Journal of Computational and Applied Mathematics 235 (2010) 74–81

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Obtaining the Quantum Fourier Transform from the classical FFT with QR decomposition

F.L. Marquezino [a,*], R. Portugal [a], F.D. Sasse [b]

[a] Laboratório Nacional de Computação Científica, Rua Getúlio Vargas, 333, Petrópolis, 25651-075, RJ, Brazil

[b] Universidade do Estado de Santa Catarina, Department of Mathematics, CCT, Joinville, 89223-100, SC, Brazil

ARTICLE INFO

Article history:
Received 30 January 2008
Received in revised form 6 December 2009

MSC:
81-08
65T50
68W40

Keywords:
Quantum computing
Quantum Fourier Transform
Quantum circuit design

ABSTRACT

We present the detailed process of converting the classical Fourier Transform algorithm into the quantum one by using QR decomposition. This provides an example of a technique for building quantum algorithms using classical ones. The Quantum Fourier Transform is one of the most important quantum subroutines known at present, used in most algorithms that have exponential speed-up compared to the classical ones. We briefly review Fast Fourier Transform and then make explicit all the steps that led to the quantum formulation of the algorithm, generalizing Coppersmith's work.

1. Introduction

For many years the spectral analysis of sampled data over a finite range, referred to as the Discrete Fourier Transform (DFT) was performed directly on computers, using $O(N^2)$ operations, where N is the number of data points. A great milestone in Fourier analysis was the paper published in 1965 in [1], where they described the so-called Fast Fourier Transform (FFT) algorithm, which can compute the DFT with only $O(N \log N)$ operations. In the next year, Rudnick presented a computer program which also required $O(N \log N)$ operations [2], being inspired by an earlier method due to [3].

Although the FFT algorithm may seem reasonably fast for classical Computer Science, it turns out that its quantum version can provide exponentially faster algorithms for some problems. In 1994, Shor [4] developed a quantum version of the Fourier Transform when the prime factors of N are not large (smaller than $\log N$). Motivated by Shor's work, Coppersmith [5] developed the quantum version of the FFT when N is a power of two, which required only $O(\log^2 N)$ operations. In the same year [6], using a recursive approach, has also shown how to implement the Fourier Transform in quantum computers.[1] This is, with no doubt, the most important quantum subroutine designed so far. Most quantum algorithms with exponential improvement when compared to the classical counterparts use the Quantum Fourier Transform (QFT) as an essential part. These algorithms solve instances of the Hidden Subgroup Problem (HSP), which has several important applications. A good survey on the HSP can be found in [8].

The quantum algorithm for an approximate QFT was developed in [5], with complexity $O(m \log N)$, where $1 \leq m \leq \log N$ is the parameter that defines the degree of approximation, usually taken as $O(\log \log N)$. Indeed, this approximate algorithm is the one with practical applications. The exact transform yields a level of accuracy that, in most of the cases, cannot even be

* Corresponding author.
E-mail addresses: franklin@lncc.br, franklin.marquezino@gmail.com (F.L. Marquezino), portugal@lncc.br (R. Portugal), fsasse@joinville.udesc.br (F.D. Sasse).

[1] Nielsen and Chuang [7] also mention an unpublished work on this subject by David Deutsch.

0377-0427/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2010.05.012

detected by the measuring device. Barenco et al. [9] showed that, in the presence of decoherence, i.e., in realistic situations, the approximate transform may achieve better results than the exact one. They also showed that the accuracy of the exact QFT can be achieved by applying the approximate QFT repeatedly $O\left(\frac{\log^3 N}{m^3}\right)$ times.

In this paper we build the QFT algorithm from the classical FFT, generalizing Coppersmith's work. In order to accomplish this generalization, we use the QR decomposition. This technique may provide insight for the development of new quantum algorithms.

In Section 2 we review the definition of DFT and the classical FFT algorithm. We also fix the notation that will be used in the generalization of the quantum algorithm. In Section 3 we perform the decomposition of the matrix form of FFT algorithm. QR decomposition plays an essential role in this task. The resulting unitary matrices can be interpreted as quantum logic gates. In Section 4 we summarize the results obtained in the previous sections and show how to build the circuit for the exact QFT.

2. The Discrete Fourier Transform

In this section we address the DFT in a form that will be useful later on. It is important to establish some notations that will be used throughout this paper. Let n be a positive integer, and let a, c be n-bit integers. The binary representations of a and c will be denoted, respectively, by

$$(a_{n-1}a_{n-2}\ldots a_0)_2 \quad \text{and} \quad (c_{n-1}c_{n-2}\ldots c_0)_2,$$

such that,

$$a = \sum_{j=0}^{n-1} a_j 2^j \quad \text{and} \quad c = \sum_{j=0}^{n-1} c_j 2^j.$$

Binary representations of numbers may have the subscript $(\cdot)_2$ omitted when context is clear. All logarithms are base two. Let X and Y be arrays[2] of $N = 2^n$ complex numbers. The integers a and c defined above will be useful for indexing X and Y. The notation used in this paper for that indexing is X_a. Let $\omega \equiv \omega_N = \exp(2\pi \mathrm{i}/N)$, be the N-root of unity.

Finally, we may define the DFT.

Definition 1 (*DFT*)**.** The Discrete Fourier Transform takes as input a complex array X, and converts it into a complex array Y, according to the expression

$$Y_c = \frac{1}{\sqrt{N}} \sum_{a=0}^{N-1} X_a \omega^{ac}. \tag{1}$$

This equation may be rewritten by making explicit the binary representation of a and c. We concentrate only on the sub-term ω^{ac}.

$$\omega^{ac} = \exp\left(\frac{2\pi \mathrm{i}}{N} \sum_{0\le j,k\le n-1} a_j c_k 2^{j+k}\right). \tag{2}$$

Using $\omega^{(2^n)} = 1$, we can state that

$$\omega^{(2^{j+k})} = \omega^{(2^n)2^{j+k-n}} = 1, \tag{3}$$

if $j + k \ge n$. Therefore, it is possible to eliminate terms from Eq. (2) when $j + k \ge n$, leading us to an equivalent definition of the DFT,

$$Y_c = \frac{1}{\sqrt{N}} \sum_{0\le a\le N-1} X_a \exp\left(\frac{2\pi \mathrm{i}}{N} \sum_{\substack{0\le j,k\le n-1 \\ j+k\le n-1}} a_j c_k 2^{j+k}\right). \tag{4}$$

Coppersmith [5] observed that we may define a more general transform by changing the range of $j + k$ in Eq. (4). If we take $n - m \le j + k \le n - 1$, where m is a parameter such that $1 \le m \le n$, we obtain the definition of the Approximate Fourier Transform. When $m = n$, we recover the definition of the exact DFT. When $m = 1$, we achieve the a slightly modified definition of the well-known Hadamard transform—with a different indexing of the elements in the array. The argument in the exponent of this new transform differs from that in the exponent of DFT just by

$$\mathrm{i}\epsilon = \frac{2\pi \mathrm{i}}{N} \sum_{\substack{0\le j,k\le n-1 \\ j+k<n-m}} a_j c_k 2^{j+k}. \tag{5}$$

[2] Some readers may prefer to see X and Y as functions, respectively $X : \mathbb{Z}_N \to \mathbb{C}$, and $Y : \mathbb{Z}_N \to \mathbb{C}$, where $N = 2^n$.