Design and Analysis of Algorithms February 29, 2012
Massachusetts Institute of Technology 6.046J/18.410J
Profs. Dana Moshkovitz and Bruce Tidor Handout 10

Problem Set 3 Solutions

This problem set is due **at 9:00pm** on **Wednesday, February 29, 2012**.

Problem 3-1. Electric Potential Problem

According to Coulomb's Law, the electric potential created by a point charge q, at a distance r from the charge, is:

$$V_E = \frac{1}{4\pi\epsilon_0}\frac{q}{r}$$

There are n charges in a square uniform grid of $m \times m$ points. For $i = 1, 2, \ldots, n$, the charge i has a charge value q_i and is located at grid point (x_i, y_i), where x_i and y_i are integers $0 \le x_i, y_i < m$. For each grid point (x, y) not occupied by a charge, the ***effective electric potential*** is:

$$V(x, y) = \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{n}\frac{q_i}{\sqrt{(x-x_i)^2 + (y-y_i)^2}} .$$

The electric potential problem is to find the effective electric potential at each of the $m^2 - n$ grid points unoccupied by a charge.

(a) Describe a simple $O(m^2 n)$ time algorithm to solve the problem.

Solution: The following naive algorithm iterates through all of the points (x, y) on the $m \times m$ grid and computes the electric potential at that point according to the formula given above:

ELECTRIC-POTENTIAL-NAIVE(m, x, y, q)

create an $m \times m$ table V // $\Theta(m^2)$
for $x = 0$ to $m-1$ // $\Theta(m)$ iterations
  for $y = 0$ to $m-1$ // $\Theta(m)$ iterations
    $V(x, y) = 0$ // $\Theta(1)$
    for $i = 1$ to n // $\Theta(n)$ iterations
      if $x_i == x$ and $y_i == y$ // $\Theta(1)$
        $V(x, y) = \text{NIL}$ // $\Theta(1)$
      else
        $V(x, y) = V(x, y) + \frac{1}{4\pi\epsilon_0} \cdot \frac{q_i}{\sqrt{(x-x_i)^2+(y-y_i)^2}}$ // $\Theta(1)$
return V

The total runtime of this code is $\Theta(m^2) + \Theta(m) \cdot \Theta(m) \cdot (\Theta(1) + \Theta(n) \cdot \Theta(1)) = \Theta(m^2) + \Theta(m^2) \cdot (\Theta(1) + \Theta(n)) = \Theta(m^2 n)$.

(b) Let $\mathbb{Z}_{2m-1} = \{0, 1, \ldots, 2m-2\}$. Find two functions $f, g : \mathbb{Z}_{2m-1} \times \mathbb{Z}_{2m-1} \rightarrow \mathbb{R}$ such that the potential at (x, y) equals the convolution of f and g:

$$\begin{aligned} V(x,y) &= (f \otimes g)(x,y) \\ &= \sum_{x'=0}^{2m-2} \sum_{y'=0}^{2m-2} f(x',y') \cdot g(x-x', y-y') \,. \end{aligned}$$

Importantly, in this definition $x - x'$ and $y - y'$ are computed in the additive group $\mathbb{Z}_{2m-1}$, which is a fancy way of saying they are computed modulo $2m - 1$.

Solution: We will use f to contain the locations of the charges, multiplied by a constant:

$$f(x,y) = \begin{cases} \dfrac{q_i}{4\pi\epsilon_0} & \text{if there is a charge } q_i \text{ with } (x_i, y_i) = (x, y) \\ 0 & \text{if } (x,y) \text{ has no charge} \\ 0 & \text{if } (x,y) \text{ lies outside the } m \times m \text{ grid} \end{cases}$$

To match up with the formula for $V(x, y)$, we want $g(x, y)$ to take as input the coordinate differences between a pair of points and we want $g(x, y)$ to produce the inverse of the distance between the original two points. However, we cannot simply use the formula for the inverse distance, because the numbers passed into g will be taken mod $2m - 1$. So any negative difference in coordinates will wrap around to become a value $> m - 1$. Squaring that positive value will not produce the correct result. To handle this correctly, we need to map the numbers in $\mathbb{Z}_{2m-1}$ so that any number greater than $m - 1$ becomes negative again. We do this with the following intermediate function:

$$r(z) = \begin{cases} z & \text{if } z \le m-1 \\ z - (2m-1) & \text{otherwise} \end{cases}$$

With this definition in place, we can now write a definition for g:

$$g(x,y) = \begin{cases} \dfrac{1}{\sqrt{(r(x))^2 + (r(y))^2}} & \text{if } (x,y) \ne (0,0) \\ 0 & \text{otherwise} \end{cases}$$

