To what extent will quantum computing systems replace high-performance computing (HPC) systems regarding computational biology?

Centre Name: NCLT New College Pontefract
Centre Number: 38185
Qualification Code: 7993

Henry Philip Groves - L0034335

February 2023

Contents

Introduction **3**
Fundamentals 3
Classical Computing 3
Quantum Computing 3
Computational Biology 4

Current Technology **6**
High Performance Computing (HPC) 6
Benefits of HPC 6
Maturity 6
Scalability 7
Ease of Setup and Maintenance 7
Drawbacks of HPC 7

Quantum Approach **9**
Benefits of a Quantum Approach 9
Parallelisation 9
Quantum Algorithms 9
Defining Complexity 9
Shor's Algorithm 10
Differential Equation solvers 10
Fast Fourier Transform 11
Quantum Fast Fourier Transform 11
Drawbacks of a Quantum Approach 12
Quantum Effects 12
Cost 12
Cooling 13
Classical Processing 13

Potential Solutions **14**
Encode the entire system in superposition 14
Perform multiple computations in superposition 14
Quantum Subroutines 14

Evaluation **15**

Introduction

Quantum Computing is a methodology of computation that harnesses quantum mechanics. The concept of simulating quantum-mechanical systems using a computational device has been around for over 40 years, [1] however, it is only recently that the technology has matured enough to show its potentially disruptive qualities. Their inherent parallel nature and exponential scaling with size means that it can solve problems outside the problem space of classical computers. The four main areas it has an advantage are: combinatorial optimisation, linear algebra, differential equations and factorisation. Differential equations are particularly useful in nearly all aspects of computer modelling. Specifically, I will be looking at their usage in the modelling of biological molecules.

Fundamentals

Classical Computing

Classical Computing, or Binary Computing, is what underpins all of our current computer systems. Information in these systems are represented in units called bits [2]. Physically, these units are represented by transistors, which are small electrical switches, that can store a charge. When the transistor is "full" of charge, that represents that bit being "on", or in the Binary number system, 1. When the transistor is "empty", that represents the bit being off, or a 0. These switches can be combined with other components to create logic gates [3], and then our modern computer systems are built from these basic gates.

Quantum Computing

Similar to Classical Systems, Quantum Computers are based on "quantum bits", or Qubits. However, unlike classical bits, qubits are two-dimensional vectors. Graphically, this can be represented using a Bloch sphere. See Figure 1.

$$0 \rightarrow |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$1 \rightarrow |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Mapping of Classical Bits to Dirac notation [4] and column vectors

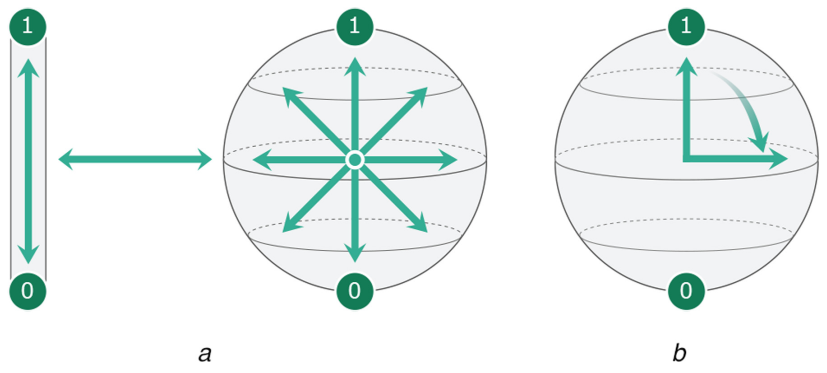

Figure 1: a. Graphical Representation of a Qubit b. Hadamard Gate [5]

The qubits $|0\rangle$ and $|1\rangle$ are referred to as $\{|0\rangle, |1\rangle\}$, the computational basis. This is the two basis states composed by any of the two distinct quantum states that the selected implementation of qubit can physically be in. For example, if we were using the spin [6] of a proton, where spin-up is our $|0\rangle$ state, and spin-down is our $|1\rangle$ state, then the computational basis of this qubit would be $\{|\uparrow\rangle, |\downarrow\rangle\}$.

