CS381 Notes on FFT and Convolution

Notation

In what follows w_n denotes $e^{2\pi\sqrt{-1}/n}$. The propoerties of w_n that we will use are (i) $(w_n)^n = 1$; (ii) $(w_n)^2 = w_{(n/2)}$; and (iii) $\sum_{k=0}^{n-1}(w_n)^{ik}$ is 0 if $i \neq 0 \bmod n$, n if $i = 0 \bmod n$.

FFT Algorithm

Input is a an n-vector A (where $n = 2^q$), whose ith entry is denoted by A_i. Output is the Fourier Transform of A, denoted $\mathcal{F}(A)$, whose ith entry is

$$(\mathcal{F}(A))_i = \sum_{k=0}^{n-1}(w_n)^{-ik}A_k$$

1. If $n = 1$ then return A_0 otherwise continue with the steps that follow.
2. Recursively call the algorithm on the $(n/2)$-vector $A^{(even)} = (A_0, A_2, \ldots, A_{n-2})$, which returns

   $$(\mathcal{F}(A^{(even)}))_i = \sum_{t=0}^{(n/2)-1}(w_{n/2})^{-it}A_{2t}$$

   for all $0 \leq i \leq (n/2) - 1$.
3. Recursively call the algorithm on the $(n/2)$-vector $(A_1, A_3, \ldots, A_{n-1})$. which returns

   $$(\mathcal{F}(A^{(odd)}))_i = \sum_{t=0}^{(n/2)-1}(w_{n/2})^{-it}A_{2t+1}$$

   for all $0 \leq i \leq (n/2) - 1$.
4. Obtain $(\mathcal{F}(A))_i$ in linear time for all $0 \leq i \leq (n/2) - 1$, as follows. Re-write the summation that defines $(\mathcal{F}(A))_i$ as two summations, one over even k values (i.e., $k = 2t$) and another over odd k values (i.e., $k = 2t + 1$):

   $$(\mathcal{F}(A))_i = \sum_{t=0}^{(n/2)-1}(w_n)^{-2it}A_{2t} + \sum_{t=0}^{(n/2)-1}(w_n)^{-i(2t+1)}A_{2t+1}$$

   which, using the fact that $(w_n)^2 = w_{n/2}$, becomes

$$= \sum_{t=0}^{(n/2)-1} (w_{n/2})^{-it} A_{2t} + (w_n)^{-i} \sum_{t=0}^{(n/2)-1} (w_{n/2})^{-it} A_{2t+1}$$

$$= (\mathcal{F}(A^{(even)}))_i + (w_n)^{-i} (\mathcal{F}(A^{(odd)}))_i$$

which can be computed in constant time using the fact that $(\mathcal{F}(A^{(even)}))_i$ was returned by the first recursive call and $\sum_{t=0}^{(n/2)-1} (w_{n/2})^{-it} A_{2t+1}$ was returned by the second recursive call.

5. Obtain $(\mathcal{F}(A))_i$ in linear time for all $n/2 \leq i \leq n-1$, as follows. This is done by writing $i = (n/2) + i'$, $0 \leq i' \leq (n/2) - 1$, and re-writing the summation that defines $(\mathcal{F}(A))_i$ as was done in the previous step, this time resulting in:

$$(\mathcal{F}(A))_{i'+(n/2)} = \sum_{t=0}^{(n/2)-1} (w_{n/2})^{-(i'+(n/2))t} A_{2t} + (w_n)^{-(i'+(n/2))} \sum_{t=0}^{(n/2)-1} (w_{n/2})^{-(i'+(n/2))t} A_{2t+1}$$

which, using the fact that $(w_{n/2})^{n/2} = 1$, gives:

$$(\mathcal{F}(A))_{i'+(n/2)} = \sum_{t=0}^{(n/2)-1} (w_{n/2})^{-i't} A_{2t} + (w_n)^{-(i'+(n/2))} \sum_{t=0}^{(n/2)-1} (w_{n/2})^{-i't} A_{2t+1}$$

$$= (\mathcal{F}(A^{(even)}))_{i'} + (w_n)^{-(i'+(n/2))} (\mathcal{F}(A^{(odd)}))_{i'}$$

which, when re-written in terms of i, becomes

$$= (\mathcal{F}(A^{(even)}))_{i-(n/2)} + (w_n)^{-i} (\mathcal{F}(A^{(odd)}))_{i-(n/2)}$$

for all $n/2 \leq i \leq n-1$.

