CHICAGO JOURNAL OF THEORETICAL COMPUTER SCIENCE 2013, Article 12, pages 1–5
http://cjtcs.cs.uchicago.edu/

A Lower Bound for Fourier Transform Computation in a Linear Model Over 2x2 Unitary Gates Using Matrix Entropy

Nir Ailon*

Received June 20, 2013; Revised October 11, 2013; Published October 18, 2013

Abstract: Obtaining a non-trivial (super-linear) lower bound for computation of the Fourier transform in the linear circuit model has been a long standing open problem. All lower bounds so far have made strong restrictions on the computational model. One of the best known results, by Morgenstern from 1973, provides an $\Omega(n\log n)$ lower bound for the *unnormalized* FFT when the constants used in the computation are bounded. The proof uses a potential function related to a determinant. The determinant of the unnormalized Fourier transform is $n^{n/2}$, and thus by showing that it can grow by at most a constant factor after each step yields the result. This classic result, however, does not explain why the *normalized* Fourier transform, which has a unit determinant, should take $\Omega(n\log n)$ steps to compute. In this work we show that in a layered linear circuit model restricted to unitary 2×2 gates, one obtains an $\Omega(n\log n)$ lower bound. The well known FFT works in this model. The main conclusion from this work is that a potential function that might eventually help proving the $\Omega(n\log n)$ conjectured lower bound for computation of Fourier transform is not related to matrix determinant, but rather to a notion of matrix entropy.

*This work has been written with the support of Marie Curie International Reintegration Grant PIRG07-GA-2010-268403.

Key words and phrases: Fourier transform, Linear circuits, Computational complexity, Matrix entropy

DOI: 10.4086/cjtcs.2013.012

1 Introduction

The (discrete) normalized Fourier transform is a complex linear mapping sending an input $x \in \mathbb{C}^n$ to $y = Fx \in \mathbb{C}^n$, where F is an $n \times n$ unitary matrix defined by

$$F(k,\ell) = n^{-1/2} e^{-i2\pi k\ell/n} .$$

The *unnormalized Fourier transform* matrix is defined as $n^{1/2}F$. (The unnormalized Fourier transform is often simply referred to, in literature, as the "Fourier transform".) The Fast Fourier Transform (FFT) of Cooley and Tukey [2] is a method for computing the Fourier transform (normalized or not - the adjustment is easy) of a vector $x \in \mathbb{C}^n$ in time $O(n \log n)$ using a so called linear algorithm. A linear algorithm, as defined in [3], is a sequence $\mathcal{F}_0, \mathcal{F}_1, \ldots$, where each $\mathcal{F}_i$ is a set of affine functions, for each $i \geq 0$ $\mathcal{F}_{i+1} = \mathcal{F}_i \cup \{\lambda_i f + \mu_i g\}$ for some $\lambda_i, \mu_i \in \mathbb{C}$ and $f, g \in \mathcal{F}_i$, and $\mathcal{F}_0$ contains (projections onto) the input variables as well as constants.

It is trivial that computing the Fourier Transform requires a linear number of steps, but no non-trivial lower bound is known. Papadimitriou, for example, computes in [4] an $\Omega(n \log n)$ lower bounds for Fourier transforms in finite fields using a notion of an information flow network. It is not clear how to extend that result to the Complex field. There have also been attempts [5] to reduce the constants hiding in the upper bound of $O(n \log n)$, while also separately counting the number of additions versus the number of multiplications (by constants). In 1973, Morgenstern proved that if the modulus of the λ_i's and μ_i's is bounded by 1 then the number of steps required for computing the *unnormalized* Fourier transform in the linear algorithm model is at least $\frac{1}{2} n \log_2 n$. It should be noted that Cooley and Tukey's unnormalized FFT indeed can be expressed as a linear algorithm with coefficients of the form e^{iz} for some real z, namely, complex numbers of unit modulus.

The main idea of Morgenstern is to define a potential function for each $\mathcal{F}_i$ in the linear algorithm sequence, equaling the maximal determinant of a square submatrix in a certain matrix corresponding to $\mathcal{F}_i$. The technical step is to notice that the potential function can at most double in each step. The determinant of the unnormalized Fourier transform is $n^{n/2}$, hence the lower bound of $\frac{1}{2} n \log_2 n$.

The determinant of the *normalized* Fourier transform, however, is 1. Morgenstern's method can therefore not be used to derive any useful lower bound for computing the normalized Fourier transform in the linear algorithm model with constants of at most unit modulus. Using constants of modulus $1/\sqrt{2}$ in the normalized version of FFT, on the other hand, does compute the normalized Fourier transform in $O(n \log n)$ steps.

