CS 180: Homework 2

Jonathan Woong

804205763

Winter 2017

Discussion 1B

Wednesday 1$^{\text{st}}$ February, 2017

Problem 1

(a) What is the FFT of (1,0,0,0)? What is the appropriate value of w in this case? And of which sequence is (1,0,0,0) FFT?

$FFT_4(1,0,0,0)$

$a = (1,0,0,0) \qquad n = 4 \qquad w = e^{2\pi i/n} = e^{2\pi i/4} = e^{\pi i/2}$

$a_0 = 1 \qquad a_1 = 0 \qquad a_2 = 0 \qquad a_3 = 0$

$\bar{s} = DFT_{n/2}(a_0, a_2) = DFT_{4/2}(1,0) = DFT_2(1,0) = (1,1)$

$a' = (1,0) \qquad n = 2 \qquad w = e^{2\pi i/n} = e^{2\pi i/2} = e^{\pi i}$

$a'_0 = 1 \qquad a'_1 = 0$

$\bar{s} = DFT_{n/2}(a'_0) = DFT_{2/2}(a'_0) = DFT_1(a'_0) = a'_0 = 1$

$\bar{s}' = DFT_{n/2}(a'_1) = DFT_{2/2}(a'_1) = DFT_1(a'_1) = a'_1 = 0$

For $j = 0$ to $[(n/2-1) = (2/2-1) = 0]$:

$r'_j = \bar{s}_j + w^j * \bar{s}'_j$

$r'_0 = \bar{s}_0 + w^0 * \bar{s}'_0 = 1 + (1*0) = 1$

$r'_{j+n/2} = \bar{s}_j + w^{n/2} * w^j * \bar{s}'_j$

$r'_{0+2/2} = r'_1 = \bar{s}_0 + w^1 * w^0 * \bar{s}'_0 = 1 + (w*1*0) = 1$

$RETURN\ r' = (r'_0, r'_1) = (1,1)$

$\bar{s}' = DFT_{n/2}(a_1, a_3) = DFT_{4/2}(0,0) = DFT_2(0,0) = (0,0)$

$a'' = (0,0) \qquad n = 2 \qquad w = e^{2\pi i/n} = e^{2\pi i/2} = e^{\pi i}$

$a''_0 = 0 \qquad a''_1 = 0$

$\bar{s} = DFT_{n/2}(a''_0) = DFT_{2/2}(a''_0) = DFT_1(a''_0) = a'_0 = 0$

$\bar{s}' = DFT_{n/2}(a''_1) = DFT_{2/2}(a''_1) = DFT_1(a''_1) = a'_1 = 0$

For $j = 0$ to $[(n/2-1) = (2/2-1) = 0]$:

$r''_j = \bar{s}_j + w^j * \bar{s}'_j$

$r''_0 = \bar{s}_0 + w^0 * \bar{s}'_0 = 0 + (1*0) = 0$

$r''_{j+n/2} = \bar{s}_j + w^{n/2} * w^j * \bar{s}'_j$

$r''_{0+2/2} = r''_1 = \bar{s}_0 + w^1 * w^0 * \bar{s}'_0 = 0 + (w*1*0) = 0$

$RETURN\ r'' = (r''_0, r''_1) = (0,0)$

For $j = 0$ to $[(n/2-1) = (4/2-1) = (2-1) = 1]$:

$r_j = \bar{s}_j + w^j * \bar{s}'_j$

$r_0 = \bar{s}_0 + w^0 * \bar{s}'_0 = 1 + (1 * 0) = 1$

$r_1 = \bar{s}_1 + w^1 * \bar{s}'_1 = 1 + (w * 0) = 1$

$r_{j+n/2} = \bar{s}_j + w^{n/2} * w^j * \bar{s}'_j$

$r_{0+4/2} = r_2 = \bar{s}_0 + w^2 * w^0 * \bar{s}'_0 = 1 + (w^2 * 1 * 0) = 1$

$r_{1+4/2} = r_3 = \bar{s}_1 + w^2 * w^1 * \bar{s}'_1 = 1 + (w^2 * w * 0) = 1$

$RETURN\ r = (1, 1, 1, 1)$

(b) Repeat for (1,0,1,-1).

$FFT_4(1, 0, 1, -1)$

$a = (1, 0, 1, -1) \qquad n = 4 \qquad w = e^{2\pi i/n} = e^{2\pi i/4} = e^{\pi i/2}$

$a_0 = 1 \qquad a_1 = 0 \qquad a_2 = 0 \qquad a_3 = 0$

$\bar{s} = DFT_{n/2}(a_0, a_2) = DFT_{4/2}(1, 1) = DFT_2(1, 1) =$

$a' = (1, 1) \qquad n = 2 \qquad w = e^{2\pi i/n} = e^{2\pi i/2} = e^{\pi i}$

$a'_0 = 1 \qquad a'_1 = 1$

$\bar{s} = DFT_{n/2}(a'_0) = DFT_{2/2}(a'_0) = DFT_1(a'_0) = a'_0 = 1$

$\bar{s}' = DFT_{n/2}(a'_1) = DFT_{2/2}(a'_1) = DFT_1(a'_1) = a'_1 = 1$

For $j = 0$ to $[(n/2 - 1) = (2/2 - 1) = 0]$:

$r'_j = \bar{s}_j + w^j * \bar{s}'_j$

$r'_0 = \bar{s}_0 + w^0 * \bar{s}'_0 = 0 + (1 * 1) = 2$

$r'_{j+n/2} = \bar{s}_j + w^{n/2} * w^j * \bar{s}'_j$

$r'_{0+2/2} = r'_1 = \bar{s}_0 + w^1 * w^0 * \bar{s}'_0 = 1 + (w * 1 * 1) = 1 + w$

