

# Efficient quantum walks over exponentially large sets of combinatorial objects for optimisation

S. Marsh\* and J. B. Wang†

Department of Physics, The University of Western Australia, Perth, Australia

(Dated: December 17, 2019)

We present a highly efficient quantum circuit for performing continuous time quantum walks (CTQWs) over an exponentially large set of combinatorial objects, provided that they can be ranked and unranked efficiently. CTQWs form the core mixing operation of a generalised version of the Quantum Approximate Optimisation Algorithm, which works by ‘steering’ the quantum amplitude into high-quality solutions. The efficient quantum circuit holds the promise of finding high-quality solutions to certain classes of NP-hard combinatorial problems such as the Travelling Salesman Problem, maximum set splitting, graph partitioning and lattice path optimisation.

## I. INTRODUCTION

Combinatorial optimisation problems are known to be notoriously difficult to solve, even approximately in general [1]. Quantum algorithms are able to solve these problems more efficiently, with a brute force quantum search offering a guaranteed polynomial speedup over the classical approach [2, 3]. Such speed-up is unfortunately insufficient to provide practically useful solutions, since these combinatorial optimisation problems scale up exponentially.

Farhi *et al.* [4] proposed the Quantum Approximate Optimisation Algorithm (QAOA), derived from approximating the quantum adiabatic algorithm on a gate model quantum computer, to find high quality solutions for general combinatorial optimisation problems [4]. More recently, we extended the QAOA algorithm to solve constrained combinatorial optimisation problems via alternating continuous-time quantum walks over efficiently identifiable feasible solutions and solution-quality-dependent phase shifts [5]. Throughout this paper, we refer to this quantum-walk-assisted generalisation as QWAO.

The core component of QWAO is the continuous time quantum walk (CTQW) [6–9], which acts as a ‘mixing’ operator for the algorithm. A CTQW is defined by the propagator  $U(t) = e^{-itC}$ , where  $C$  is the adjacency matrix of a given graph connecting all feasible solutions of the problem under study. Quantum walks have markedly different behaviour to classical random walks due to quantum effects [10–13], and they have played a central role in quantum simulation and quantum information processing [14–25].

In this paper, we discuss a significant and innovative application of QWAO to a wide range of combinatorial domains, which is defined as the set of feasible solutions to some specified combinatorial optimisation problem. In Section II we describe the QWAO procedure and its

quantum circuit implementation. In Section III, we detail our method for quantum walking over a variety of combinatorial domains. Specifically, the domains applicable to this method are those with an associated *ranking function*, which efficiently identifies each object with a unique integer rank. We show that the domain(s) of objects can be connected in any circulant way, barring some minor restrictions. We also show how to design a unitary that performs the ranking on computational basis states representing objects. In doing this, the adjacency matrix is mapped to diagonal block matrix with circulant blocks. We then show how to perform a CTQW over graphs with this form. In Section IV, we give a number of applicable combinatorial domains along with their associated NP optimisation problems, including well-known problems such as the Travelling Salesman Problem and constrained portfolio optimisation. Finally, we describe the relevance of the ranking unitary to quantum search in Section V and then make some concluding remarks.

## II. QUANTUM WALK-ASSISTED APPROXIMATE OPTIMISATION

QWAO uses the continuous time quantum walk as an ansatz, distinguished from the original derivation of the QAOA as a discretised adiabatic evolution. Specifically, in the QWAO framework the quantum system evolves as [5]

$$|\vec{\gamma}, \vec{t}\rangle = U_W(t_p)U_Q(\gamma_p)\dots U_W(t_1)U_Q(\gamma_1)|s\rangle. \quad (1)$$

In a combinatorial optimisation context, the QWAO procedure can be interpreted as follows:

1.  $|s\rangle$  is the initial state, which can be taken to be an equal superposition over all of the feasible combinatorial solutions.
2.  $U_Q(\gamma)$  applies a phase shift to each combinatorial object, proportional to  $\gamma$  and its *quality* with respect to the optimisation problem.
3.  $U_W(t)$  performs a continuous-time quantum *walk* for time  $t$  over the combinatorial domain.

