DFT and FFT implementations and proofs using ACL2

Mertcan Temel

What is Fourier Transform?

- Decomposition of a signal into the frequencies that make it up
- Applications include:
  - Differential/Difference Eqs, Filter Design, Speech Recognition, Fast Large Integer Multiplication…
- 3 main types:
  - Continuous Time Fourier Transform (CTFT)
    - Input: Continuous, Output: Continuous
  - Discrete Time Fourier Transform (DTFT)
    - Input: Discrete, Output: Continuous
  - Discrete Fourier Transform (DFT)
    - Input: Discrete, Output: Discrete
    - The one we are interested in
    - Fast Fourier Transform (FFT): some efficient algorithms to compute DFT

What is Discrete Fourier Transform (DFT)?

$$X_k = \sum_{m=0}^{N-1} x_m e^{-j2\pi mk/N}$$

DFT

$$x_m = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{j2\pi mk/N}$$

Inverse DFT (IDFT)

x: input vector of finite length N

x_m: m^{th} element of x

X: output vector of the same length N

X_k: k^{th} element of X

j: square root of -1

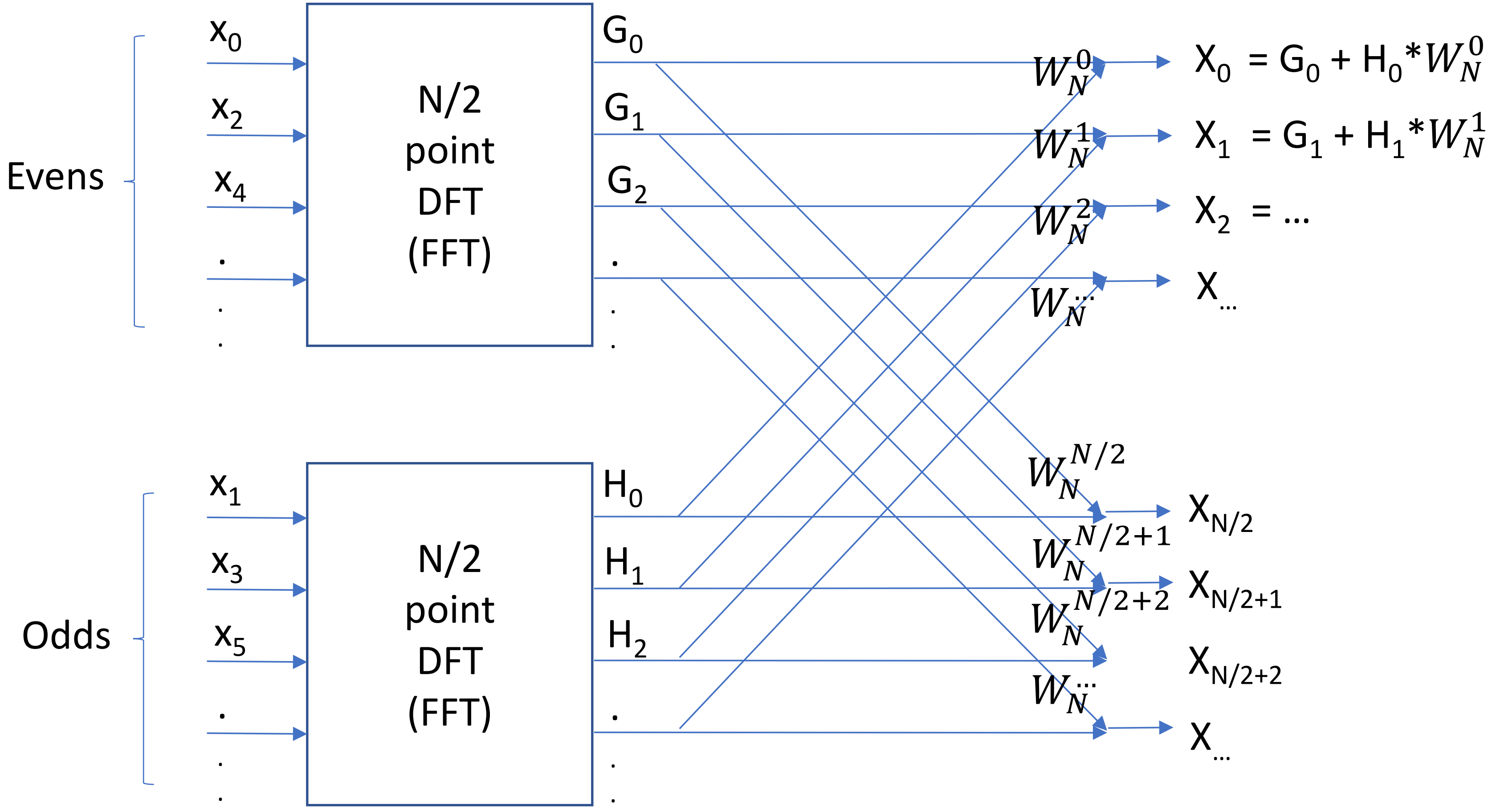
DFT Implementation:

- Calculate the sum over *m* from 0 to N-1 for every *k* in [0 N-1]
- Time complexity of $O(N^2)$

What is Fast Fourier Transform (FFT)?

