ECE 592 – Topics in Data Science

Test 2: Scientific Programming – Fall 2020

September 23, 2020

Please remember to justify your answers carefully.

Last name: _____ First name: _____

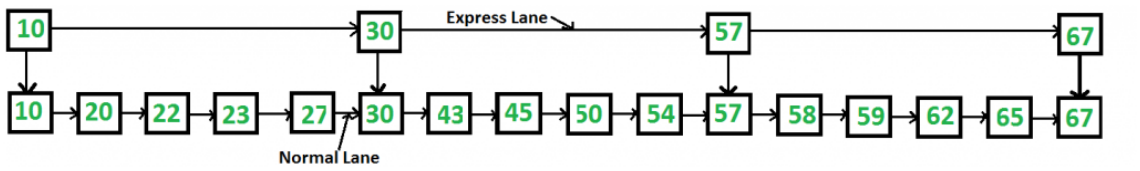
Please recall the course academic integrity policy for tests:
No cooperation or "collaboration" between students is allowed. Especially during an online course experience, it could be tempting to text or email a friend. This is not allowed. You will be allowed to use your notes, books, a browser, and software such as Matlab and/or Python.[1] However, while working on the test you should not text, email, or communicate with other people (certainly not other students) in any way, unless you are consulting with the course staff. **By submitting the test, you will be acknowledging that you completed the work on your own without the help of others in any capacity.** Any such aid would be unauthorized and a violation of the academic integrity policy.

[1] You can use the browser to access Moodle, the course webpage, and look up technical topics. Similar to a normal test, you must not communicate with other people.

Question 1 (Computational Complexity)
Please prove the following.
(a) Consider the factorial function, $N! = \prod_{n=1}^{N} n$. Show that $N! = O(N^N)$.
(b) Let $p(N) = \sum_{i=0}^{d} a_i N^i$, be a degree-$d$ polynomial in N, where $a_d > 0$. Show that $p(N) = \Theta(N^d)$. (Hint: d is constant, and $\Theta(N^d)$ implies that we are considering the rate of growth of $p(N)$ for large N.)

Question 2 (Algorithms)
Consider the problem of determining whether a sequence of N numbers contains N distinct numbers, or instead at least one number occurs multiple times. Describe an efficient algorithm and its computational complexity. (There is no need to use pseudocode; describing in words is fine if your algorithm is simple.)

Question 3 (Data structures)
We have discussed that *linked lists* are a popular data structure that lets objects be inserted and removed from the data structure in a flexible order, in contrast to stacks and queues. On the other hand, linked lists require $O(N)$ computation for sarching among N objects for one with a specific key, because we may need to scan through the entire list.



Skip lists: We can perform faster searches through the data structure using skip lists. As illustrated above, we supplement the regular linked list (normal lane) with a skip list (express lane) that lets us skip through objects. Both lanes are maintained in sorted order, allowing us to skip through the express lane and quickly identify the approximate location to scan within the normal lane. In the illustration, both lanes are singly linked lists, where objects in the skip list also include pointers to corresponding objects in the regular linked list. Each next pointer in the express lane moves us ahead (skips) K objects in the normal lane. (Although $K = 5$ in our illustration, K might not be constant in general, because objects are being added and removed. To simplify the question, we assume that K is constant.)
Example search: Suppose that we are searching for the object with key 45 (located in the regular linked list). We begin with the skip list at the object whose key is 10. Because $10 < 45$, we skip ahead to 30 in the skip list. Because $30 < 45$, we skip ahead to 57 in the skip list. However, $57 > 45$; we return to 30 in the skip list, and then move below to the 30 in the regular linked list. Because $30 < 45$, we move to 43 in the regular linked list. Because $43 < 45$, we move to 45 in the regular linked list. Our search is over.
Search computation: How much computation does searching with a skip list require? The number of objects in the express lane is $\Theta(N/K)$ Once we identify the approximate location to scan within the normal lane, that scan is $O(K)$. Therefore, the total computation for search is $O(N/K + K)$.
(a) To minimize total computation, K can be a function of N. Compute the optimal K^* that minimizes $N/K + K$. Using this K^*, what is the total computational complexity? (Hint: you may assume that N is fixed.)
(b) What about using two levels of skip lists? You can envision a normal lane with N objects, an express lane that skips K_e normal objects at a time (there are $\Theta(N/K_e)$ express lane objects), and a super lane that skips k_s express objects (there are $\Theta(\frac{N}{K_e K_s})$ super lane objects). Recall that the total computation with an express lane (and without a super lane) was $O(N/K + K)$; what is the total computation with a super lane? Express your answer as a function of K_e, K_s, N, and make sure to justify your answer.