Review: "A Comparison of Quantum and Traditional Fourier Transform Computations"

Matthias Troyer[1]

[1]Affiliation not available

August 20, 2020

The manuscript *"A Comparison of Quantum and Traditional Fourier Transform Computations"* discusses a very important and often overlooked aspect of quantum computing, namely a fair and detailed comparison of a quantum algorithm, its classical simulation, and its classical counterpart taking into account the complexity of I/O. Such an article is valuable and worth publishing. The current manuscript, however, still contains some inaccuracies that should be fixed and I will make suggestion on how the presentation can be improved.

Let me first summarize the main result in my own words: a quantum Fourier transform (QFT) can calculate the Fourier transform of a vector with time complexity $O\left(\log^2 N\right)$, compared to the complexity $O\left(N \log N\right)$ of a classical FFT. However, if one needs to read out the full vector instead, the complexity becomes $O\left(N \text{ polylog}(N)\right)$ again, without an advantage over the classical algorithm. It is actually worse than the author discusses, if one also takes into account the complexity of loading the initial state. Loading an initial classical data vector has complexity $O\left(N\right)$, and this has to be repeated at every repetition, giving a complexity of $O\left(N^2 \text{ polylog } N\right)$, worse than classical. As the author correctly mentions, the QFT is thus a useful algorithm if the input data is prepared algorithmically, and limited sampling of the result vector is sufficient, as is the case for Shor's algorithm.

This is a valuable observation that deserves a paper in CiSE, since these important considerations are not all known to people outside the quantum computing community, and are sometimes ignored even by quantum computing specialists. I thus want to encourage the author to improve the presentation to make it more accessible to a broader audience and fix a couple of technical flaws.

Before discussing presentation issues, I want to address one technical flaw and a few points where more clarity is needed:

1. This statement towards the end is too simplistic and needs to be clarified: "... if we want to measure each coefficient, we must redo our operations for each coefficient (since our wave function will collapse for every measurement)." The manuscript does not explain how to read out a specific coefficient. If one samples the wave function, one measures the result and gets a certain value $(s_1, \cdots, s_N)$ with a probability depending on the wave function. More complex amplitude-estimation algorithms are needed to read out specific coefficients, which then takes time $O(N/\epsilon)$. This needs to be better explained. This also raises another issue of normalization and precision. The quantum wave function has to be normalized to have an L_2 norm of 1. We thus need to measure to a precision of epsilon divided by the L_2 norm of the classical data, and if that increases with N the scaling is even worse. For example, if all entries are of order unity, the L_2 norm is $\sqrt{N}$.
2. The one technical flaw in the paper is in preparing the input state to the QFT, and the complexity of the classical simulation of the quantum algorithm. There, the numbers are wrong. The classical complexity can easily be estimated by realizing that any sparse quantum gate (such as a 1-qubit or 1-qubit gate) can be simulated in time $O(N) = O(2^n)$, where $N = 2^n$. Thus, the overhead is just $O(N)$,

not $O(N^2)$, and the simulation complexity is just $O(N)$ times the complexity of the quantum algorithm. This will be more apparent if, as I propose below, the author shows the quantum algorithm also through a sequence of quantum gates. However, note that in the simulation we then have the full vector, and thus do not incur the overhead of quantum state tomography. We need to simulate the algorithm only once and not $O(N/\epsilon)$ times. This means the simulation remains at $O(N\ \log^2 N)$ even if we read out all entries, which is not bad compared to the classical $O(N\ \log N)$ of the FFT. The reason why the author seems to end with a different complexity is that his Matlab simulation does not simulate the quantum algorithm but the computation of the effect of the QFT applied to just one basis state. That is a suboptimal implementation. I thus propose to replace the Matlab code by a discussion of how a quantum gate (Hadamard or controlled phase rotation) can be implemented, and that will then nicely give the scaling discussed above.

3. As mentioned in my summary at the beginning, the value of the paper can be increased if the author could also discuss the complexity of quantum state preparation from classical input data, e.g., following Shende-Bullock-Markov (Shende et al., 2006), which has a complexity of $O\left(N\ \log\ \frac{1}{\epsilon}\right)$. That means, that if the data is read from a classical vector (and not computed as, e.g., in Shor's algorithm), then the complexity of state preparation of $O\left(N\ \log\ \frac{1}{\epsilon}\right)$ completely dominates the QFT itself, and if one furthermore wants to read out the full vector instead of sampling it, the complexity becomes $O\left(N^2 \log^2\left(\frac{1}{\epsilon}\right)\right)$.
4. In the conclusion the author writes that "QEC research is still in very early development and it is currently difficult to determine how the required resources for these corrections could scale with qubit usage." That is incorrect, as the overhead is pretty well known by now for certain QEC codes, such as the surface code. The asymptotic scaling of the overhead has long been known, and also detailed resource costs have been worked out for various algorithms. I suggest to focus the QEC section on the need for fault tolerance, and the large overhead associated with it. This is definitely not something that can be done on NISQ devices at an interesting scale. I would also use another reference for QEC than the Gil Kalai paper.