\* samuel.marsh@research.uwa.edu.au

† jingbo.wang@uwa.edu.au

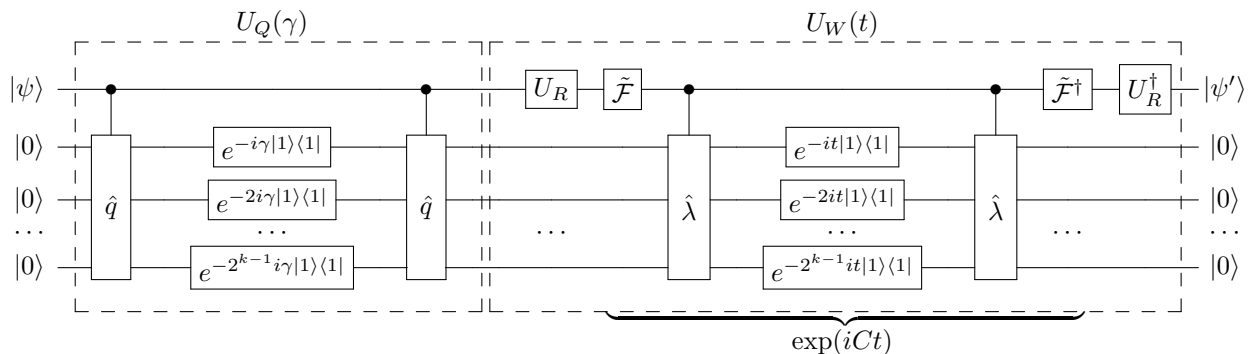


FIG. 1: An illustrative component of a QWAO circuit for optimisation over combinatorial domains having efficient ranking and unranking functions. The circuit performs  $|\psi'\rangle = U_W(t)U_Q(\gamma)|\psi\rangle$ . Here the operator  $\hat{q}$  evaluates the quality of a solution to  $k$  bits of precision, and  $\hat{\lambda}$  computes the eigenvalue of the circulant matrix to the same precision. The rotation angles  $\gamma$  and  $t$  are varied by a classical optimiser.

4. The set of  $2p$  parameters  $\vec{\gamma} = (\gamma_1, \dots, \gamma_p)$  and  $\vec{t} = (t_1, \dots, t_p)$  are varied by a classical optimiser in order to maximise the average measured solution quality. A higher choice of  $p$  leads to better solutions at the cost of a longer circuit.

The QWAO state evolution consists of an interleaved series of phase shifts, which introduce a bias to solutions dependent on their quality, and quantum walks to mix amplitude between solutions. A classical optimiser is used to vary the parameters  $\vec{\gamma}$  and  $\vec{t}$  such that the expectation value of the solution quality is maximised. A circuit diagram for a component of the QWAO is illustrated in Fig. 1, consisting of one solution quality dependent phase shift  $U_Q(\gamma)$  followed by a quantum walk over the valid solutions  $U_W(t)$ .

The  $U_Q(\gamma)$  component of the QWAO circuit is straightforward to implement for any combinatorial optimisation problem in the NPO complexity class. Put another way, given a combinatorial object  $x$ , as long as its solution quality  $q(x)$  can be efficiently computed. If so, there is an efficient quantum circuit that implements the desired solution quality-dependent phase shift [26]. We show this implementation in the left box of Fig. 1. With this in mind, the remainder of this paper proposes a generic approach to implementing  $U_W(t)$  over a wide range of combinatorial domains.

### III. QUANTUM WALKS OVER COMBINATORIAL DOMAINS

Consider a set of  $M$  combinatorial objects, each encoded as a binary string. An example is  $k$ -combinations of a set  $S = \{s_1, s_2, \dots, s_n\}$ , where each combinatorial object corresponds to a unique  $k$ -combination of  $S$ . A straightforward encoding of a specific  $k$ -combination as a length- $n$  binary string is  $x = x_1x_2 \dots x_n$ , where  $x_j = 1$  iff  $s_j$  is selected as part of the combination. In this case, the aim is to perform a quantum walk over a graph con-

necting each of the  $M = \binom{n}{k}$  possible  $k$ -combinations of  $S$ . Current approaches resort to sparse Hamiltonian simulation of the XY model Hamiltonian [27, 28], which is not ideal as the quantum circuit changes with the value of the walk time  $t$  (requiring a longer circuit to approximate the quantum walk dynamics for a longer time), and furthermore the XY model does not maintain symmetry amongst the relevant vertices, introducing bias into the optimisation algorithm.

