APPROVAL SHEET

Title of Dissertation: Solving Mathematical Problems in Quantum Regime

Name of Candidate: Abu Mohammad Omar Shehab Uddin Ayub
Doctor of Philosophy, 2016

Dissertation and Abstract Approved: S. J. Lomonaco Jr.
(Samuel J Lomonaco Jr.)
(Professor)
(Computer Science and Electrical Engineering)

Date Approved: July 7, 2016

ABSTRACT

Title of dissertation: SOLVING MATHEMATICAL PROBLEMS IN QUANTUM REGIME

Omar Shehab, Doctor of Philosophy, 2016

Dissertation directed by: Professor Samuel J Lomonaco Jr.
Department of Computer Science
and Electrical Engineering

In this dissertation, I investigate a number of algorithmic approaches in the quantum computational regime to solve mathematical problems. My problems of interest are the graph isomorphism and the graph automorphism problems, and the complexity of memory recall of Hopfield network. I show that the hidden subgroup algorithm, quantum Fourier sampling, always fails, to construct the automorphism group for the class of the cycle graphs. I have discussed what we may infer for a few non-trivial classes of graphs from this result. This raises the question, I have discussed in this dissertation, whether the hidden subgroup algorithm is the best approach for these kinds of problems. I have given a correctness proof of the Hen-Young quantum adiabatic algorithm for graph isomorphism for cycle graphs. To the best of my knowledge this result is the first of its kind. I also report a proof-of-concept implementation of a quantum annealing algorithm for the graph isomorphism problem on a commercial quantum annealing device. This is also, to the best of my knowledge, the first of its kind. I have also discussed the worst-case

for the algorithm. Finally, I have shown that quantum annealing helps us achieve exponential capacity for Hopfield networks.

SOLVING MATHEMATICAL PROBLEMS IN QUANTUM REGIME

by

Abu Mohammad Omar Shehab Uddin Ayub

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2016

Advisory Committee:
Professor Samuel J Lomonaco Jr., Chair/Advisor
Professor Milton Halem
Professor Yanhua Shih
Professor William Gasarch
Professor John Dorband

Dedication

Shaheda Begum,

the most curious and independent, and the strongest woman,

I have ever met, and my mother;

Dr. Mohammad Ayub,

the best listener I have ever met, and my father.

Acknowledgments

I gratefully acknowledge financial support from the NASA-AIST 2014 grant as a research assistant at the Center for Hybrid Multicore Productivity Research, UMBC. I also thank the department of Computer Science and Electrical Engineering, UMBC for supporting me during my first five years at UMBC.

I have received so much support in so many ways from a large number of people during my journey to finishing this dissertation. First, I thank my parents for giving me love, support, and freedom. My undergrad faculty, Muhammed Zafar Iqbal, encouraged me to study quantum information science for undergrad project which was my first introduction to the field. Samuel J Lomonaco, Jr., my PhD supervisor, taught me how to ask more questions and keep trying at answering them. He also gave me lots of freedom throughout the process. During my time at UMBC, I have had the good fortune to work with Ivan Erill, Alan Sherman, and John Dorband. These experiences have always been source of learning. Milton Halem has always been there with support and advices while I was being supported by the NASA-AIST 2014 grant and beyond. As committee members, Yanhua Shih and Bill Gasarch have always been there with their suggestions on how to improve my work. I have had lots of interactions with the participants of the StackExchange forum for mathematics, physics, and theoretical computer science. This helped me to have a great peer learning experience. I was also fortunate to be mentored by Kenneth M Zick during my internship at the Information Sciences Institute, University of Southern California, and Radhakrishnan Balu during my internship

at the US Army Research Lab. Both of these two visits have been very productive and educational. I also thank Siddhartha Santra for being a friend from whom I can learn physics.

My stay at Baltimore helped me make a number of great friends who have enriched my experience. Thank you very much Amrita, Blaise and Prajit for everything.

I have the best sister and brother in the world, Rumi and Rana. I wish my nickname also had started with an 'R' to sync with you!

Last but not the least, thank you very much, Silvia, my partner, for being there. You are the best!

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In *Quantum mechanical computers*, Feynman suggested that one of the potential applications of quantum computers will be the simulation of quantum physics. If we abstractify the idea, simulating physics is solving certain mathematical problems. Apart from that, there are mathematical problems which are interesting in their merit. So, if we are able to build quantum computers which can solve general mathematical problems, it would have wide applications in our real lives. It is natural to ask whether quantum information science help us in having better standing of yet unsolved mathematical problems. Moreover, one may also ask whether a solution in quantum regime can be better than the best known classical solutions. One such example is the Shor's algorithm. We have had very few progress since Shor's. Why are not we seeing such progress for most of the other unsolved mathematical problems? To answer to that question, we have to understand the structure of the problems in quantum regime better. Moreover, we also need to set near term goals using quantum enhancements in classical problem solving as we are still at the early stage of building universal quantum computers. In this dissertation I ask and answer when graph isomorphism, an old open mathematical problem, is easy and impossible to solve in quantum computation for selected classes.

1.1 Outline and Summary of Results

I have given a preliminary background in *Graph Theory*, *Pseudo-Boolean optimization*, *Representation Theory*, *Quantum Fourier Transformation*, and *Quantum Adiabatic Algorithms* in Chapter 2. Following the chapter, I present the following new results in the next four chapters.

Chapter 3: Graph Automorphism and Hidden Subgroup Algorithms shows that the hidden subgroup approach (both strong and weak quantum Fourier sampling) always fails for cycle graph automorphism problem. Then, I have shown how to determine non-trivial classes of graph automorphism problems for which the same technique also always fails.

Chapter 4: Graph Isomorphism and Quantum Adiabatic Algorithms gives the first correctness proof of the Hen-Young quantum adiabatic graph isomorphism algorithm for cycle graphs. I have shown that the time for adiabatic evolution to find the configuration of a graph grows in the cubic power of the size of the input graphs.

Chapter 5: Graph Isomorphism and Quantum Annealing gives the first ever experimental implementation of a quantum annealing algorithm for graph isomorphism using manufactured spins. I have also shown that, for $\frac{n-1}{2}$-regular graphs, the requirement on the number of physical spins becomes worse.

Chapter 6: Hopfield Network and Quantum Annealing shows that we can achieve exponential memory capacity for a Hopfield network if we implement the memory recall phase using quantum annealing. We have also implemented it using

a commercial quantum annealer.

Chapter 2

Preliminaries

2.1 Graph Theory

Most of the work presented in this dissertation involves the graph isomorphism and automorphism problems. So, it would be appropriate to start the background chapter with a few concepts of graph theory. Most of the materials in this section are reproduced from the very well written book by Bollobás [30]. The materials I have covered do not constitute a comprehensive coverage on graph theory, rather they are only related to the problems I have approached in this dissertation.

Definition 1 (Graph)**.** *A graph G is an ordered pair of disjoint sets (V, E) such that E is a subset of the set $V^{(2)}$ of unordered pairs of V.*

We call V as the set of vertices, and E as the set of edges. Each element of E connects two elements of V. A graph is directed of the edge (v_i, v_j) is an element of E but (v_j, v_i) is not for all i and j. A simple graph does not have loops or multi-edges. This dissertation only focuses on questions defined on simple undirected graphs.

A graph can be partitioned in more than one ways. They are defined as follows.

Definition 2 (Vertex partition)**.** *The vertex partition $P = P_1, \ldots, P_i, \ldots$ of a graph $\Gamma = (V, E)$ is a partition over V such that $\bigcup_i P_i = V$.*

Definition 3 (Edge partition). *The edge partition $Q = Q_1, \ldots, Q_i, \ldots$ of a graph $\Gamma = (V, E)$ is a partition over E such that $\bigcup_i Q_i = E$.*

Definition 4 (Regular coloring). *A coloring C of a graph Γ is called regular if i) any two vertices in P_i, are the end vertices of the same number of edges in Q_j for all i, j and ii) any two edges in Q_j have the same number of end vertices in P_i for all i and j.*

The study of the automorphism group of a graph is a very active area of research. I present a few basic related definitions as follows.

Theorem 1 (Orbit theorems [172]). *Let $Aut(\Gamma)$ be the automorphism group of the graph $\Gamma = (V, E)$. The orbits of $Aut(\Gamma)$ on vertices and edges are the partitions,*

$$P = \{\{gv | g \in Aut(\Gamma)\} | v \in V\}$$
$$Q = \{\{ge | g \in Aut(\Gamma)\} | e \in E\} \tag{2.1}$$

(P, Q) maintains the regularity condition from Definition 4.

In 1971 Conway introduced the following theorem [43]. Complete proofs of the Theorem 2 are given in [133, 23, 107].

Theorem 2. *An arc-transitive group of automorphism $Aut(\Gamma)$ of a d-valent graph Γ ($d \geq 2$) has exactly $d-1$ orbits in its action on the set of all G-consistent cycles.*

Conway's theorem was generalized in [36] as follows.

Theorem 3. *Let G be any subgroup of the symmetric group S_n. Let X be the set of all cycles occurring in the elements of G. Then G has exactly n orbits in its action on X by conjugation.*

Definition 5 (k-clique)**.** *A k-clique is defined as the completely connected subgraph of size k in a graph.*

I also define k-tree in this section [89].

Definition 6 (k-tree)**.**

- *The complete graph on k vertices is a k-tree.*
- *A k-tree G with $n+1$ vertices ($n \geq k$) can be constructed from a k-tree H with n vertices by adding a vertex adjacent to exactly k vertices that form a k-clique in H.*
- *No other graphs are k-trees.*

The partial k-tree is a generalization of k-tree.

Definition 7 (Partial k-tree)**.** *A graph G is a partial k-tree if and only if G has a treewidth at most k.*

The *tree decomposition* of a graph is defined as follows [89].

Definition 8 (Tree decomposition)**.** *A tree decomposition of a graph $G = (V, E)$ is a pair*

$$\left(\left\{X_i | i \in \mathbb{Z}^+\right\}, T = \left(\mathbb{Z}^+, M\right)\right)$$

where $\{X_i | i \in \mathbb{Z}^+\}$ *is a collection of subsets of* V *(also called bags), and* T *is a tree, such that:*

- $\bigcup_{i \in \mathbb{Z}^+} X_i = V$
- $(u, v) \in E \implies \exists i \in \mathbb{Z}^+$ *with* $u, v \in X_i$
- *For all vertices* $v \in V$, $\{i \in \mathbb{Z}^+ | v \in X_i\}$ *induces a connected subtree of* T.
- M *is the set of edges in the tree.*

The *treewidth* of a graph is defined as follows.

Definition 9 (Treewidth)**.** *Let the width of a tree decomposition* $(\{X_i | i \in \mathbb{Z}^+\}, T = (\mathbb{Z}^+, M))$ *be* $\max_{i \in \mathbb{Z}^+} |X_i| - 1$. *The treewidth of a graph* G, $tw(G)$, *is the minimum width over all tree decompositions of* G.

Now I like to define the concept of frames [115, 38].

Definition 10 (Frame)**.** *A nonempty set* S *of independent vertices of the graph* G *is called a frame of* G *if* $G - S'$ *is connected for any* $S' \subseteq S$.

If $|S| = k$ it is called a k-frame.

Kyaw [115] has given the following sufficient condition for a graph being k-tree.

Theorem 4. *Let* $G = \{V, E\}$ *be a connected graph and* $k (\geq 2)$ *an integer. If*

$$d_G(S) + \sum_{i=2}^{k+1} (k-1) |N_i(S)| \geq n - 1 \tag{2.2}$$

for every $k+1$*-frame* S *in* G, *then* G *has a* k*-tree.*

Here, d_G is defined as follows. For any nonempty subset S of V, $d_G(S) = \sum_{s \in S} d_G(s)$. $d_G(s)$ is the degree of the vertex $s \in V$ in G. Moreover, $N_i(S)$ is defined as $N_i(S) = \{v \in V : |N_G(v) \cap S| = i\}$. $N_G(u)$ is defined as $N_G(u) = \{v \in V : (u, v) \in E\}$ for a given $u \in V$.

2.1.1 Isomorphism and automorphism

The graph isomorphism and automorphism problems are two of the oldest problems in combinatorics. The formal statement of the graph isomorphism problem goes as follows as mentioned in [68].

Definition 11 (Graph isomorphism (**GI**))**.** *Given two graphs, $\Gamma_1 = (V_1, E_1)$ and $\Gamma_2 = (V_2, E_2)$, does there exist a bijection $f : V_1 \to V_2$ such that $\forall a, b \in V_1, (a, b) \in E_1 \iff (f(a), f(b)) \in E_2$?*

Here, V_1 and V_2 are the sets of vertices and E_1 and E_2 are the sets of edges of Γ_1 and Γ_2 respectively.

Graph automorphism is a special version of Definition 11 when $\Gamma_1 = \Gamma_2$.

Definition 12 (Graph automorphism (**GA**))**.** *Given a graph $\Gamma = (V, E)$, the automorphism group are $\Gamma \to \Gamma$ isomorphisms; they form the subgroup $Aut(\Gamma)$ of the symmetric group $S_{|V|}$.*

Read et al. [155] have named the tendency of incessant but unsuccessful attempts at the problem as the *graph isomorphism diseases*. This indicates the amount of interest about the problem among the mathematicians. The current best known algorithm for the general graph isomorphism problem is due to Babai

et al. [16]. The algorithm exploits graph canonization techniques through label reordering in exponential time ($exp\left(n^{\frac{1}{2}+o(1)}\right)$), where $n = |V|$. While the best known algorithm for the general graph isomorphism problem is exponential, better results have been proven for graph sub classes with special properties. In [16], Babai et al. also proved the bound for tournament graphs is $n^{\left(\frac{1}{2}+o(1)\right)\log n}$. In [121], Luks reduced the bounded valence graph isomorphism problem to the color automorphism problem, and gave a polynomial time algorithm. In another paper [15], Babai et al. created two polynomial algorithms using two different approaches, i.e., the tower of groups method, and the recursion through systems of imprimitivity respectively, for the bounded eigenvalue multiplicity graph isomorphism problem. The isomorphism problem for planar graphs is known to be in polynomial time due to Hopcroft et al. [95]. In their paper, the authors used a reduction approach to eventually tranform the graphs into five regular polyhedral graphs and check the isomorphism by exhaustive matching in a fixed finite time. Miller [134] used a different approach by finding minimal embeddings of the graphs of bounded genus and checking their isomorphism by generating codes. Babai et al. in [14] and Czajka et al. in [47] showed that the isomorphism of almost all the graphs in a class of random graphs can be tested in linear time. Both of their approaches exploit the properties of the degree sequence of a random graph. Babai et al. [13] proved that while the graph isomorphism problem for strongly regularly graphs may be solved faster than the general version it is still an exponential time algorithm.

Although we have an exponential time algorithm for the general graph isomorphism problem, it is not proven to be optimal. So, the complexity class of the

graph isomorphism problem is yet undecided. While we know that it is in **NP** [75], we don't know whether it is in **P** or **NP**-complete. This is why the graph isomorphism problem is called an **NP**-intermediate problem. Schöning [166] has shown that graph isomorphism is in L_2^P and not γ-complete under the assumption that the polynomial hierarchy does not collapse to L_2^P. Given this information, many researchers believe that the graph isomorphism problem is not **NP**-complete.

While the efforts towards finding an efficient solution for the general graph isomorphism problems have been unsuccessful, the researchers have attempted practically feasible methods to solve the problem in reasonable time frame.

2.2 Pseudo-Boolean optimization

Pseudo-Boolean optimization is an active area of study in operations research. In Chapter 5, I have represented the graph isomorphism problem as a pseudo-Boolean function. Hence, it would be appropriate to discuss the related theories. A pseudo-Boolean function can be defined as follows.

A function f is called a *set function* when it maps a set to a number. Let, $\mathbb{R}$ is the set of reals, $\mathbb{B} = \{0, 1\}$ and n is a positive integer. So, $\mathbb{B}^n$ is the set of all binary n-tuples. If the function f, such that $f : \mathbb{B}^n \rightarrow \mathbb{R}$, f is called a pseudo-Boolean function.

Definition 13 (Pseudo-Boolean function)**.** *A pseudo-boolean function, f, is a mapping $f : \mathbb{B}^n \rightarrow \mathbb{R}$ where $n \in \mathbb{Z}^+$ and $\mathbb{Z}^+$ is the set of all positive integers.*

A pseudo-Boolean function can be expressed as an algebraic expression of

degree n polynomials. When the highest degree of such polynomial is two it is called a quadratic pseudo-Boolean function. The inputs to the function are also called *configurations*.

In this report, I study only the quadratic pseudo-Boolean functions. An introductory survey on pseudo-boolean functions is available in Boros et al. [32].

As a special case of pseudo-Boolean functions, a quadratic unconstrained binary optimization (QUBO) is defined in [182]. Boolean problems are encoded as QUBO problems when one wants to solve them using the commercial quantum annealers built by the D-Wave Systems Inc. [109].

Definition 14 (Quadratic unconstrained binary optimization)**.** *Quadratic unconstrained binary optimization is a quadratic pseudo-Boolean function which determines the minimum over* $\{0,1\}^n$ *and takes the form*

$$f(x_1,\ldots,x_n) = c_0 + \sum_{i=1}^{n} c_i x_i + \sum_{1\le i\le j\le n} c_{ij} x_i x_j \tag{2.3}$$

Here, $i,j,n \in \mathbb{Z}^+$ *and* $c_i \in \mathbb{R}$*.*

The derivative of a pseudo-Boolean function is defined as follows.

Definition 15 (Derivative)**.** *The* i*-th derivative of a pseudo-Boolean function* f*,* $\Delta_i(\mathbf{x})$*, is defined as follows.*

$$\Delta_i(\mathbf{x}) = f(x_1,\ldots,x_{i-1},1,x_{i+1},\ldots,x_n) - f(x_1,\ldots,x_{i-1},0,x_{i+1},\ldots,x_n) \tag{2.4}$$

I also define the residual of a pseudo-Boolean function.

Definition 16 (Residual)**.** *The i-th residual of a pseudo-Boolean function f, $\Theta_i(\mathbf{x})$, is defined as follows.*

$$\Theta_i(\mathbf{x}) = f(\mathbf{x}) - x_i \Delta_i(\mathbf{x}) \tag{2.5}$$

The local minima of a pseudo-Boolean function is defined as follows.

Definition 17 (Local minima [44])**.** *Two configurations $\mathbf{X}$ and $\mathbf{Y}$ are neighbors if the Hamming distance between them is exactly one. $\mathbf{X} \in \mathbb{B}^n$ is a local minimum of the pseudo-Boolean function $f : \mathbb{B}^n \to \mathbb{R}$ if,*

$$f(\mathbf{X}) \leq f(\mathbf{Y}) \tag{2.6}$$

for all neighbors $\mathbf{Y}$ of $\mathbf{X}$.

I like to note that that every pseudo-Boolean function has a unique multilinear polynomial representation [82, 83, 32].

Theorem 5. *Every pseudo-Boolean function $f : \mathbb{B}^n \mapsto \mathbb{R}$ has a unique multi-linear polynomial representation of the following form.*

$$f(x_1, \ldots, x_n) = \sum_{S \subseteq \mathbf{V}} c_S \prod_{j \in S} x_j \tag{2.7}$$

By convention, we always assume that $\prod_{j \in \emptyset} x_j = 1$.

Here, $\mathbf{V} \in \mathbb{N}$ and $c_S \in \mathbb{R}$.

The *linear majorant* and *linear minorant* are defined as follows [32].

Definition 18 (Linear majorant)**.** *A linear function* $l(\mathbf{x}) = l_0 + l_1x_1 + l_2x_2 + \ldots + l_nx_n$ *is called a linear majorant of a quadratic pseudo-Boolean function* f*, if* $l(\mathbf{x}) \geq f(\mathbf{x})$ *holds for all* $\mathbf{x} \in \mathbb{B}^n$*.*

Definition 19 (Linear minorant)**.** *A linear function* $l(\mathbf{x}) = l_0 + l_1x_1 + l_2x_2 + \ldots + l_nx_n$ *is called a linear minorant of a quadratic pseudo-Boolean function* f*, if* $l(\mathbf{x}) \leq f(\mathbf{x})$ *holds for all* $\mathbf{x} \in \mathbb{B}^n$*.*

I also define a few concepts related to *roof-duality* [81].

Definition 20 (Upper plane)**.** *A linear function* $p(\mathbf{x})$ *is called an upper plane of a pseudo-Boolean function* $f(\mathbf{x})$*, if for all* $\mathbf{x} \in \mathbb{B}^n$*,* $p(\mathbf{x}) \geq f(\mathbf{x})$*.*

The upper planes are also known as *roofs*. The set of the upper planes S is complete if $f(\mathbf{x}) = \min_{p(\mathbf{x}) \in S} p(\mathbf{x})$ for all $\mathbf{x}$ in $\mathbb{B}^n$. I also define the concept of *local upper plane* for each term in f [81].

Definition 21 (Local upper plane)**.** *Let a quadratic pseudo-Boolean function be,*

$$f(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j$$

where $q_{ij} = 0$ *whenever* $i > j$*. For each term* $q_{ij}x_ix_j$*, the local upper plane is defined as*

$$p_{ij}(x_i, x_j) = a_{ij}x_i + b_{ij}x_j + c_{ij}$$

where $c_{ij} \geq 0$*,* $a_{ij} + c_{ij} \geq 0$*,* $b_{ij} + c_{ij} \geq 0$*, and* $a_{ij} + b_{ij} + c_{ij} \geq 0$*.*

Definition 22 (Duality gap)**.** *Let the function $M(f,S)$ be the function $M(f,S) = \max_{\mathbf{x}\in\mathbb{B}^n} \min_{p\in S} p(\mathbf{x})$. The duality gap for the pseudo-Boolean function is,*

$$\max_{\mathbf{x}\in\mathbb{B}^n} \min_{p\in S} p(\mathbf{x}) - \max_{\mathbf{x}\in\mathbb{B}^n} f(\mathbf{x})$$

.

An important concept associated with a pseudo-Boolean function is the *co-occurrence graph* [32].

Definition 23 (Co-occurrence graph)**.** *If a pseudo-Boolean function $f : \mathbb{B}^n \mapsto \mathbb{R}$ is given by its unique multi-linear polynomial, a graph $G_f = (V, E)$ is called its co-occurrence graph, in which $(i,j) \in E$ (for $i,j \in V, i \neq j$) iff f has a term for which $S \supseteq \{i,j\}$ and $c_S \neq 0$.*

The *tree-width* of f is the *tree-width* of G_f.

A basic algorithm for pseudo-Boolean optimization, **BASIC-ALGORITHM**, was first proposed in [82, 84] and later simplified in [83]. I present the algorithm as Algorithm 1.

Algorithm 1 BASIC-ALGORITHM

procedure BASIC-ALGORITHM(f) ▷ Input
  Let n denote the number of variables.
  if If $n = 1$ and $f(1) > f(0)$ **then** RETURN $x_1^* = 0$
  else RETURN $x_1^* = 1$
  end if
  if If $n > 1$ **then** continue
  end if
▷ Local optimality
  Label the variables, and choose x_n to be eliminated.
  Determine the pseudo-Boolean function g_n defined by

$$g_n(x_1, \ldots, x_{n-1}) = \begin{cases} 1 & \text{if } \Delta_n(x_1, \ldots, x_{n-1}) < 0, \text{and} \\ 0 & \text{otherwise.} \end{cases}$$

▷ Recursion
  Determine $f^{n-1}(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, g_n(x_1, \ldots, x_{n-1}))$
  Obtain the optimal values for $x_1^*, x_2^*, \ldots, x_{n-1}^*$ by calling **BASIC-ALGORITHM** (f^{n-1})
▷ Output
  Set $x_n^* = g_n(x_1^*, \ldots, x_{n-1}^*)$, and RETURN the binary vector $\mathbf{x}^* = (x^*, \ldots, x_n^*)$.
end procedure

The relation between the complexity of **BASIC-ALGORITHM** and the co-occurrence graph of a pseudo-Boolean function was established in [45].

Theorem 6. *If for a pseudo-Boolean function f its co-occurrence graph G_f is a partial k-tree, then* **BASIC-ALGORITHM** *can be implemented to run in polynomial time in the input size size(f) and in 2^k.*

2.3 Representation Theory

I introduce a few concepts of the representation theory in this section which are relevant to the work in Chapter 3. A more detailed introduction may be found in [72, 46].

Definition 24 (Young diagram). *For any partition $\lambda_1, \ldots, \lambda_k$ of an integer λ, there is a diagram associated called the Young diagram where there are λ_i cells in the i-th row. The cells are lined up on the left.*

Definition 25 (Hook). *For a cell (i, j) of a Young tableau λ, the (i, j)-hook $h_{i,j}$ is the collection of all cells of λ which are beneath (i, j) (but in the same column) or t the right of (i, j) (but in the same row), including the cell (i, j). The length of the hook is the number of cells appearing in the hook.*

Definition 26 (Skew hook). *A skew hook s of a Young diagram λ is a connection collection of boundary boxes such that their removal from λ results in a (smaller) diagram.*

Definition 27 (Restricted and induced representations). *If $H \subset G$ is a subgroup, any representation ρ_1 of G restricts to a representation of H, denoted $Res^G_H \rho_1$ or simple $Res\rho_1$. Let $\rho_2 \subset \rho_1$ be a subspace which is H-invariant. For any g in G, the subspace $g.\rho_2 = \{g.w : w \in \rho_2\}$ depends only on the left coset of gH of g modulo H, since $gh.W = g.(h.\rho_2) = g.\rho_2$; for a coset c in G/H, I write $c.\rho_2$ for this subspaceof ρ_1 subspace of ρ_1. I say that ρ_1 is induced by ρ_2 if every element in ρ_1 can be written uniquely as a sum of elements in such translates of ρ_2, i.e.*

$$\rho_1 = \bigoplus_{c \in G/H} c.\rho_2 \tag{2.8}$$

In this case I write the induced representation $\rho_1 = Ind^G_H \rho_2 = Ind\ \rho_2$.

A common representation we will see in later sections of this report is the

regular representation [72].

Definition 28 (Regular representation)**.** *If X is any finite set and G actson the left on X, i.e., $G \to Aut(X)$ is a homomorphism to the permutation group of X, there is a associated permutation representation: let V be the vector space with basis $\{e_x : x \in X\}$, and let G act on V by*

$$g \cdot \sum a_x e_x = \sum a_x e_{gx}. \tag{2.9}$$

The regular representation, denoted R_G or R, corresponds to the left action of G on itself.

The character of a group element is defined as follows [72].

Definition 29 (Character)**.** *If ρ is a representation of a group G, its character χ_ρ is the complex-valued function on the group defined by*

$$\chi_\rho(g) = Tr(g|_\rho) \tag{2.10}$$

the trace of g on ρ.

It is useful to define the inner product of characters [162].

Definition 30 (Inner product of characters)**.** *Let χ and ψ be the characters of a group G. The inner product of χ and Ψ is*

$$\langle \chi, \Psi \rangle = \frac{1}{|G|} \sum_{g \in G} \chi(g) \Psi^\dagger(g) \tag{2.11}$$

The character table of a finite group is defined as follows [162].

Definition 31 (Character table)**.** *Let G be a group. The character table of G is an array with rows indexed by the inequivalent irreducible characters of G and columns indexed by the conjugacy classes. The table entry in row χ and column K is χ_K:*

	$\ldots$	K	$\ldots$
$\vdots$		$\vdots$	
χ	$\cdots$	χ_K	
$\vdots$			

By convention, the first row corresponds to the trivial character, and the first column corresponds to the class of the identity, $K = \{e\}$.

I define the wreath product as follows [54].

Definition 32 (Wreath product)**.** *Let K and L be groups, let n be a positive integer, let $\phi : K \to S_n$ be a homomorphism and let H be the direct product of n copies of L. Let ψ be an injective homomorphism from S_n into $Auto(H)$ constructed by letting the elements of S_n permute the n factors of H. The composition $\psi \circ \phi$ is a homomorphism from G into $Aut(H)$. The wreath product of L by K is the semidirect product $H \rtimes K$ with respect to this homomorphism and is denoted by $L \wr K$.*

The dimension of an irreducible representation of the symmetric group is defined as follows.