Where qubits diverge from their classical analog is that qubit states are not discrete, meaning that they are not 0 OR 1. Instead, they exist in a superposition [7] of both $|0\rangle$ and $|1\rangle$ states. When the value of the qubit is measured, it "collapses" to either $|0\rangle$ or $|1\rangle$. Whilst it cannot be predicted which state the qubit will collapse, the probability of whether a qubit will collapse to a particular state can be determined.

Analogous to classical computing, quantum computers also have quantum gates, which transform the input qubit, and large sequences of these gates can be used to perform more complex operations. More specifically, a quantum gate specifies how the computation basis (for example, $\{|0\rangle, |1\rangle\}$) is transformed by the operation of a quantum gate. The most common quantum gates are the `NOT(X)`, Hadamard gate and `CNOT(CX)`. The `NOT` gate transforms $|0\rangle$ to $|1\rangle$, and vice versa. The Hadamard gate creates an equal superposition state, therefore meaning that the input qubit has an equal chance to collapse to $|0\rangle$ or $|1\rangle$. Graphically, we can represent this by a rotation of π^c in the Bloch Sphere [8] (Figure 1).

Computational Biology

Computational Biology is the use of mathematical modelling and simulations to understand biological molecules and interactions between those molecules, and aims to analyse and visualise the complex connections in biological processes.

One major usage of this is in protein 'folding', which is the prediction of the structure of a protein from its ammino acid sequence. This was particularly useful when modelling the viral proteins of SARS-CoV-2, due to its rapid mutagenesis, and allowed for mutations of the virus to be tested against vaccines and treatments without having the specific variant in a lab [9, 10].

This is just one example of what we can use computational biology for; there are potentially limitless applications, such as simulating new fertilizers and personalising medicines. If we can successfully simulate biological molecules interacting given a set of starting conditions, in a timeframe quicker than it would take to observe the actual system, then the cost and time taken to develop new pharmaceuticals could be dramatically decreased. Drug molecules could be designed to interface perfectly with the intended recipient's body to ensure maximum efficacy. It would remove the need to prototype tens if not hundreds of molecules, as any error in the mdlecule would only require the simulation being ran again, rather than having to wait for the physical system to interact again.

The potential benefits of having the capabilities to simulate any system with any number of variables are therefore very obvious.

Current Technology

High Performance Computing (HPC)

High Performance Computing (HPC) marks the boundary of our current computing capabilities, and it pushes the limits of what we can achieve with classical computing. It regards the implementation of algorithms and the the hardware that they are then ran on [11]. The most common form of HPC is the super-computer, containing thousands of compute nodes that work together to complete a task or algorithm. Modern super-computers are composed of hundreds of thousands of CPU cores and millions of GPU cores. At time of writing, *Frontier* is the fastest supercomputer, performing 1.102 ExaFLOPs [12] (1.102×10^{18} floating point operations per second)[1], and has 591, 872 CPU cores and 7,398, 400 GPU cores.

Benefits of HPC

Maturity

Supercomputing systems have been around since the 1960s, with the *Livermore Atomic Research Computer (LARC)*, built in 1960, being considered the oldest [13]. Since the invention of the transistor, supercomputing technology has only increased exponentially in speed.

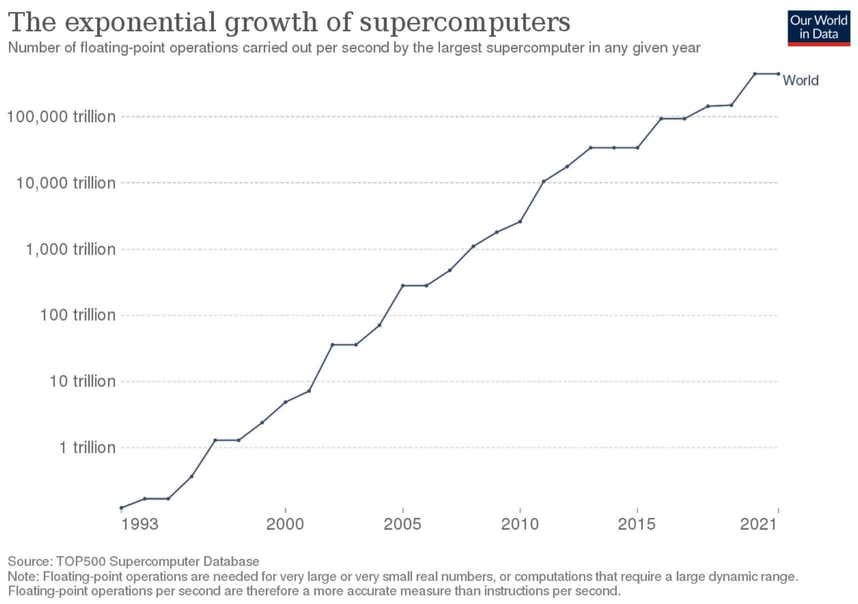

Figure 2: Visualisation of supercomputer growth [14]