The core of our approach is to unitarily map the  $M$  binary strings corresponding to combinatorial objects, which are ‘scattered’ in a larger space of  $2^n$  binary strings (i.e. 0 through to  $M - 1$ ). This is done using a so-called ‘ranking unitary’  $U_R$ . We then describe an approach for performing a continuous time quantum walk over the first  $M$  computational basis states connected as a circulant graph, where  $M$  is not necessarily a power of 2.

We choose to focus on circulant connectivity because all circulant graphs are diagonalised by the Fourier transform. Thus, the highly efficient Quantum Fourier Transform (QFT) can be used to diagonalise any choice of circulant graph, whilst the circuit structure remains largely unchanged [29–32]. After ‘unranking’ using  $U_R^\dagger$ , a CTQW over the circulant connected combinatorial objects has been performed. Our approach is summarised with

$$U_W(t) = U_R^\dagger \exp(iCt) U_R, \quad (2)$$

where  $U_R$  is the ranking unitary and  $C$  is a block-diagonal adjacency matrix with circulant blocks, which defines the connectivity of the combinatorial objects. In the following subsections we describe each component of the overall quantum walk, and the corresponding circuit implementation.

### A. Ranking with $U_R$

Any finite set of combinatorial objects with an associated ranking and unranking function can be assigned to vertices of a graph and connected in such a way that the adjacency matrix is diagonal block-circulant. Given a finite set of combinatorial objects  $S$  with cardinality  $M$ , a ranking function  $r : S \rightarrow [M]$  is a bijection that efficiently assigns a unique integer ‘rank’ to a given object. The unranking function is the inverse map  $u : [M] \rightarrow S$ .

Let  $S = \{s_1, s_2, \dots, s_M\}$  be a set of integers. We wish to perform a continuous time quantum walk (CTQW) over this set. Assume we have a bijective function  $f : S \rightarrow [M]$  that identifies each element of  $S$  with its unique integer ‘id’, where both  $f$  and  $f^{-1}$  are efficiently computable. In essence, the function  $f$  is an ordering of  $S$ . We require  $n = \lceil \log M \rceil$  qubits to encode the elements in a quantum register. Then

$$f_N(x) = \begin{cases} f(x) & x \in S \\ f^{-1}(x) & x \notin S \wedge x < M \\ x & \text{otherwise} \end{cases} \quad (3)$$

extends  $f$  to be a bijection  $[N] \mapsto [N]$ , thus embedding  $f$  in a larger space with  $N = 2^n$ .

There is a certain degree of flexibility in connecting up solutions. Most ranking functions can be expressed as ranking a particular object amongst all objects of the same size. For example, one could define a ranking function to walk exclusively over  $k$ -partitions. Alternatively, the ranking function could be extended to rank amongst all objects of any size. This leads to quantum walking over, for example, any partition with size  $\leq k$ . It is also straightforward to design alternative ranking functions that encapsulate the same-size ranking, for example to connect solutions with size  $a \leq k \leq b$ .

We now detail how to construct the ranking unitary  $U_R$ , given classical algorithms for ranking and unranking. Since we assume we have an efficient classical algorithm for ranking, we can construct a reversible circuit to perform  $U_r |x\rangle |z\rangle = |x\rangle |z \oplus y\rangle$ , where  $y = r(x)$ . Similarly, we can construct a unitary for unranking,  $U_u |y\rangle |z\rangle = |y\rangle |z \oplus x\rangle$ . It is straightforward to check that

$$U_R |x\rangle |0\rangle \equiv U_u \mathcal{S} U_r |x\rangle |0\rangle = |r(x)\rangle |0\rangle, \quad (4)$$

where  $\mathcal{S}$  swaps the first and second registers. We give the circuit diagram in Fig. 2. Clearly, unranking is performed with  $U_R^\dagger$ .

