkeit in the next semester as the properties are formally proofed in the script.