Comparison of resources required for monolithic and distributed versions of quantum phase estimation

Petra Brčić

July 2024

Abstract

A quantum algorithm is useful only if it can be successfully and efficiently executed on future, full-scale, error-corrected quantum computers. It is crucial to perform quantum resource estimation (QRE) for prospective algorithms to assess if an algorithm will be executable using realistic resources and whether it truly gives quantum advantage, and to determine the requirements for future hardware. At the same time, to achieve practical, fault-tolerant quantum computers, the main challenge is scaling to big qubit numbers. Distributed quantum computing offers a way to achieve this by connecting multiple moderately-sized quantum processors. For these reasons, we perform QRE for ordinary (monolithic) and distributed quantum phase estimation algorithms and compare the required resources. The distributed approach is found to indeed be promising in reduction of required resources, however the complexities arising from communication between multiple quantum processors might negate the improvements, and the current tools might not be suitable to properly take all physical aspects into account.

1 Introduction

A great effort is put into finding quantum algorithms that solve important classically-intractable problems. For these algorithms to be useful, they need to be executable on real, full-scale, fault-tolerant quantum computers, which we are yet to develop. It is of great significance to estimate resources required for a successful algorithm execution in order to assess the algorithm's feasibility, to compare efficiency of different algorithms, and to determine hardware requirements. This way, quantum resource estimation (QRE) can determine which quantum algorithms give quantum advantage under realistic assumptions, and allows us to specify necessary and sufficient conditions for future quantum computers to be practical.

In this work, we will explore how required resources vary for the monolithic (ordinary, non-distributed) and distributed approaches to quantum computing, in particular by implementing the quantum phase estimation (QPE) algorithm [1]. It is very difficult to scale quantum computers to large numbers of qubits, but distributed quantum computing [2, 3] equips us with a way around this. The idea is to connect multiple quantum processors (each with limited number of qubits) to work together, instead of relying on a single, big quantum computer. To implement distributed QPE, we follow the Ref. [4], and we use Qiskit [5]. To perform QRE, we use Azure Quantum Resource Estimator [6] and feed it our Qiskit circuit.

The following section provides the theoretical background of QPE, distributed QPE and QRE. Section 3 demonstrates the results and accompanied analysis. Finally, Section 4 concludes the work and offers future directions.

2 Theoretical background

Here we introduce QPE, explain how to implement distributed QPE, and finally familiarise with the QRE background.

2.1 Quantum phase estimation

The QPE algorithm evaluates an approximation of an eigenvalue of a given unitary U and a corresponding eigenvector. It has numerous applications, the most famous being Shor's algorithm [7]. Let U be a unitary operator on m qubits, and $|\psi\rangle$ an eigenstate of U. Then $U|\psi\rangle = \exp(2\pi i\phi)|\psi\rangle$. We can write ϕ in binary representation as:

$$\phi = \sum_{i=0}^{\infty} \frac{\phi_i}{2^i} = 0.\phi_1\phi_2... \tag{1}$$

We can truncate the sum in Eq. 1 to n, yielding an n-bit approximation of $\phi \approx 0.\phi_1\phi_2...\phi_n$. This is the result of QPE.

The implementation is depicted in Fig. 1. First we apply

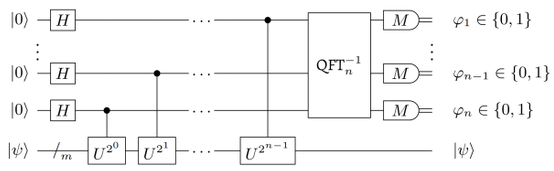

Figure 1: Circuit implementation of the quantum phase estimation algorithm for a unitary operator U. Taken from Ref. [4].

Hadamard gates to the n-qubit register to prepare an equal superposition state. The m-qubit register is prepared in $|\psi\rangle$. Afterwards, controlled-$U^{2^{n-i}}$ gates are applied, with the control qubit i in the first register and the second register as target. Then an inverse quantum Fourier transform on the first register is applied and the qubits are measured. The result is interpreted as the n-bit phase approximation of ϕ. The results in this work are for the QPE performed on the T gate:

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \tag{2}$$

and its eigenvector $|1\rangle$, for which the phase is $\phi = 1/8$.