### B. Quantum walk with $\exp(iCt)$

We now connect the ranked objects in a circulant manner, defining a circulant graph  $C_R$  of size  $M \times M$ . A circulant matrix of size  $M \times M$  is diagonalised by the Fourier matrix  $\mathcal{F}_M$ .

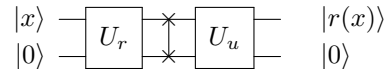


FIG. 2: Circuit for  $U_R$ , using an ancilla register of the same size as the input register.

When padded to a binary power to match the Hilbert space of the quantum register, the matrix  $C$  will have the form

$$C = C_R \oplus \mathbb{I}. \quad (5)$$

Since  $C$  is block diagonal, it is diagonalised by

$$\tilde{\mathcal{F}} = \mathcal{F}_M \oplus \mathbb{I} \quad (6)$$

which is unitary since it is a direct sum of unitaries. Hence,  $C = \tilde{\mathcal{F}} D \tilde{\mathcal{F}}^\dagger$  where  $D$  is a diagonal matrix consisting of the eigenvalues of  $C_R$ , afterwards padded with 1s.

Therefore the quantum walk over the graph with adjacency matrix  $C$  is

$$\exp(iCt) = \tilde{\mathcal{F}} \exp(iDt) \tilde{\mathcal{F}}^\dagger. \quad (7)$$

Since  $D$  is diagonal, it is straightforward to implement  $\exp(iDt)$  using techniques from [26] assuming that the eigenvalues of each  $C_i$  are efficiently computable. This is the case for any circulant graph with at most polynomial degree such as cycle graphs, as well as those with a closed-form expression for the eigenvalues such as the complete graph. Hence, here we focus on the implementation of  $\tilde{\mathcal{F}}$ .

In the extended state space, we want to perform the operation

$$\tilde{\mathcal{F}} |x\rangle = \begin{cases} \mathcal{F}_M |x\rangle & x < M, \\ |x\rangle & x \geq M. \end{cases} \quad (8)$$

The following method, based on the well-known result from [33], implements Eq. (8). Consider the action on a computational basis state  $|x\rangle |0\rangle |0\rangle$ , where the first ancilla register is the same size as the  $|x\rangle$  register, and the second is a single qubit.

1. Flip the ancilla qubit conditional on the first register holding a value less than  $M$ .
2. Create an equal superposition up to  $M$  in the second register,  $\frac{1}{\sqrt{M}} |x\rangle \sum_{y=0}^{M-1} |y\rangle$ , controlled by the ancilla qubit being  $|1\rangle$ .
3. Perform controlled phase shift,  $\frac{1}{\sqrt{M}} |x\rangle \sum_{y=0}^{M-1} e^{2\pi i xy/M} |y\rangle = |x\rangle \mathcal{F}_M |x\rangle$ .
4. Controlled on the ancilla qubit being  $|0\rangle$ , swap the first two registers.

- To uncompute the first register, perform inverse Quantum Phase Estimation (QPE) on the second register with respect to the unitary

$$U|x\rangle = \begin{cases} |x-1 \pmod M\rangle & x < M \\ |x\rangle & x \geq M. \end{cases} \quad (9)$$

- Swap the first two registers, and reset the ancilla qubit by performing another  $< M$  conditional flip.

In the above procedure, since  $\mathcal{F}_M|x\rangle$  is an eigenstate of  $U$  with eigenphase  $x/M$ , the operation  $|x\rangle \mathcal{F}_M|x\rangle \mapsto |0\rangle \mathcal{F}_M|x\rangle$  is carried out (when  $x < M$ ). If instead  $x \geq M$  the system will be in the state  $|0\rangle|x\rangle|0\rangle$  and will remain unchanged by the inverse QPE. This is a straightforward approach to implementation of Eq. (8), but if it is unnecessary to preserve the  $> M$  subspace then there are other even more quantum circuit depth-efficient methods that can be used to implement the Fourier transform with arbitrary modulus [34].