Definition 33 (Dimension of an irreducible representation)**.** *The dimension dim_{ρ_λ} of an irreducible representation ρ for a partition $\lambda = (\lambda_1 + \ldots + \lambda_i + \ldots + \lambda_k)$ of a symmetric group S_n is given as follows [72].*

$$dim_{\rho_\lambda} = \frac{n!}{l_1 \cdot \ldots \cdot l_k!} \Pi_{i<j} \left(l_i - l_j \right), \tag{2.12}$$

with $l_i = \lambda_i + k - i$.

It is suitable to mention the following theorem on the multiplicity of an irreducible representation in the regular representation [72].

Theorem 7. *Every irreducible representation* ρ *occurs* $dim\,(\rho)$ *times in the regular representation.*

Proof of Theorem 7. Let χ be the character of the regular representation. Then

$$\chi\left(g\right) = \begin{cases} n & \text{if } g = 1, \text{ and} \\ 0 & \text{otherwise} \end{cases}$$

Because, each group elements acts by a permutation matrix, and the trace of a permutation matrix is simply the number of fixed points of the permutation. Thus,

$$\begin{aligned} \langle \chi_\rho, \chi \rangle &= \frac{1}{n} \chi_\rho \left(1\right) \bar{\chi}\left(1\right) \\ &= \frac{1}{n} \dim\left(\rho\right) n \\ &= \dim\left(\rho\right) \end{aligned}$$

□

Another two important concepts in representation theory are *restriction* and

induction [162].

Definition 34 (Restriction)**.** *Let H be a subgroup of G and X be a matrix representation of G. The restriction of X to H, $X\downarrow_H^G$, is given by*

$$X\downarrow_H^G(h) = X(h)$$

for all $h \in H$.

Definition 35 (Induction)**.** *Let $H \leq G$ and $t_1, \ldots, t_l$ be a fixed transversal for the left cosets of H, i.e., $G = t_1H \sqcup \ldots \sqcup t_lH$. If Y is a representation of H, then the corresponding induced representation $Y\uparrow_H^G$ assigns to each $g \in G$ the block matrix*

$$Y\uparrow_H^G(g) = Y\left(t_i^{-1}gt_j\right)$$
$$= \begin{pmatrix} Y\left(t_1^{-1}gt_1\right) & Y\left(t_1^{-1}gt_2\right) & \cdots & Y\left(t_1^{-1}gt_l\right) \\ Y\left(t_2^{-1}gt_1\right) & Y\left(t_2^{-1}gt_2\right) & \cdots & Y\left(t_2^{-1}gt_l\right) \\ \vdots & \vdots & \ddots & \vdots \\ Y\left(t_l^{-1}gt_1\right) & Y\left(t_l^{-1}gt_2\right) & \cdots & Y\left(t_l^{-1}gt_l\right) \end{pmatrix}$$

where $Y(g)$ is the zero matrix if $g \notin H$.

It is natural to define the characters for the restricted and induced representations [170].

Definition 36 (Character of restricted representation)**.** *Let X be a matrix representation of a group G, and let $H \leq G$ be a subgroup. Then, the character of the*

restricted representation $\chi\downarrow_H^G(h)$ *is the character of the original representation* $\chi(h)$ *for all* $h \in H$.

I reproduce the character of induced representation from [10].

Definition 37 (Character of induced representation)**.** *Let* Y *be a matrix representation of a group* H *such that* $H \leq G$. *I pick a transversal of* H *in* G. *Using our formula for the induced representation we find,*

$$\begin{aligned}
\chi\uparrow_H^G(g) &= Tr\left(Y\left(t_i^{-1}gt_j\right)\right) \\
&= Tr\begin{pmatrix} Y\left(t_1^{-1}gt_1\right) & Y\left(t_1^{-1}gt_2\right) & \cdots & Y\left(t_1^{-1}gt_l\right) \\ Y\left(t_2^{-1}gt_1\right) & Y\left(t_2^{-1}gt_2\right) & \cdots & Y\left(t_2^{-1}gt_l\right) \\ \vdots & \vdots & \ddots & \vdots \\ Y\left(t_l^{-1}gt_1\right) & Y\left(t_l^{-1}gt_2\right) & \cdots & Y\left(t_l^{-1}gt_l\right) \end{pmatrix} \\
&= \sum_{i=1}^{n} Tr\left(Y\left(t_i^{-1}gt_i\right)\right) \\
&= \sum_{i=1}^{n} \chi\left(t_i^{-1}gt_i\right)
\end{aligned}$$

where $\chi(g)$ *is the zero matrix if* $g \notin H$.

Since, χ is a class function on H, conjugation by any element $h \in H$ leaves it the same. So, $\chi(h^{-1}gh) = \chi(g)$ for all $g \in G$ and $h \in H$.

We do exactly this for each element of H, add all the results together and divide by the number of elements of H. In other words, I write the above function out in $|H|$ different ways, add them all together, and divide by $|H|$ to get exactly what we started with:

$$\chi \uparrow_H^G (g) = \frac{1}{|H|} \sum_{h \in H} \sum_{i=1}^{n} \chi \left(h^{-1} t_i^{-1} g t_i h \right)$$
$$= \frac{1}{|H|} \sum_{h \in H} \sum_{i=1}^{n} \chi \left((t_i h)^{-1} g (t_i h) \right) \tag{2.13}$$

But now as t_i varies over the transversal, and as h varies over H, their product $t_i h$ varies exactly once over G. That is, every $x \in G$ can be written in exactly one way in the form t_h for some transversal element t_i and subgroup element h. Thus we have the following relation.

$$\chi \uparrow_H^G (g) = \frac{1}{|H|} \sum_{x \in G} \chi \left(x^{-1} g x \right) \tag{2.14}$$

It would be appropriate if I mention the theorem on the *Frobenius reciprocity* in this section [162].

Theorem 8 (Frobenius reciprocity)**.** *Let $H \leq G$ and suppose that ψ and χ are characters of H and G, respectively. Then*

$$\langle \psi \uparrow_H^G, \chi \rangle_G = \langle \psi, \chi \downarrow_H^G \rangle_H \tag{2.15}$$

where the left inner product is calculated in G and the right one in H.

A special case of *Frobenius reciprocity* is relevant to our discussion where the representation of H is the trivial representation $\mathbf{1}_H$ [80]. I describe the case as

follows.

Lemma 1 (Special case of Frobenius reciprocity)**.** *Let $H \leq G$ and suppose that χ_ρ is the character of the irreducible representation ρ of G. Then*

$$\langle \chi \uparrow^G_{H\mathbf{1}_H}, \chi_\rho \rangle_G = \langle \chi_{\mathbf{1}_H}, \chi_\rho \downarrow^G_H \rangle_H \tag{2.16}$$

where the left inner product is calculated in G and the right one in H.

I would like to reproduce the Example 3.13 from [72] here.

Remark 1. *The permutation representation associated to the left action of G on G/H is induced from the trivial one-dimensional representation W of H. Here, the representation of G, V has basis $\{e_\sigma : \sigma \in G/H\}$, and $W = \mathbb{C} \cdot e_H$, with $H$the trivial coset.*

Following Remark 1, we can say that $\mathbf{1}_H \uparrow^G_H$ is the permutation representation of G. So, according to the Theorem 7, the multiplicity of ρ in $\mathbf{1}_H \uparrow^G_H$ is d_ρ.

2.4 Quantum Fourier Transformation

Quantum Fourier Transformation is an integral component of the theorem presented in Chapter 3. The quantum Fourier transform of a map from a finite group to its representation is defined as follows [80].

Definition 38 (Fourier transformation of a finite group)**.** *Let $f : G \to \mathbb{C}$. The*

Fourier transform of f at the irreducible representation ρ is the $d_\rho \times d_\rho$ matrix

$$\hat{f}(\rho) = \sqrt{\frac{d_\rho}{|G|}} \sum_{g \in G} f(g)\, \rho(g) \tag{2.17}$$

In quantum Fourier transform, the superposition $\sum_{g \in G} f_g |g\rangle$ is identified with the function $f : G \to \mathbb{C}$ defined by $f(f(g)) = f_g$. Using this notation, $\sum_{g \in G} f(g) |g\rangle$ is mapped under the Fourier transform to $\sum_{\rho \in \hat{G}, 1 \le i,j \le d_\rho} \hat{f}(\rho)_{i,j} |\rho, i, j\rangle$. Here, $\hat{G}$ is the set of all irreducible representations of G and $\hat{f}(\rho)_{i,j}$ is a complex number. The probability of measuring the register $|\rho\rangle$ is

$$\sum_{1 \le i,j \le d_\rho} |\hat{f}(\rho)_{i,j}|^2 = \|\hat{f}(\rho)\|^2 \tag{2.18}$$

where $\|A\|$ is the natural norm (also known as Frobenius norm) given by $\|A\|^2 = Tr\left(A^\dagger A\right)$.

The Frobenius norm can be calculated from the characters of the group associated which is demonstrated in the following theorem reproduced from [80].

Theorem 9. *If, f is an indicator function of a left closet of H in G, i.e. for some $c \in G$,*

$$f(g) = \begin{cases} \frac{1}{\sqrt{|H|}} & \text{if } g \in cH, \text{ and} \\ 0 & 0 \text{ otherwise} \end{cases} \tag{2.19}$$

, then,

$$\| \hat{f}(\rho) \|^2 = \frac{|H|}{|G|} d_\rho \langle \chi_\rho, \chi_{\mathbf{1}_H} \rangle_H \tag{2.20}$$

Proof of Theorem 9. From Definition 38, we know that,

$$\| \hat{f}(\rho) \|^2 = \sum_{1 \leq i,j \leq d_\rho} | \hat{f}(\rho)_{i,j} |^2 \tag{2.21}$$

I seek to only measure ρ.

I use the assumption in the theorem.

$$\| \hat{f}(\rho) \|^2 = \| \rho(c) \sum_{h \in H} \rho(h) \|^2 \tag{2.22}$$

$\rho(c)$ is a unitary matrix. So, as a multiplier it does not change the norm [132].

$$\| \hat{f}(\rho) \|^2 = \| \sum_{h \in H} \rho(h) \|^2 \tag{2.23}$$

So, the probability of measuring ρ is determined by $\sum_{h \in H} \rho(h)$. If correctly normalized, $\frac{1}{|H|} \sum_{h \in H} \rho(h)$ is a projection.

$$\begin{aligned} \left(\frac{1}{|H|} \sum_{h \in H} \rho(h) \right)^2 &= \frac{1}{|H|^2} \sum_{h_1, h_2 \in H} \rho(h_1 h_2) \\ &= \frac{1}{|H|} \sum_{h \in H} \rho(h) \end{aligned} \tag{2.24}$$

because $h_1 h_2 = h$ has $|H|$ solutions $(h_1, h_2) \in H \times H$.

With the right choice of basis, $\hat{f}(\rho)$ will be diagonal and consist of ones and zeros. The probability of measuring ρ will then be the sum of ones in the diagonal. As ρ is an irreducible representation of G, we need to take into account of the sum of the matrices $\rho(h)$ for all $h \in H$. Based of the assumption of the current theorem, we may only consider to evaluate ρ on H. According to the assumption, the probability of measuring ρ when $g \notin cH$ is zero. So, we may consider consider $\rho \downarrow_H^G$ instead of G.

Then, the Fourier transform of f at ρ is comprised of blocks, each corresponding to a representation in the decomposition of $\rho \downarrow_H^G$. Such as,

$$\sum_{h \in H} \rho(h) = U \begin{bmatrix} \sum_{h \in H} \sigma_1(h) & 0 & \cdots & 0 \\ 0 & \sum_{h \in H} \sigma_2(h) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{h \in H} \sigma_t(h) \end{bmatrix} U^\dagger \tag{2.25}$$

Here, U is an arbitrary unitary transformation, σ_i is an irreducible representation of H with possible repetition. Now, as a special case of the orthogonality relation among group characters, $\sum_{h \in H} \rho(h)$ is nonzero only when the irreducible representation is trivial, in which case, it is $|H|$.

So, the probability of measuring ρ is:

$$\left\| \hat{f}(\rho) \right\|^2 = \left\| \sqrt{\frac{d_\rho}{|G|}} \sum_{h \in H} \rho(h) \right\|^2$$

$$
\begin{aligned}
&= \frac{d_\rho}{|G|} \left\| \sum_{h \in H} \rho\left(h\right) \right\|^2 \\
&= \frac{d_\rho}{|G|} \mathrm{tr} \left(\sum_{h_1 \in H} \rho(h_1) \right) \left(\sum_{h_2 \in H} \rho(h_2) \right)^\dagger \\
&= \frac{d_\rho}{|G|} \mathrm{tr} \sum_{h_1, h_2 \in H} \rho(h_1 h_2^{-1}) \\
&= \frac{d_\rho}{|G|} \frac{1}{|H|} |H|^2 \langle \chi_\rho, \chi_{\mathbf{1}_H} \rangle_H \\
&= \frac{|H|}{|G|} d_\rho \langle \chi_\rho, \chi_{\mathbf{1}_H} \rangle_H
\end{aligned} \tag{2.26}
$$

I would like to note that, by definition, ρ appears $\langle \chi_\rho, \chi_{1_H} \rangle_H$ times in the decomposition of $\mathbf{1}_H$. □

I like to refer the readers to [51] for a review on the classical complexity of Fourier transformation of the symmetric groups.

2.5 Quantum Adiabatic Algorithms

The main result of Chapter 4 is the correctness proof on an quantum adiabatic algorithm. Hence, I discuss a few related concepts in this section. Adiabatic quantum computation, introduced by Farhi et al. in 2000 [66] is a continuous time quantum computational model. Aharonov et al. [3] proved that this model is equivalent to gate based quantum computation. The state of a quantum system can be described as a vector which is a function of a time, $|\Psi\left(t\right)\rangle$. The information about the system is expressed using a time dependent Hamiltonian, $H\left(t\right)$. The evolution of the Hamiltonian is governed by the Schrödinger equation as follows.

$$i\frac{d}{dt}|\Psi(t)\rangle = H(t)\,\Psi(t) \tag{2.27}$$

To solve a problem using an adiabatic algorithm, a quantum system is initialized with a beginning Hamiltonian, H_B which is easy to construct. Then the evolution is performed in such manner that the system stays always in it's instantaneous ground state. This is guaranteed by the adiabatic theorem of quantum mechanics given by Born et al [31]. When the evolution is complete, the ground state of the final Hamiltonian encodes the solution of the problem. For this reason it is also called the problem Hamiltonian, H_P. To guide the evolution we need a knob. If T is the total time of evolution and $s = t/T$ is the normalized time let the knob be $\tau(s)$. If $H(s)$ is the instantaneous Hamiltonian defined as follows.

$$H(s) = (1-s)\,H_B + sH_P \tag{2.28}$$

and it evolves according to the following equation.

$$\frac{d}{ds}|\Psi(s)\rangle = -i\tau(s)\,H(s)\,|\Psi(s)\rangle \tag{2.29}$$

The knob turns sufficiently slow if, as show in [180],

$$\tau(s) \gg \frac{||\frac{d}{ds}H(s)||_2}{g(s)^2} \tag{2.30}$$

Here, $g(s)$ is the difference between the first two eigenvalues. When it is difficult to find the ground state, it is convenient to use the minimum gap between the first two eigenvalues, $min_s g(s)$ and the maximum $max_s||\frac{d}{ds}H(s)||_2$.

In their original paper, Farhi et al [66] described the relation between the adiabatic evolution time and the energy gap as $\tau(s) \propto g(s)^{-2}$. But, as mentioned in [144], researchers have determined tighter bound in the relationship between these two quantities [7, 49, 101]. Most of these studies put the bound at $\tau(s) \propto g(s)^{-3}$.

Chapter 3

Graph Automorphism and Hidden Subgroup Algorithms

The main result of this chapter is the Theorem 12 and the Corollary 1 where I have shown that both the weak and strong quantum Fourier sampling fails for the classically trivial cycle graph automorphism problem. I have also shown how to determine non-trivial classes of graphs for which a quantum Fourier transform always fails to construct the automorphism group.

It has already been known that the graph automorphism problem is Turing-reducible to the graph isomorphism problem [113]. Hence, most of the research using the quantum hidden subgroup approach has been conducted to solve the graph isomorphism problem. It is obvious that any efficient quantum algorithm for the graph isomorphism problem would immediately solve the graph automorphism problem.

The hidden subgroup version of the graph isomorphism problem was first defined in [105]. We can reduce the n-vertex graph isomorphism problem for graphs of n vertices to the case of the hidden subgroup problem over the symmetric group S_{2n} or more specifically the wreath product $S_n \wr \mathbb{Z}_2$ where the hidden subgroup is promised to be either trivial or of order two [142]. I follow the scheme of the hidden subgroup problem as mentioned in [106] in the following definition. Erdős et al [61] have shown that the automorphism groups of most of the graphs are trivial. So,

although, the problem was defined for all simple undirected graphs in [105], I follow the example of [78] and limit our discussion to graphs with trivial automorphisms.

Definition 39 (Graph isomorphism as a hidden subgroup problem ($\mathbf{GI}_{\mathrm{HSP}}$))**.** *Let the $2n$ vertex graph $\Gamma = \Gamma_1 \sqcup \Gamma_2$ be the disjoint union of the two graphs Γ_1 and Γ_2 such that $Aut(\Gamma_1) = Aut(\Gamma_2) = \{e\}$. A map $\varphi : S_{2n} \to Mat(\mathbb{C}, N)$ [1] from the group S_{2n} is said to have hidden subgroup structure if there exists a subgroup H_φ of S_{2n}, called a hidden subgroup, an injection $\ell_\varphi : S_{2n}/H \to Mat(\mathbb{C}, N)$, called a hidden injection, such that the diagram*

$$\begin{array}{ccc} S_{2n} & \xrightarrow{\varphi} & Mat(\mathbb{C}, N) \\ \downarrow{\scriptstyle \nu} & \nearrow{\scriptstyle \ell_\varphi} & \\ S_{2n}/H & & \end{array}$$

is a commutative diagram, where S_{2n}/H_φ denotes the collection of right cosets of H_φ in S_{2n}, and where $\nu : S_{2n}/H_\varphi$ is the natural map of S_{2n} onto S_{2n}/H_φ. I refer to the group S_{2n} as the ambient group and to the set $Mat(\mathbb{C}, N)$ as the target set.

The hidden subgroup version of the graph isomorphism problem is to determine a hidden subgroup H of S_{2n} with the promise that H is either trivial or $|H| = 2$.

I also like to provide a formal definition for the hidden subgroup representation of the graph automorphism problem.

Definition 40 (Graph automorphism as a hidden subgroup problem ($\mathbf{GA}_{\mathrm{HSP}}$))**.** *For a graph Γ with n vertices, a map $\varphi : S_n \to Mat(\mathbb{C}, N)$ [2] from the group S_n is said to have hidden subgroup structure if there exists a subgroup $Aut(\Gamma)$ of S_n,*

[1] $\mathrm{Mat}(\mathbb{C}, N)$ is the algebra of all $N \times N$ matrices over the complex numbers $\mathbb{C}$.
[2] $\mathrm{Mat}(\mathbb{C}, N)$ is the algebra of all $N \times N$ matrices over the complex numbers $\mathbb{C}$.

called a hidden subgroup, an injection $\ell_\varphi : S_n/Aut(\Gamma) \to Mat(\mathbb{C}, N)$*, called a hidden injection, such that for each* $g \in Aut(\Gamma)$*,* $g(\Gamma) = \Gamma$ *and, the diagram*

$$\begin{array}{ccc} S_n & \xrightarrow{\varphi} & Mat(\mathbb{C}, N) \\ \downarrow{\scriptstyle \nu} & \nearrow_{\ell_\varphi} & \\ S_n/Aut(\Gamma) & & \end{array}$$

is commutative, where $S_n/Aut(\Gamma)$ *denotes the collection of right cosets of* $Aut(\Gamma)$ *in* S_n*, and where* $\nu : S_n/Aut(\Gamma)$ *is the natural map of* S_n *onto* $S_n/Aut(\Gamma)$*. I refer to the group* S_n *as the ambient group and to the set* $Mat(\mathbb{C}, N)$ *as the target set.*

The hidden subgroup version of the graph automorphism problem is to determine a hidden subgroup $Aut(\Gamma)$ *of* S_n *with the promise that* $Aut(\Gamma)$ *is either of trivial or non-trivial order depending on the type of* Γ*.*

3.1 Hidden subgroup algorithm for non-abelian groups

The hidden subgroup approach for both the graph isomorphism and automorphism problems require computing the quantum Fourier sampling of the symmetric group. This has been an active area of research since Shor invented the famous Shor's algorithm, a quantum hidden subgroup algorithm for abelian groups, for prime factorization [171]. While at this moment, there is no known efficient quantum hidden subgroup algorithm for symmetric groups, researchers have shed some light on why it has been so difficult to find them.

While surveys like [41], summarizes the advances made so far in hidden subgroup algorithms, I would like to review the negative results in this section to illus-

trate why this is a difficult problem. It is noteworthy that all the positive results, so far, have been demonstrated for the synthetically created product groups. While this approach may not have immediate practical application, I have used this idea of creating synthetic groups to generalize results. One of the first results for non-abelian hidden subgroup problems was presented by Roetteler et al [159]. They proved an efficient hidden subgroup algorithm for the wreath product $\mathbb{Z}_2^k \wr \mathbb{Z}_2$ which is a non-abelian group. Similarly, Ivanyos et al proved the existence of an efficient hidden subgroup algorithm for a more general non-abelian nil-2 groups [99]. Later Friedl et al generalized the result such that there are efficient hidden subgroup algorithms for the groups whose derived series have constant length and whose Abelian factor groups are each the direct product of an Abelian group of bounded exponent and one of polynomial size [69]. Ettinger et al showed that it is possible to reconstruct a subgroup hidden inside the dihedral group using finite number of queries [64]. This result was later generalized by Ettinger et al that arbitrary groups may be reconstructed using finite queries but they did not give any specific set of measurement [65].

In [136], Moore et al proved that although weak quantum Fourier sampling fails to determine the hidden subgroups of the non-abelian groups of the form $\mathbb{Z}_q \ltimes \mathbb{Z}_p$, where $q \mid (p-1)$ and $q = p/\text{polylog}(p)$, strong Fourier sampling is able to do that. Later on, Moore et al proved the existence of $polylog(|G|)$ sized quantum Fourier circuits for the groups like S_n, $H \wr S_n$, where $|H| = \text{poly}(n)$, and the Clifford groups [135, 138]. The authors also gave circuits of subexponential size for standard groups like $\text{GL}_n(q)$, $\text{SL}_n(q)$, $\text{PGL}_n(q)$, and $\text{PSL}_n(q)$, where q is a fixed prime power. Moore

et al have also presented a stronger result where they have shown that it is not possible to reconstruct a subgroup hidden inside the symmetric group with strong Fourier sampling and both arbitrary POVM and entangled measurement [141]. At the same time, the authors did not rule out the possibility of success using other possible measurements which is still an open question. Bacon et al proved that the so called *pretty good measurement* is optimal for the dihedral hidden subgroup problem [17]. Moore et al extended this result for the case where the hidden subgroup is a uniformly random conjugate of a given subgroup [139]. Moore et al eventually proved a more general results that strong quantum Fourier sampling can reconstruct q-hedral groups [137]. Alagic et al proved a general result that strong Fourier sampling fails to distinguish the subgroup of the power of a given non-abelian simple group [4]. Moore et al later proved that arbitrary entangled measurement on $\Omega(n \log n)$ coset states is necessary and sufficient to extract non-negligible information [140]. Similar result was also proved in [79] separately. Few years later, Moore et al proved a negative result that the *quantum sieve algorithm*, i.e. highly entangled measurements across $\Omega(n \log n)$ coset states, cannot solve the graph isomorphism problem [142].

It is important to point out that all the groups used in the previously mentioned results are conveniently chosen and synthetically created. Moreover, they are sporadic so it is not clear how we can extrapolate the knowledge for the symmetric groups. Most part of the history, the researchers have assumed that as the graph automorphism problem is Turing-reducible to the graph isomorphism problem, it would be sufficient to investigate the hidden subgroup representation of the graph

isomorphism problem. With all these unsuccessful attempts for the last couple of decades presented above, one may argue whether the hidden subgroup approach is the right way to attempt the graph isomorphism problem. If it is, there would have been a Turing-reduction from the hidden subgroup representation of the graph automorphism problem to the hidden subgroup representation of the graph isomorphism problem. Unfortunately, we are not aware of any such reduction using the quantum Turing machine. So, another way of looking at the problem is to understand the hidden subgroup complexity of the graph automorphism problem and compare the results with the results for graph isomorphism.

3.2 Quantum Fourier sampling for graph isomorphism

Most of the algorithms for the non-Abelian hidden subgroup problem use a black box for φ in the same way as in the Abelian hidden subgroup problem [106]. This has come to be known as the *standard method*. The standard method begins by preparing a uniform superposition over group elements [39]:

$$|S_n\rangle := \frac{1}{\sqrt{|S_n|}} \sum_{g \in S_n} |g\rangle \tag{3.1}$$

I then compute the value of $\varphi(g)$ in an ancilla register which creates the following state.

$$\frac{1}{\sqrt{|S_n|}} \sum_{g \in S_n} |g, \varphi(g)\rangle \tag{3.2}$$

Then I discard the second register by just tracing it out. If the outcome of the second register is s then the state is projected onto the uniform superposition of those $g \in S_n$ such that $\varphi(g) = s$. By definition of φ, it is some left coset of the hidden subgroup H. Since every coset contains the same number of elements, each left coset occurs with equal probability. Thus, the standard method produces the following coset state.

$$|gH\rangle := \frac{1}{\sqrt{|H|}} \sum_{h \in H} |gh\rangle \tag{3.3}$$

or equivalently as the following mixed *hidden subgroup state*.

$$\rho_H := \frac{1}{|S_n|} \sum_{g \in S_n} |gh\rangle\langle gh| \tag{3.4}$$

I have previously mentioned that φ maps the group elements of S_n to $\mathrm{Mat}(\mathbb{C}, N)$. Here, I present more information about the space $\mathrm{Mat}(\mathbb{C}, N)$. Let the complete set of irreducible representations of S_n (which are unique up to isomorphism) be $\hat{S}_n$. The Fourier transform is a unitary transformation from the group algebra, $\mathbb{C}S_n$, to a complex vector space whose basis vectors correspond to matrix elements of the

irreducible representations of S_n, $\oplus_{\rho \in \hat{S_n}} \left(\mathbb{C}^{d_\rho} \otimes \mathbb{C}^{d_\rho}\right)$. Here, d_ρ is the dimension of the irreducible representation ρ.

$|g\rangle$ is the basis vector chosen for the group element $g \in S_n$. There will be $n!$ such basis vectors of dimension $n! \times 1$. For a given group element g, we have a particular number of matrices one for each irreducible representation ρ. $|gh\rangle$ is expressed as $|\rho, j, k\rangle$ which is the basis vector labeled by the (j,k)-th element of the irreducible representation ρ of g.

When only ρ is measured from $|\rho, j, k\rangle$ it is called *weak Fourier sampling*. In *strong Fourier sampling*, j and k are also measured.

3.2.1 Weak Fourier sampling for $\mathbf{GI}_{\mathrm{HSP}}$

This section summarizes what we already know about the weak Fourier sampling when applied to the graph isomorphism problem. The weak Fourier sampling for $\mathbf{GI}_{\mathrm{HSP}}$ attempts to measure the label of irreducible representations of the symmetric group S_{2n} when the input graphs Γ_1 and Γ_2 are of n vertices. I assume $Aut\,(\Gamma_1) = Aut\,(\Gamma_2) = \{e\}$. If $\Gamma = \Gamma_1 \sqcup \Gamma_2$, one of the following two claims is true [80]. I would like to point it out that in [80], the authors derived the success probability of measuring the label of the irreducible representations for S_n. I will be deriving it for S_{2n} to keep consistency with the definition of $\mathbf{GI}_{\mathrm{HSP}}$.

- If $\Gamma_1 \not\approx \Gamma_2$, then $\mathrm{Aut}\,(\Gamma) = \{e\}$.
- If $\Gamma_1 \approx \Gamma_2$, then $\mathrm{Aut}\,(\Gamma) = \{e, \sigma\} = \mathbb{Z}_2$, where $\sigma \in S_{2n}$ is a permutation with n disjoint 2-cycles.