$RETURN\ r' = (r'_0, r'_1) = (2, 1 + w)$

$\bar{s}' = DFT_{n/2}(a_1, a_3) = DFT_{4/2}(0, -1) = DFT_2(0, -1) = (-1, -w)$

$a'' = (0, -1) \qquad n = 2 \qquad w = e^{2\pi i/n} = e^{2\pi i/2} = e^{\pi i}$

$a''_0 = 0 \qquad a''_1 = 0$

$\bar{s} = DFT_{n/2}(a''_0) = DFT_{2/2}(a''_0) = DFT_1(a''_0) = a'_0 = 0$

$\bar{s}' = DFT_{n/2}(a''_1) = DFT_{2/2}(a''_1) = DFT_1(a''_1) = a'_1 = -1$

For $j = 0$ to $[(n/2 - 1) = (2/2 - 1) = 0]$:

$r''_j = \bar{s}_j + w^j * \bar{s}'_j$

$r''_0 = \bar{s}_0 + w^0 * \bar{s}'_0 = 0 + (1 * -1) = -1$

$$r''_{j+n/2} = \bar{s}_j + w^{n/2} * w^j * \bar{s}'_j$$

$$r''_{0+2/2} = r''_1 = \bar{s}_0 + w^1 * w^0 * \bar{s}'_0 = 0 + (w * 1 * -1) = -w$$

$RETURN\ r'' = (r''_0, r''_1) = (-1, -w)$

For $j = 0$ to $[(n/2 - 1) = (4/2 - 1) = (2 - 1) = 1]$:

$$r_j = \bar{s}_j + w^j * \bar{s}'_j$$

$$r_0 = \bar{s}_0 + w^0 * \bar{s}'_0 = 2 + (1 * -1) = -2$$

$$r_1 = \bar{s}_1 + w^1 * \bar{s}'_1 = (1 + w) + (w * -w) = 1 + w - w^2 = 2 + i$$

$$r_{j+n/2} = \bar{s}_j + w^{n/2} * w^j * \bar{s}'_j$$

$$r_{0+4/2} = r_2 = \bar{s}_0 + w^2 * w^0 * \bar{s}'_0 = 2 + (w^2 * 1 * -1) = 2 - w^2 = 3$$

$$r_{1+4/2} = r_3 = \bar{s}_1 + w^2 * w^1 * \bar{s}'_1 = (1 + w) + (w^2 * w * -w) = 1 + w - w^4 = i$$

$RETURN\ r = (-2, 2 + i, 3, i)$

Problem 2

Run the BFS algorithm for the following graph with $s = 1$ and $t = 9 : G = (V, E)$, where $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, and $E = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{3, 5\}, \{3, 7\}, \{4, 5\}, \{5, 6\}, \{8, 9\}\}$.

	1	2	3	4	5	6	7	8	9
L_0 = [s]	t	f	f	f	f	f	f	f	f
L_1 = {2,3}	t	t	t	f	f	f	f	f	f
L_2 = {4,5}	t	t	t	t	t	f	f	f	f
L_3 = {7}	t	t	t	t	t	f	t	f	f
L_4 = {6}	t	t	t	t	t	t	t	f	f
$L_5 = \emptyset$	t	t	t	t	t	t	t	f	f

Problem 3

Given a graph $G = (V, E)$ in adjacency list representation, give an algorithm that runs in time $O(|V| * |E|)$ to check if G has a 'triangle', i.e., a triple of distinct vertics $\{u, v, w\}$ such that all three edges between them are present in G.

Input: adjacency list

Output: vertices u, v, w of a triangle

Algorithm: For some edge given by (u, v) in the adjacency list, explore vertex w's list of neighbors (w is any vertex that is not u or v). If w contains neighbors u and v, then we have found a triangle.

For each edge (u, v): $O(|E|)$

    For each vertex w: $O(|V|)$

        if w has neighbors u and v: $O(1)$

            return u, v, w

return null

Total Runtime $= O(|E|) * O(|V|) * O(1) = O(|V| * |E|)$

Problem 4

Give an algorithm based on BFS that given a grpah $G = (V, E)$ (in adjacency list representation) checks whether or not G has a cycle. Your algorithm should run in time $O(|V| + |E|)$. Prove your algorithm works.

Input: adjacency list

Output: true or false (contains cycle?)

Algorithm: essentially the same as BFS, but one change:

discovered[u] = false for all $u \neq s$

discovered[s] = true

$L[0] = s$

$i \leftarrow 0$

While $L[i]$ is not empty:

    $L[i+1] \leftarrow \emptyset$

    For each vertex $u \in L[i]$

        For each neighbor v of $u(v \in A[u])$

            if discovered[v] = true, then *CYCLE DETECTED*

            else

                set discovered[v]$\leftarrow$true

                add v to $L[i+1]$

        $i \leftarrow i + 1$

check if discovered[t] = true (not necessary for our application)

We use the BFS property that "all vertices with *DISCOVERED* marked true when running BFS(s) are the vertices in the connected component of s." This means that when exploring the vertices of s's neighbor u (s connected to u by definition), and u discovers a neighbor v (u and v connected by definition) that was previously marked *DISCOVERED*, that s is also connected to u. This means that there is connectivity between s, u, v, forming a cycle. Since this algorithm takes no more time than the regular BFS, it will run in $O(|V| + |E|)$.