The unitary  $\exp(iDt)$  can be implemented efficiently using well-known methods from [26], similar to  $U_Q(\gamma)$ . We require that the diagonal elements of  $D$  be efficiently computable, which is the case for graphs having efficiently computable eigenvalues. This is the case for circulant graphs when the maximum degree grows polynomially, or when the eigenvalues are known in closed-form (e.g. complete graphs). Thus, since each unitary can be implemented efficiently, the entire walk is efficient. It is anticipated that a low choice of  $p$ , leading to a relatively shallow circuit, will have a quantum advantage for near-term NISQ applications as discussed in [35].

#### IV. APPLICATIONS

In the following, we give an overview of a number of combinatorial structures and give their associated ranking and unranking functions. Thus, a continuous time quantum walk can be efficiently performed over each of following domains using the above-described scheme.

There are a wide range of integer sequences with ranking and unranking algorithms (or equivalently an efficient closed-form expression for the  $n$ th element of the sequence  $a_n$ , where the inverse operation is also efficiently computable). A comprehensive list of such sequences can be found on The On-Line Encyclopedia of Integer Sequences (OEIS) [36], with some examples given below.

##### A. Set $k$ -combinations

Let a binary string  $x = x_1 \dots x_n$  denote a combination, where  $x_j = 1$  iff the  $j$ th element is selected. Then an efficient ranking algorithm, to rank a given  $k$ -combination amongst all other  $k$ -combinations, is given in Algorithm 1.

---

##### Algorithm 1 RANK\_COMB( $x, k$ )

---

```

 $c \leftarrow$  list of  $j$  where  $x_j = 1$ 
sort  $c$  in ascending order
return sum from  $j = 1$  to  $k$  of  $\binom{c[j]}{j}$ 

```

---

The  $k$ -combination ranking function can be ‘wrapped’ to rank combinations amongst others of chosen sizes. For example in Fig. 3, we give the  $C$  matrix for choices of either 1 or 2 elements from a 4-element set, connected using a Möbius ladder graph.

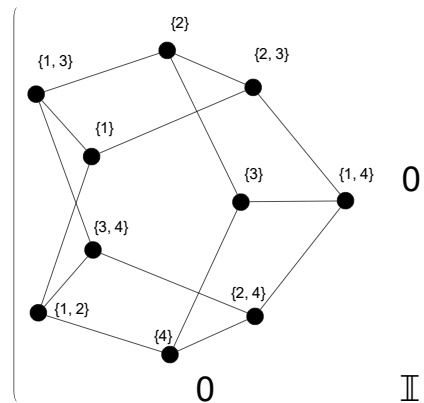


FIG. 3: An illustration of one choice of circulant connectivity of 2-combinations from a 4-element set.

By choosing  $n$  to be even and  $k = n/2$ , this is the domain of the graph partitioning problem. Similarly, this ranking algorithm can be applied to the critical node detection problem [37]. Another real-world application is to constrained financial portfolio optimisation in finance, where up to  $k$  assets are selected to create a financial portfolio with minimum risk and/or maximum return.

##### B. Permutations

A permutation can be ranked in linear time using Algorithm 2 [38], where  $\pi$  is a permutation of  $[n]$  and  $\pi^{-1}$  is the inverse permutation (which can be computed in linear time).

---

##### Algorithm 2 RANK\_PERM( $n, \pi, \pi^{-1}$ )

---

```

if  $n = 1$  then
  return 0
end if
 $s \leftarrow \pi[n-1]$ 
swap  $\pi[n-1]$  and  $\pi[\pi^{-1}[n-1]]$ 
swap  $\pi^{-1}[s]$  and  $\pi^{-1}[n-1]$ 
return  $s + n \times \text{RANK\_PERM}(n-1, \pi, \pi^{-1})$ 

```

---

As an example, the well-known NP-hard Travelling Salesman Problem has domain corresponding to the  $k!$  possible permutations of visiting each of  $k$  cities exactly

once and then returning to the starting city. A straightforward encoding is to use  $k\lceil\log k\rceil$  qubits, where each block of  $\lceil\log k\rceil$  qubits represents the next city to visit. The ranking unitary  $U_R$  can then be used to map this encoding of a tour to an integer from 0 to  $(k! - 1)$ , and vice versa.