Algorithm 2 WEAK-QUANTUM-FOURIER-SAMPLING-S_{2n}

1: **procedure** WEAK-QUANTUM-FOURIER-SAMPLING-S_{2n}(a graph Γ such that either $\mathrm{Aut}(\Gamma) = \{e\}$ or $\mathrm{Aut}(\Gamma) = \{e, \sigma\}$)
2: Compute $\frac{1}{\sqrt{(2n)!}} \sum_{g \in S_{2n}} |g, \varphi(g)\rangle$
3: Compute $\sum_{\rho \in \hat{S_{2n}}} \sqrt{\frac{d_\rho}{(2n)!}} \sqrt{\frac{1}{|\mathrm{Aut}(\Gamma)|}} \left(\sum_{h \in \mathrm{Aut}(\Gamma)} \rho(ch) \right)_{i,j} |\rho, i, j\rangle$
4: Measure ρ as in tracing it out
5: **end procedure**

Let p_ρ be the probability of sampling ρ in Algorithm 2 when $\Gamma_1 \not\approx \Gamma_2$, and q_ρ when $\Gamma_1 \approx \Gamma_2$. So, the induced representation of $\mathrm{Aut}(\Gamma)$ to S_{2n}, $\mathrm{Ind}_{\mathrm{Aut}(\Gamma)}^{S_{2n}} 1$, is the regular representation. So, $\langle \chi_\rho | \chi_{\mathrm{Ind}_{\mathrm{Aut}(\Gamma)}^{S_{2n}} 1} \rangle$, the multiplicity of ρ in the regular representation, is d_ρ. Hence, $p_\rho = \frac{d_\rho^2}{(2n)!}$.

When $\Gamma_1 \approx \Gamma_2$, $\mathrm{Aut}(\Gamma) = \{e, \sigma\}$. In this case, the probability of measuring ρ,

$$q_\rho = \frac{|\mathrm{Aut}(\Gamma)|}{(2n)!} d_\rho \langle \chi_1 | \chi_\rho \rangle_{\mathrm{Aut}(\Gamma)} \tag{3.5}$$

$\mathrm{Aut}(\Gamma)$ has only two elements, e and σ, hence

$$\begin{aligned} \langle \chi_1 | \chi_\rho \rangle_{\mathrm{Aut}(\Gamma)} &= \frac{1}{2} (\chi_\rho(e) + \chi_\rho(\sigma)) \\ &= \frac{1}{2} (d_\rho + \chi_\rho(\sigma)) \end{aligned} \tag{3.6}$$

So,

$$
\begin{aligned}
q_\rho &= \frac{|\mathrm{Aut}\,(\Gamma)\,|}{(2n)!} d_\rho \frac{1}{2}\left(d_\rho + \chi_\rho\,(\sigma)\right) \\
&= \frac{2}{(2n)!} d_\rho \frac{1}{2}\left(d_\rho + \chi_\rho\,(\sigma)\right) \\
&= \frac{d_\rho}{(2n)!}\left(d_\rho + \chi_\rho\,(\sigma)\right) \qquad (3.7)
\end{aligned}
$$

So,

$$
\begin{aligned}
|p_\rho - q_\rho| &= |\frac{d_\rho^2}{(2n)!} - \frac{d_\rho}{(2n)!}\left(d_\rho + \chi_\rho\,(\sigma)\right)| \\
&= \frac{d_\rho}{(2n)!}|\chi_\rho\,(\sigma)\,| \qquad (3.8)
\end{aligned}
$$

I use the Murnaghan-Nakayama rule [175] to approximate $|\chi_\rho\,(\sigma)\,|$. The rule is given below.

Theorem 10 (The Murnaghan-Nakayama rule)**.** *Let c be a permutation with cycle structure $(c_1, \ldots, c_t)$, $c_1 \geq \ldots \geq c_t$. Then*

$$
\chi_\lambda\,(c) = \sum_{s_1,\ldots,s_t} (-1)^{v(s_1)} \ldots (-1)^{v(s_t)}\,, \qquad (3.9)
$$

where each s_i is a skew hook of length c_i of the diagram or partition λ after $s_1, \ldots, s_{i-1}$ have been removed, and $v\,(s_i)$ denotes the number of vertical steps in s_i.

The number of unordered decompositions for the diagram λ with $2n$ cells is $2^{4n} = 16^n$. For each unordered decomposition, the number of ordered decomposition is at most $\left(2\sqrt{2n}\right)^{\frac{2n}{2}} = \left(2\sqrt{2n}\right)^n$. So, by the Murnaghan-Nakayama rule, $\chi_\rho(\sigma) \leq 16^n \left(2\sqrt{2n}\right)^n$.

So,

$$\begin{aligned} |\chi_\rho(\sigma)| &\leq 16^n \left(2\sqrt{2n}\right)^n \\ &\leq 2^{4n} 2^n \sqrt{2}^n \left(\sqrt{n}\right)^n \\ &\leq 2^{O(n)} n^{\frac{n}{2}} \end{aligned} \tag{3.10}$$

Now I compute $|p_\rho - q_\rho|$ for all irreducible representations. So,

$$\begin{aligned} |p-q|_1 &= \sum_\rho |p_\rho - q_\rho| \\ &\leq \sum_\rho \frac{d_\rho}{(2n)!} 2^{O(n)} n^{\frac{n}{2}} \\ &\leq \sum_\rho \frac{\sqrt{(2n)!}}{(2n)!} 2^{O(n)} n^{\frac{n}{2}} \\ &\leq \frac{2^{O(n)} n^{\frac{n}{2}}}{\sqrt{(2n)!}} \\ &= \frac{2^{O(n)} n^{\frac{n}{2}} \sqrt{(2n)!}}{\sqrt{(2n)!}\sqrt{(2n)!}} \\ &= \frac{2^{O(n)} n^{\frac{n}{2}} \sqrt{(2n)!}}{(2n)!} \end{aligned}$$

$$
\begin{aligned}
&= \frac{2^{O(n)} n^{\frac{n}{2}} \sqrt{(2n)^{2n}}}{(2n)!} \\
&= \frac{2^{O(n)} n^{\frac{n}{2}} (2n)^{n}}{(2n)!} \\
&= \frac{2^{O(n)} n^{\frac{n}{2}} n^{n}}{(2n)!} \\
&= \frac{2^{O(n)} n^{\frac{3n}{2}}}{(2n)!} \\
&= \frac{2^{O(n)} n^{\frac{3n}{2}}}{(2n)^{(2n)}} \\
&= \frac{2^{O(n)} n^{\frac{3n}{2}}}{n^{2n}} \\
&= \frac{2^{O(n)}}{n^{\frac{n}{2}}} \\
&= \frac{2^{O(n)}}{2^{-\frac{n}{2}} n^{\frac{n}{2}}} \\
&= \frac{2^{O(n)}}{\frac{n}{2}^{\frac{n}{2}}} \\
&\leq \frac{2^{O(n)}}{\left(\frac{n}{2}\right)!} \\
&\lll 2^{-\Omega(n)}
\end{aligned}
\tag{3.11}
$$

3.3 Weak quantum Fourier sampling for cycle graph automorphism

The automorphism group of an n-cycle graphs is the dihedral group D_n of order $2n$. I seek to study the weak Fourier sampling of cycle graphs. To provide the background, I discuss the irreducible representations of the dihedral group D_n in this section.

Definition 41 (Cycle Graph)**.** *An n-cycle graph is a single cycle with n vertices.*

The automorphism group of an n-cycle graph is the dihedral group D_n which

is of order $2n$. If D_n is even, the group can be generated as $\langle (2 \quad n)(3 \quad n-1)\dots(\frac{n}{2}-1 \quad \frac{n}{2}+1), (1\dots n)\rangle$. If D_n is odd, the group can be generated as $\langle (2 \quad n)(3 \quad n-1)\dots(\frac{n-1}{2} \quad \frac{n+1}{2}), (1\dots n)\rangle$. This is a manifestation of the presentation $D_n = \langle x, y \mid x^n = y^2 = (xy)^2 = 1, yx = y^{-1}x\rangle$. The correspondence consists of $x = (1\dots n)$ and $y = (2 \quad n)(3 \quad n-1)\dots(\frac{n}{2}-1 \quad \frac{n}{2}+1)$ if n is even, and $y = (2 \quad n)(3 \quad n-1)\dots(\frac{n-1}{2} \quad \frac{n+1}{2})$ if n is odd. The orders of x and y are n and 2 respectively.

The order of the symmetric group S_n is $n!$. The order of a dihedral group D_n is $2n$. So, the index of D_n in S_n is $\frac{n!}{2n} \approx \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{2n}$.

I would like to mention an important theorem proved in [42] in the following paragraph.

Theorem 11. *Every subgroup of D_n is cyclic or dihedral. A complete listing of the subgroups is as follows:*

- *$\langle x^d\rangle$, where $d \mid n$, with index $2d$,*
- *$\langle x^d, x^i y\rangle$, where $d \mid n$ and $0 \le i \le d-1$, with index d.*

Every subgroup of D_n occurs exactly once in this listing.

In this theorem, subgroups of the first type are cyclic and subgroups of the second type are dihedral: $\langle x^d\rangle \cong \mathbf{Z}/(n/d)$ and $\langle x^d, x^i y\rangle \cong D_{n/d}$.

Based on Theorem 11, I would like to add the following remark.

Remark 2. *The order of the subgroups $\langle x^d\rangle$ and $\langle x^d, x^i y\rangle$ are $\frac{n}{d}$ and $\frac{2n}{d}$ respectively.*

Every element of D_n is either x^i or yx^i for $0 \le i < n$. The conjugacy classes are small enough in number to be enumerated.

$$\begin{array}{lllll} \text{Conjugate} & x^i & \text{by} & x^j & : & (x^j)x^i(x^{-j}) = x^i \\ & x^i & \text{by} & yx^j & : & (yx^j)x^i(x^{-j}y) = yx^iy = x^{-i} \\ & yr^i & \text{by} & r^j & : & (r^j)yr^i(r^{-j}) = yr^{-j}r^ir^{-j} = yr^{i-2j} \\ & yx^i & \text{by} & yx^j & : & (yx^j)yx^i(x^{-j}y) = x^{i-2j}y = yx^{2j-i} \end{array}$$

The set of rotations decomposes into inverse pairs, $\left\{x^i, (x^i)^{-1}\right\}$. So, the classes are $\{1\}, \{x, x^{n-1}\}, \{x^2, x^{n-2}\}, \ldots$. When n is even, there are $\frac{n}{2}+1$, and when n is odd, there are $\frac{n+1}{2}$ conjugacy classes.

yx is conjugate to $yx^3, yx^5, \ldots$ while y is conjugate to $yx^2, yx^4, \ldots$. If n is even, these two sets are disjoint. However, yx is conjugate to yx^{n-1} (via x), so if n is odd, all the non trivial reflections are in one conjugacy class.

So, the total number of conjugacy classes are as follows. If n is even, the total number of conjugacy classes is $\left(\frac{n}{2}+1\right)+2 = \frac{n}{2}+3$. If n is odd, the total number of conjugacy classes is $\frac{n+1}{2}+1 = \frac{n+3}{2}$.

The commutators of D_n,

$$\begin{aligned} \left[x^i, yx^j\right] &= x^{-i}\left(yx^j\right)x^i\left(yx^j\right) \\ &= yx^{2i+j}yx^j \\ &= \left(x^i\right)^2 \end{aligned} \tag{3.12}$$

So, the commutators generate the subgroup of squares of rotations. When n is even, only half the rotations are squares, hence $G/[G,G]$ is of order 4. When n is odd, all rotations are squares, hence $G/[G,G]$ is of order 2. The number of

one dimensional irreducible representations is the order of $G/[G,G]$. So, when n is even, there are 4 one dimensional representations and when n is odd, there are 2 one dimensional representations.

I enumerate the representations as follows.

- When n is even:
  - The trivial representation, sending all group elements to the 1×1 matrix $\begin{pmatrix} 1 \end{pmatrix}$.
  - The representation, sending all elements in $\langle x \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and all elements outside $\langle x \rangle$ to $\begin{pmatrix} -1 \end{pmatrix}$.
  - The representation, sending all elements in $\langle x^2, y \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and x to $\begin{pmatrix} -1 \end{pmatrix}$.
  - The representation, sending all elements in $\langle x^2, xy \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and x to $\begin{pmatrix} -1 \end{pmatrix}$.
- When n is odd:
  - The trivial representation, sending all group elements to the 1×1 matrix $\begin{pmatrix} 1 \end{pmatrix}$.
  - The representation, sending all elements in $\langle x \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and all elements outside $\langle x \rangle$ to $\begin{pmatrix} -1 \end{pmatrix}$.

Now, I determine the two dimensional irreducible representations. There is an obvious subgroup $\{1, x, \ldots, x^{n-1}\}$ which is a cyclic group of order n, let's call

it $C_n < D_n$. Since C_n is abelian, it has n irreducible 1-dimensional representations over $\mathbb{C}$, namely

$$x \mapsto e^{2\pi ki/n}, \qquad 0 \leq k < n \tag{3.13}$$

which captures the idea of rotating by an angle of $2\pi k/n$. I induce these easily-described representations to D_n in order to find some possibly new representations.

We have a representation W of a subgroup $H \leq G$ (i.e. an H-linear action on W), the induced representation of W is

$$\bigoplus_{g \in G/H} g \cdot W \tag{3.14}$$

where g ranges over a set of representatives of G/H.

The induced representation of C_n to D_n for fixed k is straight forward since D_n/C_n has representatives $\{1, y\}$. So we just need to describe the D_n-vector space $\mathbb{C} \oplus y \cdot \mathbb{C}$ where $\mathbb{C}$ has basis consisting only of w_1. Now, the D_n action turns into an actual matrix representation.

Specifically, we can find out how x acts on each summand using our representation of C_n:

$$x \cdot w_1 = e^{2\pi ki/n} w_1, \text{ and} \tag{3.15}$$

$$\begin{aligned} x \cdot (y \cdot w_1) &= xy \cdot w_1 \\ &= yx^{-1} \cdot w_1 = e^{-2\pi ki/n} y \cdot w_1 \end{aligned} \tag{3.16}$$

which tells us that x acts by the matrix $\begin{pmatrix} e^{2\pi ki/n} & 0 \\ 0 & e^{-2\pi ki/n} \end{pmatrix}$.

We can also figure out how y acts. y obviously takes w_1 to $y \cdot w_1$, and y takes $y \cdot w_1$ to $y^2 w_1 = w_1$, so y simply interchanges the two summands. This tells us that y acts by the matrix $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

Here, I list the k-th two dimensional irreducible representations for the general group elements.

$$x \mapsto \begin{pmatrix} e^{\frac{2\pi ik}{n}} & 0 \\ 0 & e^{-\frac{2\pi ik}{n}} \end{pmatrix}$$

$$x^l \mapsto \begin{pmatrix} e^{\frac{2\pi ikl}{n}} & 0 \\ 0 & e^{-\frac{2\pi ikl}{n}} \end{pmatrix}$$

$$y \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$x^l y \mapsto \begin{pmatrix} 0 & e^{\frac{2\pi i k l}{n}} \\ e^{-\frac{2\pi i k l}{n}} & 0 \end{pmatrix} \tag{3.17}$$

We observe that, $0 \leq l \leq n-1$. Both $l = 0$ and $l = n$ determine the identity matrix to which the identity element, e, is mapped. When n is even, the k-th and $(n-k)$-th representations are equivalent, hence we get distinct representations only for $k = 1, 2, \ldots, \frac{n-2}{2}$. The representations for $k = 0$ and $k = \frac{n}{2}$ are not irreducible and they decompose into one dimensional representations. On the other hand, when, n is odd, there are $\frac{n-1}{2}$ irreducible representations.

Using the previous calculations, we can compute the total number of irreducible representations for D_n. When n is even, the total number is $\frac{n-2}{2} + 4 = \frac{n}{2} + 3$. When n is odd, it is $\frac{n-1}{2} + 2 = \frac{n+3}{2}$. At this point, I would like to add the following remark regarding the characters of the irreducible two dimensional representations.

Remark 3. *The characters of the representations of the elements of type y and $x^l y$ are both zeros. The representations of the elements of type x have the same character, $2\cos\left(\frac{2\pi k}{n}\right)$. Finally, the representations of the elements of type x^l have the same character, $2\cos\left(\frac{2\pi k l}{n}\right)$.*

Our interest with the dihedral group D_n of order $2n$ is based on the fact that it is the automorphism group of the n-cycle graph. A p-group is a group where the order of every group element is a power of the prime p. So, D_n can be a p-group

only when $n = 2^m$, when $m \in \mathbb{Z}$,because that is when $x^{2^m} = y^{2^1} = 1$ for $p = 2$.

The restrictions from the irreducible representations of S_n to D_n is straight forward. For any irreducible representation ρ of S_n, its restriction to D_n is $\rho(h)$ for all $h \in D_n$. This new representation may not be necessarily irreducible.

The inductions from the irreducible representations of D_n to S_n can be computed following the Definition 35. This new representation may also not be necessarily irreducible.

Following [80], I seek to compute the induced representation of $\mathbf{1}_{D_n}$ which is the trivial representation of the dihedral group D_n. As a prerequisite we need to compute a transversal for the left cosets of D_n which can be done by any of the two algorithms presented in Section 4.6.7 of [94]. The computation of a transversal of a group requires the computation of the base of a group which can be computed in polynomial time using the Schreier-Sims algorithm [169, 173, 112].

As I have mentioned previously in the current section, there are $l = \frac{n!}{2n} \approx \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{2n}$ cosets for D_n in S_n. This will also be the number of elements in a transversal for the left cosets of D_n in S_n. Let the transversal be $t_1, \ldots, t_l$. So, $S_n = t_1 D_n \sqcup \ldots \sqcup t_l D_n$. We can compute $\mathbf{1}_{D_n} \uparrow_{D_n}^{S_n}$ following Definition 35.

It would be instructive to discuss the character table of S_n and D_n here.

Computing the character table of S_n starts from computing the partitions $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r$ given $\sum_i \lambda = n$. These partitions can be partially ordered as follows. If there are two partitions $\lambda = (\lambda_1 \geq \lambda_2 \geq \ldots)$ and $\mu = (\mu_1 \geq \mu_2 \geq \ldots)$,

$\lambda \geq \mu$ if,

$$\begin{aligned} \lambda_1 &\geq \mu_1 \\ \lambda_1 + \lambda_2 &\geq \mu_1 + \mu_2 \\ \lambda_1 + \lambda_2 + \lambda_3 &\geq \mu_1 + \mu_2 + \mu_3 \\ &\vdots \end{aligned} \tag{3.18}$$

The columns of the character table are indexed by the conjugacy classes such that the partitions are arranged in increasing order. On the other hand, the rows are indexed by the characters in the decreasing order of the partitions. Each cell in the table then contains the corresponding character.

I provide a number of equivalent general procedure for computing the character table of any symmetric group S_n improvising from [77].

The most straight forward way [162] to compute the character table is given in Algorithm 3.

Algorithm 3 CHARACTER-TABLE-1

1: **procedure** CHARACTER-TABLE-1(S_n)
2: Determine all the partitions of n which will also infer the conjugacy classes.
3: Enumerate all group elements and cluster them based on their cycle types. These clusters will coincide with the conjugacy classes.
4: For each class, compute the irreducible representation for each group element.
5: For each class, determine the character of the irreducible representation. All group elements of the same cycle type will have the same character.
6: Populate the table with the characters following the prescribed order of the partitions for both column and rows.
7: **end procedure**

Enumeration of conjugacy classes becomes tedious when we are discussing

about groups larger than S_5. By using the *Murnaghan-Nakayama rule* , we can simplify the process even for larger groups [175] as shown by [77] in Algorithm 4.

Algorithm 4 CHARACTER-TABLE-2

1: **procedure** CHARACTER-TABLE-2(S_n)
2: The conjugacy classes of S_n are the permutations having a fixed number of cycles of each length, corresponding to a partition of n called the shape of the permutation. Since, the characters of a group are constant on its conjugacy classes, index the columns of the character table by these partitions. The partitions are arranged in an increasing order.
3: There are precisely as many irreducible characters as conjugacy classes, so we can also index the irreducible characters by the partitions of n. Represent each partition as a Young diagram and write them, or the characters directly down, the left of the table in a decreasing order of the partitions.
▷ The Murnaghan-Nakayama Rule
4: Calculate the entry in row λ and column μ. Define a filling of λ with content μ to be a way of writing a number in each square of λ such that the numbers are weakly increasing along each row and column and there are exactly μ_i squares labeled i for each i.
5: Consider all fillings of λ with content μ such that for each label i, the squares labeled i form a connected skew tableaux that does not contain a 2×2 square. Such a tableaux is called a *border-strip tableaux*.
6: For each label in the tableau, define the height of the corresponding border strip to be one less than the number of rows of the border strip. Weight the tableau by $(-1)^s$ where s is the sum of the heights of the border strips that compose the tableau.
7: The entry in the character table is simply the sum of these weights.
8: **end procedure**

I apply Lemma 1 to obtain the Frobenius reciprocity for $D_n < S_n$.

$$\langle \chi \uparrow^{S_n}_{D_n \mathbf{1}_{D_n}}, \chi_\rho \rangle_{S_n} = \langle \chi_{\mathbf{1}_{D_n}}, \chi_\rho \downarrow^{S_n}_{D_n} \rangle_{D_n} \tag{3.19}$$

where the left inner product is calculated in S_n and the right one in D_n. So, if we need to determine $\langle \chi_{\mathbf{1}_{D_n}}, \chi_\rho \downarrow^{S_n}_{D_n} \rangle_{D_n}$, it would be sufficient to determine $\langle \chi \uparrow^{S_n}_{D_n \mathbf{1}_{D_n}}, \chi_\rho \rangle_{S_n}$.

Following the Definition 30,

$$\begin{aligned}\langle \chi \uparrow^{S_n}_{D_n \mathbf{1}_{D_n}}, \chi_\rho \rangle_{S_n} &= \sum_{g_i \in S_n} \chi \uparrow^{S_n}_{D_n \mathbf{1}_{D_n}} (g_i) \chi^\dagger_\rho(g_i) \\ &= \sum_{g_i \in S_n} \left(\sum_{i=1}^{n} \delta_{\chi \uparrow^{S_n}_{D_n \mathbf{1}_{D_n}} (g_i)_{i,i}} \right) \chi^\dagger_\rho(g_i) \end{aligned} \tag{3.20}$$

I apply Theorem 9 to determine the probabilities of measuring the irreducible representations of D_n through proving a few lemmas.

Lemma 2. *The probability of measuring the labels of one dimensional irreducible representations of D_n is zero for non-trivial representations.*

Proof of Lemma 2. First, I assume that n is even. So, there are 4 one dimensional representations and $\frac{n-2}{2}$ two dimensional representations. I compute the probability of measuring the one dimensional representations. Let us denote them as ρ_1, ρ_2, ρ_3, and ρ_4. So, their dimensions are all the same i.e. $d_{\rho_1} = d_{\rho_2} = d_{\rho_3} = d_{\rho_4} = 1$. Let the probability of measuring ρ_i be p_{ρ_i}. So, following Theorem 9,

$$\begin{aligned} p_{\rho_i} &= \frac{|D_n|}{|S_n|} d_{\rho_i} \langle \chi_{\rho_i}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} \\ &= \frac{2n}{n!} \langle \chi_{\rho_i}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} \end{aligned} \tag{3.21}$$

D_n has $2n$ elements. So,

$$\begin{aligned}\langle \chi_{\rho_i}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} &= \frac{1}{|D_n|} \sum_{g \in D_n} \chi_{\rho_i}(g)\, \chi^{\dagger}_{\mathbf{1}_{D_n}}(g) \\ &= \frac{1}{2n} \sum_{g \in D_n} \chi_{\rho_i}(g) \end{aligned} \tag{3.22}$$

When $i = 1$, ρ_1 is the trivial representation which sends all group elements to the 1×1 matrix $\begin{pmatrix} 1 \end{pmatrix}$. The probability of measuring this representation is given below.

$$\begin{aligned}\langle \chi_{\rho_1}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} &= \frac{1}{2n} \sum_{g \in D_n} \chi_{\rho_1}(g) \\ &= 1 \end{aligned} \tag{3.23}$$

So,

$$\begin{aligned} p_{\rho_1} &= \frac{2n}{n!} \langle \chi_{\rho_1}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} \\ &= \frac{2n}{n!} \end{aligned} \tag{3.24}$$

When $i = 2$, ρ_2 is the representation which sends all elements in $\langle x \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and all elements outside $\langle x \rangle$ to $\begin{pmatrix} -1 \end{pmatrix}$. The probability of measuring this representation is given below.

$$\langle \chi_{\rho_2}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} = \frac{1}{2n} \sum_{g \in D_n} \chi_{\rho_2}(g) \tag{3.25}$$

As observed from the presentation of D_n, the number of elements in $\langle x \rangle$ is n. So, n elements will be mapped to 1 and n elements will e mapped to -1. So,

$$\begin{aligned} \langle \chi_{\rho_2}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} &= \frac{1}{2n} \left(n(1) + n(-1) \right) \\ &= 0 \end{aligned} \tag{3.26}$$

So,

$$\begin{aligned} p_{\rho_2} &= \frac{2n}{n!} \langle \chi_{\rho_2}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} \\ &= 0 \end{aligned} \tag{3.27}$$

So, the weak Fourier sampling will not be able to determine the labels of the sign representation which sends all elements in $\langle x \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and all elements outside $\langle x \rangle$ to $\begin{pmatrix} -1 \end{pmatrix}$.

When $i = 3$, ρ_3 is the representation which sends all elements in $\langle x^2, y \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and x to $\begin{pmatrix} -1 \end{pmatrix}$. The probability of measuring this representation is given below.

$$\langle \chi_{\rho_3}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} = \frac{1}{2n} \sum_{g \in D_n} \chi_{\rho_3}(g) \tag{3.28}$$

Following Remark 2, ρ_3 sends n elements of D_n to $\begin{pmatrix} 1 \end{pmatrix}$ and $2n - n = n$ elements of D_n to $\begin{pmatrix} -1 \end{pmatrix}$. So,

$$\langle \chi_{\rho_3}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} = 0 \tag{3.29}$$

So, the weak Fourier sampling will not be able to determine the labels of the sign representation which sends all elements in $\langle x^2, y \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and all elements outside $\langle x \rangle$ to $\begin{pmatrix} -1 \end{pmatrix}$.

When $i = 4$, ρ_4 is the representation which sends all elements in $\langle x^2, xy \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and x to $\begin{pmatrix} -1 \end{pmatrix}$. The probability of measuring this representation is given below.

$$\langle \chi_{\rho_4}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} = \frac{1}{2n} \sum_{g \in D_n} \chi_{\rho_4}(g) \tag{3.30}$$

Following Remark 2, ρ_4 sends n elements of D_n to $\begin{pmatrix} 1 \end{pmatrix}$ and $2n - n = n$ elements of D_n to $\begin{pmatrix} -1 \end{pmatrix}$. So,

$$\langle \chi_{\rho_4}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} = 0 \tag{3.31}$$

So, the weak Fourier sampling will not be able to determine the labels of the sign representation which sends all elements in $\langle x^2, xy \rangle$ to $\begin{pmatrix} 1 \end{pmatrix}$ and all elements outside $\langle x \rangle$ to $\begin{pmatrix} -1 \end{pmatrix}$.

The case when n is odd may be considered as a special case of when n is even and show that only the trivial representation can be sampled with non-zero probability. □

Now, I would like to compute the probability of measuring the labels of two dimensional irreducible representations in weak Fourier sampling.

Lemma 3. *The probability of measuring the labels of two dimensional irreducible representations of D_n is always zero.*

Proof of Lemma 3. When n is even, there are $\frac{n-2}{2}$ such irreducible representations. Let the irreducible representations be denoted as $\sigma_1, \sigma_2, \ldots, \sigma_k, \ldots, \sigma_{\frac{n-2}{2}}$.

