Arithmetic Algorithms: Fast Fourier Transform Algorithm

Unlike most algorithm tours in Algovision, the present one has just one scene FFT, which means Fast Fourier Transform.

The goal of the Discrete Fourier Transform (DFT) is to transform a given vector $< a_0, \ldots, a_{N-1} >$ of the dimension N over the field of complex numbers into a vector $< A_0, \ldots, A_{N-1} >$, defined as follows:

$$A_k = \sum_{0}^{N-1} a_\ell \omega^{\ell k} \quad \text{for } k = 0, \ldots, N-1, \tag{1}$$

where ω is an N-th root of unity, i.e., the number such that $\omega^N = 1$. In the field of complex numbers, one possible N-th root of unity (used most frequently) is

$$\omega = e^{2\pi i/N} = \cos\frac{2\pi}{N} + i \sin\frac{2\pi}{N}.$$

Another N-th root of unity is $e^{-2\pi i/N}$.

It is not the goal of this tour to give a motivation for DFT; another tour in Algovision explains how DFT can be used to analyze periodic signals.

When computing DFT of an N dimensional vector according to the above formula, N^2 multiplications and $N(N-1)$ additions are needed.

Fast Fourier Transform (FFT) is an algorithm that computes DFT of an N-dimensional vector using $N \log_2 n$multiplications and the same number of additions, provided N is an integer power of 2. The FFT algorithm is attributed to Cooley and Tukey (1965).

The present tour starts with a description of DFT according to the above formula and modifies it into the FFT in a sequence of phases:

Ininial Matrix

At the beginning, the window shows the above formula in an open form for the dimension N selected at the control bar. $N = 8$ is the best value, $N = 2, 3$ are to small, for $N = 16$ the letters are already quite small and the animation could be slow, and for $N = 32, 64, 128$ the terms of formula asre so small that they are replaced by simple dots.

Note that the exponent of ω in each term is the product of the column index and the row index.

Click Step.

Column Partition

The column are partitioned to even columns (black) and odd (violet). Note that the leftmost clumn is considered to be 0-th, i.e., even.

Column Shuffle

Separate even and odd columns so that the even ones are on the left, the odd ones are on the right.

Taking out Omega Power Suggested

The next step is to take out the factor ω^k from the sum of terms of the odd columns that belong to the k-th row. As we will see, the main reason is to change the exponents of ω so that the first factor od the exponent is even not only in even columns, but also in odd columns.

Taking out Omega Power

In this phase, the change suggested in the previous phase is performed.

Key Step Suggested

The changes made in this phase are the most important and represent the main idea of the FFT algorithm. It is here where we use the fact that $\omega^N = 1$.

Consider two terms in the same column such that the difference of their row indices is $N/2$. Click any term in the matrix; the clicked term and its mate have now a white background. The computation that appears in blue stripes shows that such two terms are *equal*. Let us suppose that the term, which belong to the upper half of the matrix is in ℓ-th column and k-th row (which means that $k < N/2$. Its mate is in the same column and in the $(k + N/2)$-th row.

If ℓ is even, the upper term is $a_\ell \omega^{\ell k}$, and the lower term is $a_\ell \omega^{\ell(k+N/2)}$; otherwise, after the previous modifications of the formula, the upper term is $a_\ell \omega^{(\ell-1)k}$, and the lower term is $a_\ell \omega^{(\ell-1)(k+N/2)}$.

So there is an even m (equal to ℓ for even ℓ and equal to $\ell - 1$ if ℓ is odd), such that the upper term is $a_\ell \omega^{mk}$ and the lower term is

$$a_\ell \omega^{m(k+N/2)} = a_\ell \omega^{mk} \omega^{mN/2} = a_\ell \omega^{mk} \left(\omega^N\right)^{m/2} = a_\ell \omega^{mk}, \tag{2}$$

because $\omega^N = 1$ and $m/2$ is an integer, because m is even.