2.2 Distributed controlled U-gate

When converting a monolithic algorithm into a distributed one, the immediate issue is implementing 2-qubit gates between two different quantum processors. A set of universal quantum gates for local operations is given by a CNOT-gate and single qubit rotations [8]. It follows that a universal gate-set for non-local operations is given by adding the non-local CNOT gate to the the local universal gate-set. One possible implementation of a distributed controlled-U gate between two qubits $|\psi\rangle$ and $|\phi\rangle$ on two different processors is depicted in Fig. 2 [4]. We require each device to have an extra *communication* qubit which will share an entangled Bell state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and have access to classical communication. The operation E_2 entangles the communication qubits $|0\rangle_1$ and $|0\rangle_2$, and M is the measurement. Double lines indicate that a gate is controlled on a classical value (the measurement result).

2.3 Distributed quantum Fourier transform

The final piece to distributed QPE is the distributed quantum Fourier transform (QFT). It transforms an n-qubit state $|k\rangle$ to $\sum_{j=0}^{2^n-1} e^{2\pi ijk/2^n}|j\rangle$. We will utilise its recursive implementation given in Fig. 3. We notice that the

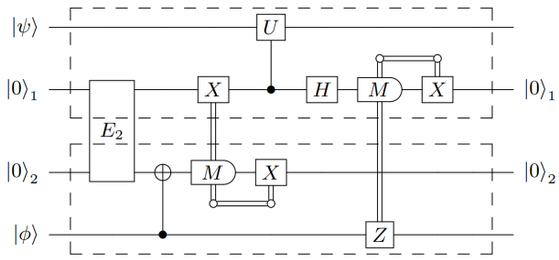

Figure 2: Circuit implementation of a distributed controlled-U gate acting on qubits $|\phi\rangle$ and $|\psi\rangle$. Taken from Ref. [4].

decomposition is confined to Hadamard and controlled-R_k gates, so that:

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix} \tag{3}$$

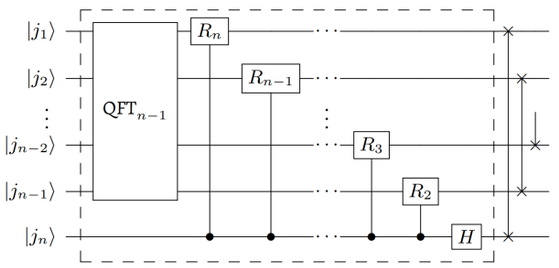

Figure 3: Recursive implementation of the quantum Fourier transform. Taken from Ref. [4].

Ref.[4] notes that the SWAP gates can be omitted because they introduce a high computational overhead, while they are easily accounted for classically (by rearranging the measurement results). To obtain the distributed QFT, we need to use the non-local version of the controlled-R_k gates. We use the method in Fig. 2, but instead of repeating it for each individual gate, we combine all operations that use the same control qubit, as illustrated in Fig. 4 (the "combined" approach in Ref. [4]). This way, we use one entangled pair for each non-local QFT "recursive layer", and we significantly reduce the communication overhead.

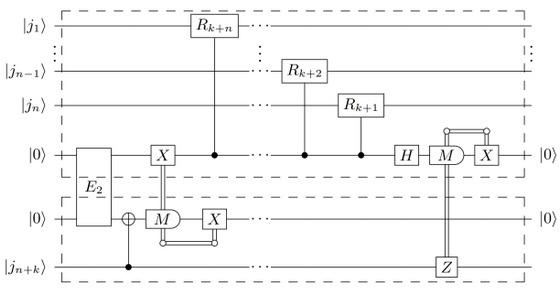

Figure 4: The "combined" implementation of a non-local recursive QFT layer. Taken from Ref. [4].

2.4 Quantum resource estimation

QRE [9, 10] explores the number of physical qubits and time required to execute a quantum algorithm given a specific fault-tolerant quantum computer (with realistic physical qubit parameters and error correction schemes). QRE is explained in more detail in Ref. [6]. In this work, we are concerned about physical qubits, runtime, comparing qubit technologies, and the impact of error budgets.