The probability of measuring σ_k is given below.

$$\langle \chi_{\sigma_k}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} = \frac{1}{2n} \sum_{g \in D_n} \chi_{\sigma_k}(g) \tag{3.32}$$

Following Remark 3, σ_k maps n number of elements to the matrices for which the characters of the representations are zero. For the rest n number of the group

elements, the character is $2\cos\left(\frac{2\pi kl}{n}\right)$ where $0 \leq l \leq n-1$. So,

$$\begin{aligned}\langle \chi_{\sigma_k}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} &= \frac{1}{2n} \sum_{l=0}^{n-1} 2\cos\left(\frac{2\pi kl}{n}\right) \\ &= \frac{1}{n} \sum_{l=0}^{n-1} \cos\left(\frac{2\pi kl}{n}\right) \end{aligned} \tag{3.33}$$

I would like to mention the following formula for the sum of series of cosines when they are in arithmetic progression as they have been proven in both [111] and [93].

$$\sum_{l=1}^{n} \cos(l\theta) = \frac{\sin(n\theta/2)}{\sin(\theta/2)} \cos((n+1)\theta/2), \quad \sin(\theta/2) \neq 0. \tag{3.34}$$

When $\theta = \dfrac{2\pi k}{n}$,

$$\sum_{l=1}^{n} \cos\left(\frac{2\pi kl}{n}\right) = \frac{\sin(\pi k)}{\sin(\pi k/n)} \cos((n+1)\pi k/n) = 0 \tag{3.35}$$

If we change the interval of l from $[1, n]$ to $[0, n-1]$ the sum still remains zero.

$$\sum_{l=0}^{n-1} \cos\left(\frac{2\pi kl}{n}\right) = \cos(0) + 0 - \cos(2\pi k) = 0. \tag{3.36}$$

So, the probability of measuring the labels of the two dimensional irreducible representations is:

$$\begin{aligned}\langle \chi_{\sigma_k}, \chi_{\mathbf{1}_{D_n}} \rangle_{D_n} &= \frac{1}{n} \sum_{l=0}^{n-1} \cos\left(\frac{2\pi kl}{n}\right) \\ &= 0 \end{aligned} \tag{3.37}$$

The case of n being odd can be considered as a special case of n being even and the same result can be proved. □

So, the weak Fourier sampling algorithm cannot determine the labels of any of the two dimensional irreducible representations. To summarize, weak Fourier sampling cannot determine any irreducible representation other than the trivial one. So, it cannot determine the automorphism group of the cycle graph which is a trivial problem in the classical paradigm. Lemmas 2 and 3 may be consolidated into the following theorem.

Theorem 12. *Weak quantum Fourier sampling fails to solve the cycle graph automorphism problem.*

Proof. Follows directly from the proofs of the Lemmas 2 and 3. □

I observe that the success probability of strong Fourier sampling is a conditional probability which depends on the success probability of measuring the labels of representation. As the success probability of measuring the non-trivial represen-

tations of D_n is zero, the success probability of measuring their individual matrix elements is also zero.

Corollary 1. *Strong quantum Fourier sampling fails to solve the cycle graph automorphism problem.*

3.4 Non-trivial graph automorphism problems

In Section 3.3, I have shown that the weak Fourier sampling fails to determine the automorphism group of the cycle graphs. It is natural to ask whether same is the case for the graphs which have the dihedral group as a subgroup in it's automorphism group. I answer to the question in the affirmative. This is based of the following theorem reproduced from [53] and also mentioned as Theorem 10 in Section 3.2 in [170].

Theorem 13. *The direct product of two irreducible representations of groups H and K yields an irreducible representation of the direct product group so that all irreducible representations of the direct product group can be generated from the irreducible representations of the original groups before they are joined.*

While the formal proof is omitted, I would like to present a sketch of the proof here.

Let F be a field and G be a group which is expressed as the direct product $G = H \times K$. Let ρ and σ be representations of H and K over F, respectively. Then a corresponding representation of G over F may be constructed from ρ and σ by using tensor products.

Suppose that ρ and σ arise from an $FH-$module M and an $FK-$module N, respectively. Form the tensor product $T = M \otimes_F N$ and make T into a right $FG-$module by the rule $(a \otimes b)(x, y) = (ax) \otimes (by)$, where $a \in M$, $b \in N$, $x \in H$ and $y \in K$. Then T affords an $F-$representation $\rho\sharp\sigma$ called the Kronecker (or outer tensor) product of ρ and σ. The degree of $\rho\sharp\sigma$ equals the product of the degrees of ρ and σ.

It is easy to show that if ρ has character χ and σ has character ψ, the character ϕ of $\rho\sharp\sigma$ is given by $(x, y)\phi = (x)\chi(y)\psi$. If F is an algebraically closed field, G is finite, the characteristic of F does not divide the order of G and $\{\rho_1, \ldots, \rho_h\}$ and $\{\sigma_1, \ldots, \sigma_k\}$ are complete sets of inequivalent irreducible $F-$representations of H and K, then the $\rho_i\sharp\sigma_r$, for $i = 1, \ldots h$ and $r = 1, \ldots, k$, form a complete set of inequivalent irreducible $F-$representations of G.

With the discussion above, a complete set of inequivalent irreducible $\mathbb{C}-$characters of $D_n \times G$, where G is a finite group, can be constructed by the rule $(x, y)\phi_j = (x)\chi_i(y)\psi_r$, where $i = 1, \ldots h$, $r = 1, \ldots, k$, $x \in D_n$, $y \in G$, and $\{\chi_1, \ldots, \chi_h\}$ and $\{\psi_1, \ldots, \psi_k\}$ are complete sets of inequivalent irreducible $\mathbb{C}-$characters of D_n and G.

Before going further, I would like to present the Frucht's theorem below [70].

Theorem 14. *Every abstract group is isomorphic to the automorphism group of some graph.*

So, any group which is a product of D_n and a finite group G is isomorphic to the automorphism group of some finite graph. It has also been shown in [11]

that every finite group as the group of symmetries of a strongly regular graph. It indicates that there is a class of strongly regular graph whose automorphism group is isomorphic to $D_n \times G$. According to Theorem 13 and the result presented in the previous section, I argue that quantum Fourier sampling fails to construct the automorphism group of a subclass of strongly regular graphs.

Another example may be the Cayley graph automorphism problem [186]. It is well known that the automorphism group of the Cayley graph $\mathrm{Aut}\left(C\left(G, X\right)\right)$ of a group G over a generating set X contain an isomorphic copy of G acting via left translations [100]. In that case, the automorphism group of the Cayley graph of a dihedral group D_n contains D_n as a subgroup. So, following the result of the previous section, quantum Fourier sampling fails to compute the automorphism group of $\mathrm{Aut}\left(C\left(G, X\right)\right)$.

I would like to draw the final case from the theory of *universal structures*. A class $\mathcal{C}$ of structures is called *universal* if every finite group is the automorphism group of a structure in $\mathcal{C}$ [37]. A series of efforts by Frucht, Sabidussi, Mendelsohn, Babai, Kantor, and others [37] has shown the following classes of graphs to be universal - graphs of valency k for any fixed $k > 2$ [71]; bipartite graphs; strongly regular graphs [130]; Hamiltonian graphs [161]; k-connected graphs [161], for $k > 0$; k-chromatic graphs, for $k > 1$; switching classes of graphs; lattices [24]; projective planes (possibly infinite); and Steiner triple systems [131]; and symmetric designs (BIBDs). It indicates that each of these classes has at least one graph which has its automorphism group isomorphic to D_n. So, quantum Fourier sampling will fail to compute the automorphism group of each of these cases.

3.5 Summary

We have seen that, while $\mathbf{GI}_{\mathrm{HSP}}$ is equivalent to determining order 2 subgroup of a symmetric group, $\mathbf{GA}_{\mathrm{HSP}}$ is equivalent to determining a hidden subgroup of higher order. I have identified a class of graph automorphism problem for which the quantum Fourier transform algorithm always fails. I have also shown how we can determine non-trivial classes of graphs for which the same algorithm always fails. With these negative results, one may be interested to ask whether the hidden subgroup representation is a practical representation of the graph isomorphism and automorphism problems in quantum regime.

Chapter 4

Graph Isomorphism and Quantum Adiabatic Algorithms

4.1 Adiabatic algorithm for graph isomorphism

The main result of this chapter is the correctness proof of the Hen-Young quantum adiabatic algorithm for graph isomorphism. Adiabatic algorithms for graph isomorphism is a relatively new area of research which started with the publication [91] by Hen et al. Later on two more publications proposed two more adiabatic algorithms ([73, 90]). These three algorithms are highly diverse in terms of the construction of the driver Hamiltonian, encoding of the solution, locality, and the size of the quantum system. In this section, I focus on the earliest Hen-Young algorithm [91]. While the authors presented the implementation of the algorithm for a few graph classes, the proof of the correctness of the algorithm was yet to be presented. Our main result in this chapter is the proof of the correctness of the Hen-Young algorithm.

4.2 The Hen-Young algorithm

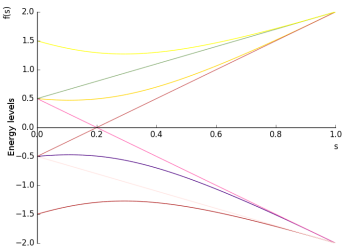
I convert the original descriptive version of the algorithm from [91] into a pseudo-code outlined in Algorithm 5.

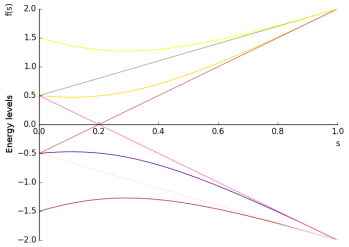
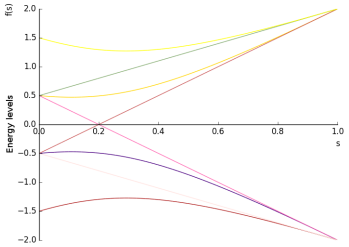
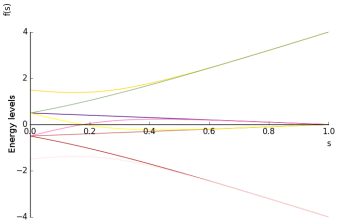
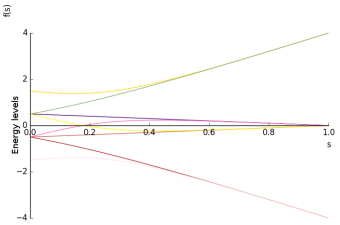
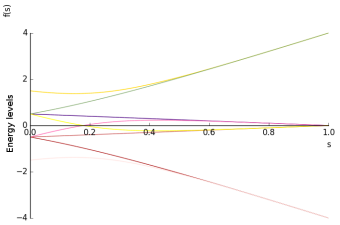
Algorithm 5 HEN-YOUNG

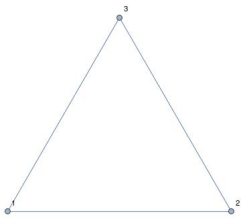
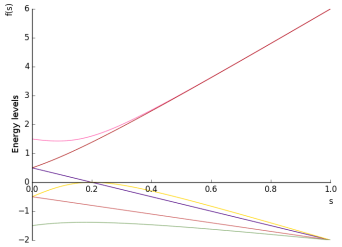
1: **procedure** HEN-YOUNG(Γ_1, Γ_2) ▷ Define driver Hamiltonians for Γ_1 and Γ_2
2: Construct driver Hamiltonians $\hat{H}_{d_1} = \hat{H}_{d_2} = \frac{1}{2}\sum_i^n \sigma_i^x$ ▷ Define problem Hamiltonians for Γ_1 and Γ_2
3: Construct the problem Hamiltonian $\hat{H}_{p_1} = \sum_{\langle ij\rangle\in\Gamma_1} \sigma_i^z\sigma_j^z$
4: Construct the problem Hamiltonian $\hat{H}_{p_2} = \sum_{\langle ij\rangle\in\Gamma_2} \sigma_i^z\sigma_j^z$ ▷ Construct the adiabatic Hamiltonians
5: Construct the adiabatic Hamiltonian $\hat{H}_1(s) = (1-s)\hat{H}_{d_1} + s\hat{H}_{p_1}$
6: Construct the adiabatic Hamiltonian $\hat{H}_2(s) = (1-s)\hat{H}_{d_2} + s\hat{H}_{p_2}$
7: Adiabatically evolve $\hat{H}_1(s)$ and $\hat{H}_2(s)$
8: Measure average energy, spin-glass order parameter, and x-magnetization
9: **if** None of the quantity matches **then** False
10: **else**True
11: **end if**
12: **end procedure**

The success of the Hen-Young algorithm depends on the two adiabatic evolutions conducted in the step 7 of Algorithm 5. It implied that the time $\tau(s)$ needed to finish the evolution should be proportional to $g(s)^{-3}$ for the algorithm to succeed. To my knowledge, this relation is yet to be proven. To illustrate the idea, I study the adiabatic evolutions of all 3-vertex graphs (except the disconnected graph) in Table 4.1.

Table 4.1: Adiabatic evolution of the Hamiltonians of 3-vertex graphs

Graph	Ground states of H_P	Spectrum of $H(s)$
	$\|010\rangle$, $\|011\rangle$, $\|100\rangle$, $\|101\rangle$	

	$\|001\rangle$, $\|010\rangle$, $\|101\rangle$, $\|110\rangle$	
	$\|001\rangle$, $\|011\rangle$, $\|100\rangle$, $\|110\rangle$	
	$\|010\rangle$, $\|101\rangle$	
	$\|001\rangle$, $\|110\rangle$	
	$\|011\rangle$, $\|100\rangle$	

	$\|001\rangle$, $\|010\rangle$, $\|011\rangle$, $\|100\rangle$, $\|101\rangle$, $\|110\rangle$	

For all these graphs, the driver Hamiltonian is always the same. In the original algorithm, the authors mentioned that the problem Hamiltonians reflect the structure of the graph. But, so far there is no rigorous proof that this structural information is sufficient for the adiabatic algorithm to succeed. Let us consider the fourth graph from the Table 4.1. In this graph, the valency of the vertex 2 is two. The ground state is degenerate with multiplicity two. The vertices with the same valency retains the same spin value. We may also consider the case of the two isomorphic triplets i.e. the first three graphs and the second three graphs from Table 4.1.

I would also like to compute the problem Hamiltonians for the Isomorphic pair in the Figure 5.7 and the non-Isomorphic pair in the Figure 5.8.

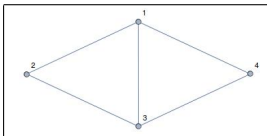
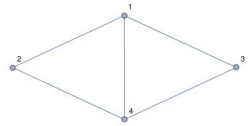
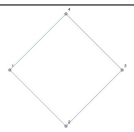
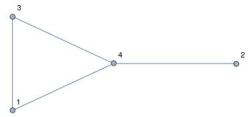
Graph	Ground states of H_P	Eigenvalues
	$\|0101\rangle$, $\|1010\rangle$	-6
	$\|0110\rangle$, $\|1001\rangle$	-6
	$\|0101\rangle$, $\|1010\rangle$	-8
	$\|0001\rangle$, $\|0011\rangle$, $\|0110\rangle$, $\|1001\rangle$, $\|1100\rangle$, $\|1110\rangle$	-4

Table 4.2: Problem Hamiltonian of 4-vertex isomorphic and non-isomorphic graphs

From Tables 4.1 and 4.2, I observe that the ground state of the problem Hamiltonian can be grouped in complementary pairs hence they are symmetric. I assume that it will be the same for any generic graph isomorphism problem.

According to our definitions in Algorithm 5, the adiabatic Hamiltonian of an n-vertex simple undirected cycle graph $\Gamma = (V, E)$ by the Hen-Young algorithm is given below. I assume n to be an even number.

$$
\begin{aligned}
\hat{H}(s) &= (1-s)\,\hat{H}_d + s\hat{H}_p \\
&= (1-s)\,\frac{1}{2}\sum_{i}^{n}\sigma_i^x + 2s\sum_{\langle ij\rangle\in\Gamma}\sigma_i^z\sigma_j^z \\
&= (1-s)\,\frac{1}{2}\sum_{i}^{n}\sigma_i^x + 2s\left(\sigma_1^z\sigma_2^z + \sigma_2^z\sigma_3^z + \ldots + \sigma_k^z\sigma_{k+1}^z + \ldots + \sigma_{n-1}^z\sigma_n^z + \sigma_n^z\sigma_1^z\right) \\
&= (1-s)\,\frac{1}{2}\sum_{i}^{n}\sigma_i^x + 2s\sum_{i}^{n}\sigma_i^z\sigma_{i+1}^z
\end{aligned}
$$

$$= \sum_{i}^{n} \left((1-s)\,\frac{1}{2}\sigma_i^x + 2s\sigma_i^z\sigma_{i+1}^z \right) \tag{4.1}$$

I adopt the periodic boundary condition for this 1-D spin ring where $i+n=i$.

To investigate the complexity of the adiabatic algorithm, I need to determine the gap between the eigenvalues of the ground state and the first excited state of $\tilde{H}(s)$.

Taking inspiration from section 4.1 of [67], I define an operator G which commutes with $\hat{H}(s)$. So,

$$\begin{aligned} G &= \prod_{j=1}^{n} \sigma_{(j)}^x \\ &= \sigma_{(1)}^x \otimes \sigma_{(2)}^x \ldots \otimes \sigma_{(n)}^x \\ &= ((\sigma^x \otimes \mathbb{I}_{n-1}) \cdot (\mathbb{I} \otimes \sigma^x \otimes \mathbb{I}_{n-2}) \cdot (\mathbb{I}_2 \otimes \sigma^x \otimes \mathbb{I}_{n-3}) \cdot \ldots \cdot (\mathbb{I}_{n-1} \otimes \sigma^x)) \end{aligned} \tag{4.2}$$

It is obvious that G negates the value of each qubit in the z basis. The first summation term in $\hat{H}(s)$ obviously commutes with all σ^x variables because it's a function of σ^x only and they commute with each other. Moreover, as $G^2 = (\prod_{j=1}^{n} \sigma_x^{(j)})^2 = \prod_{j=1}^{n} (\sigma_x^{(j)})^2 = \mathbb{I}_n$, the eigenvalues are ± 1.

As $G|x=0\rangle = |x=0\rangle$, I may restrict the computation for the states that are invariant under G. Based on our observation in the Tables 4.1 and 4.2, I assume that the space of the ground states of the generic graph isomorphism problem is invariant under G.

Now, I can write the definition of $\tilde{H}(s)$ in the invariant sector as a sum of $\frac{n}{2}$ commuting 2×2 Hamiltonians that I can diagonalize.

The second summation term in $\hat{H}(s)$ also commutes with the product of all σ^x because the term $\sigma_j^z \sigma_{j+1}^z$ anticommutes both with σ_j^x and σ_{j+1}^x because σ^x, σ^z anticommute, and it therefore commutes with the product of two σ^x as two minus signs give a plus. I use the Jordan-Wigner transformation [184, 104] to convert the Pauli operators of the adiabatic Hamiltonian into fermionic operators. To do that, I define the creation and annihilation operators from the Pauli operators.

$$\sigma^{\pm} = \sigma^x \pm i\sigma^y \tag{4.3}$$

I define the fermionic operators as follows.

$$\begin{aligned} b_j &= \sigma_1^x \sigma_2^x \ldots \sigma_{j-1}^x \sigma_j^- \mathbf{1}_{j+1} \ldots \mathbf{1}_n \\ b\dagger_j &= \sigma_1^x \sigma_2^x \ldots \sigma_{j-1}^x \sigma_j^+ \mathbf{1}_{j+1} \ldots \mathbf{1}_n \end{aligned} \tag{4.4}$$

If can be verified that,

$$\begin{aligned} \{b_j, b_k\} &= 0 \\ \{b_j, b\dagger_k\} &= \delta_{jk}. \end{aligned} \tag{4.5}$$

Moreover, I observe that,

$$b_j^\dagger b_j = \frac{1}{2}\left(1 - \sigma_j^x\right) \tag{4.6}$$

for $j = 1, \ldots, n$,

$$\left(b_j^\dagger - b_j\right)\left(b_{j+1}^\dagger + b_{j+1}\right) = \sigma_j^z \sigma_{j+1}^z. \tag{4.7}$$

I can also explicitly calculate to show that

$$\left(b_n^\dagger - b_n\right)\left(b_1^\dagger + b_1\right) = -G\sigma_n^z \sigma_1^z. \tag{4.8}$$

As G acts on the qubits which are in computational basis i.e. along the z-axis, Equations 4.7 and 4.8 are consistent only when $b_{n+1} = -b_1$ which is the antiperiodic boundary condition. So, it may be used as the definition of b_{n+1}.

So, I plug in the values in Eq. 4.1.

$$\hat{H}(s) = \sum_j^n (1-s)\frac{1}{2}\left(1 - 2b_j^\dagger b_j\right) + \sum_j^n 2s\left(b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1} + b_j^\dagger b_{j+1} - b_j b_{j+1}^\dagger\right) \tag{4.9}$$

Now, I like to define the Fourier transform of the fermionic operators as follows.

$$\beta_p = \frac{1}{\sqrt{n}} \sum_{j=1}^n e^{i\pi pj/n} b_j \tag{4.10}$$

, for $p = \pm 1, \pm 3, \ldots, \pm\,(n-1)$. The the terms for the even values of p can also be determined using the periodic boundary condition as I am investigating a spin chain on a ring of even length.

The reverse Fourier transform is defined as follows.

$$b_j = \frac{1}{\sqrt{n}} \sum_{p=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{-i\pi pj/n} \beta_p \tag{4.11}$$

This definition is consistent with $b_{n+1} = -b_1$. Moreover, I also observe that,

$$\begin{aligned} \{\beta_p, \beta_q\} &= 0 \\ \left\{\beta_p, \beta_q^\dagger\right\} &= \delta_{pq} \end{aligned} \tag{4.12}$$

Now, I replace the fermionic operators in the quadratic terms, $b_j^\dagger b_j$, $b_j^\dagger b_{j+1}^\dagger$, $b_j b_{j+1}^\dagger$, $b_j^\dagger b_{j+1}$, $b_j b_{j+1}$, of Eq. 4.24 with their Fourier transformations as defined in Eq. 4.11.

$$\begin{aligned} b_j^\dagger b_{j+1}^\dagger &= \frac{1}{n} \sum_{p,q} e^{\pi i(p+q)j/n} e^{\pi iq/n} \beta_q^\dagger \beta_p^\dagger \\ &= \sum_p e^{-\pi ip/n} \beta_{-p}^\dagger \beta_p^\dagger \end{aligned} \tag{4.13}$$

Similarly,

$$b_j b_{j+1} = \left(\frac{1}{\sqrt{n}} \sum_p e^{-i\pi pj/n} \beta_p \right) \left(\frac{1}{\sqrt{n}} \sum_q e^{-i\pi q(j+1)/n} \beta_q \right)$$

$$= \frac{1}{n} \sum_{p,q} e^{\pi i p/n} e^{-i\pi(p+q)j/n} \beta_p \beta_q$$

$$= \sum_{p} e^{\pi i p/n} \beta_p \beta_{-p} \tag{4.14}$$

,

$$\begin{aligned} b_j^\dagger b_j &= \frac{1}{n} \sum_{p,q} e^{\frac{ij\pi(q-p)}{n}} \beta_q^\dagger \beta_q \\ &= \frac{1}{n} \sum_{p,q} n\delta_{pq} \beta_q^\dagger \beta_p \\ &= \sum_{p} \beta_p^\dagger \beta_p \\ &= \beta_p^\dagger \beta_p + \beta_{-p}^\dagger \beta_{-p} \end{aligned} \tag{4.15}$$

$$\begin{aligned} b_j b_{j+1}^\dagger &= \left(\frac{1}{\sqrt{n}} \sum_{p=\pm1,\pm3,\ldots,\pm(n-1)} e^{-i\pi pj/n} \beta_p \right) \left(\frac{1}{\sqrt{n}} \sum_{p=\pm1,\pm3,\ldots,\pm(n-1)} e^{i\pi p(j+1)/n} \beta_p^\dagger \right) \\ &= \frac{1}{n} \left(\sum_{p,q=\pm1,\pm3,\ldots,\pm(n-1)} e^{-i\pi pj/n} e^{i\pi q(j+1)/n} \beta_p \beta_q^\dagger \right) \\ &= \frac{1}{n} \left(\sum_{p,q=\pm1,\pm3,\ldots,\pm(n-1)} e^{-i\pi(pj-qj-q)/n} \beta_p \beta_q^\dagger \right) \\ &= \frac{1}{n} \left(\sum_{p,q=\pm1,\pm3,\ldots,\pm(n-1)} e^{i\pi q/n} e^{-i\pi(pj-qj)/n} \beta_p \beta_q^\dagger \right) \\ &= \frac{1}{n} \left(\sum_{p,q=\pm1,\pm3,\ldots,\pm(n-1)} e^{i\pi q/n} n\delta_{pq} \beta_p \beta_q^\dagger \right) \end{aligned}$$

$$= \sum_p e^{i\pi p/n} \beta_{-p} \beta^\dagger_{-p} \tag{4.16}$$

$$\begin{aligned}
b^\dagger_j b_{j+1} &= \left(\frac{1}{\sqrt{n}} \sum_{p=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{i\pi p j/n} \beta^\dagger_p \right) \left(\frac{1}{\sqrt{n}} \sum_{p=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{-i\pi p(j+1)/n} \beta_p \right) \\
&= \frac{1}{n} \left(\sum_{p,q=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{i\pi p j/n} e^{-i\pi q(j+1)/n} \beta^\dagger_p \beta_q \right) \\
&= \frac{1}{n} \left(\sum_{p,q=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{i\pi (pj - qj - q)/n} \beta^\dagger_p \beta_q \right) \\
&= \frac{1}{n} \left(\sum_{p,q=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{-i\pi q/n} e^{i\pi (pj - qj)/n} \beta^\dagger_p \beta_q \right) \\
&= \frac{1}{n} \left(\sum_{p,q=\pm 1, \pm 3, \ldots, \pm(n-1)} e^{-i\pi q/n} n \delta_{pq} \beta^\dagger_p \beta_q \right) \\
&= \sum_p e^{-i\pi p/n} \beta^\dagger_p \beta_p
\end{aligned} \tag{4.17}$$

Now I plug in the values in $\left(b^\dagger_j - b_j\right)\left(b^\dagger_{j+1} + b_{j+1}\right)$.

$$\left(b^\dagger_j - b_j\right)\left(b^\dagger_{j+1} + b_{j+1}\right) = b^\dagger_j b^\dagger_{j+1} - b_j b^\dagger_{j+1} + b^\dagger_j b_{j+1} - b_j b_{j+1} \tag{4.18}$$

I compute the value of the terms pairwise.

$$b^\dagger_j b_{j+1} - b_j b^\dagger_{j+1} = \sum_p \left(e^{-i\pi p/n} \beta^\dagger_p \beta_p - e^{i\pi p/n} \beta_{-p} \beta^\dagger_{-p} \right)$$

$$= \sum_p e^{-i\pi p/n} \left(\beta_p^\dagger \beta_p - e^{2i\pi p/n} \beta_{-p} \beta_{-p}^\dagger\right)$$

$$= \sum_p e^{-i\pi p/n} \left(\beta_p^\dagger \beta_p - \beta_{-p} \beta_{-p}^\dagger\right)$$

$$= \sum_p \left(\cos\frac{\pi p}{n} - i \sin\frac{\pi p}{n}\right) \left(\beta_p^\dagger \beta_p - \beta_{-p} \beta_{-p}^\dagger\right) \tag{4.19}$$

Let's check the term $\sum_p \left(\cos\frac{\pi p}{n} - i\sin\frac{\pi p}{n}\right) \left(\beta_p^\dagger \beta_p - \beta_{-p}\beta_{-p}^\dagger\right)$ for $p = \pm 1$ setting $\frac{1}{n} = m$.

$$\sum_{p=1} (\cos m\pi p - i \sin m\pi p) \left(\beta_p^\dagger \beta_p - \beta_{-p} \beta_{-p}^\dagger\right)$$

$$= (\cos m\pi - i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$+ (\cos(-m\pi) - i \sin(-m\pi)) \left(\beta_{-1}^\dagger \beta_{-1} - \beta_1 \beta_1^\dagger\right)$$

$$= (\cos m\pi - i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$+ (\cos m\pi + i \sin m\pi) \left(\beta_{-1}^\dagger \beta_{-1} - \beta_1 \beta_1^\dagger\right)$$

$$= (\cos m\pi - i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$+ (\cos m\pi + i \sin m\pi) \left(\left(1 - \beta_{-1}\beta_{-1}^\dagger\right) - \left(1 - \beta_1^\dagger \beta_1\right)\right)$$

$$= (\cos m\pi - i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$+ (\cos m\pi + i \sin m\pi) \left(1 - \beta_{-1}\beta_{-1}^\dagger - 1 + \beta_1^\dagger \beta_1\right)$$

$$= (\cos m\pi - i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$+ (\cos m\pi + i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$= (\cos m\pi - i \sin m\pi + \cos m\pi + i \sin m\pi) \left(\beta_1^\dagger \beta_1 - \beta_{-1} \beta_{-1}^\dagger\right)$$

$$= 2\cos m\pi \left(\beta_1^\dagger \beta_1 - \beta_{-1}\beta_{-1}^\dagger\right) \tag{4.20}$$