The recurrence is the familiar $T(n) = 2T(n/2) + cn$ and $T(1) = c'$, whose solution is $O(n \log n)$.

Inverse Fourier Transform

The Inverse Fourier Transform of an n-vector T, denoted $\mathcal{F}^{-1}(T)$, is defined as having, as its ith entry

$$(\mathcal{F}^{-1}(T))_i = (1/n) \sum_{k=0}^{n-1} (w_n)^{ik} T_k$$

The $O(n \log n)$ time algorithm for computing the Inverse Fourier Transform is very similar to the one for computing the Fourier Transform (we omit its details). We now prove that computing the inverse of $T = \mathcal{F}(A)$ gives A, i.e.,

$$\mathcal{F}^{-1}(\mathcal{F}(A)) = A$$

In the summation that defines $(\mathcal{F}^{-1}(T))_i$, replacing T_k by $\mathcal{F}(A)_k$ gives

$$(\mathcal{F}^{-1}(T))_i = (1/n)\sum_{k=0}^{n-1}(w_n)^{ik}(\sum_{\beta=0}^{n-1}(w_n)^{-k\beta}A_\beta)$$

$$= (1/n)\sum_{\beta=0}^{n-1}A_\beta(\sum_{k=0}^{n-1}(w_n)^{(i-\beta)k})$$

The summation $\sum_{k=0}^{n-1}(w_n)^{(i-\beta)k}$ is zero unless $i-\beta = 0$, in which case it is n. Using this observation in the above gives

$$(\mathcal{F}^{-1}(T))_i = (1/n)A_i(n) = A_i$$

Proof of Convolution Theorem

We want to prove the following for any two n-vectors A and B:

$$\mathcal{F}(A * B) = \mathcal{F}(A) \cdot \mathcal{F}(B)$$

where $*$ denotes convolution and $\cdot$ denotes the componentwise product. The ith component of $A * B$, denoted by $(A * B)_i$, is defined as

$$(A * B)_i = \sum_k A_k B_{i-k}$$

where all indices are modulo n, and the summation index k is understood to range over all possible values modulo n (i.e., 0 to $n-1$).

We want to prove that, for every i, we have

$$(\mathcal{F}(A * B))_i = (\mathcal{F}(A))_i(\mathcal{F}(B))_i$$

where the notation V_i denotes, as usual, the ith component of a vector V. Our strategy will be to start with the right-hand side (RHS) of the above, and manipulate it until we end up with the left-hand side (LHS) of the above. The RHS, after we use the definition of Fourier Transform, looks like this:

$$RHS = (\sum_\alpha (w_n)^{-i\alpha}A_\alpha)(\sum_\beta (w_n)^{-i\beta}B_\beta) = \sum_{\alpha,\beta}(w_n)^{-i(\alpha+\beta)}A_\alpha B_\beta$$

where all summation indices are understood to range over all possible values modulo n (i.e., 0 to $n-1$). Note also that the $\alpha+\beta$ summation in the exponent of w_n can be carried out modulo n because of the $(w_n)^n = 1$ identity. Making the change of variable $k = \alpha+\beta$ and re-writing the above summation in terms of k and α gives:

$$RHS = \sum_{\alpha,k}(w_n)^{-ik}A_\alpha B_{k-\alpha} = \sum_k (w_n)^{-ik}(\sum_\alpha A_\alpha B_{k-\alpha}) = \sum_k (w_n)^{-ik}(A*B)_k = (\mathcal{F}(A*B))_i = LHS$$