To see why this is correct, we must plug our formulae for f and g into the equation for convolution:

$$(f \otimes g)(x,y) = \sum_{x'=0}^{2m-2} \sum_{y'=0}^{2m-2} f(x',y') \cdot g(x-x', y-y')$$

First, note that $f(x', y')$ is only nonzero at the locations (x_i, y_i) on the grid that have some charge q_i. In other words, we can rewrite the summation in the following way:

$$\begin{aligned}(f \otimes g)(x, y) &= \sum_{i=1}^{n} f(x_i, y_i) \cdot g(x - x_i, y - y_i) \\ &= \sum_{i=1}^{n} \frac{q_i}{4\pi\epsilon_0} \cdot g(x - x_i, y - y_i)\end{aligned}$$

The differences $(x - x_i)$ and $(y - y_i)$ will be taken mod $2m - 1$ before they are passed into g. However, once they are passed into g, g will undo the effect of the modulus by passing those differences through the function r, and so the final formula for $(f \otimes g)(x, y)$ will be:

$$(f \otimes g)(x, y) = \sum_{i=1}^{n} \frac{q_i}{4\pi\epsilon_0} \cdot \frac{1}{\sqrt{(x - x_i)^2 + (y - y_i)^2}} = V(x, y)$$

Hence, we can use the convolution of these functions to compute the effective electric potential at all points that do not contain charges.

For positive integer k, the ***discrete Fourier transform*** of a function $h : \mathbb{Z}_k \times \mathbb{Z}_k \to \mathbb{R}$ is the function $\widehat{h} : \mathbb{Z}_k \times \mathbb{Z}_k \to \mathbb{C}$ defined as follows:

$$\widehat{h}(a, b) = \frac{1}{k^2} \sum_{x=0}^{k-1} \sum_{y=0}^{k-1} h(x, y) \omega_k^{-ax-by} ,$$

where ω_k is a kth root of unity.

The corresponding ***inverse discrete Fourier transform*** of $\widehat{h} : \mathbb{Z}_k \times \mathbb{Z}_k \to \mathbb{C}$ is defined as follows:

$$h(x, y) = \sum_{a=0}^{k-1} \sum_{b=0}^{k-1} \widehat{h}(a, b) \omega_k^{ax+by} .$$

(c) Prove that for any two functions $f, g : \mathbb{Z}_k \times \mathbb{Z}_k \to \mathbb{R}$ and for any point $(a, b) \in \mathbb{Z}_k \times \mathbb{Z}_k$, we have

$$\widehat{(f \otimes g)}(a, b) = k^2 \cdot \widehat{f}(a, b) \cdot \widehat{g}(a, b) .$$

Solution: To see that this is true, we begin by writing out the formula for $k^2 \cdot \widehat{f}(a, b) \cdot \widehat{g}(a, b)$ and substituting in the definition for $\widehat{f}(a, b)$:

$$\begin{aligned}k^2 \cdot \widehat{f}(a, b) \cdot \widehat{g}(a, b) &= k^2 \cdot \left(\frac{1}{k^2} \sum_{x'=0}^{k-1} \sum_{y'=0}^{k-1} f(x', y') \cdot \omega_k^{-ax'-by'} \right) \cdot \widehat{g}(a, b) \\ &= \sum_{x'=0}^{k-1} \sum_{y'=0}^{k-1} \left(f(x', y') \cdot \omega_k^{-ax'-by'} \cdot \widehat{g}(a, b) \right)\end{aligned}$$

Next we want to substitute for $\widehat{g}(a,b)$. But to more closely match the formula for convolution, we would like to rewrite $\widehat{g}(a,b)$ to include the term $g(x-x', y-y')$. To do so, we set $s = x - x'$ and $t = y - y'$. We will write the formula for g in terms of s and t, then substitute in the values $s = x - x'$ and $t = y - y'$. This means that $x = s + x'$ and $y = t + y'$.

$$\begin{aligned}
\widehat{g}(a,b) &= \frac{1}{k^2}\sum_{s=0}^{k-1}\sum_{t=0}^{k-1} g(s,t)\cdot\omega_k^{-as-bt} \\
&= \frac{1}{k^2}\sum_{x=x'+0}^{x'+k-1}\sum_{y=y'+0}^{y'+k-1} g(x-x', y-y')\cdot\omega_k^{-a(x-x')-b(y-y')}
\end{aligned}$$

The values x and y are drawn from $\mathbb{Z}_k$, so the values will wrap around. The order of the summation doesn't matter. So we can rewrite those summations as sums from 0 to $k-1$:

$$\widehat{g}(a,b) = \frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1} g(x-x', y-y')\cdot\omega_k^{-a(x-x')-b(y-y')}$$