- An efficient implementation of DFT
- Two most commonly known ways:
  - Decimation-in-time (DIT)
  - Decimation-in-frequency (DIF)
- Restriction: vector length N should be power of 2. If not, fill with 0s.
- A recursive algorithm

What is Fast Fourier Transform (FFT)?



- where $W_N^m = e^{-j2\pi m/N}$
- Decimation-in-time
- Recursive definition with base N=2
- N=2 step is also called butterfly step
- Time complexity of O(Nlog(N))

Work Done in ACL2

- DFT Implementation
  - Proof for (idft (dft x N) N) = x
- FFT decimation-in-time implementation
  - Proof for (fft x N) = (dft x N)

DFT Implementation in ACL2

$$X_k = \sum_{m=0}^{N-1} x_m e^{-j2\pi mk/N}$$

DFT

$$x_m = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{j2\pi mk/N}$$

Inverse DFT (IDFT)

```
(defun dft_sum (x N k m)
  (declare (xargs :measure (nfix (- N m))))
  (if (zp (- N m))
      0
    (+ (* (number-fix (nth m x))
          (exp- (* #c(0 1)
                   2
                   (PI-)
                   m
                   -1
                   k
                   (/ N))))
       (dft_sum x N k (+ m 1)))))
```

```
(defun dft_eachk (x N k)
  (declare (xargs :measure (nfix (- n k))))
  (if (zp (- N k))
      nil
    (cons (dft_sum x N k 0)
          (dft_eachk x N (1+ k)))))

(defun dft (x N)
  (dft_eachk x N 0))
```

Proof for IDFT of DFT of X is X

Goal:

Prove that inverse DFT of DFT of a vector gives the original vector.

i.e. (idft (dft x)) = x?

ACL2 theorem:

```
(DEFTHM IDFT-OF-DFT-OF-X-IS-X
  (IMPLIES (AND (ACL2-NUMBER-LISTP X)
                (EQUAL N (LEN X)))
           (EQUAL (IDFT (DFT X N) N) X)))
```

Proof for IDFT of DFT of X is X - Steps

Step 1: Plug DFT sum into IDFT sum. 3 variables: p, q, and m

$$\frac{1}{N}\sum_{q=0}^{N-1}\left(\sum_{p=0}^{N-1} x_p e^{-j2\pi qp/N}\right)e^{j2\pi mq/N}$$

Step 2: Merge exponentials

$$\frac{1}{N}\sum_{q=0}^{N-1}\sum_{p=0}^{N-1} x_p e^{-j2\pi qp+j2\pi mq/N}$$

Proof for IDFT of DFT of X is X - Steps

Step 2: Merge exponentials

$$\frac{1}{N}\sum_{\boldsymbol{q}=0}^{N-1}\sum_{\boldsymbol{p}=0}^{N-1} x_p e^{-j2\pi qp+j2\pi mq/_N}$$

Step 3: Change summation order

$$\frac{1}{N}\sum_{\boldsymbol{p}=0}^{N-1}\sum_{\boldsymbol{q}=0}^{N-1} x_p e^{j2\pi q(m-p)/_N}$$

Proof for IDFT of DFT of X is X - Steps

Step 3: Change summation order

$$\frac{1}{N}\sum_{p=0}^{N-1}\sum_{q=0}^{N-1} x_p e^{j2\pi q(m-p)/_N}$$

Step 4: Take x_p out

$$\frac{1}{N}\sum_{p=0}^{N-1} x_p \sum_{q=0}^{N-1} e^{j2\pi q(m-p)/_N}$$

Proof for IDFT of DFT of X is X - Steps

Step 4: Take x_p out

$$\frac{1}{N}\sum_{p=0}^{N-1} x_p \sum_{q=0}^{N-1} e^{j2\pi q(m-p)/_N}$$

Step 5: Take 1/N in

$$\sum_{p=0}^{N-1} x_p (\frac{1}{N}\sum_{q=0}^{N-1} 1 * e^{j2\pi q(m-p)/_N})$$

Proof for IDFT of DFT of X is X - Steps

Step 5: Take 1/N in

$$\sum_{p=0}^{N-1} x_p \left(\frac{1}{N}\sum_{q=0}^{N-1} 1 * e^{j2\pi q(m-p)/_N}\right)$$