### C. Set partitioning

Given a set  $X$  of size  $n$ , we can give a straightforward binary encoding of a  $k$ -partition as follows. For each object  $x \in X$  we assign  $\lceil\log k\rceil$  qubits to encode the partition to which  $x$  belongs. For example, the string 01 10 10 00 01 corresponds to the 3-partition  $\{4|1, 5|2, 3\}$ . We show how to rank a partition in Algorithm 3. Since an object is removed from the set at each iteration, this ranking function works in linear time.

---

#### Algorithm 3 RANK\_PARTITION( $\mathcal{S}, X, k$ )

---

```

sort  $X$ 
sort each subset in  $\mathcal{S}$ 
sort  $\mathcal{S}$  using the first element of each subset
if each subset in  $\mathcal{S}$  only consists of a single element then
  return 0
end if
 $x \leftarrow$  first element of  $X$ 
 $\mathcal{S}', X' \leftarrow$  remove  $x$  from  $\mathcal{S}, X$ 
sort  $\mathcal{S}', X'$  as above
if the subset of  $\mathcal{S}$  containing  $x$  is  $\{x\}$  then
  return RANK_PARTITION( $\mathcal{S}', X', k$ )
end if
 $j \leftarrow$  index of  $X'$  in  $\mathcal{S}'$ 
return  $\binom{n-1}{k-1} + kr(\mathcal{S}', X') + j$ 

```

---

As an example, this domain is relevant to the NP optimisation problem *maximum set splitting*. Given a collection of subsets  $C$  of a set  $S$ , the aim is to partition  $S$  into  $S_1$  and  $S_2$  such that the number of sets in  $C$  that are subsets of  $S_1$  or  $S_2$  is minimised.

### D. Lattice paths

A Dyck path is a lattice path from  $(0, 0)$  to  $(n, n)$  which never goes above the diagonal  $y = x$ . They are an example of a combinatorial structure characterised by the Catalan numbers, a sequence appearing in a number of combinatorial applications. Catalan numbers have an efficient ranking function provided in Algorithm 4, where the function NUM\_DYCK( $i, j$ ) counts the total number of Dyck paths between  $(0, 0)$  and  $(i, j)$  [39].

---

#### Algorithm 4 RANK\_CATALAN( $x$ )

---

```

 $n \leftarrow$  number of bits of  $x$ 
 $c_1 \leftarrow 2$ 
for  $j = 2$  to  $n$  do
   $c_j \leftarrow \max(x_{j-1} + 1, 2j)$ 
end for
 $r \leftarrow 1$ 
for  $j = 1$  to  $n - 1$  do
  for  $k = c_j$  to  $b_j - 1$  do
     $r \leftarrow r + \text{NUM\_DYCK}(n - j, n + j - k)$ 
  end for
end for
return  $r$ 

```

---

There are other lattice path applications. For example, an arbitrary length- $n$  path in 3D space can be characterised by an  $n$ -letter word where each letter is chosen from an alphabet of six directions. Any ‘word’ composed of a sequence of letters from some specified alphabet can be ranked, and constraints on some/all of the letters can also be incorporated [40].

## V. QUANTUM SEARCH

Finally, we briefly make explicit the connection with quantum search. The aim here is to search for one or more combinatorial objects having a desired property out of a set of  $M$  objects. Again,  $M$  is not necessarily a power of 2. Typically, one would encode each solution using  $n$  bits, for sufficiently large  $n$ , and perform a Grover search over the space of  $N = 2^n$  binary strings (ignoring the binary strings which do not correspond to any combinatorial object). Using the ranking unitary however, the search space can be cut down from  $N$  to  $M$ . As per Brassard *et al.* [3], the Grover iteration over a set of size  $M$  takes the form

$$Q = -\mathcal{F}_M S_0 \mathcal{F}_M^{-1} S_f, \quad (10)$$

where  $S_f$  conditionally negates the amplitude of objects meeting the search criteria and  $S_0$  conditionally negates the amplitude of the all-zero state. The only modification required is  $S_f \mapsto U_R S_f U_R^\dagger$ . This unranks the integers to their corresponding combinatorial object, so each object can be meaningfully interpreted and marked according to some combinatorial criteria and then be mapped back to an integer rank. Thus, the modified Grover iteration is

$$Q' = -\mathcal{F}_M S_0 \mathcal{F}_M^{-1} U_R S_f U_R^\dagger. \quad (11)$$

Quantum search over a set of  $M$  combinatorial objects with an associated ranking function can be performed in  $\mathcal{O}(\sqrt{M/k})$  for  $k$  marked combinatorial objects.