The equations (2) are what you can see for particular values of k, ℓ, m in the lower blue stripe in the screen.

This means that the lower term can be replaced by its upper mate; in other words, the second factor of any exponent of ω in the lower half of the matrix can be decreased by $N/2$, as suggested in the present phase.

Key Step

In this phase, the change suggested in the previous phase is performed.

Circuit-like Modification

Up to now, our formalism has been purely mathematical. However, I want to show that the FFT can advantageously implemented by a certain circuit (usually called a butterfly circuit). The picture describes the same expressions, but in each row the terms of even column and the terms of the odd columns are added in a mathematical way, using + operators, but the sum of the odd columns (the right sum) is passed by a blue wire to a multiplier by ω^k (where k is the index of the row) and the even sum and the output of the multiplier are imputs for an adder that sends its output to A_k.

NE and SE Submatrices Compared

In this phase we compare two square submatrices of size $(N/2) \times (N/2)$ - one is now with an ocre background, another one with a greenish background. And we can see that, after modifications in the key phase, they are equal.

NE and SE Submatrices Overlayed

Since the submatrices are equal, we can overlay them. This means that we will not compute first sums in the one submatrix and the sums in another one, but we will evaluate only sums in one of the submatrices, and each result is used twice.

NW and SW Submatrices Compared

The other two submatrices are equal es well.

NW and SW Submatrices Overlayed

Similarly as for the east matrices, the west ones are overlayed, too.

Omega Dimension Shown

During the previous phases, we have divided the problem to four subproblems of the size $(N/2) \times (N/2)$, and showed that, in fact, we have only *two* such subproblems to be solved and combined using a column of adders and multipliers. The two subproblems look very similar to Discrete Fourier Transforms of vectors $< a_0, a_2, a_4, \ldots, N-2 >$ and $< a_1, a_3, a_5, \ldots, N-1 >$. We are going to show that the subproblems in fate *are* DFT of these vectors.

In the present and the following phases, we will be using different roots of unity, all of them denoted by ω. To avoid confusion, this and the next phase denote N-th root of unity as ω_N. Thus, $\omega_N^N = 1$ (note that ω_N represents one symbol, while the upper N represents an arithmetic operation). E.g., if $N = 8$, then ω_8 is the 8th root of unity - its eighth power is 1; ω_4 is 4th power of unity - its fourth power is 1.

This phase shows all ω's explicitly as n-th roots of unity.

Dimension Change Suggested

Note that is ω is an N-th root of unity, ω^2 (the square of ω, its second power) is $N/2$-th root of unity, because

$$(\omega^2)^{N/2} = \omega^{2N/2} = \omega^N = 1.$$

Thus, this phase suggests using $\omega^2 = \omega_{N/2}$ (the $(N/2)$-th root of unity) instead of ω_N together with halving exponents of ω_N's, which is done here hy halving the first factor of the product representing the exponent (we know that this first factor is always even, i.e., its hals is an integer).

When the suggested change is performed, we can see that the upper submatrix is nothing else than a DFT of the vector $< a_0, a_2, a_4, \ldots, N-2 >$ (using $(N/2)$-th root of unity $\omega_{N/2}$) and similarly the lower matrix is a DFT of the vector $< a_1, a_3, a_5, \ldots, N-1 >$ of the odd items of the original vector. This is why the next phase is again called "Initial Matrix".

This picture gives the FFT, a recursive algorithm for DFT: compute DFT's of the vectors $< a_0, a_2, a_4, \ldots, N-2 >$ and $< a_1, a_3, a_5, \ldots, N-1 >$, and combine them as shown to get $< A_0, A_1, \ldots, A_{N-1} >$.

Continue in the same way to get eventually a butterfly network for computing the DFT. It is clear that the network consists of $\log_2 N$ columns, each having N adders and N multipliers, which also says that, using FFT, we can compute a DFT of an N-dimensional vector using N additions and N multiplicaions by (possible pre-computed) powers of ω.