I. QUANTUM SEARCH

The problem of finding specific entries in unstructured data spaces is a ubiquitous one. This class of *search algorithms* have amongst the broadest applicability of any class of algorithms. Computer scientists have invested excruciating man-hours[1] into organising and structuring data so as to minimise the resource overhead (in time and/or space) associated with extracting desired components. However, the methodology for achieving this, and the favourability of associated resource overheads, is highly dependent on the structure of the underlying data, or whether there even is any. To this end, numerous data structures and algorithms have been developed, accommodating for every mutation and variation of the posed problem imaginable. Often, there is a tradeoff between the overheads induced in time and memory, as well as in pre-processing and data structure maintenance requirements.

For example, *hash tables* enable theoretical $O(1)$ lookup times on data with a *key-value pair* data structure. In a key-value pair each data entry (value) is tagged with a unique identifier (key) used for lookup. The value can observe any structure whatsoever, whereas the key is designed so as to minimise search times. When storing telephone numbers one might represent entries as key-value pairs, where the keys are people's names, and the values their respective telephone numbers. An efficient algorithm for mapping keys to physical memory addresses would imply efficient lookup of telephone numbers by name.

In the absence of a key-value representation one might simply store data in sorted form. However, this requires pre-sorting the entire data space, which may become costly for large data sets, and requires continual rearrangement whenever the data space is modified, making it computationally costly for mutable datasets.

For the end user, who wishes to find data elements, the worst-case data space is one with no order or underlying structure. Suppose we want to find whether a number exists in the telephone directory, but we don't know it's associated name. In this instance, it can easily be seen that the best one can hope for, in terms of algorithmic runtime, is to simply look through the data space brute-force until we find what we are looking for. It is clear that with an unstructured space of N elements, this brute-force search algorithm requires on average $O(N)$ queries to find the desired entry. We call this the *unstructured search problem*.

The brute-force classical algorithm, despite already being technically 'efficient' (i.e $O(N)$ linear runtime), could nonetheless become unwieldy for very large datasets. Google doesn't want to exhaustively scan their entire collection of data-centres each time they want to lookup a database element. The quantum search algorithm, first presented by Grover [**?**], provides a solution to this problem using only $O(\sqrt{N})$ runtime (oracle queries), a quadratic enhancement. Whilst this falls far short of the exponential quantum enhancement one might have hoped for, which has also shown to be optimal [2**?**], it is nonetheless still extremely helpful for many purposes, given the broad applications for this algorithm.

We will formulate the quantum search algorithm as an oracular algorithm, where the oracle takes as input an n-bit string, and outputs 1 if the input matches the entry we are looking for, otherwise 0. This formulation of the problem makes the algorithm naturally suited to solving satisfiability problems (many of which are **NP**-complete and of great practical interest).

The Grover quantum search algorithm is shown explicitly in Alg. I.

Since M denotes the number of targets in the search space, when $M = 1$ ($M > 1$) we refer to this as the single (multiple) target quantum search algorithm.

II. INTEGER FACTORISATION

Today, the security of the most widely used public-key cryptography scheme, RSA scheme, is based the difficulty of factoring large numbers for classical computer. Thus, the problem of efficient factoring numbers has attracted widespread attention in the field of computer and information Science. However, there is still no no classical algorithm that can factoring numbers in polynomial time [3]. In 1994, a major breakthrough was made by Peter Shor that quantum computers could efficiently factor numbers in polynomial time [4, 5], posing a serious threat to information security in business transactions on the Internet, such as e-commerce.

Suppose we want to factor N using Shor's algorithm, for a randomly chose number a ($0 < a < N$) that is co-prime to N, Shor's algorithm can output the minimum integer r that satisfies $a^r \mod N = 1$. From this period r, the prime factors of N are given by the greatest common divisor (GCD) of $a^{r/2} \pm 1$ and N, which can be solved classically. The Shor's algorithm can be broken into the following simple steps:

1. Creat two registers, register 1 and register 2, which have $n = 2\left\lceil \log_2^N \right\rceil$ qubits and $m = \left\lceil \log_2^N \right\rceil$, respectively. Initialize the quantum register 2 to $|0, ..., 0, 1\rangle$, and the quantum register 1 to

function Grover(f,n):

1. Using a Hadamard transform, prepare the n-qubit equal superposition of all 2^n logical basis states,

$$\begin{aligned} |\varphi\rangle &= \hat{H}^{\otimes n}|0\rangle^{\otimes n} \\ &= \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle, \end{aligned} \tag{1}$$

where x denotes a bit-string of length n.