This 60 year period of innovation and improvement in super-computing architectures, and processor technology has caused this technology to mature very quickly and is now used across the globe for a huge

[1] That is, 1.1 quintillion

number of tasks each day. Stock markets, bank transfers, digital currencies, weather prediction and modelling, climate research are all run by HPC systems.

Scalability

Common supercomputing architectures are now based around huge arrays of smaller computer systems, each using mostly off-the-shelf components [15]. This enables new supercomputing arrays to be set up very quickly and effectively, and at a considerably lower cost than a fully custom array would (though those systems do still exist) . It also makes them much more easier to upgrade and scale horizontally. Therefore, if a specific biological system requires more processing then it would be easy to simply add more nodes to the array and increase the capabilities of the system.

Ease of Setup and Maintenance

The environmental requirements of classical computers are not great. The major limitation (and cost) of HPC systems is the vast amounts of heat generated. The faster they are running, the more heat generated, requiring large amounts of cooling, either via HVAC or liquid cooling. Other than that, supercomputers can be set up anywhere there is enough space to fit it, and power for it to use.

Additionally, due to the modular nature of modern parallel systems, if a node fails, it can be easily hot-swapped out with a replacement.

Drawbacks of HPC

A lot of current models for modelling biological systems are designed to model inert particles are not applicable to chemically inhomogeneous systems. The complexity of both the biological molecules and their interactions, due to the entropy of such systems. Calculating the energy levels and transfers between particles requires very detailed calculations, the number of which required scales exponentially as the number of molecules being simulated increases to fully model the system [16].

Current Molecular Dynamics (MD) simulations can only simulate very small particles for very small periods of time. Using 32 CPUs to simulate a small protein, it is possible to simulate 100ns of the system in 24 hours of running the program. Therefore, to simulate that same protein with 162 ammino acids - approximately 3×10^4 atoms - for $10\mu s$ would take 3200 CPU days [17].

Larger Molecules, for example, a ribosome, with $\approx 2.6 \times 10^6$ atoms, being simulated for 20ns on 768 CPUs took 10^6 CPU hours in 2006 [18]. Ribosomes produce one ammino acid every 100ms, and so to simulate the production with the same processing power, would take 10^6 times longer, leading to almost 1.5 million years.

This is obviously impractical, and it is much more faster to just observe the actual system, partially due to the small time scales that biological molecules operate on and partially due to the complexity of the mathematics that underpins it.

Classical Processors are also, at their very core, linear. They can only process one instruction on one piece of data at a time. Whist running processors simultaneously can let one achieve parallel processing, there is a large amount of overhead required to coordinate each processor, which only increases as the number of processors increases.

Quantum Approach

Benefits of a Quantum Approach

Parallelisation

A lot of the speed-ups from quantum computing relies on quantum parallelism, which arises from the ability for a quantum register (simply a place where qubits are stored) to exist in the superposition of base states. Applying a function/gate is performed on each of the components of the superposition, but due to the register being in a superposition, the function is only applied one time. The number of possible states is 2^n, with n being the number of qubits in the register. Therefore, you can perform one operation on a quantum computer that would take an exponential number of operations on a classical computer.

For example, take the function (1), applied over a quantum register in an equally weighted superposition of the states 0 through 6, represented by $Q : \{0, 1, 2, 3, 4, 5, 6\}$:

$$f(x) = x^2 + 1 \tag{1}$$

$$\implies f(Q) = \{1, 2, 5, 10, 26, 37\} \tag{2}$$

Although we have only performed the function once, every single component of the superposition has had the function applied to it. To perform the same operation on a classical computer, on the same data would require iterating over each element and performing the function on each number, 0 through 6, taking n iterations to complete, n being the number of elements performing the operation on.

