CS41 Lab 13: Alternate Models of Computation, Quantum Computing

Thursday, December 1

1. **Quantum Teleportation.** In this problem, you'll develop a quantum notion of *teleportation.*

   Alice and Bob once lived in the same town, but Bob has since moved far away. Alice wants to send Bob a care package of a single qubit $|x\rangle$, but is only able to communicate using classical bits. How can she enable Bob to have $|x\rangle$?

   (a) As a warmup, consider the notion of copying bits. This can be easily done in classical computation, with (among other things) a controlled-NOT (CNOT) gate.

   i. First, feed in a classical bit b into the top line of a CNOT gate, and a zero in the bottom line. Verify that indeed output is two copies of b.

   ii. What happens if you do the same with quantum bits? Feed $|x\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|0\rangle$ into a quantum CNOT gate. What is output?

   (b) Next, revisit how the quantum gates (X,Z, and H)[1] manipulate quantum states. For example, we saw in class that the NOT gate X maps $|0\rangle$ to $|1\rangle$ and maps $|1\rangle$ to $|0\rangle$.

   i. How do the Z and H gates map $|0\rangle$ and $|1\rangle$?

   ii. How do the X, Z, and H gates map an arbitrary qubit $|x\rangle = \alpha|0\rangle + \beta|1\rangle$?

   iii. Now, suppose you start with one of the four qubits below, and you'd like to apply gates to get the qubit $|x\rangle = \alpha|0\rangle + \beta|1\rangle$. What gate(s) should you apply? The first two are solved for you.

   - $\alpha|0\rangle + \beta|1\rangle$. **Solution.** do nothing (the qubit is already in the state you want.)
   - $\beta|0\rangle + \alpha|1\rangle$. **Soln.** Apply the X gate, since it maps $|0\rangle \to |1\rangle$ and vice versa.
   - $\alpha|0\rangle - \beta|1\rangle$
   - $-\beta|0\rangle + \alpha|1\rangle$

   (c) The kind of teleportation Alice and Bob want to achieve is possible, as long as they start with a bit of *entanglement.* Suppose that, prior to moving away, Alice and Bob created the following two qubit state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$. This is called an *EPR* pair.[2] When Bob left, he took one of these qubits and left Alice the other.

   Before teleportation, we can view the qubits as a three qubit system, with Alice having the first two ($|x\rangle$ and the first qubit from the EPR pair), and Bob having the third (the second qubit form the EPR pair). At this point, Alice can manipulate only her qubits.

   i. First, Alice applies a CNOT gate to her qubits, using $|x\rangle$ for the control, and her qubit from the EPR pair as the second qubit.
   What is the state of our system?

   ii. Next, Alice applies an H gate to her first qubit. What is the state of the system now?

   iii. Now, group terms by Alice's qubits. For each possible state of Alice's qubits, what is Bob's qubit in?

[1]see front of board for diagrams of each gate.

[2]EPR pairs are named after their inventors Einstein, Podolsky, and Rosen.

iv. Although we know what state Bob's qubit is in for each possible state of Alice, Bob does not. He knows only his qubit. Alice now measures both of her qubits, and sends the measurements to Bob. Bob then applies one or more gates on his qubit based on Alice's message. What gates should Bob apply to recover $|x\rangle$?

2. **External Memory Analysis.** In class this week, we also saw the External Memory a.k.a. I/O model. Recall that in this model, basic computation and memory references are free; we are charged only for the number of disk accesses we use. You're given M words of memory, and each disk access reads or writes a block of B words from disk to memory (or vice versa), and your input size is N words.

   For the problems below, assume that $N = 2^{40}, M = 2^{21}$, and $B = 2^{13}$. Also assume that each disk access takes 1ms. For simplicity, assume the constant hidden inside the $O(\cdot)$ notation is always 1.

   (a) Imagine doing a linear traversal through a linked list. In a naive implementation, each node will reside on a different block, so $O(N)$ references are necessary. If the nodes are packed tightly into blocks, only $O(N/B)$ accesses are required. How long does each traversal take?

   (b) In class, you saw that it's possible to sort in this model using $O(\frac{N}{B}\log_{M/B}(N/B))$ disk accesses. How many disk accesses do you use? How does this compare to a linear traversal?

3. **External Memory MergeSort.** In this problem, you will develop an $O(\frac{N}{B}\log_{M/B}(N/B))$ algorithm for Mergesort in the external memory model.

   (a) First, consider as a base case when $N \leq M$. Show that $O(M/B)$ disk accesses suffice to sort the array.

   (b) In the general case of the standard Mergesort, one divides the array in to, runs MergeSort on each half, and performs a linear time merge step. In the I/O model, it makes sense to do a multi-way merge. Show that it's possible to merge together M/B sorted arrays of total size N using $O(N/B)$ total disk accesses.

   (c) Use the previous answers to design an external-memory MergeSort algortihm along the lines of the standard MergeSort.

   (d) State a recurrence relation for the number of disk accesses used by your external memory MergeSort, as a function of the number of blocks that your array uses (e.g., if your array is N elements, your array takes up N/B blocks of space)

   (e) Solve this recurrence relation.

4. **Encoding Lower Bounds for Sorting.** Before Thanksgiving break, you saw the Encoding Lower Bound technique, and we used it in class to show an $\Omega(n \log n)$ lower bound for comparison-based sorting algorithms by encoding permutations of $\{1, \ldots, n\}$.

   In this problem, you'll practice this technique by actually encoding permutations. For this problem, you must work with a partner.

   (a) Pick a deterministic comparison-based sorting algorithm.

(b) Write out pseudocode for this algorithm. (It will be helpful to err on the side of precision in the pseudocode.)

(c) (each parnter should do this step privately) For some small value(s) of n, use the sorting algorithm to encode some arbitrary permutation of $\{1, \ldots, n\}$.

(d) Exchange this encoding with your partner, and decode your partner's encoding to recover their permutation. Did you decode the permutation your partner encoded?

(e) Argue that the choice of sorting algorithm didn't matter as long as it was deterministic and comparison-based, and that the encoder and decoder both knew the algorithm.