## VI. CONCLUSION

In this paper we have proposed an approach for efficient continuous time quantum walk over finite sets of combinatorial objects having an efficient ranking algorithm. The quantum algorithm requires that the choice of graph connecting the combinatorial objects is circulant. Our scheme is specifically beneficial for QAOA-based algorithms to optimise over combinatorial domains, since quantum walks are used as a mixing operator between feasible solutions to the problem. The variational parameter in this case is the quantum walk time. With this in mind, a specific benefit of our approach is that the size of the circuit is independent of the walk time  $t$ , so the quantum circuit does not need to be re-compiled each time the variational parameter is updated. In this paper we also briefly discuss the relevance to Grover search over combinatorial domains.

We note that although each  $U_W$  in the QWAO state evolution represents the same operator, this does not need to be the case. One could consider different connectivities amongst combinatorial objects for each quantum walk. For example, it may be beneficial to start with an initial quantum walk that is highly connected (c.f. complete graph) and increase the inter-solution connectivity each time, ending with a quantum walk over the objects connected as a cycle graph. This approach may be beneficial to ‘hone in’ on a high-quality solution, and we leave this to future work.

Finally, it is worth noting that this approach does not work if the combinatorial domain cannot be efficiently ranked, or if its size cannot be efficiently determined. For example, independent sets do not have an efficient ranking function; even the problem of counting the number of independent sets of a graph belongs to the complexity class  $\#P$ .

