HW 5

Due on **Wednesday, October 6, 2021 at 10am**

CS 473: Algorithms, Fall 2021 Version: **1.0**

Submission guidelines and policies as in homework 1.

13 (100 PTS.) FFT Applications

13.A. (25 PTS.) You are given two sets B and C each containing integers from $[\![n]\!]$. Design an algorithm, as fast as possible, that computes

$$S = \{b + c \mid b \in B \text{ and } c \in C\}.$$

(The running time of your algorithm will be a function of n).

13.B. (75 PTS.) You are given two binary strings $S \in \{0,1\}^*$ (text string) and $P \in \{0,1\}^*$ (pattern string) of length n and m where $m \ll n$. Our goal is to find the close occurrences of the pattern string P in S. In particular, for all $i \in [\![n-m+1]\!]$, the algorithm should output the hamming distance between $S_1[i \ldots i+m-1]$ and S_2. Design an algorithm, as fast as possible.

14 (100 PTS.) Randomized Algorithms

14.A. (25 PTS.) You are given two n dimensional binary vectors $u, v \in \{0,1\}^n$. Consider a random vector $r \in \{-1,+1\}^n$ (each coordinate is picked independently and uniformly). Observe that if $u = v$ then $\langle r, u\rangle = \langle r, v\rangle$. Prove that if $u \neq v$ then $\mathbb{P}[\langle r, u\rangle = \langle r, v\rangle] \leq 1/2$.

14.B. (25 PTS.) You are given two $n \times n$ matrices $B, C \in \{0,1\}^{n\times n}$, and a parameter $p \in (0,1)$. You are given an oracle, such that for a vector $v \in \{-1,+1\}^n$, and an $n \times n$ matrix D, it computes vD in $O(n)$ time.

Design an algorithm (as fast as possible) that outputs "equal" correctly if $B = C$, with probability at least $1-p$. If it outputs "unequal" then B and C are not equal. That is, most of the time, the algorithm returns a correct answer (such algorithms are called Monte-Carlo algorithms).

(Hint: Use **14.A.**.)

14.C. (25 PTS.) In QuickSort, we invoke the function rand(n), which returns a random integer between 1 and n. In this exercise, we want to investigate how to implement rand(n) with parameters smaller than n.

Let n and m be two positive integers and $n \leq m$. Show how to implement rand(n) using rand(m) in expected constant time.

14.D. (25 PTS.) Do part **14.C.** for the following variant. Let $\sqrt{n} \leq m \leq n$. Show that one can implement rand(n) using rand(m) in expected constant time.

15 (100 PTS.) Sorting networks in matrix form.

Consider an $n \times n$ matrix (assume all the values in the matrix are distinct). One can sort it by repeating the following procedure several times:

(I) Sort each odd row in increasing order.
(II) Sort each even row in decreasing order.
(III) Sort each column in increasing order.

Here is an example of this procedure execution when executed three times:

$$
\begin{array}{l}
1001111110\\
0011100111\\
1101100100\\
1110111001\\
0010101111\\
1101111111\\
1111100011\\
0111100101\\
1111110101\\
1000101000
\end{array}
\xrightarrow{(I)}
\begin{array}{l}
0001111111\\
0011100111\\
0000011111\\
1110111001\\
0000111111\\
1101111111\\
0001111111\\
0111100101\\
0011111111\\
1000101000
\end{array}
\xrightarrow{(II)}
\begin{array}{l}
0001111111\\
1111110000\\
0000011111\\
1111111000\\
0000111111\\
1111111110\\
0001111111\\
1111110000\\
0011111111\\
1110000000
\end{array}
\xrightarrow{(III)}
\begin{array}{l}
0000000000\\
0000010000\\
0000110000\\
0001111000\\
0011111110\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111
\end{array}
\xrightarrow{(I)}
\begin{array}{l}
0000000000\\
0000010000\\
0000000011\\
0001111000\\
0001111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111
\end{array}
$$

$$
\xrightarrow{(II)}
\begin{array}{l}
0000000000\\
1000000000\\
0000000011\\
1111000000\\
0001111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111
\end{array}
\xrightarrow{(III)}
\begin{array}{l}
0000000000\\
0000000000\\
0000000000\\
1001000011\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111
\end{array}
\xrightarrow{(I)}
\begin{array}{l}
0000000000\\
0000000000\\
0000000000\\
1001000011\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111
\end{array}
\xrightarrow{(II)}
\begin{array}{l}
0000000000\\
0000000000\\
0000000000\\
1111000000\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111\\
1111111111
\end{array}
$$

15.A. (40 PTS.) Suppose the matrix contains only 0's and 1's. We repeat the above procedure again and again until no changes occur. In what order should we out the entries of the matrix to get sorted output (i.e., all the numbers in the matrix output in increasing order)? Prove that any $n \times n$ matrix of 0's and 1's will be finally sorted.

15.B. (40 PTS.) Prove, by reproving the zero-one principle in this case (see class notes), that by repeating the above procedure, any matrix of real numbers can be sorted.

15.C. (20 PTS.) Suppose k iterations are required for this procedure to sort the $n \times n$ numbers. Give an upper bound for k. The tighter your upper bound the better (prove you bound). [Hint: $k \lll n$.]