When we plug this formula for $\widehat{g}(a,b)$ into the formula for $k^2\cdot\widehat{f}(a,b)\cdot\widehat{g}(a,b)$, we get the following:

$$\begin{aligned}
&k^2\cdot\widehat{f}(a,b)\cdot\widehat{g}(a,b) \\
&= \sum_{x'=0}^{k-1}\sum_{y'=0}^{k-1}\left(f(x',y')\omega_k^{-ax'-by'}\left(\frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1} g(x-x', y-y')\omega_k^{-a(x-x')-b(y-y')}\right)\right) \\
&= \frac{1}{k^2}\sum_{x'=0}^{k-1}\sum_{y'=0}^{k-1}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1}\left(f(x',y')\omega_k^{-ax'-by'} g(x-x', y-y')\omega_k^{-a(x-x')-b(y-y')}\right) \\
&= \frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1}\sum_{x'=0}^{k-1}\sum_{y'=0}^{k-1}\left(f(x',y') g(x-x', y-y')\omega_k^{-ax'-by'-a(x-x')-b(y-y')}\right) \\
&= \frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1}\left(\sum_{x'=0}^{k-1}\sum_{y'=0}^{k-1} f(x',y') g(x-x', y-y')\right)\omega_k^{-ax-by} \\
&= \frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1}(f\otimes g)(x,y)\cdot\omega_k^{-ax-by} \\
&= \widehat{(f\otimes g)}(a,b)
\end{aligned}$$

This is precisely what we wanted to show.

(d) Design an $O(k^2 \lg k)$ time algorithm to compute the discrete Fourier transform and its inverse.

Solution: For the purposes of this problem, we will be using the definition of the FFT that matches the definition more commonly used outside of CS — the inverse of the FFT seen in class. More formally, we say that $\widehat{h}$ is the one-dimensional FFT of h if:

$$\widehat{h}(a) = \frac{1}{k} \cdot \sum_x h(x) \cdot \omega_k^{-ax}$$

Hence, for this problem, the result h of inverting the FFT on $\widehat{h}$ is defined to be:

$$h(a) = \sum_x \widehat{h}(x) \cdot \omega_k^{ax}$$

With these definitions in place, we can derive algorithms for two-dimensional FFT using the one-dimensional algorithm as a black box.

The algorithm that we will use to compute the two-dimensional FFT involves reducing the problem to computing $\Theta(k)$ different FFTs for one-dimensional functions. More specifically, we use the following algorithm to compute the two-dimensional FFT:

1. Use $f(x, y)$ to create k different functions f_x that operate on the domain $\mathbb{Z}_k$. The function $f_x(y)$ is defined to be $f(x, y)$.
2. Run one-dimensional FFT k times, once on each function f_x. This yields k functions $\widehat{f_x}$ that operate on the domain $\mathbb{Z}_k$.
3. Use the computed functions $\widehat{f_x}$, to create k different functions g_b that operate on the domain $\mathbb{Z}_k$. The function $g_b(x)$ is defined to be $\widehat{f_x}(b)$.
4. Run one-dimensional FFT k times, once on each function g_b. This yields k functions $\widehat{g_b}$ that operate on the domain $\mathbb{Z}_k$.
5. Define the function $\widehat{f}$ such that $\widehat{f}(a, b) = \widehat{g_b}(a)$. Return the function $\widehat{f}$.

We begin by analyzing the runtime of this algorithm. Step 1 requires us to create k functions with domains of size k, and calculating each value requires $\Theta(1)$ time, so the total runtime required is $\Theta(k^2)$. Step 2 requires us to run FFT k times, each time on a function with domain k. Each time we run FFT requires $\Theta(k \log k)$ time, for a total of $\Theta(k^2 \log k)$. Step 3, like Step 1, requires us to construct k functions with domains of size k. Calculating each value only requires a lookup, for a total of $\Theta(k^2)$ time. Step 4 has the same runtime as Step 2, $\Theta(k^2 \log k)$. Finally, step 5 requires us to construct a function with domain of size k^2, and each value of the function is looked up elsewhere. So the total runtime is $\Theta(k^2 \log k)$, just as we wanted.

Next, we must examine the correctness of this algorithm. We may do so by starting with step 5 and gradually expanding the definition of $\widehat{f}$ using the definition of the FFT and the equalities resulting from the various steps in our algorithm.

$$\begin{aligned}
\widehat{f}(a,b) &= \widehat{g_b}(a) && \text{(definition from step 5)}\\
&= \frac{1}{k}\sum_{x=0}^{k-1} g_b(x)\cdot\omega_k^{-ax} && \text{(FFT performed in step 4)}\\
&= \frac{1}{k}\sum_{x=0}^{k-1} \widehat{f_x}(b)\cdot\omega_k^{-ax} && \text{(definition from step 3)}\\
&= \frac{1}{k}\sum_{x=0}^{k-1}\left(\frac{1}{k}\sum_{y=0}^{k-1} f_x(y)\cdot\omega_k^{-by}\right)\cdot\omega_k^{-ax} && \text{(FFT performed in step 2)}\\
&= \frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1} f(x,y)\cdot\omega_k^{-by}\cdot\omega_k^{-ax} && \text{(definition from step 1)}\\
&= \frac{1}{k^2}\sum_{x=0}^{k-1}\sum_{y=0}^{k-1} f(x,y)\cdot\omega_k^{-ax-by} && \text{(rearranging terms)}
\end{aligned}$$

This is precisely the definition of two-dimensional convolution, so our algorithm will compute the correct results.