Step 6: Rewrite impulse from idft of ones

$$\sum_{p=0}^{N-1} x_p \delta[m-p]$$

where

$$\delta[a] = \begin{cases} 1, & a = 0 \\ 0, & a \neq 0 \end{cases}$$

Proof for IDFT of DFT of X is X - Steps

Step 6: Rewrite impulse from idft of ones

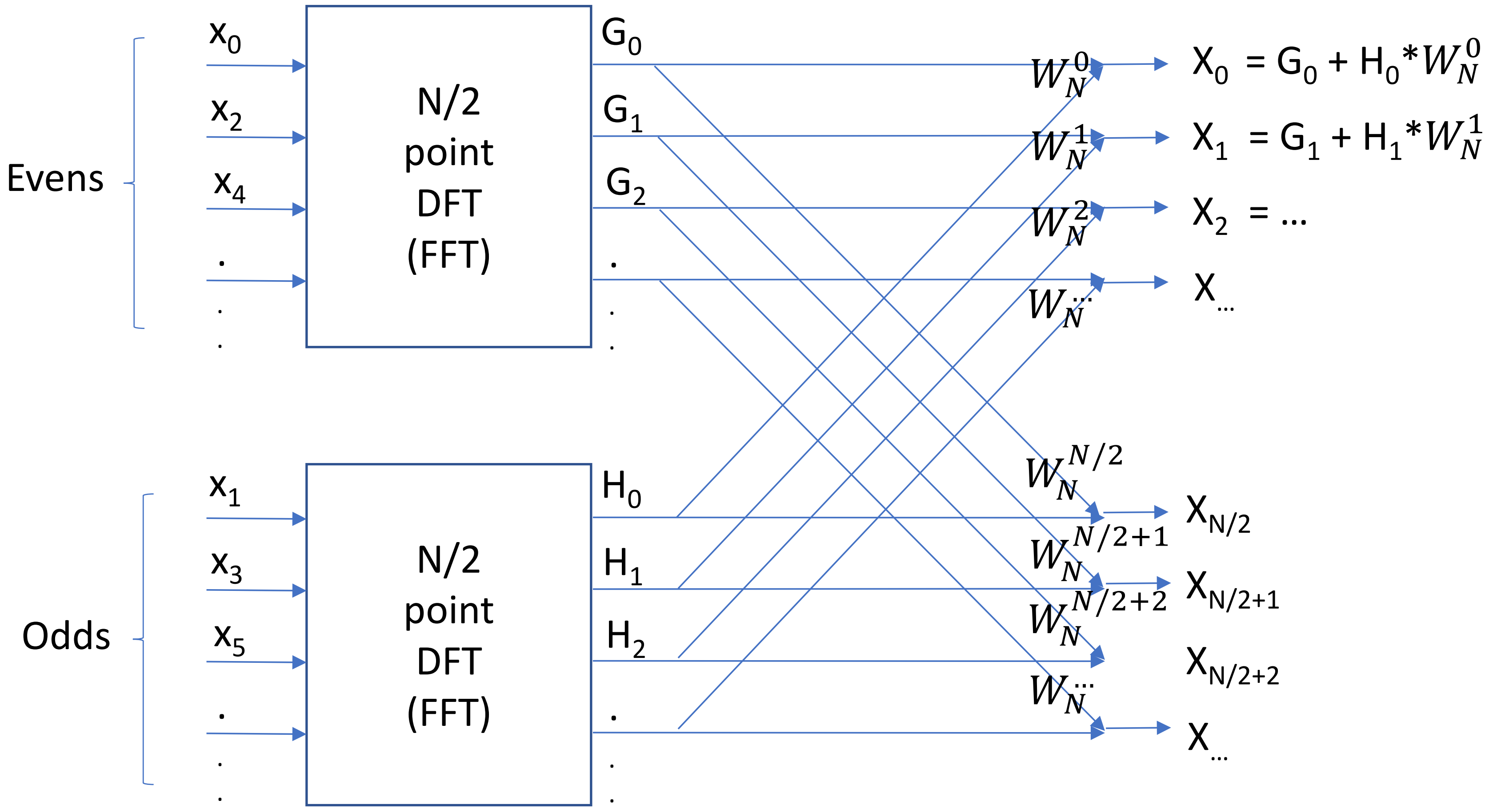
$$\sum_{p=0}^{N-1} x_p \delta[m-p]$$

where

$$\delta[a] = \begin{cases} 1, & a = 0 \\ 0, & a \neq 0 \end{cases}$$

Step 7: Equate the term to x_m. This concludes the proof.