So,

$$\begin{aligned} b_j^\dagger b_{j+1} - b_j b_{j+1}^\dagger &= \sum_p e^{-i\pi p/n}\left(\beta_p^\dagger\beta_p - \beta_{-p}\beta_{-p}^\dagger\right) \\ &= \sum_p \left(\cos\frac{\pi p}{n} - i\sin\frac{\pi p}{n}\right)\left(\beta_p^\dagger\beta_p - \beta_{-p}\beta_{-p}^\dagger\right) \\ &= 2\sum_p \cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p - \beta_{-p}\beta_{-p}^\dagger\right) \end{aligned} \tag{4.21}$$

Similarly,

$$\begin{aligned} b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1} &= \sum_p [e^{-\pi ip/n}\beta_{-p}^\dagger\beta_p^\dagger - e^{\pi ip/n}\beta_p\beta_{-p}] \\ &= \sum_p e^{-\pi ip/n}[\beta_{-p}^\dagger\beta_p^\dagger - e^{2\pi ip/n}\beta_p\beta_{-p}] \\ &= \sum_p e^{-\pi ip/n}[\beta_{-p}^\dagger\beta_p^\dagger - \beta_p\beta_{-p}] \quad \boxed{\text{using } e^{2\theta} = 1} \\ &= \sum_p (\cos\frac{\pi p}{n} - i\sin\frac{\pi p}{n})[\beta_{-p}^\dagger\beta_p^\dagger - \beta_p\beta_{-p}] \\ &= -2i\sum_p \sin\left(\frac{\pi p}{n}\right)[\beta_{-p}^\dagger\beta_p^\dagger - \beta_p\beta_{-p}] \end{aligned} \tag{4.22}$$

So,

$$\left(b_j^\dagger - b_j\right)\left(b_{j+1}^\dagger + b_{j+1}\right) = b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1}^\dagger + b_j^\dagger b_{j+1} - b_j b_{j+1}$$

$$
\begin{aligned}
&= \left(b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1}\right) + \left(b_j^\dagger b_{j+1} - b_j b_{j+1}^\dagger\right) \\
&= \left(-2i \sum_p \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger \beta_p^\dagger - \beta_p \beta_{-p}\right)\right) \\
&+ \left(2 \sum_p \cos\frac{\pi p}{n}\left(\beta_p^\dagger \beta_p - \beta_{-p}\beta_{-p}^\dagger\right)\right) \\
&= -2i \sum_p \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger \beta_p^\dagger - \beta_p \beta_{-p}\right) \\
&+ 2 \sum_p \cos\frac{\pi p}{n}\left(\beta_p^\dagger \beta_p - \beta_{-p}\beta_{-p}^\dagger\right) \qquad (4.23)
\end{aligned}
$$

So, the adiabatic Hamiltonian can be written as follows.

$$
\begin{aligned}
\tilde{H}(s) &= \sum_j^n \left((1-s)\frac{1}{2}\left(1 - 2b_j^\dagger b_j\right) + 2s\left(\left(b_j^\dagger - b_j\right)\left(b_{j+1}^\dagger + b_{j+1}\right)\right)\right) \\
&= \frac{1-s}{2}\sum_j^n \left(1 - 2b_j^\dagger b_j\right) + 2s\sum_j^n \left(b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1}^\dagger + b_j^\dagger b_{j+1} - b_j b_{j+1}\right) \\
&= \frac{1-s}{2}\sum_j^n \left(1 - 2b_j^\dagger b_j\right) + 2s\sum_j^n \left(b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1}^\dagger + b_j^\dagger b_{j+1} - b_j b_{j+1}\right) \\
&= \sum_{p=1,3,\ldots,(n-1)} A_p(s) \qquad (4.24)
\end{aligned}
$$

where,

$$
\begin{aligned}
A_p(s) &= \frac{1-s}{2}\left(1 - 2b_j^\dagger b_j\right) + 2s\left(b_j^\dagger b_{j+1}^\dagger - b_j b_{j+1}^\dagger + b_j^\dagger b_{j+1} - b_j b_{j+1}\right) \\
&= \frac{1-s}{2}\left(1 - 2\left(\beta_p^\dagger \beta_p + \beta_{-p}^\dagger \beta_{-p}\right)\right) \\
&+ 2s\left(-2i\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger \beta_p^\dagger - \beta_p \beta_{-p}\right) + 2\cos\frac{\pi p}{n}\left(\beta_p^\dagger \beta_p - \beta_{-p}\beta_{-p}^\dagger\right)\right)
\end{aligned}
$$

$$= \frac{1-s}{2}\left(1 - 2\left(\beta_p^\dagger\beta_p + \beta_{-p}^\dagger\beta_{-p}\right)\right)$$
$$+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p - \beta_{-p}\beta_{-p}^\dagger\right) - 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger - \beta_p\beta_{-p}\right) \qquad (4.25)$$

Let $|\Omega_p\rangle = |0\rangle$ be the ground state of A_p. So, $\beta_{\pm p}|\Omega_p\rangle = \beta_{\pm p}|0\rangle = 0$. Among $\beta_p^\dagger\beta_p$, $\beta_{-p}^\dagger\beta_{-p}$, $\beta_{-p}^\dagger\beta_p^\dagger$, $\beta_p\beta_{-p}$, $\beta_{-p}\beta_{-p}^\dagger$, and $\beta_p^\dagger\beta_p$, only $\beta_{-p}^\dagger\beta_p^\dagger$ takes $|\Omega_p\rangle$ to $|\Sigma_p\rangle$.

Now I compute $A_p|\Omega_p\rangle = A_p|0\rangle$.

$$\begin{aligned}
A_p|\Omega_p\rangle &= A_p|0\rangle \\
&= \frac{1-s}{2}\left(1 - 2\left(\beta_p^\dagger\beta_p + \beta_{-p}^\dagger\beta_{-p}\right)\right)|0\rangle \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p - \beta_{-p}\beta_{-p}^\dagger\right)|0\rangle - 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger - \beta_p\beta_{-p}\right)|0\rangle \\
&= \frac{1-s}{2}\left(|0\rangle - 2\left(\beta_p^\dagger\beta_p + \beta_{-p}^\dagger\beta_{-p}\right)|0\rangle\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p|0\rangle - \beta_{-p}\beta_{-p}^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}|0\rangle\right) \\
&= \frac{1-s}{2}\left(|0\rangle - 2\left(\beta_p^\dagger\beta_p|0\rangle + \beta_{-p}^\dagger\beta_{-p}|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(0 - \beta_{-p}\beta_{-p}^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - 0\right) \\
&= \frac{1-s}{2}\left(|0\rangle - 2\left(0+0\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(0 - \beta_{-p}\beta_{-p}^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - 0\right) \\
&= \frac{1-s}{2}|0\rangle - 4s\cos\frac{\pi p}{n}\left(1 - \beta_{-p}^\dagger\beta_{-p}\right)|0\rangle
\end{aligned}$$

$$
\begin{aligned}
&- 4si \sin\left(\frac{\pi p}{n}\right) \beta^\dagger_{-p}\beta^\dagger_p|0\rangle \\
&= \frac{1-s}{2}|0\rangle - 4s\cos\frac{\pi p}{n}\left(|0\rangle - \beta^\dagger_{-p}\beta_{-p}|0\rangle\right) \\
&- 4si \sin\left(\frac{\pi p}{n}\right)|1\rangle \\
&= \frac{1-s}{2}|0\rangle - 4s\cos\frac{\pi p}{n}\left(|0\rangle - 0\right) - 4si \sin\left(\frac{\pi p}{n}\right)|1\rangle \\
&= \left(\frac{1-s}{2} - 4s\cos\frac{\pi p}{n}\right)|0\rangle - 4si \sin\frac{\pi p}{n}|1\rangle
\end{aligned} \tag{4.26}
$$

Before I compute $A_p|\Sigma_p\rangle = A_p|1\rangle$, I observe the following.

$$
\begin{aligned}
\beta_{-p}|1\rangle &= \beta_{-p}\beta^\dagger_{-p}\beta^\dagger_p|0\rangle \\
&= \beta_{-p}\left(-\beta^\dagger_p\beta^\dagger_{-p}\right)|0\rangle \\
&= -\beta_{-p}\beta^\dagger_p\beta^\dagger_{-p}|0\rangle \\
&= -\left(-\beta^\dagger_p\beta_{-p}\right)\beta^\dagger_{-p}|0\rangle \\
&= \beta^\dagger_p\beta_{-p}\beta^\dagger_{-p}|0\rangle \\
&= \beta^\dagger_p\left(1 - \beta^\dagger_{-p}\beta_{-p}\right)|0\rangle \\
&= \beta^\dagger_p\left(|0\rangle - \beta^\dagger_{-p}\beta_{-p}|0\rangle\right) \\
&= \beta^\dagger_p|0\rangle
\end{aligned} \tag{4.27}
$$

Now I compute $A_p|\Sigma_p\rangle = A_p|1\rangle$.

$A_p|\Sigma_p\rangle = A_p|1\rangle$

$$
\begin{aligned}
&= \frac{1-s}{2}\left(1-2\left(\beta_p^\dagger\beta_p+\beta_{-p}^\dagger\beta_{-p}\right)\right)|1\rangle \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p-\beta_{-p}\beta_{-p}^\dagger\right)|1\rangle - 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger-\beta_p\beta_{-p}\right)|1\rangle \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(\beta_p^\dagger\beta_p+\beta_{-p}^\dagger\beta_{-p}\right)|1\rangle\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p|1\rangle-\beta_{-p}\beta_{-p}^\dagger|1\rangle\right) - 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger|1\rangle-\beta_p\beta_{-p}|1\rangle\right) \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(\beta_p^\dagger\beta_p|1\rangle+\beta_{-p}^\dagger\beta_{-p}|1\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p|1\rangle-\beta_{-p}\beta_{-p}^\dagger|1\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger|1\rangle-\beta_p\beta_{-p}|1\rangle\right) \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle+\beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(\beta_p^\dagger\left(-\beta_{-p}^\dagger\beta_p\right)\beta_p^\dagger|0\rangle+\beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(-\beta_p^\dagger\beta_{-p}^\dagger\beta_p\beta_p^\dagger|0\rangle+\beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(-\beta_p^\dagger\beta_{-p}^\dagger\left(1-\beta_p^\dagger\beta_p\right)|0\rangle+\beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle-\beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right) \\
&= \frac{1-s}{2}\left(|1\rangle-2\left(-\beta_p^\dagger\beta_{-p}^\dagger\left(|0\rangle-\beta_p^\dagger\beta_p|0\rangle\right)+\beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right)
\end{aligned}
$$

$$
\begin{aligned}
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&= \frac{1-s}{2}\left(|1\rangle - 2\left(-\beta_p^\dagger\beta_{-p}^\dagger\left(|0\rangle - 0\right) + \beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right)\\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&= \frac{1-s}{2}\left(|1\rangle - 2\left(-\beta_p^\dagger\beta_{-p}^\dagger|0\rangle + \beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right)\\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&= \frac{1-s}{2}\left(|1\rangle - 2\left(-\left(-\beta_{-p}^\dagger\beta_p^\dagger\right)|0\rangle + \beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right)\\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&= \frac{1-s}{2}\left(|1\rangle - 2\left(\beta_{-p}^\dagger\beta_p^\dagger|0\rangle + \beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right)\\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&= \frac{1-s}{2}\left(|1\rangle - 2\left(|1\rangle + \beta_{-p}^\dagger\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\right)\\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&= \frac{1-s}{2}\left(-|1\rangle - 2\beta_{-p}^\dagger\left(1 - \beta_{-p}^\dagger\beta_{-p}\right)\beta_p^\dagger|0\rangle\right)\\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_p^\dagger\beta_p\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_{-p}\beta_{-p}^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)\\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^\dagger\beta_p^\dagger\beta_{-p}^\dagger\beta_p^\dagger|0\rangle - \beta_p\beta_{-p}\beta_{-p}^\dagger\beta_p^\dagger|0\rangle\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1-s}{2}\left(-|1\rangle - 2\beta_{-p}^{\dagger}\left(\beta_{p}^{\dagger}|0\rangle - \beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_{p}^{\dagger}\beta_{p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= \frac{1-s}{2}\left(-|1\rangle - \left(2\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - 2\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_{p}^{\dagger}\beta_{p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= \frac{1-s}{2}\left(-|1\rangle - \left(2\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - 2\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_{p}^{\dagger}\beta_{p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= \frac{1-s}{2}\left(-|1\rangle - (2|1\rangle - 0)\right) \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_{p}^{\dagger}\beta_{p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_{p}^{\dagger}\beta_{p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle \\
&+ 4s\cos\frac{\pi p}{n}\left(\beta_{p}^{\dagger}\left(-\beta_{-p}^{\dagger}\beta_{p}\right)\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle \\
&+ 4s\cos\frac{\pi p}{n}\left(-\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}\beta_{p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)
\end{aligned}
$$

$$
\begin{aligned}
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle\\
&+ 4s\cos\frac{\pi p}{n}\left(-\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\left(1-\beta_{p}^{\dagger}\beta_{p}\right)|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle\\
&+ 4s\cos\frac{\pi p}{n}\left(-\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\left(|0\rangle - \beta_{p}^{\dagger}\beta_{p}|0\rangle\right) - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle\\
&+ 4s\cos\frac{\pi p}{n}\left(-\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\left(|0\rangle - 0\right) - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle\\
&+ 4s\cos\frac{\pi p}{n}\left(-\beta_{p}^{\dagger}\beta_{-p}^{\dagger}|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle\\
&+ 4s\cos\frac{\pi p}{n}\left(-\left(-\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\right)|0\rangle - \beta_{-p}\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}\left(|1\rangle - 0\right)\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)\\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle\\
&- 4si \sin\left(\frac{\pi p}{n}\right)\left(\beta_{-p}^{\dagger}\left(-\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\right)\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right)
\end{aligned}
$$

$$
\begin{aligned}
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(-\beta_{-p}^{\dagger}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&- 4si\sin\left(\frac{\pi p}{n}\right)\left(0 - \beta_{p}\beta_{-p}\beta_{-p}^{\dagger}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\beta_{p}\left(1 - \beta_{-p}^{\dagger}\beta_{-p}\right)\beta_{p}^{\dagger}|0\rangle \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\beta_{p}\left(\beta_{p}^{\dagger}|0\rangle - \beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\left(\beta_{p}\beta_{p}^{\dagger}|0\rangle - \beta_{p}\beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\left(\left(1 - \beta_{p}^{\dagger}\beta_{p}\right)|0\rangle - \beta_{p}\beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\left(\left(|0\rangle - \beta_{p}^{\dagger}\beta_{p}|0\rangle\right) - \beta_{p}\beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\left(\left(|0\rangle - 0\right) - \beta_{p}\beta_{-p}^{\dagger}\beta_{-p}\beta_{p}^{\dagger}|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\left(|0\rangle - \beta_{p}\beta_{-p}^{\dagger}\left(-\beta_{p}^{\dagger}\beta_{-p}\right)|0\rangle\right) \\
&= -\frac{3\left(1-s\right)}{2}|1\rangle + 4s\cos\frac{\pi p}{n}|1\rangle \\
&+ 4si\sin\left(\frac{\pi p}{n}\right)\left(|0\rangle + 0\right)
\end{aligned}
$$

$$= \left(-\frac{3\left(1-s\right)}{2} + 4s\cos\frac{\pi p}{n} \right) |1\rangle + 4si\sin\frac{\pi p}{n}|0\rangle \tag{4.28}$$

So, I reexpress A_p in the Ω_p, Σ_p basis, it becomes as follows.

$$A_p\left(s\right) = \begin{pmatrix} \left(\frac{1-s}{2} - 4s\cos\frac{\pi p}{n}\right) & -4si\sin\frac{\pi p}{n} \\ 4si\sin\frac{\pi p}{n} & \left(-\frac{3(1-s)}{2} + 4s\cos\frac{\pi p}{n}\right) \end{pmatrix} \tag{4.29}$$

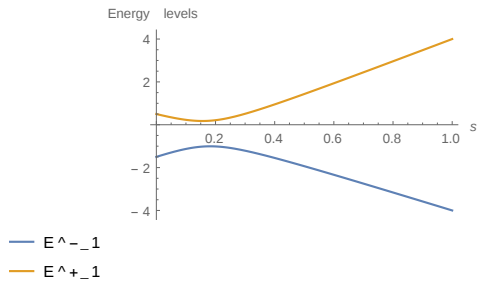
For each p, the eigenvalues of $A_p\left(s\right)$ are:

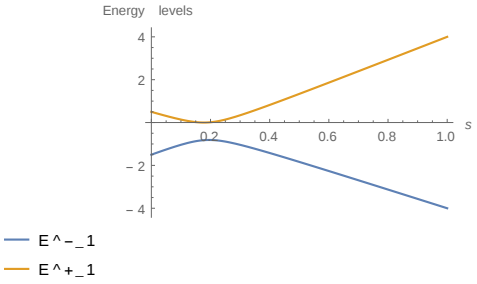
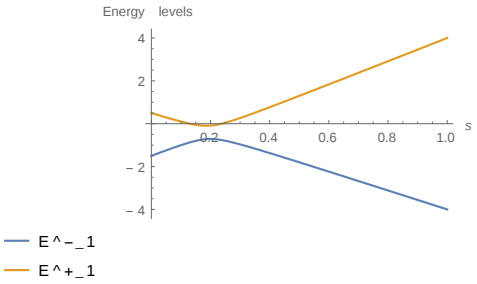
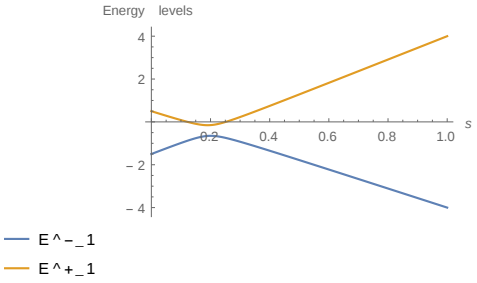
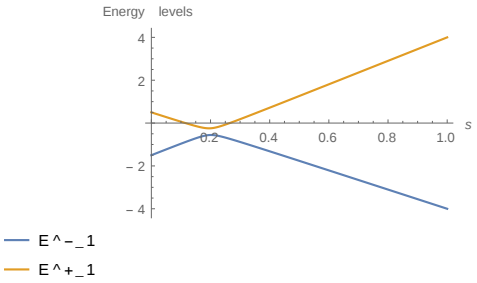
$$E_p^{\pm} = \frac{1}{2}\left(\pm 2\sqrt{8s^2\cos\left(\frac{\pi p}{n}\right) - 8s\cos\left(\frac{\pi p}{n}\right) + 17s^2 - 2s + 1} + s - 1 \right) \tag{4.30}$$

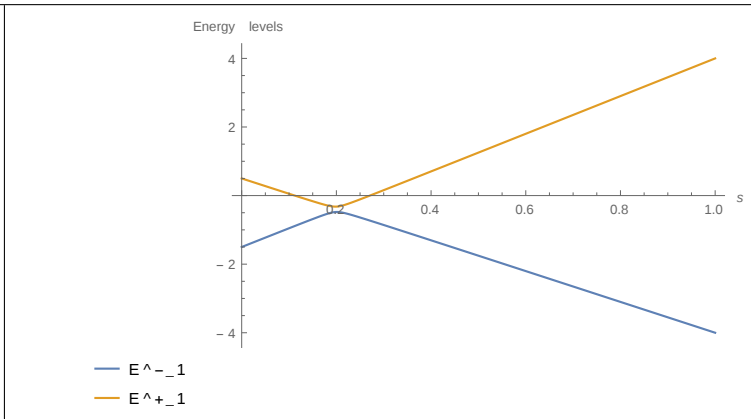
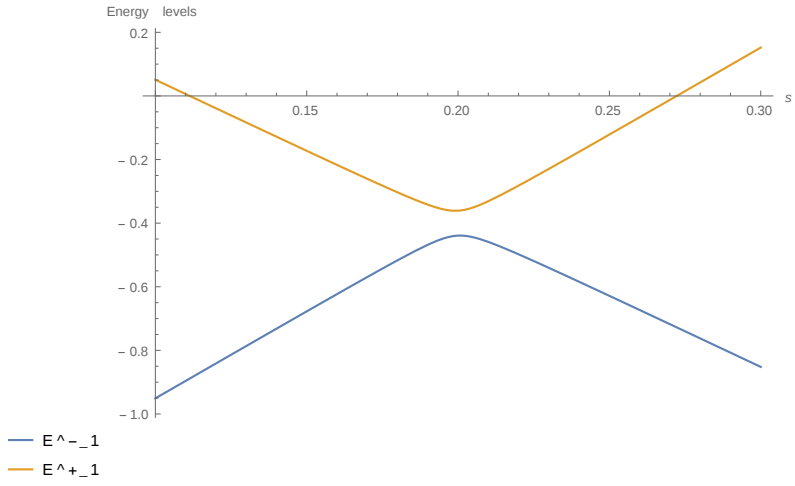
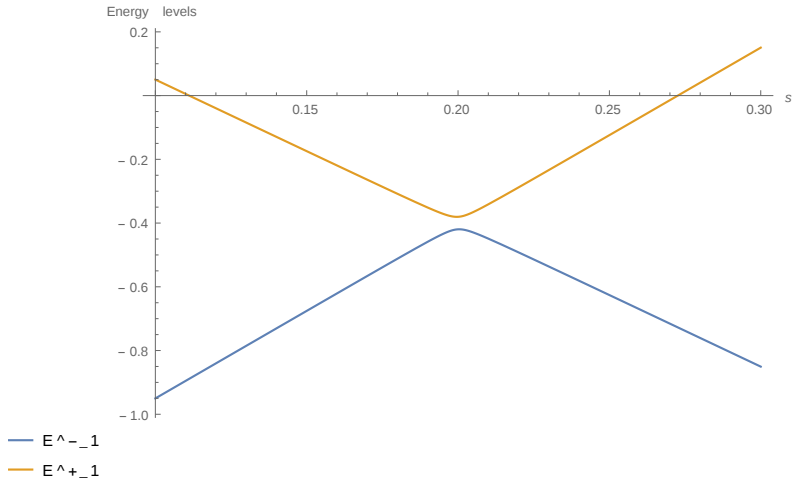
So, the ground state is $E_1^-\left(s\right)$. The first excited state is $E_1^+\left(s\right) + \sum_{p=3\ldots} E_p^-\left(s\right)$. I plot the eigenvalue for $E_1^-\left(s\right)$ and $E_1^+\left(s\right)$.

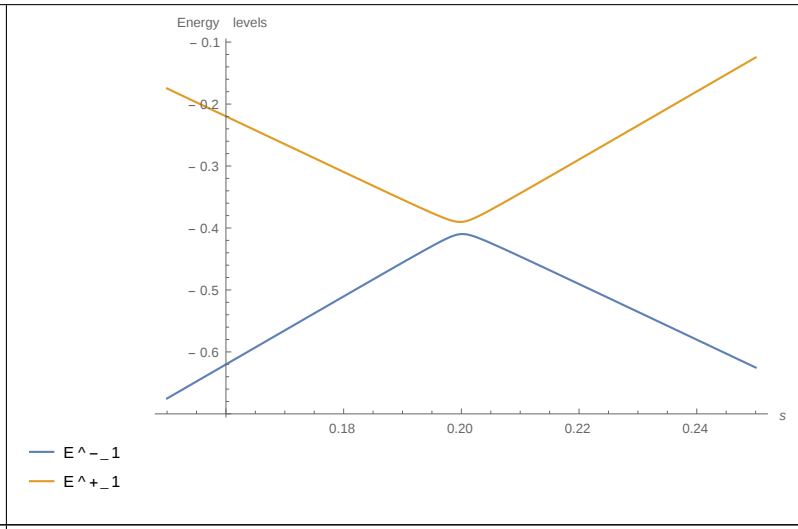
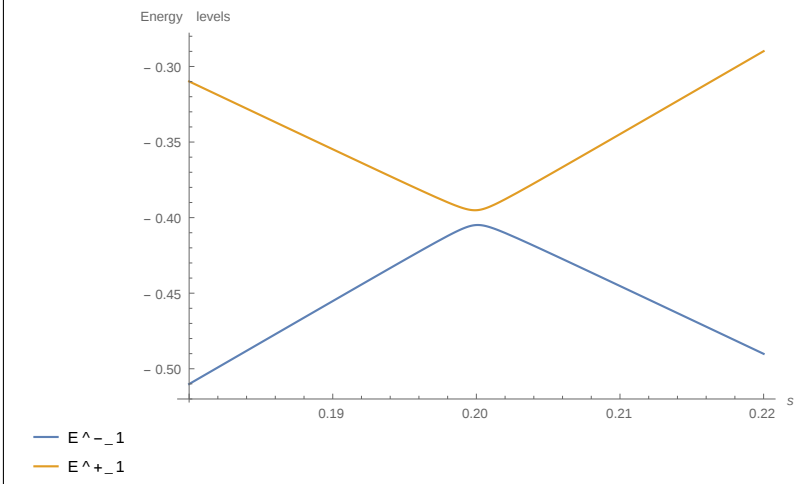
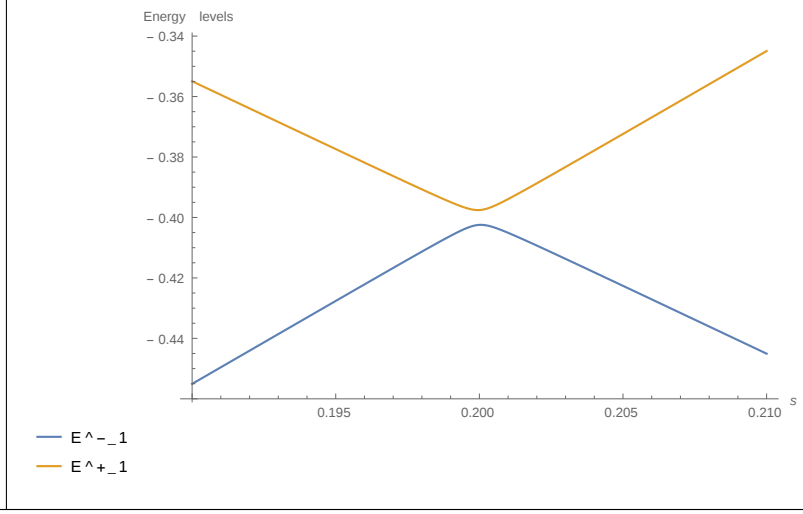
I plot the eigenvalue for $E_1^-\left(s\right)$ and $E_1^+\left(s\right)$ for $n = 4, 6, 8, 10, 16, 32, 64, 128, 256, 512, 1024$ to have initial idea about the spectrum of the Hamiltonian.

Table 4.3: Energy gaps for C_n

C_n	$E_1^-\left(s\right), E_1^+\left(s\right)$ vs. s
C_4	

C_6	 Energy levels 4 2 0.2 0.4 0.6 0.8 1.0 s – 2 – 4 E^–_1 E^+_1
C_8	 Energy levels 4 2 0.2 0.4 0.6 0.8 1.0 s – 2 – 4 E^–_1 E^+_1
C_{10}	 Energy levels 4 2 0.2 0.4 0.6 0.8 1.0 s – 2 – 4 E^–_1 E^+_1
C_{16}	 Energy levels 4 2 0.2 0.4 0.6 0.8 1.0 s – 2 – 4 E^–_1 E^+_1

C_{32}	 Energy levels 4 2 − 2 − 4 0.2 0.4 0.6 0.8 1.0 s E^−_1 E^+_1
C_{64}	 Energy levels 0.2 − 0.2 − 0.4 − 0.6 − 0.8 − 1.0 0.15 0.20 0.25 0.30 s E^−_1 E^+_1
C_{128}	 Energy levels 0.2 − 0.2 − 0.4 − 0.6 − 0.8 − 1.0 0.15 0.20 0.25 0.30 s E^−_1 E^+_1

C_{256}	
C_{512}	
C_{1024}	

For a few plots in the table above, I have plotted for a smaller window of s

as the curves diverges on both sides beyond the window. The minimum gap occurs very close to $s = \frac{1}{5}$ and is,

$$\begin{aligned} g_{min} &\approx E_1^+\left(\frac{1}{5}\right) - E_1^-\left(\frac{1}{5}\right) \\ &= \frac{8}{5}\sqrt{2 - 2\cos\left(\frac{\pi}{n}\right)} \end{aligned} \tag{4.31}$$

When n is large,

$$\begin{aligned} g_{min} &\approx \frac{8}{5}\sqrt{\frac{\pi^2}{n^2} + 1} \\ &\approx \frac{8\pi}{5n} \end{aligned} \tag{4.32}$$

In Section 2.5, I have mentioned that the required evolution time T must be much greater than $\frac{E}{g_{min}^3}$ where for this problem E scales like n so $T \gg cn^3$ where c is a constant. I have shown that for any cycle graph, quantum adiabatic evolution will find the graph configuration in a time which grows as a fixed power of n.