How can we compute the inverse two-dimensional FFT? It's possible to compute it by using ω_k^{-1} instead of ω_k in all of the FFT calculations, and multiplying by a constant. For completeness, we give a more detailed algorithm, closely resembling the algorithm for two-dimensional FFT. More formally, the following algorithm can be used to compute the inverse FFT of a two-dimensional function $\widehat{f}$:

1. Use $\widehat{f}(x,y)$ to create k different functions $\widehat{f_x}$ that operate on the domain $\mathbb{Z}_k$. The function $\widehat{f_x}(y)$ is defined to be $\widehat{f}(x,y)$.
2. Run the one-dimensional inverse FFT k times, once on each function $\widehat{f_x}$. This yields k functions f_x that operate on the domain $\mathbb{Z}_k$.
3. Use the computed functions f_x, to create k different functions $\widehat{g_b}$ that operate on the domain $\mathbb{Z}_k$. The function $\widehat{g_b}(x)$ is defined to be $f_x(b)$.
4. Run one-dimensional FFT k times, once on each function $\widehat{g_b}$. This yields k functions g_b that operate on the domain $\mathbb{Z}_k$.
5. Define the function f such that $f(a,b) = g_b(a)$. Return the function f.

The runtime analysis of this function proceeds analogously to the runtime analysis of the FFT algorithm given above, for a total runtime of $\Theta(k^2\log k)$.

We use a similar technique to show the correctness of our algorithm:

$$\begin{aligned}
f(a,b) &= g_b(a) && \text{(definition from step 5)} \\
&= \sum_{x=0}^{k-1} \widehat{g_b}(x) \cdot \omega_k^{ax} && \text{(inverse FFT in step 4)} \\
&= \sum_{x=0}^{k-1} f_x(b) \cdot \omega_k^{ax} && \text{(definition from step 3)} \\
&= \sum_{x=0}^{k-1} \left(\sum_{y=0}^{k-1} \widehat{f_x}(y) \cdot \omega_k^{by} \right) \cdot \omega_k^{ax} && \text{(inverse FFT in step 2)} \\
&= \sum_{x=0}^{k-1} \sum_{y=0}^{k-1} \widehat{f}(x,y) \cdot \omega_k^{by} \cdot \omega_k^{ax} && \text{(definition from step 1)} \\
&= \sum_{x=0}^{k-1} \sum_{y=0}^{k-1} \widehat{f}(x,y) \cdot \omega_k^{ax+by} && \text{(rearranging terms)}
\end{aligned}$$

So we have correctly computed the inverse FFT of $\widehat{f}$.

(e) Design an $O(m^2 \lg m)$ time algorithm to solve the electric potential problem for a grid of size $m \times m$.

Solution: The following algorithm can be used to compute $V(x,y)$ for all points (x,y) not occupied by a charge.

1. Compute the values of f and g, as defined in part (b), on $\mathbb{Z}_{2m-1} \times \mathbb{Z}_{2m-1}$. This requires the computation of $\Theta((2m-1)^2) = \Theta(m^2)$ values, each of which requires $\Theta(1)$ time to compute.
2. Compute the values of $\widehat{f}$ and $\widehat{g}$ using the two-dimensional FFT algorithm from part (d). This requires $\Theta(m^2 \log m)$ time in total.
3. Compute the values of $\widehat{h}$, defined to be $\widehat{h}(x,y) = \widehat{f}(x,y) \cdot \widehat{g}(x,y)$ for all values of x and y. This requires the computation of $\Theta((2m-1)^2) = \Theta(m^2)$ values, each of which is the product of two values that have already been computed. As a result, this step requires $\Theta(m^2)$ time.
4. Compute the values of h using the two-dimensional inverse FFT algorithm from part (d). This requires $\Theta(m^2 \log m)$ time.

The results in part (c) show that steps 2, 3, and 4 are computing the convolution of f and g. We showed in part (b) that the convolution of f and g gives us $V(x,y)$ for all (x,y) not occupied by a charge. Therefore, this algorithm will compute $V(x,y)$ correctly. The total runtime of this algorithm is $\Theta(m^2 \log m)$.