FFT Implementation in ACL2



- where $W_N^m = e^{-j2\pi m/N}$
- Recursively calculate N/2 point FFTs of even and odd indices
- Perform G + H*W over each element
  (G and H values are used twice)

FFT Implementation in ACL2

```
(defun fft2-dit-multi (G H i N)
  (declare (xargs :measure (nfix (- N i))))
  (if (zp (- N i))
      nil
    (let ((j (if (>= i (/ N 2)) (- i (/ N 2)) i)))
      (cons (+ (number-fix (nth j G))
               (* (number-fix (nth j H))
                  (WNk i N)))
            (fft2-dit-multi G H (1+ i) N)))))

(defun fft2-dit (x N)
  (declare (xargs :measure (if (> N 1) (floor N 1/2) 0)))
  (if (or (not (integerp N)) (<= N 1))
      (list (number-fix (car x)))
    (let ((evenfft (fft2-dit (getevens x) (/ N 2)))
          (oddfft (fft2-dit (getodds x) (/ N 2))))
      (fft2-dit-multi evenfft oddfft 0 N))))
```

Proof for FFT is DFT

- Basic idea of FFT:
  - Get rid of redundant/repeated multiplications
  - Remember intermediate results
- Different derivations for Decimation-in-time (DIT) and Decimation-in-frequency (DIF). Only DIT will be discussed here.
- ACL2 Theorem:

```
(DEFTHM DFT-IS-FFT-DIT
  (IMPLIES (POWER-OF-2 N)
           (EQUAL (DFT X N) (FFT2-DIT X N)))
```

Proof for FFT is DFT - Steps

DFT formula:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi nk/N}$$

Step 1: Divide into two summations

$$X_k = \sum_{n=0}^{N/2-1} x_{2n} e^{-j2\pi(2n)k/N} + \sum_{n=0}^{N/2-1} x_{2n+1} e^{-j2\pi(2n+1)k/N}$$

Proof for FFT is DFT - Steps

Step 1: Divide into two summations

$$X_k = \sum_{n=0}^{N/2-1} x_{2n} e^{-j2\pi(2n)k/N} + \sum_{n=0}^{N/2-1} x_{2n+1} e^{-j2\pi(2n+1)k/N}$$

Step 2: Distribute and commute constants

$$X_k = \sum_{n=0}^{N/2-1} x_{2n} e^{-j2\pi k/(N/2)} + \sum_{n=0}^{N/2-1} x_{2n+1} e^{-j2\pi kn/(N/2)} e^{-j2\pi k/N}$$

Proof for FFT is DFT - Steps

Step 2: Distribute and commute constants

$$X_k = \sum_{n=0}^{N/2-1} x_{2n} e^{-j2\pi k/(N/2)} + \sum_{n=0}^{N/2-1} x_{2n+1} e^{-j2\pi kn/(N/2)} e^{-j2\pi k/N}$$

Step 3: Take the constant exponential out

$$X_k = \sum_{n=0}^{N/2-1} x_{2n} e^{-j2\pi k/(N/2)} + (e^{-j2\pi k/N}) \sum_{n=0}^{N/2-1} x_{2n+1} e^{-j2\pi kn/(N/2)}$$

Proof for FFT is DFT - Steps

Step 3: Take the constant exponential out

$$X_k = \sum_{n=0}^{N/2-1} x_{2n} e^{-j2\pi k / (N/2)} + (e^{-j2\pi k / N}) \sum_{n=0}^{N/2-1} x_{2n+1} e^{-j2\pi kn / (N/2)}$$

Observation 1:

$$X_k = (N/2\ point\ DFT\ of\ evens\ x) + (e^{-j2\pi k / N}) * (N/2\ point\ DFT\ of\ odds\ x)$$

Observation 2:

N point DFT is periodic with N (i.e. $X_k = X_{k+N}$)

Proof for FFT is DFT - Steps

Observation 1:

$$X_k = (N/2\, point\ DFT\ of\ evens\ x) + (e^{-j2\pi k/N}) * (N/2\, point\ DFT\ of\ odds\ x)$$

Observation 2: N point DFT is periodic with N (i.e. $X_k = X_{k+N}$)

=> 1st and 2nd DFTs give the same result for k and k+N/2

(e.g. $evendft(x)_k = evendft(x)_{k+N/2}$)

=> Instead of calculating separately for all k ∈ [0 N-1],

calculate 1st and 2nd DFTs for k ∈ [0 N/2-1]

remember and use those values twice for k and k+N/2

Applying these conclude the proof.

Future Work

Implement this FFT web,

- in the DE system in ACL2
- as a self-timed, asynchronous circuit