2. The oracle is defined as a unitary black-box, which tags a target element T using a phase-flip,

$$\begin{aligned} \hat{U}_T|x\rangle &= (-1)^{f(x)}|x\rangle \\ &= (\hat{I} - 2|T\rangle\langle T|)|x\rangle, \end{aligned} \tag{2}$$

where $f(x) = \{0,1\}$ is the black-box function determining whether input x is the target element T ($f(x) = 1$) or not ($f(x) = 0$).

3. The Grover diffusion operator is defined to implement,

$$\hat{U}_s = \hat{I} - 2|T\rangle\langle T|. \tag{3}$$

4. repeat($O(N)$)):

5. $|\varphi_{i+1}\rangle = \hat{U}_s \cdot \hat{U}_T|\varphi_i\rangle$.

6. ???BOX

TABLE I. .

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$$

2. Apply the modular exponential function $f(x) = a^x \bmod N$ on register 2 when register 1 is in state $|x\rangle$, and then obtain the state

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle|a^x \bmod N\rangle$$

Due to the quantum parallelism, the quantum computer will calculate the function $a^x \bmod n$ for all $x \in \{0, ..., 2^n\}$ in parallel by only one step.

3. Apply quantum Fourier transformation (QFT) on the register 1, yielding

$$\frac{1}{2^n} \sum_{y=0}^{2^n-1} \sum_{x=0}^{2^n-1} e^{2\pi i xy/2^n} |y\rangle|a^x \bmod N\rangle$$

4. Measure the registers. Due to the QFT in step 3, register 1 will output $|y\rangle$ with high probability, where $y = c2^n/r$ (for integer c), which could enable to deduce the period r by some post processing on a classical computer. Then, the factor of n can be determined by taking $\gcd(x^{r/2} + 1, n)$ and $\gcd(x^{r/2} - 1, n)$.

Given an n-bit integer, the best rigorously proven upper bound on the classical complexity of factoring is $O(2^{n/4+o(1)})$ [3, 6]. However, the Shor's algorithm can solve this task in $O(n^3)$ time [4, 5], which achieves an exponential speedup over all classical algorithms. Furthermore, for some special case, such as factoring safe semiprimes (an important class of numbers used in cryptography), more feaster quantum algorithm is developed by changing the classical part of Shor's algorithm [7]. In the experimental side, some proof-of-principle experiments with a small number of qubits have already been achieved [8–16].

III. TOPOLOGICAL DATA ANALYSIS

Topological data analysis (TDA) [17] is a tool for extracting useful information from unstructured data by studying the shape of data. In particular, it can be used to estimate the certain topological features of data, such as the Betti numbers, which count the number of holes and voids of various dimensions in a scatterplot. In 2016, Lloyd, Garnerone, and Zanardi proposed a quantum algorithms for TDA, quantum TDA, which can provide an exponential speedup over the best currently known classical algorithms [18].

Generally, the k-th Betti number refers to the number of k-dimensional holes. Consider n data points, we will introduce the steps of the quantum TDA algorithm for calculating the k-th Betti number:

1. Simplicial complex construction. Implement the Grover's algorithm with a membership oracle function $\{f_k^\epsilon(s_k) = 1 \text{ if } s_k \in S_k^\epsilon\}$ to construct the simplicial complex state,

$$|\psi\rangle_k^\epsilon = \frac{1}{\sqrt{|S_k^\epsilon|}} \sum_{s_k \in S_k^\epsilon} |s_k\rangle.$$

where the k-simplex $|s_k\rangle$ is an n-qubit quantum state with $k+1$ 1s at positions $j_0, j_1, ..., j_k$ and 0s at the other remaining positions, which means a connected graph with points $j_0, j_1, ..., j_k$, and the Vietoris-Rips simplicial complex S_k^ϵ is the set of k-simplices where all points are within distance ϵ of each other.

2. Mixed state construction. Add an ancillary register consisted of n qubits and then perform controlled-NOT (CNOT) gates to copy $|\psi\rangle_k^\epsilon$ to construct $\frac{1}{\sqrt{|S_k^\epsilon|}} \sum_{s_k \in S_k^\epsilon} |s_k\rangle \otimes |s_k\rangle$, finally trace out the ancillary register to obtain mixed state ρ_k^ϵ.

$$\rho_k^\epsilon = \frac{1}{|S_k^\epsilon|} \sum_{s_k \in S_k^\epsilon} |s_k\rangle\langle s_k|.$$

3. Topological analysis. Define the boundary map ∂_k^ϵ as $\partial_k^\epsilon |s_k\rangle = \sum_l (-1)^l |s_{k-1}(l)\rangle$, where $|s_{k-1}(l)\rangle$ is obtained from s_k with vertices $j_0 ... j_l ... j_k$ by omitting the l-th point j_l from s_k. Then, transform the boundary map to a Hermitian matrix

$$B_k^\epsilon = \begin{pmatrix} 0 & \partial_k^\epsilon \\ \partial_k^{\epsilon\,\dagger} & 0 \end{pmatrix}.$$