The normalized and unnormalized Fourier transforms are proportional to each other, and hence we don't believe there should be a difference between their computational complexities in any reasonable computational model. The fact that Morgenstern's method doesn't give us a useful lower bound for the unnormalized Fourier transform is due to the weakness of the model, which does not allow us to use constants of modulus that grows with n. If such constants were allowed at any location in the circuit (and not just, say, as a preprocessing or postprocessing "scaling up" or "scaling down" step) then his arguments would break down. It is important also to note that, due to the model's weakness, Morgenstern's result teaches us, upon inspection of the proof, that both matrices $\sqrt{n}F$ (the unnormalized Fourier transform) and $\sqrt{n}\,\mathrm{Id}$ are in the same complexity class. More generally, it tells us that all unitary matrices scaled up by the same constant ($\sqrt{n}$ in this case) are in the same complexity class. It is therefore necessary to understand the complexity of the Fourier transform *within* the unitary group.

This note presents a method of computing complexity *within* the unitary group. It is important to note that we are in the classical setting, not quantum. A quantum version of the Fourier transform can be computed in time $O(\log^2 n)$ using an algorithm by Shor (refer e.g. to Chapter 5, [1]) but that setting is different. The model of computation which we present in Section 2 allows the algorithm, in each step, to apply a 2×2 unitary operator on two coordinates of an n dimensional complex vector, serving as both input and output. In Section 4.5.4 in [1], it is shown that in the same model we consider here, most (with respect to the Häar measure) unitary operators require $\Omega(n^2)$ steps using a simple counting argument. This work presents a tight analysis for the special case of Fourier transform.

2 The Unitary Layered Circuit

Our model of computation consists of layers $L_0, \ldots, L_m$, each containing exactly n nodes and representing a vector in $\mathbb{C}^n$. The first layer, $L_0 \in \mathbb{C}^n$, is the input. The last layer $L_m \in \mathbb{C}^n$ is the output. For each layer $i \geq 1$ there are two indices $k_i, \ell_i \in [n]$, $k_i < \ell_i$, and a complex unitary matrix

$$A_i = \begin{pmatrix} a_i(1,1) & a_i(1,2) \\ a_i(2,1) & a_i(2,2) \end{pmatrix} .$$

For each $j \notin \{k_i, \ell_i\}$, $L_i(j) = L_{i-1}(j)$. The values of $L_i(k_i)$ and $L_i(\ell_i)$ are given as

$$\begin{pmatrix} L_i(k_i) \\ L_i(\ell_i) \end{pmatrix} = A_i \begin{pmatrix} L_{i-1}(k_i) \\ L_{i-1}(\ell_i) \end{pmatrix} .$$

In words, the next layer is obtained from the current layer by applying a 2-by-2 unitary transformation on two coordinates. Compared to Morgenstern's model of computation the unitary layered circuit is strictly weaker. To see why it is not stronger, notice that the matrix elements of A_i all have modulus at most 1. It is strictly weaker because it uses only unitary transformations, but also because it has a bounded memory of n numbers at any given moment. Indeed, it is not possible in layer L_{i+1} to use a coordinate of $L_{i'}$ for $i' < i$. Still, the normalized FFT is implemented as a unitary layered circuit with $m = O(n \log n)$.

Theorem 2.1. If a layered circuit given by $A_1, \ldots, A_m \in \mathbb{C}^{2\times 2}$, $k_1, \ldots, k_m \in [n]$ and $\ell_1, \ldots, \ell_m \in [n]$ computes the normalized Fourier transform, then $m \geq \frac{1}{2} n \log_2 n$.

Proof. For a matrix M and a set $I \subseteq [n]$ of indices, let $M[I]$ denote the principal minor corresponding to the set I. For $i = 1, \ldots, m$ let $\tilde{A}_i$ denote the matrix defined so that $\tilde{A}_i[\{k_i, \ell_i\}] = A_i$, $\tilde{A}_i[[n] \setminus \{k_i, \ell_i\}] = \mathrm{Id}$ and $\tilde{A}_i(p,q) = 0$ whenever exactly one of p, q is in $\{k_i, \ell_i\}$. It is clear that

$$L_i = \tilde{A}_i \tilde{A}_{i-1} \cdots \tilde{A}_1 L_0 .$$

It hence makes sense to define $M_i = \tilde{A}_i \tilde{A}_{i-1} \cdots \tilde{A}_1$. Note that $M_m = F$, where F is the normalized FFT matrix. We also define $M_0 = \mathrm{Id}$. For a matrix M, we now define a potential function

$$\Phi(M) = -\sum_{p,q} |M(p,q)|^2 \log |M(p,q)|^2 ,$$

where we formally define $0\log 0$ to be $\lim_{x\to 0^+} x\log x = 0$. For a unitary matrix M, we notice that $\Phi(M)$ is the sum of the Shannon entropies of the probability vectors given by the squared moduli of the elements of each row. Also notice that $\Phi(\mathrm{Id}) = 0$ and $\Phi(F) = n\log_2 n$, as all elements have modulus $1/\sqrt{n}$. We now show that for any $i \geq 1$,

$$|\Phi(M_i) - \Phi(M_{i-1})| \leq 2\ . \tag{2.1}$$

This clearly implies the theorem statement.