So, for cycle graph isomorphism problem, the adiabatic gap doesn't decrease exponentially for the Hen-Young algorithm. Hence, it succeeds for cycle graph isomorphism.

4.3 Other algorithms

Since the Hen-Young algorithm, two more adiabatic algorithms have been proposed for the graph isomorphism problem. Correctness proof for any of them is yet to be presented. Here I would like to briefly mention about these two algorithms to illustrate their differences with the original Hen-Young algorithm. It could be mentioned that the Hen-Sarandy algorithm draws its inspiration from [120] and one of my work with a few others [187] which is the center of discussion in Chapter 5 of this dissertation.

4.3.1 The Gaitan-Clark algorithm

I reproduce the scheme of the Gaitan-Clark algorithm from [73]. A permutation π of a finite set $\mathcal{S} = \{0, \ldots, N-1\}$, expressed in the cycle notation, can be mapped into an integer string $P(\pi) = \pi_0 \cdots \pi_{N-1}$, with $\pi_i \in \mathcal{S}$ and $\pi_i \neq \pi_j$ for $i \neq j$.

For reasons that will become clear later in this section, I like to represent the integer string $P(\pi) = \pi_0 \cdots \pi_{N-1}$ into a binary string P_π. This can be done by replacing each π_i in $P(\pi)$ by the unique binary string formed from the coefficients appearing in its binary decomposition

$$\pi_i = \sum_{j=0}^{U-1} \pi_{i,j} \, (2)^j \, . \tag{4.33}$$

Here $U \equiv \lceil \log_2 N \rceil$. So, the integer string $P(\pi)$ is converted into the binary string

$$P_\pi = (\pi_{0,0} \cdots \pi_{0,U-1}) \cdots (\pi_{N-1,0} \cdots \pi_{N-1,U-1}) \, , \tag{4.34}$$

where $\pi_{i,j} \in \{0,1\}$. The binary string P_π has length NU, where

$$N \leq 2^U \equiv M + 1. \tag{4.35}$$

Thus the permutation π corresponds with the binary string P_π in Eq. (4.34).

I defined $\mathcal{H}$ be the Hamming space of binary strings of length NU. It contains 2^{NU} strings. Among them, $N!$ strings P_π encode valid permutations π. Finally, I define a mapping from $\mathcal{H}$ to the space of $N \times N$ matrices σ with binary matrix elements $\sigma_{i,j} = 0, 1$.

- Let $s_b = s_0 \cdots s_{NU-1}$ be a binary string in $\mathcal{H}$. s_b can be written as N substrings of length U as follows:

$$s_b = (s_0 \cdots s_{U-1})\ (s_U \cdots s_{2U-1}) \cdots \left(s_{(N-1)U} \cdots s_{NU-1}\right). \tag{4.36}$$

- For each substring $s_{iU} \cdots s_{(i+1)U-1}$, define s_i as:

$$s_i = \sum_{j=0}^{U-1} s_{iU+j}\,(2)^j \leq 2^U - 1 = M. \tag{4.37}$$

- Define the integer string $s = s_0 \cdots s_{N-1}$, and define the $N \times N$ matrix $\sigma(s)$ to have matrix elements

$$\sigma_{i,j}(s) = \begin{cases} 0, & \text{if } s_j > N-1 \\ \delta_{i,s_j}, & \text{if } 0 \leq s_j \leq N-1, \end{cases} \tag{4.38}$$

where $i, j \in \mathcal{S}$, and $\delta_{x,y}$ is the Kronecker delta.

I observe that when the binary string s_b corresponds to a permutation, the matrix $\sigma(s)$ is a permutation matrix since the s_i formed in step 2 will obey $0 \leq s_i \leq N-1$ and $s_i \neq s_j$ for $i \neq j$. Here, if A is the adjacency matrix for a graph G, then $A' = \sigma(s) A \sigma^T(s)$ will be the adjacency matrix for a graph G' isomorphic to G. On the other hand, if s_b does not correspond to a permutation, then the adjacency matrix $A' = \sigma(s) A \sigma^T(s)$ must correspond to a graph G' which is not isomorphic to G.

So, I have constructed a map from binary strings of length NU to $N \times N$ matrices (viz. linear maps) with binary matrix elements. When the string is (is not) a valid permutation, the matrix produced is (is not) a permutation matrix. Finally, follow from Stirling's formula that $\log_2 N! \sim N \log_2 N - N$ which is the number of bits needed to represent $N!$. This encoding of permutations requires $N\lceil \log_2 N \rceil$ bits and so approaches asymptotically as required by Stirling's formula.

Now I transform an instance of the graph isomorphism problem into an instance of a combinatorial optimization problem (COP) whose cost function has a minimum value of zero if and only if G and G' are isomorphic.

The search space for the COP is the Hamming space $\mathcal{H}$ of binary strings s_b of length NU which are associated with the integer strings s and matrices $\sigma(s)$. The COP cost function $C(s)$ contains three terms.

$$C(s) = C_1(s) + C_2(s) + C_3(s). \tag{4.39}$$

The first two terms on the RHS penalize integer strings $s = s_0 \cdots s_{N-1}$ whose associated matrix $\sigma(s)$ is not a permutation matrix,

$$C_1(s) = \sum_{i=0}^{N-1} \sum_{\alpha=N}^{M} \delta_{s_i,\alpha} \tag{4.40}$$

$$C_2(s) = \sum_{i=0}^{N-2} \sum_{j=i+1}^{N-1} \delta_{s_i,s_j}, \tag{4.41}$$

where $\delta_{x,y}$ is the Kronecker delta. Here, $C_1(s) > 0$ when $s_i > N-1$ for some i, and $C_2(s) > 0$ when $s_i = s_j$ for some $i \neq j$. Thus $C_1(s) + C_2(s) = 0$ if and only if $\sigma(s)$ is a permutation matrix. The third term $C_3(s)$ adds a penalty when $\sigma(s) A \sigma^T(s) \neq A'$:

$$C_3(s) = \|\sigma(s) A \sigma^T(s) - A'\|_i. \tag{4.42}$$

Here $\|M\|_i$ is the L_i-norm of M. So, when G and G' are isomorphic, $C_3(s) = 0$, and $\sigma(s)$ is the permutation of vertices of G that maps $G \to G'$. So, if $C(s) = 0$ for some integer string s, then G and G' are isomorphic and $\sigma(s)$ is the permutation that maps them. On the other hand, if $C(s) > 0$ for all strings s, then G and G' are non-isomorphic.

I have thus converted an instance of GI into an instance of the following COP:

Definition 42 (Graph Isomorphism COP:)**.** *Given the N-vertex graphs G and G' and the associated cost function $C(s)$ defined above, find an integer string s_* that minimizes $C(s)$.*

By definition: (i) $C(s_*) = 0$ if and only if G and G' are isomorphic and $\sigma(s_*)$ is the permutation matrix mapping $G \to G'$; and (ii) $C(s_*) > 0$ if and only if G and

G' are non-isomorphic.

Like the authors, I also like to point it out that if $G = G'$, then $C(s_*) = 0$ since G is certainly isomorphic to itself. In this case $\sigma(s_*)$ is an automorphism of G.

To map the GI COP onto an adiabatic quantum computation, I construct the computational basis states (CBS) $|s_b\rangle$ with the labels s_b. So, each bit in s_b is represented by a qubit so that the quantum register contains $L = NU = N\lceil \log_2 N \rceil$ qubits. The CBS are defined to be the 2^L eigenstates of $\sigma_z^0 \otimes \cdots \otimes \sigma_z^{L-1}$. The problem Hamiltonian H_P is defined to be diagonal in the CBS with eigenvalue $C(s)$, where s is the integer string associated with s_b:

$$H_P|s_b\rangle = C(s)|s_b\rangle. \tag{4.43}$$

The ground-state energy of H_P will be zero if and only if the graphs G and G' are isomorphic. The initial Hamiltonian H_i is chosen to be

$$H_i = \sum_{l=0}^{L-1} \frac{1}{2}\left(I^l - \sigma_x^l\right), \tag{4.44}$$

where I^l and σ_x^l are the identity and x-Pauli operator for qubit l, respectively. The ground-state of H_i is the easily constructed uniform superposition of CBS.

The quantum algorithm for GI begins by preparing the L qubit register in the ground-state of H_i and then driving the qubit register dynamics using the time-dependent Hamiltonian $H(t)$. At the end of the evolution the qubits are measured

in the computational basis. The outcome is the bit string s_b^* so that the final state of the register is $|s_b^*\rangle$ and its energy is $C(s^*)$, where s^* is the integer string derived from s_b^*. In the adiabatic limit, $C(s^*)$ will be the ground-state energy, and if $C(s^*) = 0$ (> 0) the algorithm decides G and G' are isomorphic (non-isomorphic).

4.3.2 The Hen-Sarandy algorithm

I reproduce the scheme of the Hen-Sarandy algorithm from [90]. In this algorithm, the construction of the driver and problem Hamiltonian are more straightforward. A problem Hamiltonian can then be written in terms of positive energy contributions to invalid mappings, i.e., each time an edge appears in one graph but not in the other. This yields

$$H_p = \sum_{ij \notin E_1} \sum_{i'j' \in E_2} \frac{(1+\sigma_{i,i'}^z)}{2} \frac{(1+\sigma_{j,j'}^z)}{2} + \sum_{ij \in E_1} \sum_{i'j' \notin E_2} \frac{(1+\sigma_{i,i'}^z)}{2} \frac{(1+\sigma_{j.j'}^z)}{2}\,, \tag{4.45}$$

where E_1 and E_2 are the edge sets of G_1 and G_2, respectively. Moreover, another term is also added to the problem Hamiltonians which penalizes invalid permutations. This constraint is translated into penalty terms in the problem Hamiltonian, explicitly,

$$H^{\text{pen}} = \sum_i \Big[\big(\sum_j \frac{\big(1+\sigma_{i,j}^z\big)}{2} - 1 \big)^2 + \big(\sum_j \frac{\big(1+\sigma_{j,i}^z\big)}{2} - 1 \big)^2 \Big]\,. \tag{4.46}$$

At this point, I invite the readers to compare these terms with the terms I

have defined in the objective function (Eq. 5.2) minimized by a quantum annealing algorithm I have proposed in Chapter 5. This illustrates how researches are taking inspirations from classical regime to design new quantum algorithms.

The minimal set of hopping terms required to hop from one allowed configuration to another is given by the driver Hamiltonian:

$$H_d = -\sum_{i,j}\sum_{j<j'}\Big(|\uparrow\rangle\langle\downarrow|_{(i,j)} \otimes |\downarrow\rangle\langle\uparrow|_{(i+1,j)} \otimes |\uparrow\rangle\langle\downarrow|_{(i,j')} \otimes |\downarrow\rangle\langle\uparrow|_{(i+1,j')} + \text{c.c.}\Big), \tag{4.47}$$

where c.c. denotes complex conjugate terms. The interpretation of the Hamiltonian above in terms of hopping particles is illustrated as follows. The hopping terms swap the location of particles (or up spins) in neighboring rows.

By rewriting H_d in Eq. (4.47) in terms of Pauli operators, we get

$$H_d = -\sum_{i,j}\sum_{j<j'}\left(\sigma^+_{(i,j)}\sigma^-_{(i+1,j)}\sigma^+_{(i+1,j')}\sigma^-_{(i,j')} + c.c.\right). \tag{4.48}$$

The H_d as I have just defined involves four-body terms. In particular, four-body interactions are enough to implement the solution of any n-vertex GI problem; i.e., the non-locality of the interaction does not scale with the size of the problem. The advantage of this approach is that, at the cost of a four-body quantum driver Hamiltonian, no additional constraints are needed in the problem Hamiltonian, meaning that $N_E = O(n^2)$ edges are already enough to embed an n-vertex graph. Obviously,

this will require a quantum computer with fundamentally different architecture than the previous ones mentioned in this chapter.

4.4 Summary

In this section, I have given a correctness proof of the Hen-Young algorithm for the cycle graphs. It is already known that [3], for any efficient quantum adiabatic algorithm, there is always an efficient quantum gate algorithm. So, it is natural too ask, if quantum Fourier transform fails to solve graph isomorphism and automorphism problems, as shown in Chapter 3, what is the equivalent gate model algorithm when there is an efficient quantum adiabatic algorithm for a particular class of graph isomorphism problem? In future, one may investigate efficient designs of Hamiltonian clocks to perform this conversion between two paradigms.

Chapter 5

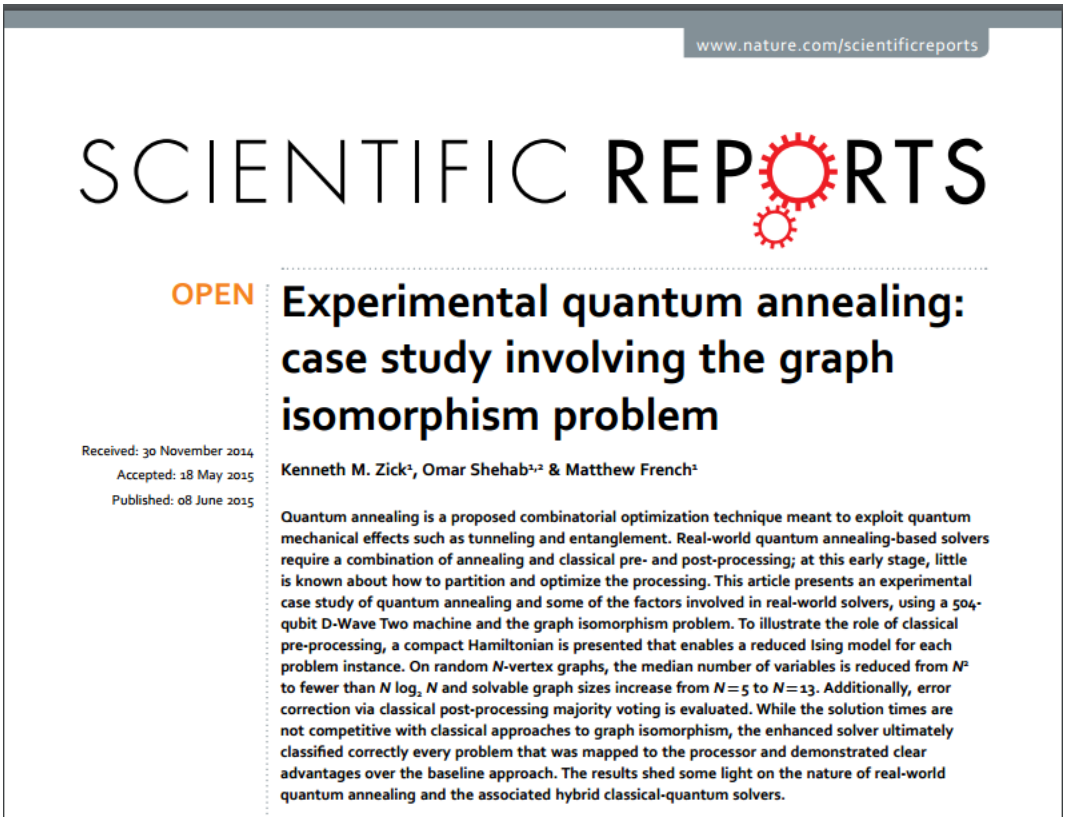
Graph Isomorphism and Quantum Annealing

5.1 Introduction

In this chapter, I implement a quantum annealing algorithm for graph isomorphism, first published in [187], and study its asymptotic behavior. I also describe the setup, and methodology of the proof-of-concept experiment and discuss the results. To the best of my knowledge, this is the first of its kind.

Quantum annealing algorithm for the graph isomorphism problem is an active research area [187, 120, 74, 91, 98, 179, 181]. The idea is to define an objective function f which encodes the connectivity of two input graphs Γ_1 and Γ_2. When the function is minimized on a quantum annealing device the value and the corresponding configuration tells us whether Γ_1 and Γ_2 are isomorphic. I assume the graphs in this report are simple and undirected.

So far, only [187] co-authored by me and [181] have demonstrated proof-of-concept experiments of solving the graph isomorphism problem. It was a successful collaboration with Kenneth M. Zick and Matthew French of the Information Sciences Institute, University of Southern California.



www.nature.com/scientificreports

SCIENTIFIC REPORTS

OPEN

Experimental quantum annealing: case study involving the graph isomorphism problem

Received: 30 November 2014
Accepted: 18 May 2015
Published: 08 June 2015

Kenneth M. Zick[1], Omar Shehab[1,2] & Matthew French[1]

Quantum annealing is a proposed combinatorial optimization technique meant to exploit quantum mechanical effects such as tunneling and entanglement. Real-world quantum annealing-based solvers require a combination of annealing and classical pre- and post-processing; at this early stage, little is known about how to partition and optimize the processing. This article presents an experimental case study of quantum annealing and some of the factors involved in real-world solvers, using a 504-qubit D-Wave Two machine and the graph isomorphism problem. To illustrate the role of classical pre-processing, a compact Hamiltonian is presented that enables a reduced Ising model for each problem instance. On random N-vertex graphs, the median number of variables is reduced from N^2 to fewer than $N \log_2 N$ and solvable graph sizes increase from $N=5$ to $N=13$. Additionally, error correction via classical post-processing majority voting is evaluated. While the solution times are not competitive with classical approaches to graph isomorphism, the enhanced solver ultimately classified correctly every problem that was mapped to the processor and demonstrated clear advantages over the baseline approach. The results shed some light on the nature of real-world quantum annealing and the associated hybrid classical-quantum solvers.

Quantum annealing (QA) is a proposed combinatorial optimization technique meant to exploit quantum mechanical effects such as tunneling and entanglement [125]. Machines purportedly implementing a type of quantum annealing have recently become available [25]. While the extent of quantumness in these implementations is not fully understood, some evidence for quantum mechanical effects playing a useful role in the processing has been appearing [116, 6, 26]. Aside from the debate over quantumness, there are interesting questions regarding how to effectively solve a real-world problem using a quantum annealer. Quantum annealing-based solvers require a combination of annealing and classical pre- and post-processing; at this early stage, little is known about how to partition and optimize the processing. For instance, current quantum annealers have severe practical limitations on the size of

problems that can be handled. Can the pre-processing algorithms be modified in order to improve scalability? A second question involves post-processing. Quantum annealers provide solutions to an embedded version of a problem involving physical qubits. Post-processing is generally required for translating these to solutions to the original problem involving logical qubits (aka variables). Occasionally, a chain of physical qubits representing a single variable resolves to an inconsistent state, a scenario known as a broken chain. Studies are needed regarding broken chains and the possibility of classical error correction during post-processing.

This chapter presents an experimental case study of quantum annealing and some of the factors involved in real-world solvers, using a 504-qubit D-Wave Two machine. An example of parsimonious pre-processing is considered, along with post-processing majority voting. Through experiments on a 504-qubit D-Wave Two machine, I quantify the QA success probabilities and the impact of the methods under study. I use the graph isomorphism (GI) problem as the problem of focus. The GI problem is to determine whether two input graphs $G_{1,2}$ are in essence the same, such that the adjacency matrices can be made identical with a relabeling of vertices. This problem is an interesting candidate for several reasons. First, an accurate quantum annealing-based solver for GI has never been implemented. Second, quantum approaches can sometimes provide new insight into the structure of a problem, even if no speedup over classical approaches is achieved or even expected. Third, the GI problem is mathematically interesting; though many sub-classes of the problem can be solved in polynomial time by specialized classical solvers, the run time of the best general solution is exponential and has remained at $e^{O\left(\sqrt{N \log N}\right)}$ since 1983 [128, 16].

The classical computational complexity of the problem is currently considered to be **NP**-intermediate [156], and the quantum computational complexity of the problem is unknown. Graph isomorphism is a non-abelian hidden subgroup problem and is not known to be easy in the quantum regime [141, 79]. Lastly, the GI problem is of practical interest. It appears in fields such as very large-scale integrated circuit design, where a circuits layout graph must be verified to be equivalent to its schematic graph [114], and in drug discovery and bio-informatics, where a graph representing a molecular compound must be compared to an entire database, often via a GI tool that performs canonical labeling [128].

This chapter relates to previous works as follows. A pre-print by King and McGeoch discusses tuning of quantum annealing algorithms, including the use of low-cost classical post-processing majority voting similar to what is evaluated in this article [108]. Our study goes further regarding pre-processing (designing a Hamiltonian to generate compact Ising models) and covers graph isomorphism rather than problems such as random not-all-equal 3-SAT. A work by Rieffel et al. maps real-world problems such as graph coloring to a D-Wave quantum annealer [157]. Regarding the graph isomorphism problem in particular, multiple attempts have been made using adiabatic quantum annealing. One of the first attempts assigned a Hamiltonian to each graph and conjectured that measurements taken during each adiabatic evolution could be used to distinguish non-isomorphic pairs [91]. A subsequent experimental study using a D-Wave quantum annealer found that using quantum spectra in this manner was not sufficient to distinguish non-isomorphic pairs [181]. A second approach converted a GI problem to a combinatorial opti-

mization problem whose non-negative cost function has a minimum of zero only for an isomorphic pair. The approach required problem variables and additional ancillary variables. It was numerically simulated up to $N = 7$ but not validated on a quantum annealing processor [73]. An alternative GI Hamiltonian was proposed by Lucas [119].

5.2 Graph isomorphism as pseudo-Boolean function

Quantum annealing minimized the quadratic uncontrained binary optimization representation of the graph isomorphism problem which is a special case of pseudo-Boolean optimization. To express the graph isomorphism as a pseudo-Boolean function, I intuitively choose a subset of the class of pseudo-Boolean functions which is $f : \mathbb{B}^n \to \mathbb{Z}$. The basic idea is to create a function which will penalize input binary strings at two levels. Firstly, it will penalize if the input strings do not correspond to valid permutations. Secondly, it will penalize if the input strings do not isomorphically maps one graph to another. This approach was first used in [74]. The family of such functions can be described as follows.

$$S_{2n} \times SO\left(n, \mathbb{F}_2\right) \xrightarrow{f} \mathbb{Z} \tag{5.1}$$

One such pseudo-Boolean function representing the graph isomorphism problem was first defined in [120]. This is still an open question whether this is the optimal way to capture the graph isomorphism problems given the family. Later it

has been refined in [187] by the author with others. The refinement allows one to solve larger graph isomorphism problem with lesser logical and physical qubits. I define the pseudo-Boolean function f in Definition 43.

Definition 43. *Let $\Gamma_1 = (V_1, E_1)$ and $\Gamma_2 = (V_2, E_2)$ be two simple, undirected*[1] *n-vertex graphs expressed as adjacency matrices A_1 and A_2 respectively. Also, let, the boolean variable $x_{u,i}$ be one for a permutation σ if $\sigma\,(i \in V_1) \mapsto u \in V_2$. I define pseudo-Boolean function $f : \mathbb{B}^{n^2} \to \mathbb{R}^+$ as follows,*

$$\begin{aligned} &\forall\,(deg(i) = deg(u) > 0; deg(j) = deg(v) > 0) : \\ f\,(x_{1,1}, \ldots, x_{n,n}) = &\sum_{u=1}^{n} \left(1 - \sum_{i=1}^{n} x_{u,i}\right)^2 + \sum_{u=1}^{n} \left(1 - \sum_{i=1}^{n} x_{i,u}\right)^2 \\ &+ \sum_{(i,j) \notin E_1, i \neq j} \quad \sum_{(u,v) \in E_2} x_{u,i} x_{v,j} + \sum_{(i,j) \in E_1} \quad \sum_{(u,v) \notin E_2, u \neq v} x_{u,i} x_{v,j} \end{aligned} \tag{5.2}$$

such that if Γ_1 and Γ_2 are isomorphic f is zero else positive.

In this report, I call the first two terms as the *permutation terms* and the last two terms in f as the *edge discrepancy penalty terms*. The *permutation terms* penalize f if the input is not a valid permutation. The *edge discrepancy penalty terms* penalize when the equivalent Ising model is not symmetric. In the quantum adiabatic version of the function, the *edge discrepancy penalty terms* are considered as the problem Hamiltonian and the *permutation terms* are considered as the penalty Hamiltonian [90].

[1] A simple undirected graph doesn't have any multi-edge or loop.

5.3 Mathematical perspective

In this section, I discuss about f from Definition 43 with a mathematical perspective.

5.3.1 Energy landscape

For an n-vertex graph isomorphism problems, f takes up to n^2 binary variables as arguments. A permutation σ maps the vertices, V_1, of the graph Γ_1 to the vertices V_2 of the graph Γ_2. All the input configurations in $\mathbb{B}^{n^2}$ do not correspond to valid permutations. For example, I consider the input
$(x_{1,1} = 1, x_{1,2} = 1, \ldots)$. In this input, both the vertices 1 and 2 of Γ_1 are mapped to the vertex 1 of Γ_2. This is not a valid permutation.

The *permutation terms* of f sets the baseline of the energy landscape. For an n-vertex graph isomorphism problem, there are 2^{n^2} possible input configurations for f. Among these configurations, there are $n!$ configurations which correspond to valid permutations. Figure 5.1, shows how the number of valid permutations grows against the number of all possible configurations as n increases.

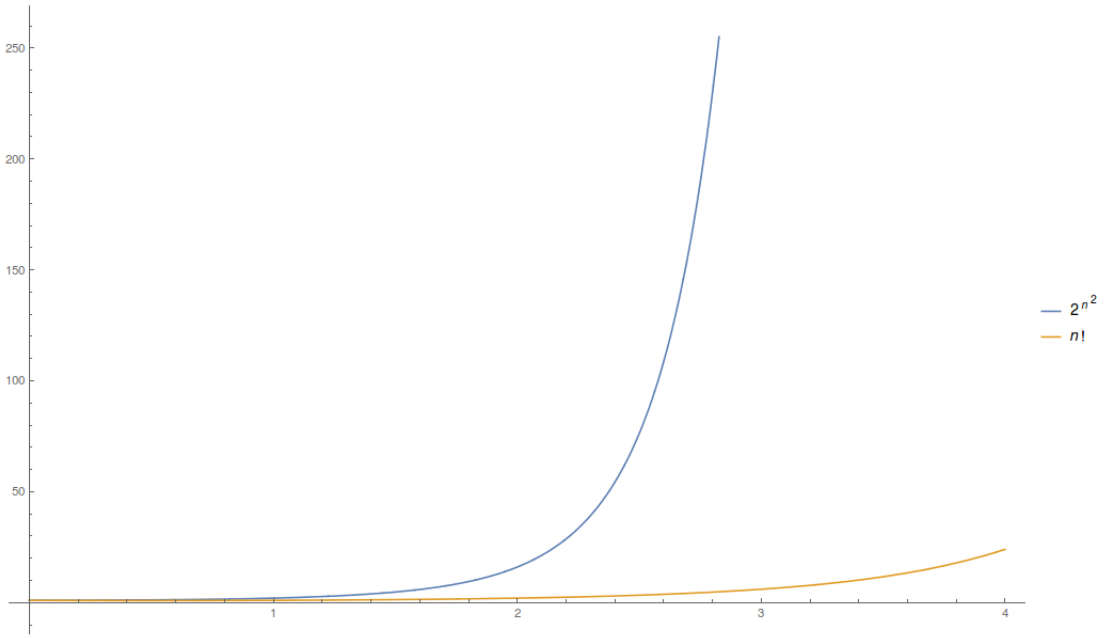


Figure 5.1: All possible configurations vs. valid permutations

The first two terms in f, $\sum_{u=1}^{n}\left(1-\sum_{i=1}^{n} x_{u,i}\right)^2$ and $\sum_{u=1}^{n}\left(1-\sum_{i=1}^{n} x_{i,u}\right)^2$, check whether a given input corresponds to a valid permutation. For valid permutations, the value of these two terms are zero. f is penalized when they are not. It would be helpful if we could figure out a way to input only the binary tuples which correspond to valid permutations. If we can do that, the *permutation terms* might be discarded and the function will be simpler to be implemented on a quantum annealer. At this moment, I think that we do not have enough control on a quantum annealing system which allows us to input only the configurations corresponding to valid permutations. Another way to speed up the annealing process is to modify the energy landscape so that it is easier to reach the global minima. But, modifying the landscape may lead to solving the problem itself while designing the function which will be inherently hard. Figure 5.2 shows the landscape of the values of the

permutation terms of f for a 4-vertex graph isomorphism problem.