3 Results and discussion

When running the estimator, we can specify whether we want to minimise the runtime or the number of physical qubits. We specify the total number of (logical) qubits, n, available (these include both the qubits needed for computation and the communication qubits), we find all factors of n smaller than n and divide n into the corresponding number of quantum processor units (QPUs). Each QPU has to consist of at least 2 qubits (one for computation and one for communication). The default qubit technology is superconducting, and we take the realistic parameters.

3.1 Dependence of time and physical qubits on the number of QPUs

Here we explore how minimising for time and physical qubits differ.

3.1.1 Minimising for time

We set the estimator setting to minimise for runtime; the results can be seen in Fig. 5. We notice that introducing a small number of QPUs instead of using a single big QPU dramatically increases the required execution time, while physical qubits are not significantly affected. However, as we progress with increasing the QPU number (and in turn making the QPUs have less qubits), the runtime tapers off and physical qubits show a little decrease.

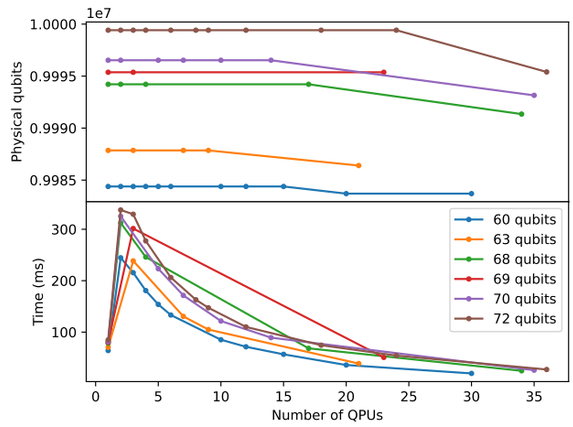

Figure 5: Quantum resource estimation performed for minimum runtime. A total number of qubits is split across an increasing number of quantum processors (of equal sizes).

3.1.2 Minimising for physical qubits

Now, let us minimise for the number of physical qubits. The results are shown in Fig. 6. We notice that physical qubits are now an order of magnitude smaller than in the runtime minimisation case, while the runtime increased by 1-2 orders of magnitude. Since we are now dealing with runtime in terms of seconds, we deem this trade-off acceptable and we minimise for physical qubits for the rest of the report. Increasing the number of QPUs among which the qubits are shared, we observe a much more prominent decrease in physical qubits (compared to Fig. 5), and a general decreasing trend for the runtime (particularly steep for bigger total qubit numbers).

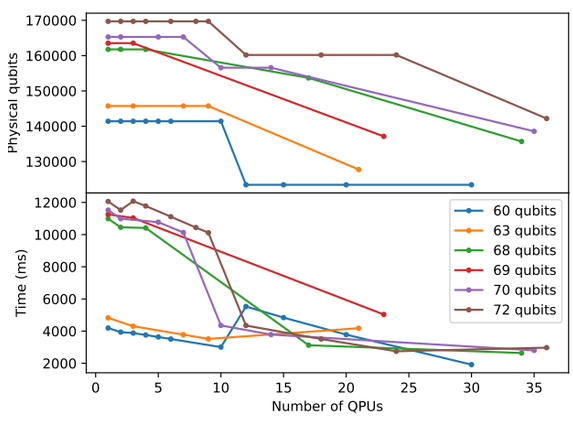

Figure 6: Quantum resource estimation performed while minimising for the number of physical qubits.

3.2 Comparison of qubit technologies

The Azure QRE tool features default qubit models that include parameter sets that describe superconducting qubits, trapped ion qubits, and Majorana qubits, with each model consisting of two different regimes corresponding to realistic and optimistic estimates of the physical error rates. The first two models are gate-based, while Majorana qubits [11, 12] offer a topological approach, and are still under research. Here we compare the three qubit technologies using realistic (Fig. 7) and optimistic parameters (Fig. 8) for a total number of qubits of 84.

For the realistic case, Majorana qubits perform the best.