However, as we increase the number of superposition states, we decrease the change that, upon measuring the quantum register and hence collapsing the superposition of states, that the desired value is measured. However, quantum algorithms can be designed to use properties of the output as a whole, not just a particular one.

This parallelism therefore can be utilised to speed up many problems and algorithms that, whilst possible on a classical system, would take a very large amount of time as the size of the problem increases.

Quantum Algorithms

Defining Complexity

Algorithms and problems can be sorted into different complexities, which help one understand what problems can be efficiently solved, and the time and space their solutions take up. In Figure 3, we can see the relation between each class. P and NL are the two classes where problems can be solved in polynomial time, and a general rule of thumb is that any solution in these two classes can be efficiently solved and are tractable.

Quantum Algorithms, when properly designed, can exist outside of this P-complexity space, referred to as BQP (Bounded Quantum Polynomial). This contains any solution to a problem that can be solved in polynomial time on a quantum computer, but not a classical computer. [19]

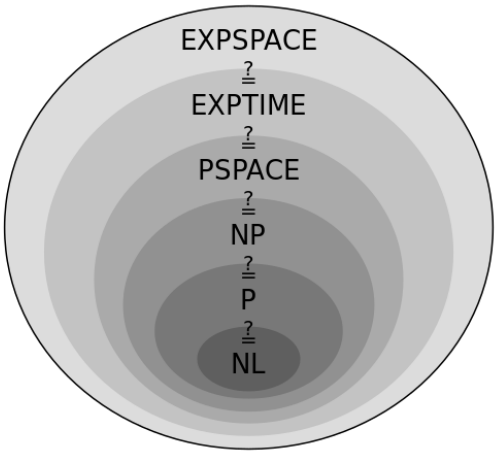

Figure 3: Complexity Theory Classes [20]

For example, integer factorisation. This is relied upon for many cryptography systems, such as RSA public-key encryption. RSA keys are composed of two semiprimes, numbers that are the product of two prime numbers. Whilst it has not yet been proven that the factorisation of prime integers is not a P-complexity space problem, the best publicly available algorithm currently has a time complexity of

$$O(\exp((\sqrt[3]{\frac{64}{9}}+o(1))(\ln n)^{\frac{1}{3}}(\ln\ln n)^{\frac{2}{3}})) \tag{3}$$

for n number of bits. Whilst this is sub exponential, and therefore not in EXPTIME-complexity space, it does significantly exceed polynomial time. [21]

Shor's Algorithm

There is however, an algorithm devised by Peter Shor, which, using a quantum computer, can factorise large prime integers in polynomial time [22].

$$O((\log n)^2(\log\log n)) \tag{4}$$

This algorithm was first implemented in 2001, using a 7 qubit NMR quantum computer [23]. This is just one example of a problem that can be significantly sped up by designing quantum algorithms. If this algorithm can be scaled up, it provides a significant threat to cryptography.

Differential Equation solvers

Many, if not all mathematical and scientific models are underpinned by differential equations. They are used to describe how functions interact with their derivative and change in relation to other variables. Some common differential equations include:

$$\frac{d^2u}{du^2} + \omega^2 u = 0 \qquad \text{Describing a harmonic oscillator} \tag{5}$$

$$L\frac{d^2}{du^2} + g\sin u = 0 \qquad \text{Describing the motion of a pendulum, length L} \tag{6}$$

$$\nabla^2 f(x,y,z) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0 \qquad \text{3-dimensional Laplace equation [24]} \tag{7}$$

The speed with which we can either, find a solution to a differential equation (that is, find functions that represent each of the variables involved), or find a numeric solution to the differential equation, therefore plays a large role in the speed that we can model biological systems.

For both linear (the system only contains linear equations of u and it's derivatives), and non-linear (system contains non-linear functions of u and it's derivatives) differential equations, there exists quantum algorithms for solving both. Classical algorithms for solving differential equations typically grow exponentially with the number of variables involved. However, using quantum algorithms this can be reduced to poly-logarithmic time or better [25, 26], and for linear differential equations, even to quadratic time ($O(n^2)$) [27].

Fast Fourier Transform

The Fourier Transform is a mathematical transform that decomposes a function into it's frequency components. For example, splitting a musical waveform into the intensity of the individual pitches and harmonics. However, it has many uses, including solving partial differential equations.