Figure 5.2: Values of f for the permutation terms

I would like to comment that plotting energies against the configurations is mostly schematic and does not reflect the actual dynamics of the function. In future, I would like to plot the energies against time for both simulated and quantum annealing of the function. The Fourier transformation of the dataset of Figure 5.2 is shown in Figure 5.3.

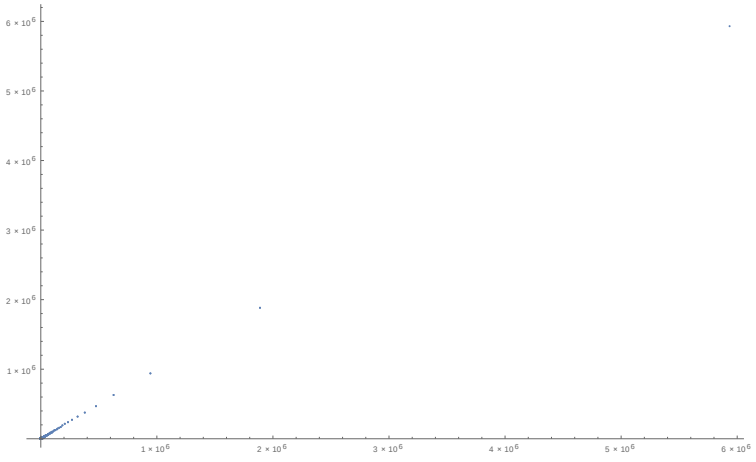


Figure 5.3: Fourier transformation of Figure 5.2

It is instructive to study the basins around the valid permutations in the energy

landscape of f. The 24 basins around $f = 0$ from Figure 5.2 are shown in figure 5.4.

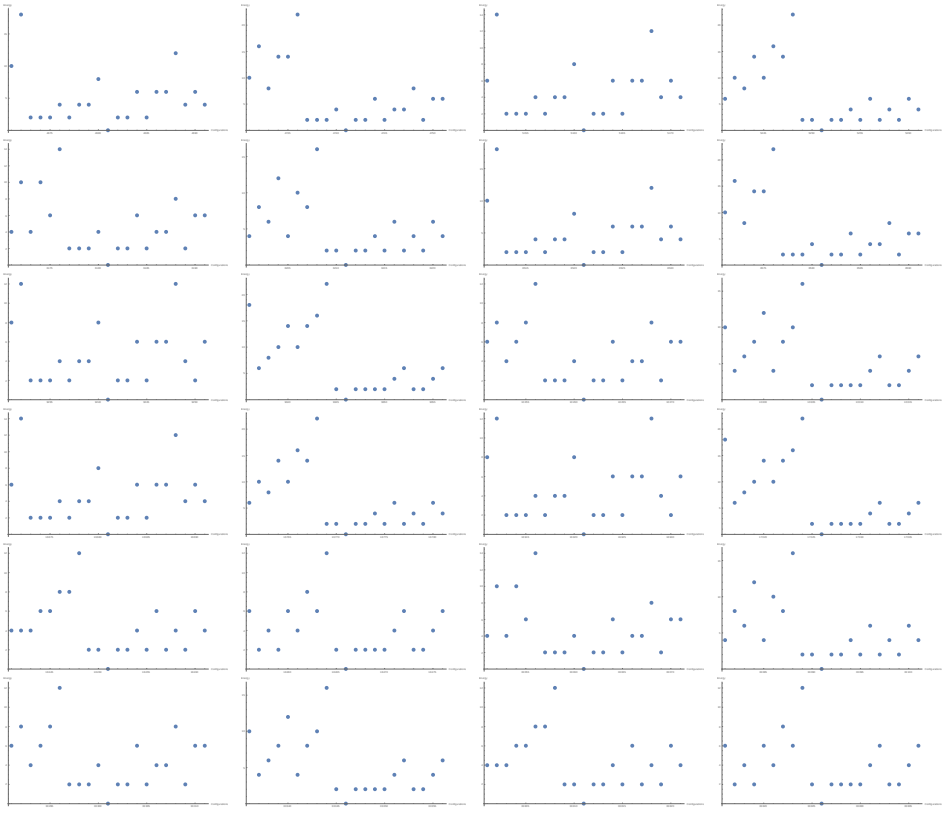


Figure 5.4: Basins of $f = 0$ for the permutation terms

The distribution of the energies of the *permutation terms* of f has a long tail to the right which is shown in Figure 5.5 for 4-vertex graph isomorphism problem.

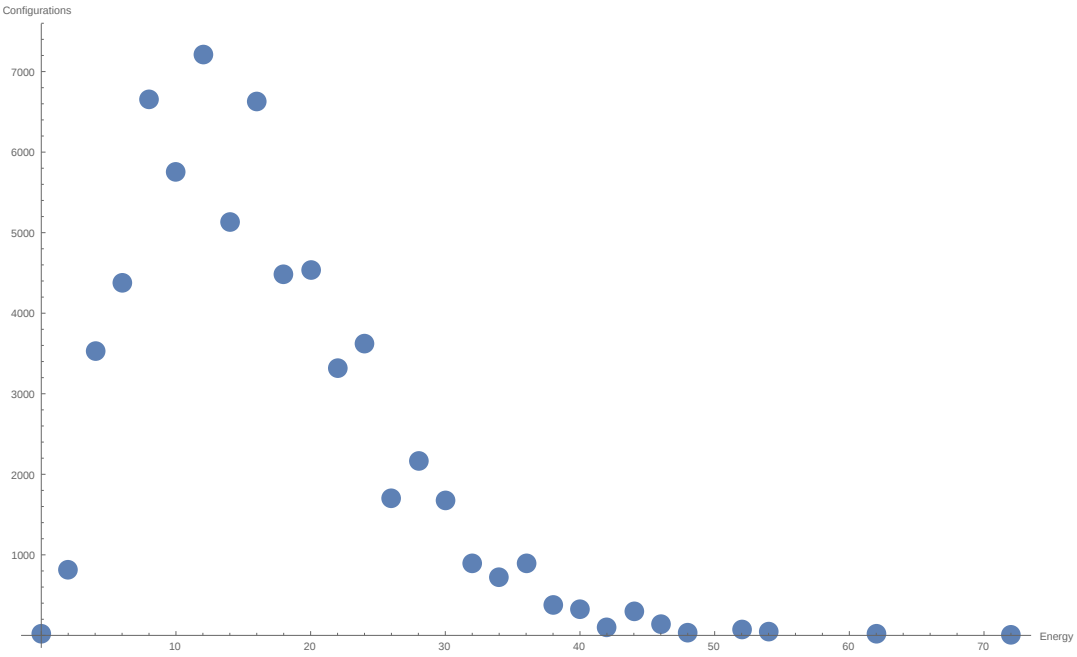


Figure 5.5: Distribution of energies of the *permutation terms.*

The distribution in Figure 5.5 is a skewed normal distribution with parameters 12, 7, and 0.1 as shown in Figure 5.6. I also need to verify whether this dataset is better represented by gamma or inverse gamma distribution.

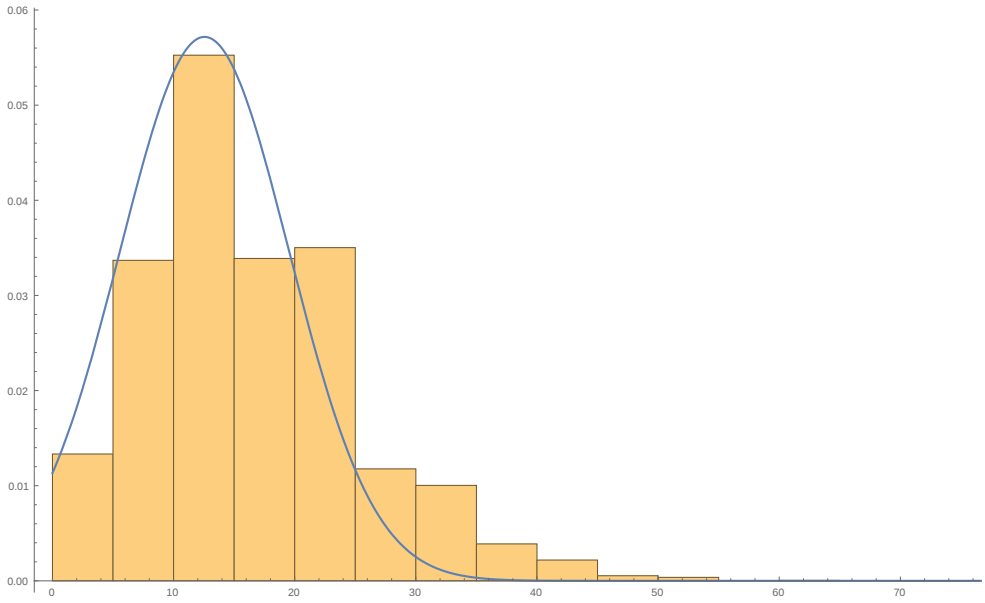


Figure 5.6: Figure 5.5 is skewed normal

I like to mention that, both *permutation terms*, $\sum_{u=1}^{n}(1-\sum_{i=1}^{n}x_{u,i})^2$ and $\sum_{u=1}^{n}(1-\sum_{i=1}^{n}x_{i,u})^2$, are required to penalize all possible invalid permutations. I also mention that f is a partial function hence it is not a bijection.

Among all the valid permutations, only a few or none could be isomorphic permutations. The *edge discrepancy penalty terms* filter out the permutations which are not isomorphic. If Γ_1 and Γ_2 are not isomorphic, the *edge discrepancy penalty terms* will penalize all the valid permutations evaluating f at a value greater than zero. When I are solving a graph isomorphism problem for Γ_1 and Γ_2, it is important to try to match a node from Γ_1 only with the nodes of same degree from Γ_2. The *edge discrepancy penalty terms* penalize f when a non-existing edge in Γ_1 is mapped to an existing edge in Γ_2. This kind of permutations tries to map a vertex from Γ_1 with a vertex with different degree from Γ_2 hence should be penalized.

The *edge discrepancy penalty terms* are different for different input graph pairs. Here, I consider two graph pairs - one isomorphic,

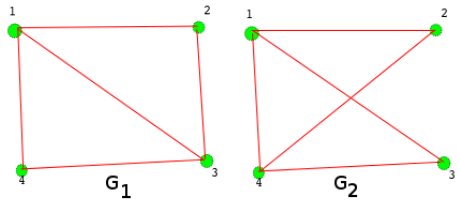


Figure 5.7: Isomorphic pair

and one non-isomorphic,

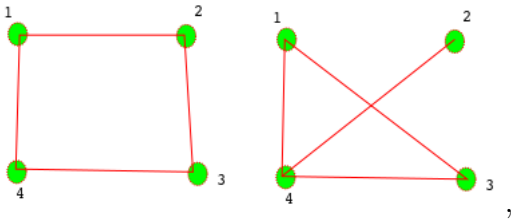
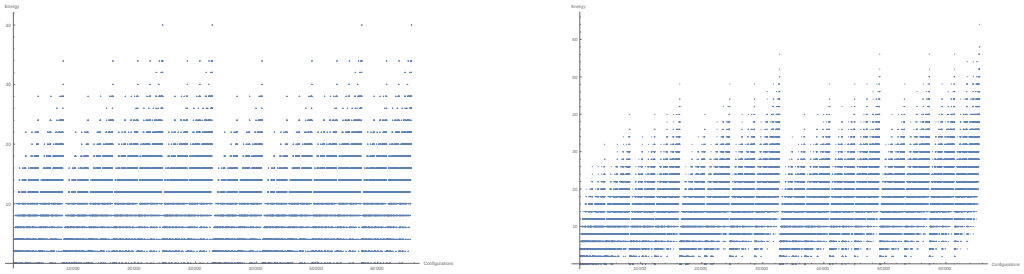


Figure 5.8: Non-isomorphic pair

both are instances of 4-vertex graph isomorphism problem.

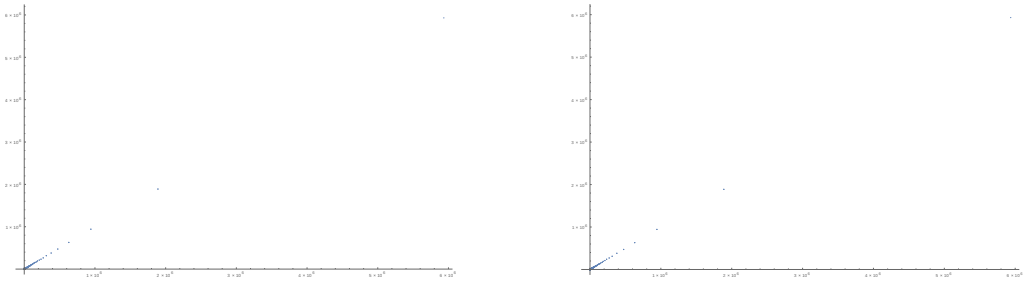
The energy landscape of the *edge discrepancy penalty terms* for the isomorphic and non-isomorphic pair is given in Figure 5.9.



(a) Isomorphic (b) Non-isomorphic

Figure 5.9: *Edge discrepancy penalty terms* energy landscape

The Fourier transformation of the dataset in Figure 5.9 is show in Figure 5.10.



(a) Isomorphic (b) Non-isomorphic

Figure 5.10: Fourier transformation of *Edge discrepancy penalty terms* energy landscape

As it is not obvious from Figure 5.10 how the Fourier transforms differ I plot

them jointly in Figure 5.11.

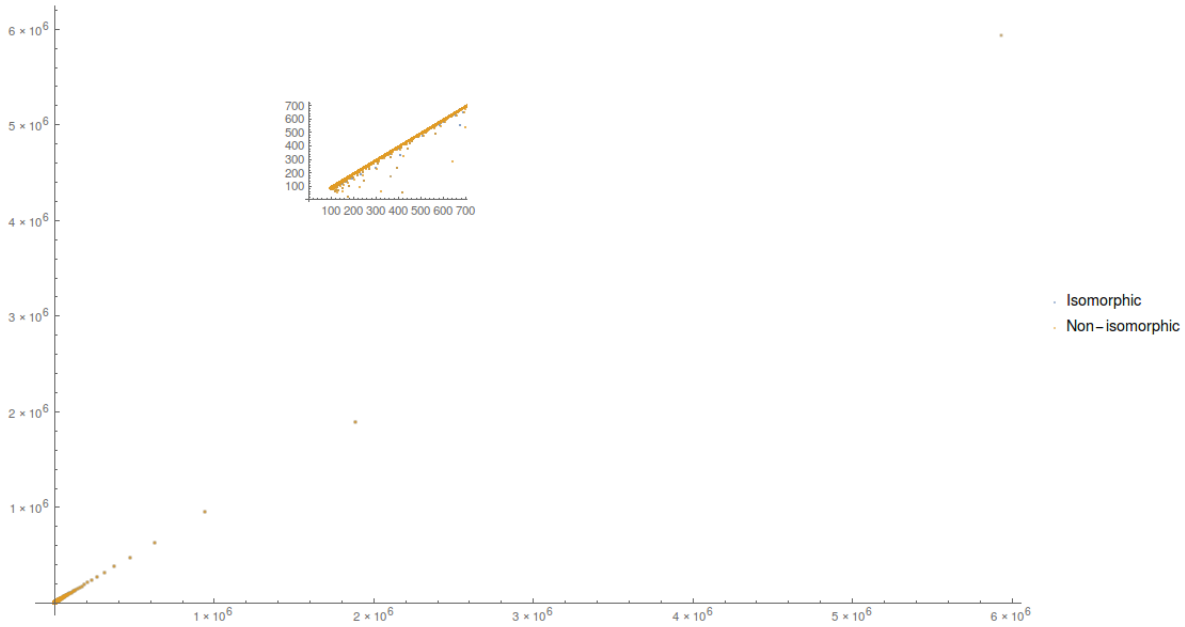
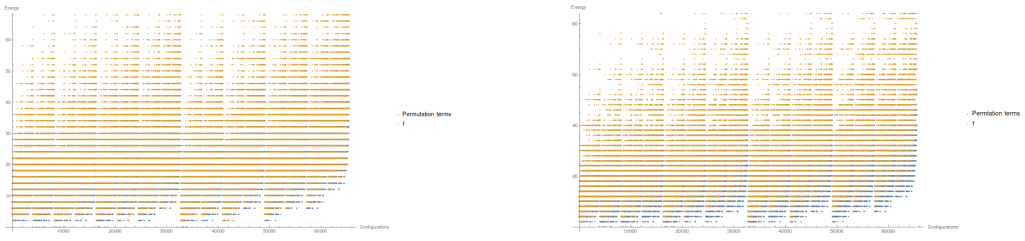


Figure 5.11: Fourier transformation of *Edge discrepancy penalty terms* energy landscape (superimposed)

Now I demonstrate how the *edge discrepancy penalty terms* modify the energy landscape of the *permutation terms* when both combine to construct f in Figure 5.12.



(a) Isomorphic (b) Non-isomorphic

Figure 5.12: *Permutation terms* vs. complete f

I would like to mention that, the plot for f (yellow) in the Figure 5.12 b never touches the X-axis as the *edge discrepancy penalty terms* lifts all the global minima above zero. On the other hand, f touches the X-axis fewer times than before (24)

in Figure 5.12 (a) as the *edge discrepancy penalty terms* lift the energies of all the non-isomorphic but valid permutations above zero.

I revisit the basins of the surviving global minimas of Figure 5.4 remaining in the Figure 5.12 in Figure 5.13.

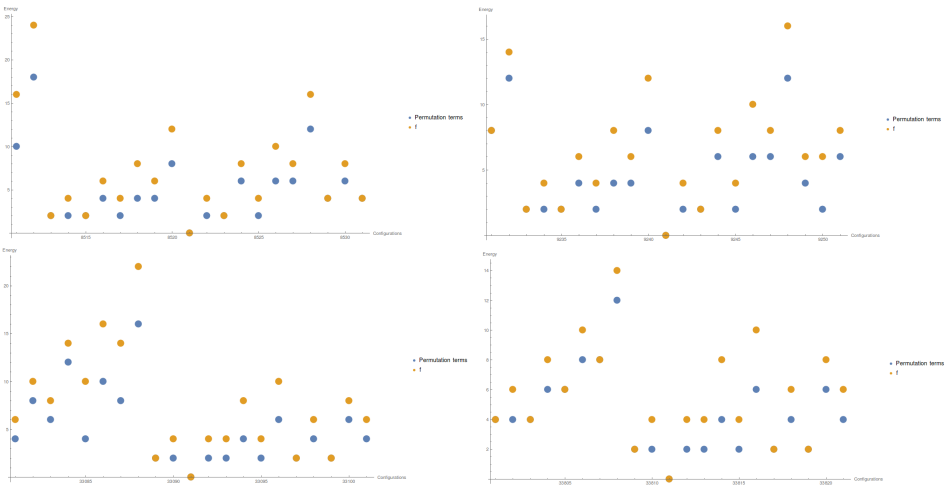


Figure 5.13: Basins of global minima of f for 4-vertex isomorphic pair

In Figure 5.13, the yellow curves correspond to the basins of the global minima of f and the blue curves are the basins of the global minima of the *permutation terms*. While the point of minima is fixed at 0, the basin is lifted upwards by the *edge discrepancy penalty terms*. At this moment, I do not know how to interpret the dataset for non-isomorphic pair other than the fact that all the energy values are greater than zero hence the input graphs are non-isomorphic. I observe that the global minima for non-isomorphic pair may contain both valid and invalid permutations. I propose to investigate in future whether more information could be extracted from the dataset.

The *edge discrepancy penalty terms* also redistribute the probabilities of different configurations from Figure 5.5 to Figure 5.14.

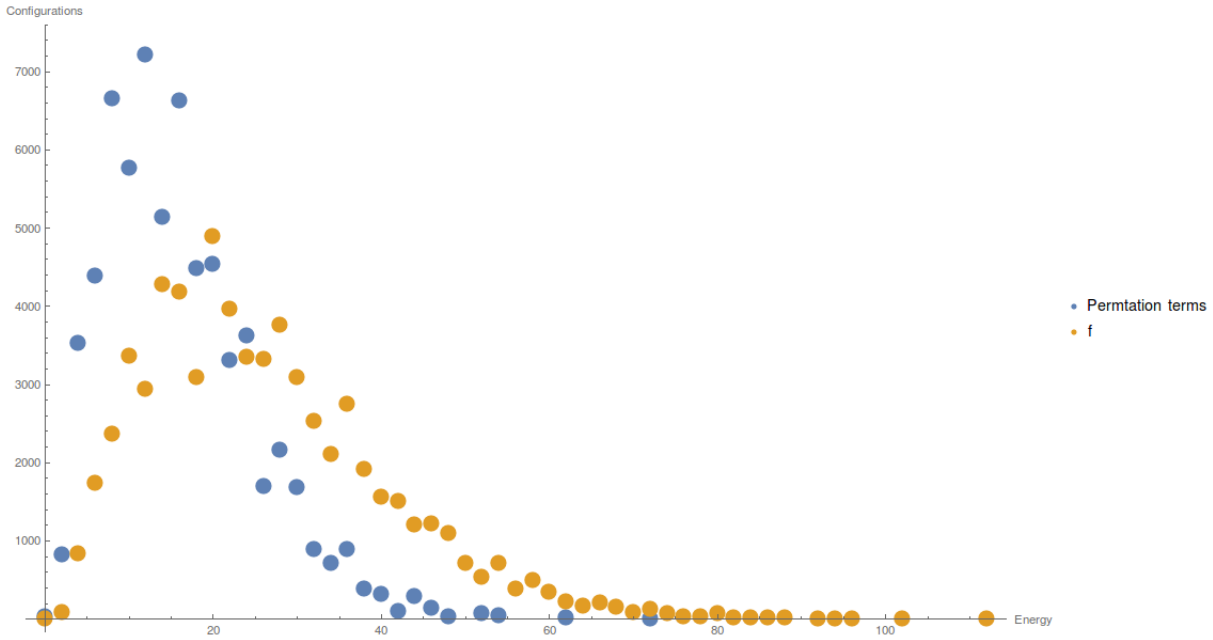


Figure 5.14: Distribution of energies of f (yellow)

5.4 Generating the function

I reproduce the pseudo-code of the algorithm, a previous result by my collaborators and I, to generate the corresponding pseudo-Boolean function, f, from the adjacency matrix representation of a graph isomorphism problem from [187] in Algorithm 6.

Algorithm 6 QUBO matrix generator algorithm

procedure QUBO-MATRIX-GENERATE(Γ_1, Γ_2) ▷ Define QUBO variables
  for each node v in Γ_2 **do**
    for each node v in Γ_1 **do**
      if $deg(v) == deg(i)$ and $deg(i) > 0$ **then** ▷ Create QUBO variable $x_{v,i}$
      end if
    end for
  end for
  ▷ Populate QUBO matrix with penalty terms
  for each pair of different QUBO variables $x_{u,i}$ and $x_{v,j}$ **do** ▷ Assign penalty for node mapping conflict
    if $i == j$ **then**
      penalizing mapping $u \to i$, $v \to j$ ▷ Two nodes in Γ_2 map to the same node in Γ_1
    else if $u == v$ **then**
      penalizing mapping $u \to i$, $v \to j$ ▷ A node in Γ_2 maps to two nodes in Γ_1 ▷ Assign penalty for edge discrepancy
    else if $\Gamma_1(i,j) \neq \Gamma_2(u,v)$ **then**
      penalizing mapping $u \to i$, $v \to j$ ▷ Edge in one graph, not edge in other
    end if
  end for
  for each QUBO variable **do**
    assign value along diagonal of QUBO matrix
  end for
end procedure

Since the model is restricted to pairwise interactions over binary variables, it represents a quadratic unconstrained binary optimization (QUBO) formulation, which can be readily converted to an Ising model.

The approach embodied in the Hamiltonian defined in [119] suffers from a severe lack of scalability. For N-vertex input graphs, it requires N^2 logical variables. Moreover, due to the limited direct connections between qubits in the D-Wave Chimera architecture, problems are often given a minor embedding into the

processor working graph. This typically involves replicating variables across multiple qubits. Thus, the qubit requirements can reach $O(N^4)$. Problems mapped in this way to a 500-qubit processor tend to be limited to $N = 5$ or 6. Our Hamiltonian is more effective. The idea is that many variables and interactions are unnecessary, and information indicating so can be leveraged up front during the requisite pre-processing. Note that an isomorphic mapping requires the vertices in each matched pair to have the same degree. Thus, degree information can be used to decide whether two vertices are eligible to be matched. I propose the new Hamiltonian which avoids creating variables for vertices of different degree. A second, minor simplification deals with isolated vertices (degree=0). If $G_{1,2}$ each have k isolated vertices, an isomorphic mapping of such vertices is trivial and thus no variables or penalties for those vertices need be modeled. If $G_{1,2}$ have a different number of isolated vertices, then they also have a different number of non-isolated vertices and existing variables and penalties for those will suffice. Thus, I we only create variables and penalties for vertices with degree greater than zero.

Figure 5.15 shows an example of problem instances generated using the baseline and the compact Hamiltonians on the same input; it illustrates how the number of variables and the number of non-zero interactions - as seen in the off-diagonal entries in the associated QUBO matrix Q can be much smaller when using the compact Hamiltonian. The variable scaling of each Hamiltonian is quantified and compared later.

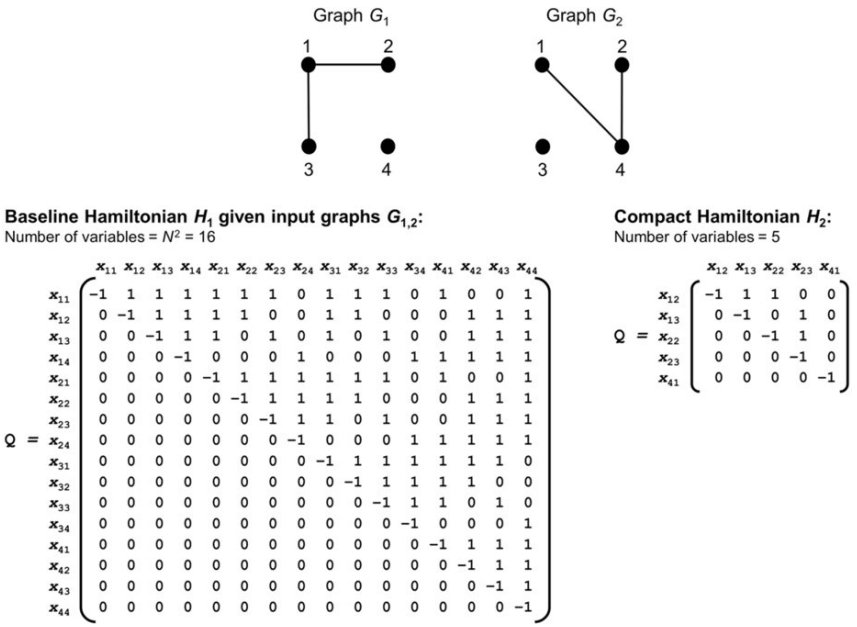


Figure 5.15: Illustration of problem instances generated using baseline Hamiltonian H1 and compact Hamiltonian H2 on the same input.

5.5 Larger graph isomorphism problems

The quantum annealing of a pseudo-Boolean function is limited by the architectural primitives of the hardware. In [187], it has been shown that the pseudo-Boolean function can solve only up to 13 node graph isomorphism problems. Practical graph isomorphism may involve graphs with even million nodes [127, 128]. So, to circumvent the limitation due to architectural primitives (e.g. number of physical qubits, connectivity of the hardware graphs, precision of external magnetic field or coupling strength), I need to apply divide-and-conquer. Applying divide-and-conquer to scale up graph isomorphism algorithm has already been a known technique [121, 127]. The ideas have been streamlined in Babai's recent paper [12] which proves that the graph isomorphism problem can be solved in quasipolynomial time. An intuitive approach for divide-and-conquer is to compute the degree

sequences of the input graphs, partition them based on the degrees, find isomorphism between the equivalent partitions and if the partitions are still to big go on recursively. Algorithm 7 illustrates the idea.