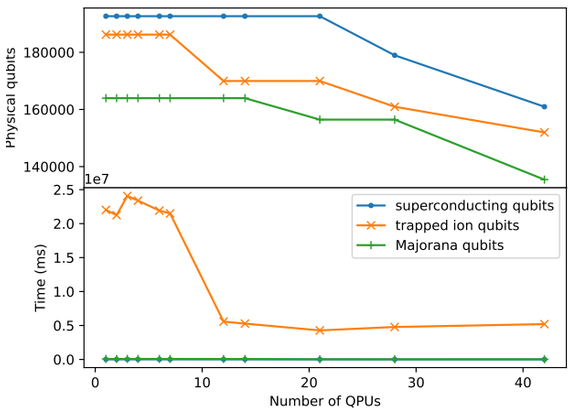

Figure 7: Quantum resource estimation performed for the total number of qubits of 84, and for minimum physical qubits. The estimator was fed realistic parameters corresponding to different qubit technologies.

Superconducting share the same runtime, but use up the most physical qubits overall. Trapped ions come in-between in terms of physical qubits, however their execution time is significantly worse for small numbers of QPUs.

In the optimistic case, Majorana qubits outperform by

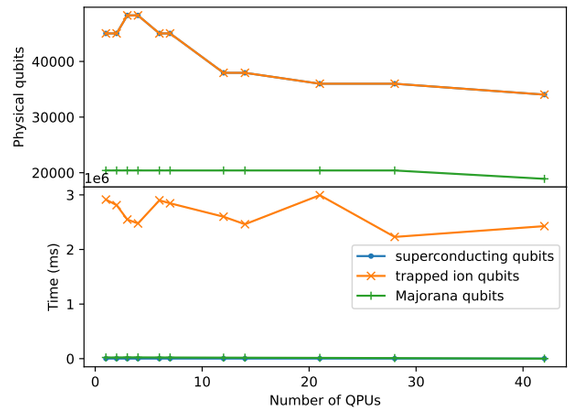

Figure 8: Comparison of resources needed for different qubit technologies repeated for optimistic parameters.

far. Superconducting and trapped ion have the same physical qubit requirements, while the latter again significantly underperforms in terms of the runtime.

3.3 Error budget

Error budget specifies the maximum acceptable failure rate of an algorithm. It determines the parameters of the quantum error correction scheme to be deployed, which in turn affects the required resources. If the algorithm we want to execute yields results which are easy to check for correctness, we can increase the error budget. This will reduce the accuracy (which does not matter because we can easily verify), but we expect that the required resources will reduce as well. We might use the QPE, for example, in context of Shor's algorithm which finds prime factors of an integer. It is easy to check whether the output is a prime factor of the input, so it makes sense to explore different error budgets for QPE. In figure Fig. 9, where we ran QRE for 84 total realistic superconducting qubits, we can see that increasing the error budget indeed significantly reduces required resources with the smallest QPU sizes being the most effective.

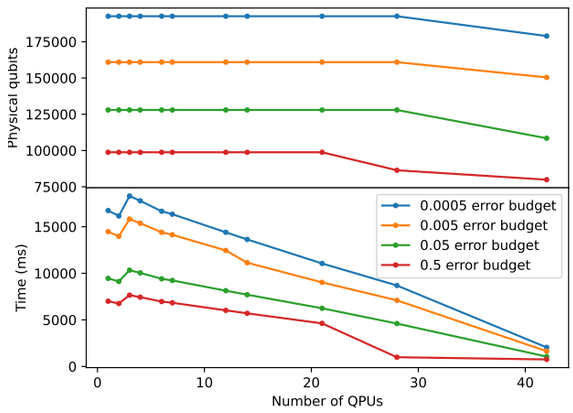

Figure 9: Quantum resource estimation performed for the total number of qubits of 84, while minimising physical qubits. Resources required for different error budgets are shown.

3.4 Realistic distributed quantum computing

It is necessary to discuss how realistic our model is. The first issue arises from communication between two non-local QPUs – generating entanglement takes time; it requires a purification protocol [13], and if we are using photons there is transmission loss in optical fibers. Our model does not take into account any of these. Accounting for purification and loss is not trivial using our tools. A way to account for extra time is to increase Azure's two-qubit gate time parameter which affects only two-qubit Clifford gates - ideal for CNOT and CZ involved in quantum teleportation. Furthermore, we do not take into account how the QPUs would be interconnected in a realistic scenario, instead we assume we can communicate directly between any two QPUs at any distance. Decreasing the size of QPUs while increasing their number increases the number of connections required and the network becomes more complex. Our analysis does not take this into account, and perhaps the improvements offered by the distributed approach would therefore be negated to some degree.