The Fast Fourier Transform (FFT) was (re)discovered by James Cooley and John Turkey in 1965, thought the be an original discovery at the time, but it was later found that Carl Gauss had already devised it over 160 years earlier [28]. This algorithm was pivotal in a giant number of fields, reducing the time complexity to perform a Fourier transform from $O(n^2)$ to $O(n \log n)$. In Figure 4 it is clear that the FFT scales much more slowly than traditional methods, as the dimensions of the input matrix increases.

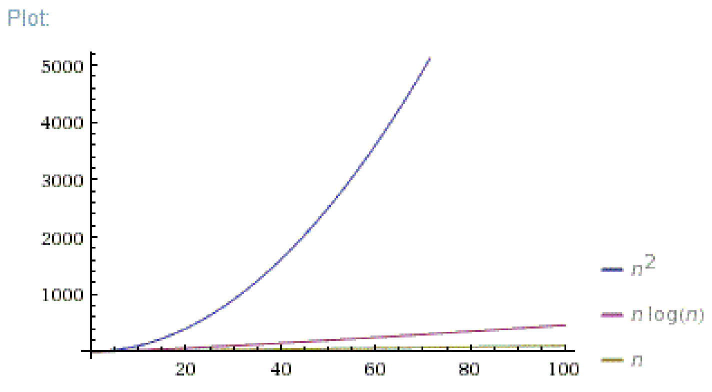

Figure 4: $O(n^2)$ vs $O(n \log n)$ vs $O(n)$

Quantum Fast Fourier Transform

A quantum algorithm also exists to perform an FFT on a quantum system, and when implemented properly can scale at $O(n)$ [29, 16]. See Figure 4

The exponential speed up - along with the quantum parallelism discussed earlier - provided by quantum algorithms can vastly increase the speed and efficiency that we can simulate biological systems with, especially as the number of variables and particles simulated grows.

Drawbacks of a Quantum Approach

Whilst this paints a rather favourable picture for quantum computing matching and superseding HPC in the field of computational biology, there exists a number of significant drawbacks.

Quantum Effects

Whilst quantum computing as a concept has been around for over 4 decades, the actual capabilities to construct and maintain quantum computers have only come to fruition recently, with huge corporations and small start-ups producing systems using various designs. However, regardless of the design philosophy used in a quantum computer, it is still very difficult to maintain the entanglement of many qubits at once, and this only becomes more difficult as the number of qubits increases. Entanglement of qubits is critical to many quantum gates functioning. The greater the number of entanglement qubits, the greater the number of states that computer can represent and be in a superposition of.

Additionally, qubits need to be entangled with each other, but not entangled with anything else; They must be designed very carefully not to entangle with any other physical system and therefore become decoherent.

Qubits must also be shielded from any kind of interferences, such as radiation or other particles. Some decoherence and noise is inevitable in any quantum system, however our techniques to manage these factors are relatively immature and limit the ability to scale quantum systems. For example, we can create a ”logical” qubit that is composed of many entangled qubits to represent one noise-free qubit. However, it is estimated that 100 to 1000 physical qubits are required to create one logical qubit, which brings us back to the issue of maintaining coherence between qubits.

Qubits are not just isolated units. Quantum computers need to be able to control and measure each qubit's state, requiring large amounts of circuitry, and this increases linearly as the number of qubits increases. If using a logical qubit system, this would exacerbate this issue even more.

Decoherence limits the amount of time we can run quantum computers for before the error rate becomes too high to take any meaningful reading. If we wanted to simulate a biological system that acts over a couple of days, then we would currently have significant challenge maintaining coherence during that timescale.

Cost

Whilst the construction of quantum computers is not cheap - Dwave's 2000 qubit quantum computer has a price tag of $15 million, it is the cooling and power requirements of quantum systems that really increases the cost.

Cooling

Qubits must be cooled to temperatures just above absolute zero in order to reduce the effect of heat radiation, and also to stabilise the qubit. In quantum systems that use superconducting magnets, cooling is also required for those magnets to superconduct. Cooling lasers are used to freeze qubits and held, whist being manipulated and measured using other qubits. The costs are due to the high energy requirements of these lasers, and cost of supercooled fluids like Helium.

