Motivation: Representing an $n - 1$ degree polynomial $A(x)$ with the $n$ coefficients $a_0$ through $a_{n-1}$ allows polynomial evaluation and addition to be done in $\Theta(n)$ time, but the straightforward method for polynomial multiplication takes $\Theta(n^2)$ time.

Representing an $n - 1$ degree polynomial $A(x)$ with $n$ point-value pairs $(x_0, A(x_0))$ through $(x_{n-1}, A(x_{n-1}))$ allows polynomial addition to be done in $\Theta(n)$ time, assuming the two polynomials to be added are represented by pairs with the same $x$'s. Polynomial multiplication can also be done in $\Theta(n)$ time assuming, in addition, that $2^n - 1$ pairs are available for each polynomial, since the product polynomial has degree $2n - 2$. However, evaluating the polynomial for an arbitrary $x$ takes time $\Omega(n^2)$.

We'd like to have the best of both worlds: Represent polynomials using the coefficients. When multiplication must be done, evaluate the polynomials at $2^n$ carefully chosen points in $O(n \log n)$ time using the FFT algorithm, do multiplication using the resulting pairs, and then extract the coefficients from the resulting pairs (called interpolation) in $O(n \log n)$ time using the inverse FFT algorithm.

The DFT: Let $m = 2n$. For convenience, assume $m$ is a power of 2.

The carefully chosen points are powers of the complex number

$$\omega_m = e^{2\pi i/m} = \cos(2\pi/m) + i \cdot \sin(2\pi/m),$$

where $i$ is the square root of $-1$. The powers of interest are $\omega_m^0, \omega_m^1, \omega_m^2, \ldots, \omega_m^{m-1}$.

The Discrete Fourier Transform of $m$-vector $a$, denoted $\text{DFT}(a)$, is defined to be the $m$-vector $y$ whose $k$-th element is

$$y_k = \sum_{j=0}^{m-1} a_j \cdot (\omega_m^k)^j$$

for $k = 0, \ldots, m - 1$.

Therefore, when $a$ consists of the coefficients $a_0$ through $a_{n-1}$ of an $n - 1$ degree polynomial $A$, padded with $n$ zeroes, the $k$-th element of $\text{DFT}(a)$ equals $A(\omega_m^k)$, the polynomial $A$ evaluated at $\omega_m^k$.

The FFT: The Fast Fourier Transform (FFT) is an algorithm to compute the DFT. Here’s the recursive version of the FFT. The input is an $m$-vector $a$ and the output is an $m$-vector $y$ such that $y = \text{DFT}(a)$.

function Recursive-FFT($a$):
  if $m = 1$ then return $a$ endif
  $p := \text{Recursive-FFT}([a_0, a_2, a_4, \ldots, a_{m-2}])$
  $q := \text{Recursive-FFT}([a_1, a_3, a_5, \ldots, a_{m-1}])$
  for $k := 0$ to $m/2 - 1$ do
    $y_k := p_k + \omega_m^k \cdot q_k$ (line 1)
    $y_{k+m/2} := p_k - \omega_m^k \cdot q_k$ (line 2)
  endfor
  return $y$
**Running Time:** Let $T(m)$ be the running time on input of size $m$. Then $T(1) = 1$ and $T(m) = 2 \cdot T(m/2) + c \cdot m$ for some constant $c$. Solving this recurrence using the standard techniques shows that $T(m) = O(m \log m)$.

**Correctness:** We must show that $y = \text{DFT}(a)$. We use induction on $m$, the size of $a$.

**Basis:** $m = 1$. When $m = 1$, $a = [a_0]$. The vector DFT($a$), by definition, consists solely of $a_0 \cdot (\omega^0_m)$. This equals $a_0$, which is what the algorithm returns.

**Inductive hypothesis:** Assume that for input of size $m/2$, the algorithm returns the DFT of its input.

**Inductive step:** We must show that for input of size $m$, the algorithm returns the DFT of its input. By the inductive hypothesis, the recursive calls that define the vectors $p$ and $q$ are correct. Thus, for $k$ between 0 and $m - 1$,

$p_k$ equals the polynomial $P$ evaluated at $\omega^k_{m/2}$, where $P(x) = a_0 + a_2 \cdot x + a_4 \cdot x^2 + \ldots + a_{m-2} \cdot x^{m/2-1}$, and

$q_k$ equals the polynomial $Q$ evaluated at $\omega^k_{m/2}$, where $Q(x) = a_1 + a_3 \cdot x + a_5 \cdot x^2 + \ldots + a_{m-1} \cdot x^{m/2-1}$.

Check the values computed in line 1:

$y_k = p_k + \omega^k_m \cdot q_k$ by the code

$= a_0 + a_2 \cdot \omega^k_{m/2} + a_4 \cdot (\omega^k_{m/2})^2 + \ldots + a_{m-2} \cdot (\omega^k_{m/2})^{m/2-1}$

$+ \omega^k_m (a_1 + a_3 \cdot \omega^k_{m/2} + a_5 \cdot (\omega^k_{m/2})^2 + \ldots + a_{m-1} \cdot (\omega^k_{m/2})^{m/2-1})$, by the inductive hypothesis.

This expression is in terms of $\omega^k_{m/2}$. We want an expression in terms of $\omega^k_m$. To convert, use the identity that $\omega^k_{m/2} = e^{2\pi i k/(m/2)} = e^{2\pi i 2k/m} = \omega^k_m$.

After plugging in and doing some algebra, we get that

$y_k = a_0 + a_1 \cdot \omega^k_m + \ldots + a_{m-1} \cdot (\omega^k_m)^{m-1}$. Since this equals $A(\omega^k_m)$, this is correct.

Now check the values computed in line 2:

$y_{k+m/2} = p_k - \omega^k_m \cdot q_k$ by the code

$= p_k + \omega^k_{m/2+m} \cdot q_k$, since $(-1) \cdot \omega^k_m = \omega^m_{m/2} \cdot \omega^k_m$

$= a_0 + a_2 \cdot \omega^k_{m/2} + a_4 \cdot (\omega^k_{m/2})^2 + \ldots + a_{m-2} \cdot (\omega^k_{m/2})^{m/2-1}$

$+ \omega^k_{m/2} (a_1 + a_3 \cdot \omega^k_{m/2} + a_5 \cdot (\omega^k_{m/2})^2 + \ldots + a_{m-1} \cdot (\omega^k_{m/2})^{m/2-1})$, by the inductive hypothesis.

This expression is in terms of $\omega^k_{m/2}$. We want an expression in terms of $\omega^{k+m/2}_m$. To convert, use the identity that $\omega^k_{m/2} = \omega^{2k}_{m} = 1 \cdot \omega^{2k}_m = \omega^m_{m} \cdot \omega^{2k}_m = \omega^{2k+m}_m = (\omega^{k+m/2}_m)^2$.

After plugging in and doing some algebra, we get that

$y_{k+m/2} = a_0 + a_1 \cdot \omega^{k+m/2}_m + \ldots + a_{m-1} \cdot (\omega^{k+m/2}_m)^{m-1}$. Since this equals $A(\omega^{k+m/2}_m)$, this is correct.