4 Conclusions and outlook

A comparison of required resources for monolithic and distributed QPE implementations was performed. Estimations were obtained under constraints of minimum runtime and minimum physical qubits, and for different quantum technologies and error budgets. The distributed approach demonstrated big potential in decreasing the required resources. Since it is harder to build quantum processors the more qubits are added, the distributed paradigm of connecting multiple smaller processor is promising. However, the more QPUs are added, the more complex our processor network becomes, which might counter-act the improvements. Maybe the best approach is somewhere in the middle, using just a handful of middle-sized QPUs. However, there is no conclusive evidence for this in this work, which is also influenced by the fact that it doesn't fully account for a realistic scenario. It is possible that Qiskit and Azure's estimator are not the right tools for this problem, and more suitable tools should be investigated.

References

[1] A. Yu. Kitaev. *Quantum measurements and the Abelian Stabilizer Problem*. 1995. arXiv: quant-ph/9511026 [quant-ph]. URL: https://arxiv.org/abs/quant-ph/9511026.

[2] Richard Cleve and Harry Buhrman. "Substituting quantum entanglement for communication". In: *Phys. Rev. A* 56 (2 Aug. 1997), pp. 1201–1204. DOI: 10.1103/PhysRevA.56.1201. URL: https://link.aps.org/doi/10.1103/PhysRevA.56.1201.

[3] Marcello Caleffi et al. *Distributed Quantum Computing: a Survey*. 2022. arXiv: 2212.10609 [quant-ph]. URL: https://arxiv.org/abs/2212.10609.

[4] Niels M. P. Neumann, Roy van Houte, and Thomas Attema. "Imperfect Distributed Quantum Phase Estimation". In: *Computational Science – ICCS 2020* 12142 (2020), pp. 605–615. URL: https://api.semanticscholar.org/CorpusID:219876111.

[5] Ali Javadi-Abhari et al. *Quantum computing with Qiskit*. 2024. DOI: 10.48550/arXiv.2405.08810. arXiv: 2405.08810 [quant-ph].

[6] Wim van Dam, Mariia Mykhailova, and Mathias Soeken. *Using Azure Quantum Resource Estimator for Assessing Performance of Fault Tolerant Quantum Computation*. 2024. arXiv: 2311.05801 [quant-ph]. URL: https://arxiv.org/abs/2311.05801.

[7] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. ISSN: 1095-7111. DOI: 10.1137/s0097539795293172. URL: http://dx.doi.org/10.1137/S0097539795293172.

[8] A. Yu Kitaev. "Quantum computations: algorithms and error correction". In: *Russian Mathematical Surveys* 52.6 (Dec. 1997), pp. 1191–1249. DOI: 10.1070/RM1997v052n06ABEH002155.

[9] Michael E. Beverland et al. *Assessing requirements to scale to practical quantum advantage*. 2022. arXiv: 2211.07629 [quant-ph]. URL: https://arxiv.org/abs/2211.07629.

[10] Nils Quetschlich et al. *Utilizing Resource Estimation for the Development of Quantum Computing Applications*. 2024. arXiv: 2402.12434 [quant-ph]. URL: https://arxiv.org/abs/2402.12434.

[11] David Aasen et al. "Milestones Toward Majorana-Based Quantum Computing". In: *Phys. Rev. X* 6 (3 Aug. 2016), p. 031016. DOI: 10.1103/PhysRevX.6.031016. URL: https://link.aps.org/doi/10.1103/PhysRevX.6.031016.

[12] Ramón Aguado and Leo P. Kouwenhoven. "Majorana qubits for topological quantum computing". In: *Physics Today* 73.6 (June 2020), pp. 44–50. ISSN: 0031-9228. DOI: 10.1063/PT.3.4499. eprint: https://pubs.aip.org/physicstoday/article-pdf/73/6/44/10124104/44_1_online.pdf. URL: https://doi.org/10.1063/PT.3.4499.

[13] Charles H. Bennett et al. "Purification of Noisy Entanglement and Faithful Teleportation via Noisy Channels". In: *Physical Review Letters* 76.5 (Jan. 1996), pp. 722–725. ISSN: 1079-7114. DOI: 10.1103/physrevlett.76.722. URL: http://dx.doi.org/10.1103/PhysRevLett.76.722.