Classical Processing

Whilst many quantum algorithms can provide exponential speed-ups over classical equivalents, many of these algorithms require additionally processing, either via another quantum algorithm or via processing on a classical system. This is due to qubits being superposed, and the need for individual state's probabilities needing to be measured. This post-processing can actually nullify the benefits one gains when running quantum algorithms, and can even make some algorithms scale worse than their classical counterparts.

Additionally, the scaling of a quantum algorithm does not show the whole picture. Whilst in theory a quantum algorithm may scale more efficiently that a classical equivalent, it does not mean that the specific implementation of a quantum computer can complete the computation quicker than a classical computer.

Potential Solutions

Encode the entire system in superposition

This would use quantum parallelism to mimic the way that current classical algorithms work. That single classical computation could be carried out using an exponentially smaller amount of memory, but in turn, only provide a very small portion of data. This is therefore only suitable for simulations where the desired result is some global average that can be extracted from the probability of each state in memory. It also would require the whole computation to maintain coherence for the entire computation, and therefore is not likely to be suitable for modelling large biological systems where intricate detail is required.

Perform multiple computations in superposition

The system would be encoded the same way a classical simulation would be, resulting in no saved memory, but the quantum superposition still allows one to perform several computations in parallel. A single simulation could processes different initial conditions, each encoded separately simultaneously. The most favourable outcomes could then be selected using destructive interference of less favourable outcomes. Multiple simultaneous computations could also be used to calculate the entropy of each molecule in the system, a variable that is often poorly estimated in classical models due to its high complexity, which would vastly increase the accuracy of simulations.

Quantum Subroutines

A hybrid approach could be used, in which computations that are inefficient on a HPC are ran on a quantum computer. This means that quantum coherence only has to be maintained for the length of the timestep of the simulation, and so may be feasible sooner. The running time of the simulation would be decreased by the time factor that each time-step is decreased by.

Evaluation

High Performance Computing is going nowhere anytime soon, and will continue being the backbone of our computational needs for the forseeable future. It is low cost, highly scalable and can be set up in most environments. To be able to utilise the abilities that computational biology would provide us in the near future, HPC is the most viable option, and as processor and GPU technology improves, so will the performance of HPC.

Whilst quantum computing can provide us with incredible benefits and speed-up over classical computing, the technology still is not mature enough to provide much usefulness. There are many problems, such as the issue of coherence and scalability, that will have to be solved before quantum computing will become competitive. Additionally, even if we do eventually develop large scale quantum systems, I do not believe that they will fully replace our HPC systems that are already deployed. Some classical algorithms just cannot be sped up using quantum computers and so the additionally costs and complexity that comes with running said algorithm on a quantum system is not worth it.

After all, quantum computers are a completely different type of computer, not just an evolution of our current technology. There are so many problems that can be solved on a quantum computer that are just inconceivable on a classical computer, and so it is not injust to look at it in an opposite way.

When modelling biological systems, there are no quick and easy solutions when simulating large, complex systems with many variables. However, quantum computers have very special capabilities and features that, when properly developed and mature, could make a significant difference to our prediction and modelling of biological systems. Most important of those features is the ability for a quantum system to process multiple versions of simulations simultaneously, that would allow the finding of the minimum free energy. Modern MD simulations now include repeat simulations, in which a number of initial confirmations are investigated in parallel to check the robustness of any conclusions against thermal noise [16]. The potentially huge parallelisation provided by quantum computers could take this to the extreme, modelling hundreds of different starting conditions simultaneously.

I therefore believe that Quantum Computers will only partially replace high performance classical computing in the field of computational biology. Whilst quantum computers can theoretically perform algorithms and functions critical to biological modelling, such as the Fast Fourier Transform and solving differential equations with a much greater efficiency than classical computers, there are many issues that prevent quantum computers fully replacing classical systems:

1. Maintaining Coherence - Quantum systems cannot maintain coherence for the long timescales that some modelling may require.
2. Cooling - To ensure that the amount of noise in the quantum system is as low as possible, quantum systems must be cooled to as low a temperature as possible. This requires large amounts of extra infrastructure and considerations when designing large scale quantum systems.

3. Nature of Superposition - Whilst some algorithms can exploit the features of superposed qubits, as the number of states contained within the superposition increases, it becomes more and more difficult to extract the desired value without having to do extra processing on top, which will scale with the number of superposed states in itself, potentially nullifying the benefits gained from quantum parallelism.