Now, to suggestions for improved presentation:

1. In the abstract, I suggest to present fewer technical details. "radix-2 DIT case of the Cooley-Tukey Algorithm" can just be called Fast Fourier transform, the QUBIT4MATLAB package does not need to be mentioned in the abstract, and neither does the "Master Theorem" have to be mentioned.
2. In the abstract and in the rest of the paper, it is also better to talk about quantum speedup and not quantum supremacy. The paper is about asymptotic scaling (the author uses the big-O notation throughout), and not about supremacy, which is a specific size problem that is solved better on quantum hardware rather than classical hardware. A supremacy claim needs all constants to be worked out, and assumptions for the specific classical and quantum hardware. Replacing quantum supremacy by quantum speedup can fix this. The observations are still valuable and correct.
3. The first section is called "The shortcomings of classical computing and the early developments of quantum computing". That title sounds strange to me. What are the shortcomings? The author may rather want to stress the additional capabilities of quantum computers.
4. Given that most readers will not be physicists and have no experience with quantum mechanics, it would be valuable to expand this first section. Discuss equation (3) before equation (1), and then introduce equation (1), to show that this means that the state of a qubit is actually a two-dimensional complex vector, After that only go to the many-quit state. After equation 4 it may be good to talk about amplitudes first, and then probabilities. Saying that the state is determined by "the qubit simply has a probability…", ignores the all important phases that are the crucial difference between quantum computing and probabilistic classical computing.
5. "Quantum gates, unlike classical gates, do not delete information and are fully reversible." Needs further explanation. This comes from the reversibility of the microscopic laws of physics. Here it may be worthwhile to mention that, as quantum gates are time evolution of a quantum system, they are

unitary operators on the Hilbert space that was introduced above. The discussion in the next sentence about them not using energy is misleading as well. In fact, the control and operation of the quantum gate always needs energy. It is only the free evolution of a quantum system that conserves energy. The part on " subatomic systems such as a superposition of quantum states. " would also need more explanation. Does the author here want to describe specific implementations of qubits? If so, more detail is needed. If not, I propose to drop this part.

6. "The definition and use of the classical discrete Fourier transform": here is an abrupt jump from Shor's algorithm to the DFT. Since Shor's algorithm again gets mentioned later on as a case where the QFT is better than a classical DFT, it make sense to discuss more details of Shor's algorithm, and that in essence it solvers the factor problem by using a QFT to find the period of a function. In light of the above complexity discussion this is important: we don't deal with classical data that one has to load but with a state that is computed (the function f(x) = aˆx mod N), and one is not interested in the full vector but just in finding the period. This can be done just by measuring the result of the QFT, which in this particular case will return a random multiple of the base frequency, and with a few repeated measurements one can then extract the period).
7. The Fourier transform should be well known to the reader, and it is thus not obvious to me that equation 6 is needed. I would instead introduce \omega in equation 5. For the discussion of the implementation I suggest to mention that the rest of the paper limits itself to the case of powers of 2, i.e. N=2ˆn, for which the radix-2 version applies. Since the implementation is well known for most computational scientists, it could be moved from to supplementary material, as can be the derivation of the O(N log N) complexity, which is also well known.
8. In the discussion of the QFT, unfortunately the quantum algorithm is not presented. What is presented is the mathematical unitary operation. It would be very valuable for the reader to see an actual implementation of the quantum algorithm, i.e. the sequence of O(nˆ2) Hadamard gates and controlled rotations – either as a circuit drawing, or more useful as quantum code, either using a quantum programming language that allows loops (such as Q#), or as some quantum pseudo code with loops. Then the reader will see the sequence of O(nˆ2) operations and how it creates the unitary operation of the QFT.
9. "The implementation of the QFT in MATLAB...". This section can be improved as well., As discussed above, the Matlab code does not implement the quantum algorithm but rather implements the application of the unitary operation to one of the basis states. It would be more useful for the reader to instead get implementations of the Hadamard and controlled phase gates, which could then be combined with the above quantum algorithm (maybe all in Matlab even?) for a full simulation of the algorithm – and in time O(N nˆ2), not O(Nˆ2 nˆ2).
10. "Evaluating the complexity of the QFT Implementation" – I discussed the issues with the complexity estimates already above. If one replaces it by an estimate of the complexity of simulating the H and controlled phase gate each in O(N) operations then it both becomes simpler and the complexity becomes just O(N) times that of the quantum version
11. "Evaluating the complexity of the theoretical form of the QFT Implementation". I assume that by "theoretical" the author means the actual quantum operation? The comparison section needs to be fixed, since that's where there is the mentioned flaw based on the suboptimal Matlab code, and that will also change table 1. I suggest that table 1 could have three cases: A) ignoring state preparation and one only wants to sample the output a constant number of times; B) ignoring state preparation but one wants to read the full vector; C) including state preparation from classical data and one only wants to sample the output a constant number of times; D) including state preparation from classical data and one wants to read the full vector
12. The reader will then see that only case A has a speedup, and that in the other cases even the simulation of the quantum algorithm is faster than the quantum algorithm on a quantum computer.

The conclusions should then be updated, taking these slightly modified comparisons into account. The case of Shor's algorithm, which falls into category A) above, is indeed interesting but deserves a deeper explanation

so that the reader understands that Shor's algorithm can compute the input with complexity $O\left(N^3\right)$ (up to log factors), and why only a few samples of the output are needed. That way, the exponential speedup over the FFT remains, and the super polynomial speedup over the name sieve. However, if classical data has to be loaded or should be returned then there is no speedup. I call this the I/O problem of quantum algorithms, which indeed—as the author writes—means that in these cases quantum computing is no better than classical computing.

This is an important observation that the paper beautifully works out. Fixing the flaw in the simulation section and improving the presentation will make this a very nice paper.

References

Synthesis of Quantum Logic Circuits. (2006). https://arxiv.org/abs/quant-ph/0406176. https://arxiv.org/abs/quant-ph/0406176