Apply the phase-estimation algorithm to decompose ρ_k^ϵ in terms of the eigenvectors and eigenvalues of B_k^ϵ, and then measure the eigenvalue register. Employing the probability of measuring zero η_k^ϵ, the dimension of the kernel of ∂_k^ϵ could be calculated as $\dim(\mathrm{Ker}\,\partial_k^\epsilon) = \eta_k^\epsilon \cdot |S_k^\epsilon|$. Using the similar approach for calculating $\dim(\mathrm{Ker}\,\partial_{k+1}^\epsilon)$, then we can reconstruct the k-th Betti number by,

$\beta_k^\epsilon = \dim(\mathrm{Ker}\,\partial_k^\epsilon) - \dim(\mathrm{Im}\,\partial_{k+1}^\epsilon) = \dim(\mathrm{Ker}\,\partial_k^\epsilon) + \dim(\mathrm{Ker}\,\partial_{k+1}^\epsilon) - |S_{k+1}^\epsilon|$.

Above is the basic idea of calculating the k-th Betti number. In practical application, we can define the full Hermitian boundary map, Dirac operator, to be $B^\epsilon = B_1^\epsilon \oplus B_2^\epsilon \oplus ... \oplus B_n^\epsilon$, and then use the Dirac operator to estimate Betti numbers to all orders (see [18] for details).

In theory, the quantum TDA algorithm could estimate approximate values of Betti numbers to all orders and to accuracy δ in time $O(n^5/\delta)$ [18], by contrast, the best classical algorithms for estimating Betti numbers to all orders to accuracty δ takes time at least $O(2^n log(1/\delta))$ [19–21].

---

[1] Presently, most computer science research institutions are equal opportunity employers.
[2] C. Zalka, Physical Review A **60**, 2746 (1999).
[3] J. M. Pollard, in *Mathematical Proceedings of the Cambridge Philosophical Society*, Vol. 76 (Cambridge University Press, 1974) pp. 521–528.
[4] P. W. Shor, SIAM Journal on Computing **26**, 1484 (1997).
[5] P. W. Shor, in *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on* (Ieee, 1994) pp. 124–134.
[6] V. Strassen, Jahresbericht der Deutschen Mathematiker-Vereinigung **78**, 1 (1976).
[7] F. Grosshans, T. Lawson, F. Morain, and B. Smith, arXiv preprint arXiv:1511.04385 (2015).

[8] T. Monz, D. Nigg, E. A. Martinez, M. F. Brandl, P. Schindler, R. Rines, S. X. Wang, I. L. Chuang, and R. Blatt, Science **351**, 1068 (2016).
[9] E. Lucero, R. Barends, Y. Chen, J. Kelly, M. Mariantoni, A. Megrant, P. O'malley, D. Sank, A. Vainsencher, J. Wenner, *et al.*, Nature Physics **8**, 719 (2012).
[10] E. Martin-Lopez, A. Laing, T. Lawson, R. Alvarez, X.-Q. Zhou, and J. L. O'brien, Nature Photonics **6**, 773 (2012).
[11] A. Politi, J. C. Matthews, and J. L. O'brien, Science **325**, 1221 (2009).
[12] C.-Y. Lu, D. E. Browne, T. Yang, and J.-W. Pan, Physical Review Letters **99**, 250504 (2007).
[13] B. Lanyon, T. Weinhold, N. K. Langford, M. Barbieri, D. James, A. Gilchrist, and A. White, Physical Review Letters **99**, 250505 (2007).
[14] L. M. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, Nature **414**, 883 (2001).
[15] N. Johansson and J.-Å. Larsson, arXiv preprint arXiv:1706.03215 (2017).
[16] H.-L. Huang, Q. Zhao, X. Ma, C. Liu, Z.-E. Su, X.-L. Wang, L. Li, N.-L. Liu, B. C. Sanders, C.-Y. Lu, *et al.*, Physical Review Letters **119**, 050503 (2017).
[17] G. Carlsson, Bulletin of the American Mathematical Society **46**, 255 (2009).
[18] S. Lloyd, S. Garnerone, and P. Zanardi, Nature communications **7**, 10138 (2016).
[19] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer, Discrete & Computational Geometry **37**, 103 (2007).
[20] S. Basu, Discret. Comput. Geom. **22**, 1 (1999); **30**, 65 (2003); Found. Comput. Math. **8**, 45 (2008); arXiv:1409.1534.
[21] J. Friedman, Algorithmica **21**, 331 (1998).