I think the most probable outcome is that we see hybrid quantum-classical supercomputers, where computations that can be made more efficient (that is, scale less as the number of inputs increase) by using a quantum approach are offloaded to quantum systems, where as computations that would provide no benefit, or even be slower on a quantum computer are kept on a classical machine. We have already seen this demonstrated by various companies that provide access to quantum computers over the cloud. Users interface with a classical computer to write the code to be executed on the quantum computer, and the classical system then measures the quantum state, providing an output to the user.

References

[1] R. P. Feynman, "Simulating physics with computers - International Journal of Theoretical Physics," *SpringerLink*, Jun 1982.

[2] Wikipedia, "Bit."

[3] Wikipedia, "Logic Gate."

[4] Wikipedia, "Dirac Notation."

[5] V. Hassija, V. Chamola, V. Saxena, V. Chanana, P. Parashari, S. Mumtaz, and M. Guizani, "Present landscape of quantum computing," *IET Quantum Communication*, vol. 1, no. 2, pp. 42–48, 2020.

[6] Wikipedia, "Spin."

[7] Wikipedia, "Quantum Superposition."

[8] Wikipedia, "Bloch Sphere."

[9] S. Bhowmick, T. Jing, W. Wang, E. Y. Zhang, F. Zhang, and Y. Yang, "In silico protein folding prediction of covid-19 mutations and variants," *Biomolecules*, vol. 12, no. 11, 2022.

[10] A. Kumar, P. K. Gupta, and A. Srivastava, "A review of modern technologies for tackling covid-19 pandemic," *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, vol. 14, no. 4, pp. 569–573, 2020.

[11] G. Hager and G. Wellein, *Introduction to high performance computing for scientists and engineers*, p. xvii. CRC Press, 2010.

[12] TOP500, "Frontier," 2022.

[13] E. G. Swedin, *Computers : the life story of a technology*, ch. 3, p. 57. Greenwood Press, 2005.

[14] Our World In Data, " https://ourworldindata.org/grapher/supercomputer-power-flops," 2021.

[15] National Academy of Sciences, A. R. Hoffman, J. F. Traub, Academy Industry Program, and National Research Council, *Supercomputers: Directions in Technology and Applications*, pp. 35–47. National Academies Press, 1989.

[16] S. A. Harris and V. M. Kendon, "Quantum-assisted biomolecular modelling," *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1924, pp. 3581–3592, 2010.

[17] P. L. Freddolino, F. Liu, M. Gruebele, and K. Schulten, "Ten-microsecond molecular dynamics simulation of a fast-folding ww domain," *Biophys*, vol. 94, pp. L75–77, 2008.

[18] K. Y. Sanbonmatsu and C.-S. Tung, "Large-scale simulations of the ribosome: a new landmark in computational biology," *Journal of Physics: Conference Series*, vol. 46, p. 334, sep 2006.

[19] E. Bernstein and U. Vazirani, “Quantum complexity theory,” *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1411–1473, 1997.

[20] Qef and Wikipedia, “Wikimedia Commons.”

[21] D. E. Knuth, *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*, p. 403. Addison-Wesley Longman Publishing Co., Inc., 1997.

[22] P. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, 1994.

[23] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, “Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance,” *Nature*, vol. 414, pp. 883–887, dec 2001.

[24] Wikipedia, “Laplace’s Equation.”

[25] S. Lloyd, G. De Palma, C. Gokler, B. Kiani, Z.-W. Liu, M. Marvian, F. Tennie, and T. Palmer, “Quantum algorithm for nonlinear differential equations,” 2020.

[26] S. K. Leyton and T. J. Osborne, “A quantum algorithm to solve nonlinear differential equations,” 2008.

[27] D. W. Berry, “High-order quantum algorithm for solving linear differential equations,” *Journal of Physics A: Mathematical and Theoretical*, vol. 47, p. 105301, feb 2014.

[28] J. W. Cooley and J. W. Tukey, “An algorithm for the machine calculation of complex fourier series,” *Mathematics of Computation*, Jan 1965.

[29] R. Asaka, K. Sakai, and R. Yahagi, “Quantum circuit for the fast fourier transform - quantum information processing,” *SpringerLink*, Aug 2020.