CSCI 570 - HW 1

Solutions

1. Arrange the following functions in increasing order of growth rate with g(n) following *f*(*n*) in your list if and only if *f(n) = O(g(n))*

$$n + 10,\ 100^n,\ \sqrt{n},\ n^{2.5},\ 10^n,\ n^2\log n$$

Solution: We can start approaching this problem by putting 10^n and 100^n at the end of the list, because these functions are exponential and will grow the fastest. $10^n < 100^n$ because 10 < 100. Other four functions are polynomial and will grow slower than exponential. We can represent $n^{2.5}$ and $\sqrt{n}$ as $n^{2.5} = n^2 * \sqrt{n}$; and $\sqrt{n} = n^{0.5}$. Now, we can say that out of all polynomial functions $\sqrt{n}$ will be the slowest because it has the smallest degree. Moreover, $\sqrt{n}$ will be bounded by $n + 10$ because it has a higher degree of 1. Furthermore, $n^{2.5}$ and $n^2\log n$ will be between exponential 10^n and 100^n and polynomial $\sqrt{n}$ and $n + 10$, because polynomial functions grow slower and both 10^n and 100^n and have the highest degree of 2 out of all other polynomial functions. And $n^2\log n$ will be bounded by $n^{2.5}$ because $n^2\log n$ and $n^{2.5} = n^2\sqrt{n}$ and $\log n = O\left(\sqrt{n}\right)$.

Therefore the final order will be: $\sqrt{n} < n + 10 < n^2\log n < n^{2.5} < 10^n < 100^n$.

2. Arrange the following functions in increasing order of growth rate with g(n) following *f*(*n*) in your list if and only if *f(n) = O(g(n))*

$$2^{\log n},\ 2^n,\ n\,(\log n)^3,\ n^{4/3},\ 2^{2^n},\ n\log n,\ 2^{n^2}$$

Solution: Assuming all the logarithms are base 2, $2^{\log n} = n$. Therefore the final order will be: $2^{\log n} < n\log n < n\,(\log n)^3 < n^{4/3} < 2^n < 2^{n^2} < 2^{2^n}$.

3. Suppose that *f(n)* and *g(n)* are two positive non-decreasing functions such that *f(n)* = O(*g*(*n*)). Is it true that log *f*(*n*) = O(log *g*(*n*))? Give a proof or counterexample.

Solution: True. f(n) = O(g(n)) ➔ f(n) ≤ O(g(n))
➔ log(f(n)) ≤ log(c g(n))
➔ log(f(n)) ≤ log(c) + log(g(n))

➔ $\log(f(n)) \leq \log(c) \log(g(n)) + \log(g(n))$
➔ $\log(f(n)) \leq (\log(c)+1) \log(g(n))$
➔ $\log(f(n)) = O(\log(g(n))$

4. Give a liner time algorithm based on BST to detect whether a given undirected graph contains a cycle. If the graph contains a cycle, then your algorithm should output one. It should not output all cycles in the graph, just one of them.

Solution: We run BFS starting from an arbitrary node s, obtaining a BFS tree T. Now, if every edge of G appears in the BFS tree, then G = T, so G is a tree and contains no cycles. Otherwise, there is some edge e = (v, w) that belongs to G but not to T. Consider the least common ancestor u of v and w in T; we obtain a cycle from the edge e, together with the u-v and u-w paths in T.

5. Let $G = (V, E)$ be a connected undirected graph and let v be a vertex in G. Let T be the depth-first search tree of G starting from v, and let U be the breadth-first search tree of G starting from v. Prove that the depth of T is at least as great as the depth of U.

Solution: Let the depth of U be d and let w by a vertex on level d of U. We know that the BFS tree from v indicates the shortest-path distance from v to every node (counting each edge as distance 1). Thus there is no path in G of length less than d from v to w. If the depth of T were less than d, there would be a path in G of length less than d from v to w, given by the path in T. This is impossible, so T cannot have depth less than d.