To see (2.1), notice that M_i is obtained from M_{i-1} by replacing rows k_i and ℓ_i as follows. If x and y denote rows k_i and ℓ_i of M_{i-1}, respectively, and x', y' the corresponding rows of M_i , then $x' = a_i(1,1)x + a_i(1,2)y$ and $y' = a_i(2,1)x + a_i(2,2)y$. All other rows remain untouched. We also have by orthonormality that $\|x\|^2 = \|y\|^2 = \|x'\|^2 = \|y'\|^2 = 1$ and that for all $j \in [n]$,

$$|x'(j)|^2 + |y'(j)|^2 = |x(j)|^2 + |y(j)|^2 =: r(j)\ .$$

Now let $\mathcal{P}_r$ denote the set of pairs of vectors $(\alpha, \beta) \in [0,1]^n \times [0,1]^n$ satisfying:

1. $\sum_{j=1}^n \alpha(j) = 1$
2. $\sum_{j=1}^n \beta(j) = 1$
3. $\alpha(j) + \beta(j) = r(j)$ for $j \in [n]$.

For $(\alpha, \beta) \in \mathcal{P}_r$ now let

$$\Phi(\alpha, \beta) = -\sum \alpha(j)\log_2 \alpha(j) - \sum \beta(j)\log_2 \beta(j)$$

(abusing notation). Then it suffices to show that

$$\sup_{(\alpha,\beta)\in\mathcal{P}_r} \Phi(\alpha,\beta) - \inf_{(\alpha,\beta)\in\mathcal{P}_r} \Phi(\alpha,\beta) \leq 2\ .$$

Indeed, for each $j \in [n]$, the expression $-\alpha(j)\log_2\alpha(j) - \beta(j)\log_2\beta(j)$ is minimized, subject to $\alpha(j) + \beta(j) = r(j)$, when $\alpha(j) = 0$ and $\beta(j) = r(j)$ (or vice versia), in which case its value is $-r(j)\log_2 r(j)$. It is maximized when $\alpha(j) = \beta(j) = r(j)/2$, in which case its value is $-r(j)\log_2 r(j) + r(j)$. Hence,

$$\begin{aligned} \sup \Phi(\alpha,\beta) &\leq -\sum_{j=1}^n r(j)\log_2 r(j) + 2 \\ \inf \Phi(\alpha,\beta) &\geq -\sum_{j=1}^n r(j)\log_2 r(j)\ . \end{aligned}$$

(Note that the RHS of the second inequality might not be feasibly obtained by a pair $(\alpha,\beta) \in \mathcal{P}_r$, but this does not matter.) The difference is at most $\sum_{j=1}^n r(j) = 2$. By the above discussion, both vector pairs

$$(\alpha,\beta) = ((|x(1)|^2, \ldots, |x(n)|^2), (|y(1)|^2, \ldots, |y(n)|^2))$$

and

$$(\alpha',\beta') = ((|x'(1)|^2, \ldots, |x'(n)|^2), (|y'(1)|^2, \ldots, |y'(n)|^2))$$

are in $\mathcal{P}_r$. This implies (2.1), as required. □

References

[1] ISAAC L. CHUANG AND MICHAEL A. NIELSEN: *Quantum Computation and Quantum Information*. Cambridge University Press, 2010. 3

[2] J. W COOLEY AND J. W TUKEY: An algorithm for the machine computation of complex Fourier series. *J. of American Math. Soc.*, pp. 297–301, 1964. 2

[3] JACQUES MORGENSTERN: Note on a lower bound on the linear complexity of the fast Fourier transform. *J. ACM*, 20(2):305–306, April 1973. 2

[4] CHRISTOS H. PAPADIMITRIOU: Optimality of the fast Fourier transform. *J. ACM*, 26(1):95–102, January 1979. 2

[5] S. WINOGRAD: On computing the discrete Fourier transform. *Proc. Nat. Assoc. Sci.*, 73(4):1005–1006, 1976. 2

AUTHOR

Nir Ailon
Assistant Professor
Department of Computer Science
Technion Israel Institute of Technology
Haifa, Israel
nailon@cs.technion.ac.il