- 
- [1] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 4th ed. (Springer-Verlag Berlin Heidelberg, 2007).
- [2] L. K. Grover, *Chaos Solitons and Fractals* **10**, 1695 (1999).
- [3] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, in *Quantum computation and information (Washington, DC, 2000)*, Contemp. Math., Vol. 305 (Amer. Math. Soc., Providence, RI, 2002) pp. 53–74.
- [4] E. Farhi, J. Goldstone, and S. Gutmann, arXiv e-prints (2014), [arXiv:1411.4028 \[quant-ph\]](https://arxiv.org/abs/1411.4028).
- [5] S. Marsh and J. B. Wang, *Quantum Information Processing* **18**, 61 (2019).
- [6] E. Farhi and S. Gutmann, *Physical Review A* (3) **58**, 915 (1998).
- [7] J. Kempe, *Contemporary Physics* **44**, 307 (2003).
- [8] A. M. Childs, D. Gosset, and Z. Webb, *Science* **339**, 791 (2013).
- [9] K. Manouchehri and J. Wang, *Physical implementation of quantum walks*, Quantum Science and Technology (Springer, Heidelberg, 2014).
- [10] H. Tang, C. Di Franco, Z.-Y. Shi, T.-S. He, Z. Feng, J. Gao, K. Sun, Z.-M. Li, Z.-Q. Jiao, T.-Y. Wang, M. S. Kim, and X.-M. Jin, *Nature Photonics* **12**, 754 (2018).
- [11] A. M. Childs and J. Goldstone, *Physical Review A* **70**, 022314 (2004).
- [12] G. S. Engel, T. R. Calhoun, E. L. Read, T.-K. Ahn, T. Mancal, Y.-C. Cheng, R. E. Blankenship, and G. R. Fleming, *Nature* **446**, 782 (2007).
- [13] S. D. Berry and J. B. Wang, *Physical Review A* **82**, 042333 (2010).
- [14] S. Lloyd, *Science* **273**, 1073 (1996).
- [15] A. Ambainis, *SIAM J. Comput.* **37**, 210 (2007).
- [16] B. L. Douglas and J. B. Wang, *Journal of Physics A: Mathematical and Theoretical* **41**, 075303 (2008).
- [17] S. D. Berry and J. B. Wang, *Physical Review A* **83**, 042317 (2011).
- [18] A. Schreiber, A. Gábris, P. P. Rohde, K. Laiho, M. Štefaňák, V. Potoček, C. Hamilton, I. Jex, and C. Silberhorn, *Science* **336**, 55 (2012).
- [19] J. A. Izaac, J. B. Wang, P. C. Abbott, and X. S. Ma, *Physical Review A* **96**, 032305, 17 (2017).
- [20] F. Levi, *Nature Physics* **13**, 926 (2017).
- [21] Y. Ming, C.-T. Lin, S. D. Bartlett, and W.-W. Zhang, *npj Computational Materials* **5**, 88 (2019).
- [22] M. E. Tai, A. Lukin, M. Rispoli, R. Schittko, T. Menke, D. Borgnia, P. M. Preiss, F. Grusdt, A. M. Kaufman, and M. Greiner, *Nature* **546**, 519 (2017).
- [23] N. C. Harris, G. R. Steinbrecher, M. Prabhu, Y. Lahini, J. Mower, D. Bunandar, C. Chen, F. N. C. Wong, T. Baehr-Jones, M. Hochberg, S. Lloyd, and D. Englund, *Nature Photonics* **11**, 447 (2017).
- [24] Z. Yan, Y.-R. Zhang, M. Gong, Y. Wu, Y. Zheng, S. Li, C. Wang, F. Liang, J. Lin, Y. Xu, C. Guo, L. Sun, C.-Z. Peng, K. Xia, H. Deng, H. Rong, J. Q. You, F. Nori, H. Fan, X. Zhu, and J.-W. Pan, *Science* **364**, 753 (2019).
- [25] M.-A. Miri and A. Alù, *Science* **363**, 7709 (2019).
- [26] A. M. Childs, *Quantum information processing in continuous time*, Ph.D. thesis, Massachusetts Institute of Technology (2004).
- [27] S. Hadfield, Z. Wang, B. O’Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, *Algorithms* **12**, 34 (2019).
- [28] Z. Wang, N. C. Rubin, J. M. Dominy, and E. G. Rieffel, arXiv e-prints (2019), [arXiv:1904.09314 \[quant-ph\]](https://arxiv.org/abs/1904.09314).
- [29] X. Qiang, T. Loke, A. Montanaro, K. Aungskunsiri, X. Zhou, J. L. O’Brien, J. B. Wang, and J. C. F. Matthews, *Nature Communications* **7**, 11511 (2016).
- [30] S. S. Zhou and J. B. Wang, *R. Soc. Open Sci.* **4**, 160906, 12 (2017).
- [31] S. S. Zhou, T. Loke, J. A. Izaac, and J. B. Wang, *Quantum Information Processing* **16**, 82 (2017).
- [32] X. Qiang, X. Q. Zhou, J. W. Wang, C. M. Wilkes, T. Loke, S. O’Gara, L. Kling, G. D. Marshall, R. Santagati, T. C. Ralph, J. B. Wang, J. L. O’Brien, M. G. Thompson, and J. C. F. Matthews, *Nature Photonics* **12**, 534 (2018).
- [33] A. Y. Kitaev, arXiv e-prints (1995), [arXiv:quant-ph/9511026 \[quant-ph\]](https://arxiv.org/abs/quant-ph/9511026).
- [34] R. Cleve and J. Watrous, in *Proceedings 41st Annual Symposium on Foundations of Computer Science* (2000)

- pp. 526–536.
- [35] S. Bravyi, D. Gosset, and R. König, *Science* **362**, 308 (2018).
  - [36] OEIS Foundation Inc., “The On-Line Encyclopedia of Integer Sequences,” <http://oeis.org> (2019).
  - [37] M. Lalou, M. A. Tahraoui, and H. Kheddouci, *Computer Science Review* **28**, 92 (2018).
  - [38] W. Myrvold and F. Ruskey, *Information Processing Letters* **79**, 281 (2001).
  - [39] Z. Kása, *Acta Univ. Sapientiae, Inform.* **1**, 109 (2009).
  - [40] N. A. Loehr, *Bijjective combinatorics*, Discrete Mathematics and its Applications (Boca Raton) (CRC Press,

Boca Raton, FL, 2011) pp. 187–188.

#### ACKNOWLEDGEMENTS

This research was supported by a Hackett Postgraduate Research Scholarship and an Australian Government Research Training Program Scholarship at the University of Western Australia.