Algorithm 7 DIVIDE-AND-CONQUER-GRAPH-ISOMORPHISM

```
1: procedure DIVIDE-AND-CONQUER-GRAPH-ISOMORPHISM(Γ_1, Γ_2)   ▷ Define QUBO variables
2:     Compute the degree sequence of Γ_1
3:     Compute the degree sequence of Γ_2
4:     Partition Γ_1 and Γ_1 based on their ordered degrees.
5:     for All degrees do
6:         if The partition size is not limited by architectural primitives then
7:             Run the pseudo-Boolean representation of the partition isomorphism problem on quantum annealing computer
8:         else
9:             Invoke DIVIDE-AND-CONQUER-GRAPH-ISOMORPHISM recursively on the partitions
10:        end if
11:    end for
12: end procedure
```

A major obstacle for Algorithm 7 appears when the graphs are regular. In that case, all the partitions are of same size hence there is no advantage in divide-and-conquer. In that situation one could partition the graphs into equitable partitions which is a partition of vertices by degree relative to the vertices in other blocks of the partition [2].

A phenomenal results of [12] is the Theorem 1.3.3. In the theorem, Babai proves that any non-trivial regular graph, $\Gamma = (V, E)$, can be partitioned into $\{Y_i\}$'s at a quasipolynomial multiplicative cost as follows:

[2]So, a vertex which has degree 3 but two degree 2 neighbors will be in a different block than a vertex with degree 3 and only 1 neighbor of degree 2.

- A coloring of V with no color-class larger than $0.9n$;
- A coloring of V with a color-class C of $size \geq 0.9n$ and a nontrivial equipartition of C (the blocks of the partition are of equal $size \geq 2$ and there are at least two blocks);
- A coloring of V with a color-class C of $size \geq 0.9n$ and a Johnson graph $J(v,t)\,(t \geq 2)$ with vertex-set C,

such that the index of the subgroup $Aut(X) \cap Aut(Y)\, in Aut(X)$ is quasipolynomially bounded. This is the famous *split-or-Johnson* technique due to Babai.

In future, I would like to investigate whether a Babai-inspired divide and conquer approach is feasible for solving graph isomorphism problems on a quantum annealing computer. One may also be interested to investigate the Johnson graph isomorphism problem via quantum annealing in future.

5.6 Empirical scalability

The scalability of the pseudo-Boolean representation of the graph isomorphism problem, f, has been studied empirically in [187]. The number of variables in f has been estimated to be the function of n, $0.748n^{1.45}$. If I let n grow comparable to the size of the inputs in practical graph isomorphism problems [127], i.e. $n = 1000$, the number of variables in f will be $\lfloor 0.7481000^{1.45} \rfloor = 16745$. On a 512-qubit D-Wave computer, it may take $\lfloor 0.07231000^{3.29} \rfloor = 5.35967 \times 10^8$ physical qubits which is impractical to be build with current technologies. So, it would be useful to

understand the asymptotic behavior of f to see if a better scaling behavior may be investigated from further theoretical understanding of the problem.

5.7 Asymptotic analysis of Algorithm 6

In the asymptotic analysis of the Algorithm 6, it is assumed that the inputs are randomly generated simple undirected graphs. The number of possible simple undirected graphs with n vertices is $2^{\binom{n}{2}}$ [150]. This constructs the super space of the input space.

The best, average and worst cases for Algorithm 6 are identified based on the *total number of variables* and *the numbers of first and second order terms* in f. The total number of variables determine the number of logical qubits needed to be embedded on a quantum annealing computer. For example, the recently decommissioned 512-qubit D-Wave computer was able to embed only up to 50 logical qubits through heuristics [1]. The number of second order terms involving a variable determines the length of the chain of the physical qubits needed to encode the corresponding logical qubit. The longer the chain, the more it tends to break down, i.e. all the physical qubits in a chain not having the same value, requiring post-quantum-annealing error correction procedure, for example, *gauge averaging* [25] or *majority voting* [187].

The best possible input scenario to Algorithm 6 is when the input graphs are either *completely connected* i.e. with degree $n-1$ or *totally disconnected* i.e. with degree 0 if the total number of vertices is n. In either case, the number of penalty

terms will be zero. So, I will have fewest terms possible in f.

Proposition 1. *Completely connected or totally disconnected graphs are the best input scenarios for Algorithm 6.*

Proof of Proposition 1. For any graph of fixed size, the number of the *permutation terms* are always the same. So, the best case means when I have best case for the number of the *edge discrepancy penalty terms*. For a completely connected or totally disconnected graph, the number of the *edge discrepancy penalty terms* is the lowest possible i.e. zero. □

I assume that the input graphs are generated in random. There is more than one model of generating random graphs. The two common models of generating random graphs are the *uniform random graph model* , $\mathbb{G}(n, M)$ where n is the number of vertices and M is the number of edges [55, 56, 58, 60, 62, 63, 59], and the *binomial random graph model*, $\mathbb{G}(n, p)$ where n is the number of vertices and p is the probability of two vertices being connected with an edge [76]. It has been shown that both models are *asymptotically equivalent* [102]. The name Erdős-Rényi model is used interchangeably for both models. I list a few facts about random graphs as follows.

In [58], using the $\mathbb{G}(n, M)$ model, Erdős et al., proved that the probability of a random graph being without any isolated vertex is $e^{-e^{-2c}}$. Here c is a constant when the number of vertices is n and the number of edges is $\left[\frac{1}{2}n \log n + cn\right]$. This is related to our discussion as it is assumed that Algorithm 6 takes as input only graphs without any isolated vertex. The total number of possible random graphs

constitutes the input space for Algorithm 6.

Let $d_1 \geq \ldots \geq d_m \geq \ldots \geq d_n$ be the degree sequence of a random graph with n vertices. The worst case for the total number of variables in the output of Algorithm 6 occurs when $d_1 = \ldots = d_m = \ldots = d_n = r$ i.e. it is an r-regular graph [3]. As I already have discussed the best cases it is assumed that $0 < r < n - 1$.

Lemma 4. *The total number of variables in the output of Algorithm 6 is maximum if the input graphs are r-regular and same for any value of r when $0 < r < n - 1$.*

Proof of Lemma 4. The total number of the first order terms in f is determined by the first outer for loop of Algorithm 6. When the input graphs are r-regular, i.e. $d_1 = \ldots = d_m = \ldots = d_n = r$, for all $i \in V(\Gamma_1))$ and $v \in V(\Gamma_2))$, $deg(v) = deg(i) = r$. So the if condition is true all the time and the number of QUBO variables is n^2 for any value of r. □

Remark 4. *The total number of first order terms in the output of Algorithm 6 is completely determined by the total number of variables in f. Hence the worst case for the total number of first order terms occurs when the input graphs are r-regular.*

The frequency of r-regular graphs from a random graph generator model is a well studied area of research [153, 154, 150, 86, 21, 57, 147, 19, 20, 27, 29]. It is noteworthy that different models other than $\mathbb{G}(n, M)$ and $\mathbb{G}(n, p)$ have been used to generate *random regular graphs*. Those models turn out to have different asymptotic properties from the two basic models $\mathbb{G}(n, M)$ and $\mathbb{G}(n, p)$ [102]. The **RANGRAPH** algorithm in [185] can generate a random r-regular graph with

[3] An r-regular graph is a graph which has all vertices with degree r[85].

probability of success about $e^{\frac{1-r^2}{4}}$ which is prohibitively small for large r. [174] gives a scheme based on Markov processes to generate random graphs with given degrees in polynomial time as long as the degrees are bounded above by a polynomial function of the number of edges. [129] uses a switching scheme to generate r-regular graphs on n vertices uniformly at random in $O(nr^3)$ time as long as $r = O\left(n^{\frac{1}{3}}\right)$. [160] provides an algorithm which generates random regular graphs *asymptotically almost surely* but not in a uniformly distributed manner.

The standard model for generating random r-regular graph is Bollobás 's version of the *configuration model* [21, 27, 29]. According to the model, the total number of random regular graphs for a fixed r is $L_n \sim \sqrt{2}e^{-\frac{\left(r^2-1\right)}{4}} \left(\frac{r^{\frac{r}{2}}e^{-\frac{r}{2}}}{r!}\right)^n n^{\frac{rn}{2}}$ when $n \to \infty$.

So, among the $2^{\binom{n}{2}}$ simple undirected graphs over n vertices, only $\sim \sqrt{2}e^{-\frac{\left(r^2-1\right)}{4}} \left(\frac{r^{\frac{r}{2}}e^{-\frac{r}{2}}}{r!}\right)^n n^{\frac{rn}{2}}$ are r-regular. These fraction of graphs constitute the worst case input space for Algorithm 6.

The first two terms in f also generates several second order terms which are always same for any input graph pair over n vertices. So, the difference over the second order terms comes from the last two or *edge discrepancy penalty terms*.

Along with the highest number of the variables, I identify it as the worst case when the numbers of first order and second order terms in f are maximum. The number of first order terms is maximum when the input graphs are regular. Now I will determine when the number of second order terms is maximum.

Before going further I prove a lemma here.

Lemma 5. *The product of the bipartition of an integer is maximum when the partitions are the closest to half of the integer.*

Proof of Lemma 5. Let n be the integer and $(r, (n-r))$ be a partition. So, the product is $r\,(n-r) = nr - r^2$. This is parabolic function of r where the maximum is at $r = \frac{n}{2}$. □

Lemma 6. *Given both graphs are r-regular, the total number of second order terms created from the edge discrepancy penalty terms in f is highest when r is closest to $\frac{n-1}{2}$.*

Proof of Lemma 6. The total number of second order terms in the *edge discrepancy penalty terms*, $\sum_{(i,j)\notin E_1, i\neq j} \sum_{(u,v)\in E_2} x_{u,i}x_{v,j}$ and $\sum_{(i,j)\in E_1} \sum_{(u,v)\notin E_2, u\neq v} x_{u,i}x_{v,j}$, of f, is maximum when all the edge-non-existing-edge combinations are considered for penalization in both terms.

If the total number of edges in a complete graph is $\frac{n(n-1)}{2}$, let $|\bar{E}_1|$ be $|\bar{E}_1| = \frac{n(n-1)}{2} - |E_1|$. In that case, $|E_1|$ will also be the number of edges of the $n-1-r$-regular graph complement to E_1 [152].

So, the total number of second order terms is $|\bar{E}_1||E_2| + |E_1||\bar{E}_2|$.

The number of edges in the r-regular n-vertex graph is $|E_1| = \frac{nr}{2}$ [183]. Similarly, the number of edges in an $n-1-r$-regular n-vertex graph, Γ_2, which is the complement of $|\Gamma_1|$, is $|E_2| = \frac{n(n-1-r)}{2}$.

The number of *edge discrepancy penalty terms* is $|\bar{E}_1||E_2| + |E_1||\bar{E}_2|$. So,

$$|\bar{E}_1||E_2| + |E_1||\bar{E}_2| = \frac{n\,(n-1-r)}{2}\frac{nr}{2} + \frac{nr}{2}\frac{n\,(n-1-r)}{2}$$
$$= \frac{n^2 r\,(n-1-r)}{2} \tag{5.3}$$

As n is constant, the value will be maximum when $r\,(n-1-r)$ is maximum. It is important to mention that $n-1$ is also constant and $(r, (n-1-r))$ is a bipartition of $n-1$. So, the maximization of the number of second order terms boils down to the maximization of the product of bipartition of the integer $n-1$.

Using Lemma 5, I can say that the maximization occurs when $r = \frac{n-1}{2}$. □

Theorem 15. *The worst case of f occurs when the input graphs are $\frac{n-1}{2}$-regular.*

Proof of Theorem 15. From Lemma 5 and 6, I can say that the worst case for f occurs when the input graphs are $\frac{n-1}{2}$-regular graphs. □

In future, it may also be relevant to know the asymptotic results for the *strongly regular graphs*.

5.8 Experimental setup

My Hamiltonian was validated using a software solver (D-Wave Systems ising-heuristic version 1.5.2) that provides exact results for problems with low tree width. In exhaustive testing of all 2^{12} $N = 4$ pairs and 2^{20} $N = 5$ pairs, the ground state energy of our Hamiltonian was confirmed to be zero for isomorphic cases and greater than zero for non-isomorphic.

The high-level flow of the GI solver I am using is shown in Figure 5.16. The problem input is a graph pair $G_{1,2}$. In this article, the graph types considered are random, simple, undirected graphs. Graphs are generated using the Erdős-Rényi model [59] $G(N, p)$ where I set the probability of an edge being present $p = 0.5$. An advantage of $G(N, 0.5)$ graphs for an initial study is that all graphs are equally likely. This type has been used in classical graph isomorphism work as well [128]. In step 1, the input graphs and the Hamiltonian formulation of interest are used to generate a QUBO problem which is then converted to an Ising problem $[h, J]$. An example of an algorithm for generating the QUBO problem is shown in Algorithm 6. The Ising problem is then compiled to a specific quantum annealing processor in step 2. A main task is to find sets of physical qubits to represent the problem variables (aka logical qubits); this is achieved by providing the J matrix and the processor working graph to the D-Wave `findEmbedding()` heuristic [35]. Subsequently, the parameters of the embedded Ising problem $[h', J']$ are set following certain strategies such as the use of a random spin gauge. The embedded Ising problem, sometimes referred to as a machine instruction, is submitted to the quantum annealing machine along with several job parameters. The quantum annealing job is executed in step 3 and solutions are returned in the form of strings of two-valued variables. These solutions and energies are associated with the embedded problem, not the original Ising problem. Therefore a post-processing step is necessary (step 4), in which the state of each qubit chain is plugged into the cost function of the Ising problem. A difficulty arises when the states of the qubits in a chain are inconsistent, a case referred to as a broken chain. In the proposed solver, broken chains can be handled

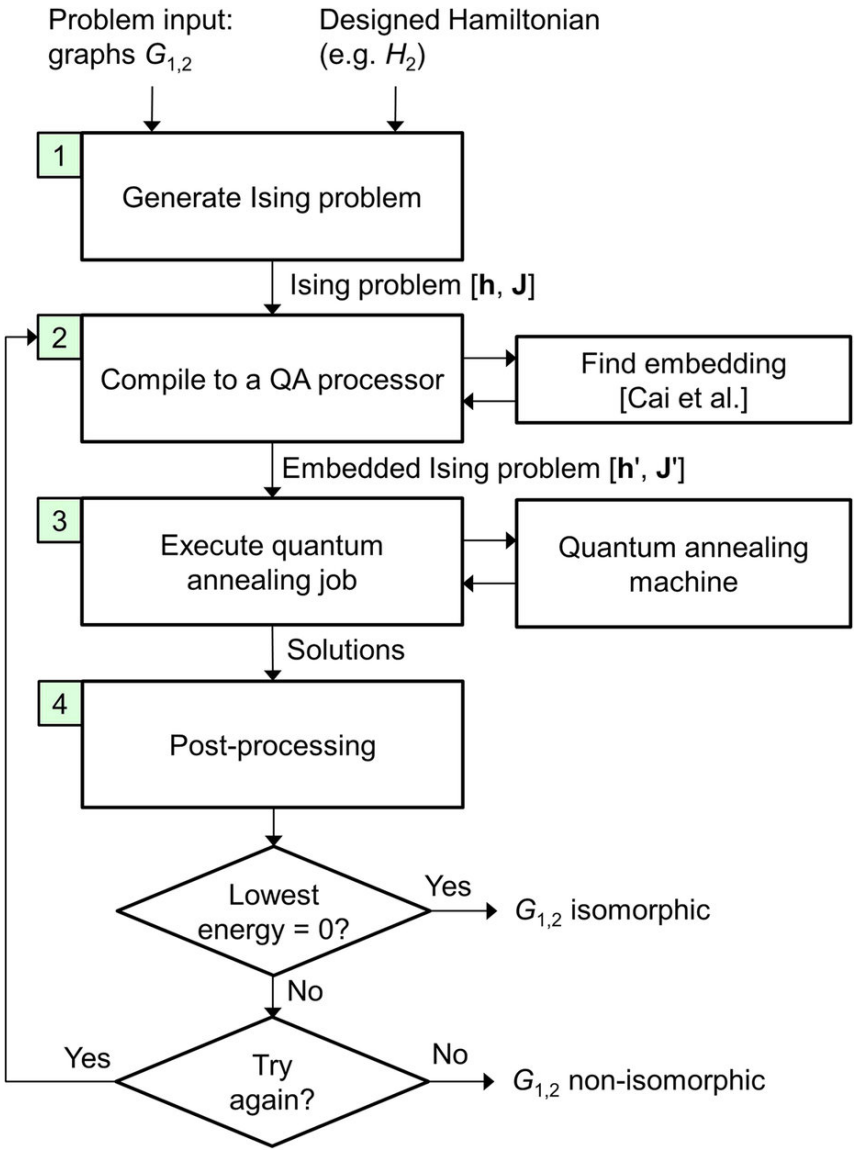


Figure 5.16: Experimental quantum annealing: case study involving the graph isomorphism problem.

by either discarding the associated solution, or by performing majority voting over each chain. The two strategies are compared empirically in Results. Given a solution to the original Ising problem, the solution energy can be calculated. If the lowest energy is zero, then the input pair can be declared isomorphic and no further jobs are necessary. Otherwise, a decision must be made whether to repeat the process from step 2 or to stop and declare that isomorphism could not be established.

5.9 Results

5.9.1 Ising Model Scaling

To compare the resource requirements of the two proposed Hamiltonians, 100 pairs of graphs are used as inputs to Step 1 of the solver flow (Figure 5.16), where 50 pairs are isomorphic and 50 are non-isomorphic for each size up to $N = 100$. In a few places of the rest of the chapter, the Lucas Hamiltonian is denoted as H_1 and our proposed Hamiltonian is denoted as H_2 also as the compact Hamiltonian. So, H_2 and f, defined in Eq. 5.2, will be used interchangeably and H_1 will correspond to f when all the conditions on the degree of vertices in Eq. 5.2 will be removed. So, Since the original Lucas Hamiltonian models a variable for each possible vertex pair, N^2 variables are required by definition. Ising problems generated using our proposed Hamiltonian are found to use fewer variables; scaling of the median problems fits to $0.748N^{1.45}$. Incidentally, this indicates that most problems have fewer variables than with the Gaitan et al. approach, which entails plus ancillary variables [74]. The variable scaling is illustrated in Figure 5.17. In addition to the number of

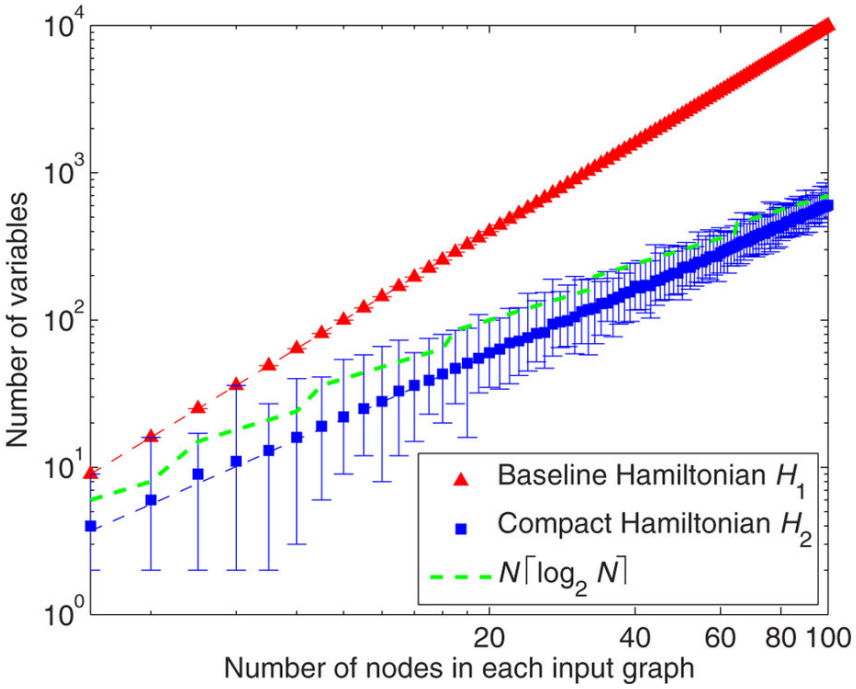


Figure 5.17: Scaling of the number of Ising model variables.

variables, a second resource metric is the number of non-zero interactions between variables; dense interactions make the minor embedding problem more difficult. I find that the scaling of variable interactions has been improved from $O(N^4)$ for the original Lucas Hamiltonian to $O(N^{2.9})$ for our Hamiltonian (where $R^2 = 0.9991$).

5.9.2 Embeddability

Next, I compare the embeddability of the two approaches, in other words the extent to which Ising problems can be minor-embedded in a given processor graph. The processor of choice is the D-Wave Two Vesuvius-6 processor housed at USC ISI. At the time of this experiment, the working graph contained 504 qubits and 1427 couplers. Embedding is attempted using the D-Wave `findEmbedding()` heuristic [35] with default parameter values such as 10 tries per function call. As shown in

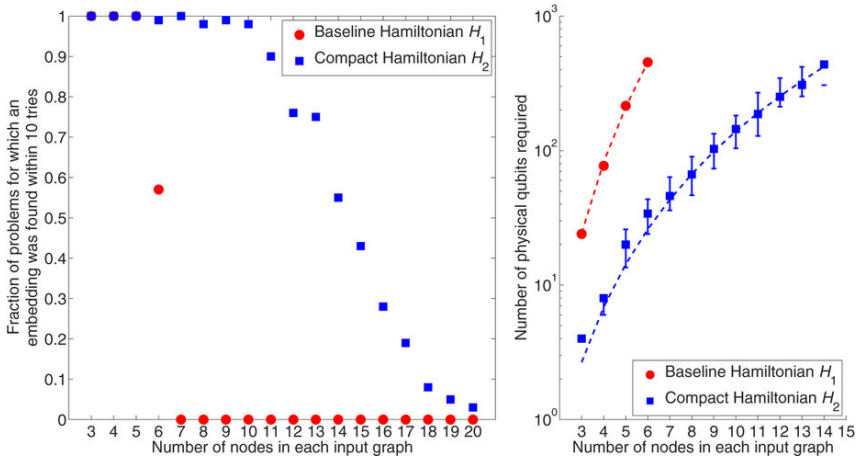


Figure 5.18: Embeddability when targeting the USC-LM Vesuvius processors 504-qubit, 1427-coupler working graph.

Figure. 5.18 a, embeddings are found for the majority of problems only for sizes $N \leq 6$ when using the Lucas Hamiltonian, but sizes $N \leq 14$ with the Hamiltonian proposed by us (Figure. 5.18a). The median number of qubits across all problems scales as $O(N^{4.22})$ for the Lucas Hamiltonian and has been reduced to $O(N^{3.29})$ for our proposed Hamiltonian (Figure 5.18b).

5.9.3 Experimental Quantum Annealing for Graph Isomorphism

The accuracy of the solver described in the previous section was measured via trials conducted on a D-Wave Two Vesuvius quantum annealing processor. Several alternative strategies were compared - the use of Hamiltonians H_1 vs. H_2, running a single job per problem vs. multiple jobs, and the use of chain majority voting during post-processing. Note that by construction of the Ising models using a penalty Hamiltonian, problems with non-isomorphic input graphs cannot achieve a zero energy state, regardless of annealing results. The main challenge for the solver is to find the zero energy state for isomorphic pairs. Thus, I first focus on the isomorphic

case. One hundred isomorphic pairs were input into the solver for each size N from 3 to 20.

For one strategy in particular the zero energy state was always eventually achieved - the use of Hamiltonian H_2 combined with multiple jobs and chain majority voting. Thus, with this strategy there were no false negatives and classification accuracy reached 100% of the embeddable problems, as shown in Table 5.1. For the most difficult problem, the zero energy state was achieved on the 9th job. All other strategies incurred false negatives. For the successful strategy, the expected total annealing time was calculated (as described later). Results are shown in Figure 5.

Table 5.1: **Number of isomorphic-input problems embedded and correctly classified as isomorphic via quantum annealing.** One hundred problems were attempted at each problem size. All embedded problems were solved when using H_2, error correction, and multiple jobs.

	Size of input graphs (number of vertices)															
	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Number of problems for which an embedding was found using H_1	100	100	100	74	0	0	0	0	0	0	0	0	0	0	0	0
Number of problems for which an embedding was found using H_2	100	100	100	99	99	96	95	93	87	83	54	39	20	7	4	1
Number of problems solved using H_1, no error correction	100	99	98	13	0	0	0	0	0	0	0	0	0	0	0	0
Number of problems solved using H_1, error correction	100	100	99	43	0	0	0	0	0	0	0	0	0	0	0	0
Number of problems solved using H_2, no error correction	100	100	100	98	96	93	86	75	67	53	34	23	9	1	2	0
Number of problems solved using H_2, error correction	100	100	100	99	99	96	90	80	73	65	40	25	12	3	2	1
Number of problems solved using H_2, error correction, multiple jobs	100	100	100	99	99	96	95	93	87	83	54	39	20	7	4	1

For completeness, non-isomorphic pairs were run as well, using Hamiltonian H_2 and chain majority voting. Since in the worst case nine jobs were required to correctly classify the isomorphic pairs above, nine jobs were submitted for each non-isomorphic problem. One hundred non-isomorphic $G(N, 0.5)$ problems were tested at each size between $N = 3$ to 14; of the 1200 problems, embeddings were found for 1186. In addition, pairs of isospectral non-isomorphic graphs (PINGs) were tested. All $N = 5$ PINGs were tested (150 permutations), as well as 100 random $N = 6$ PINGs. As expected, none of the non-isomorphic problems achieved a zero energy

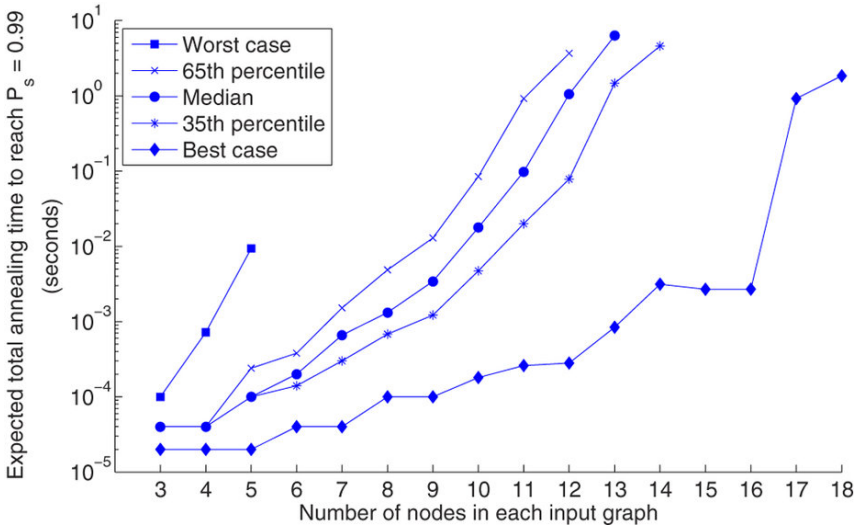


Figure 5.19: Total expected annealing time when using Hamiltonian H_2, multiple jobs, and classical majority voting.

state and thus none were classified as isomorphic. In other words, there were no false positives.

5.10 Discussion

Several observations can be made from this case study. First, the formulation of the cost function (Hamiltonian) can have a noticeable impact on quantum annealing results. For the graph isomorphism problem, the baseline approach (embodied in Hamiltonian H_1 and in [120]) blindly creates QUBO variables for every possible vertex pair, whereas the proposed Hamiltonian H_2 is more parsimonious. Variable requirements decreased from N^2 to fewer than $N \log_2 N$ (Figure 5.17) on the graph type under study, allowing larger problems to be solved (Figure 5.18 and Table 5.1). Along with Rieffel et al [157], this is one of the first quantum annealing studies to experimentally quantify the effect of alternative Hamiltonian formulations.

One of the impacts of this observation is increased appreciation for the fact that all quantum annealing-based solvers are actually classical-quantum hybrids and that focus must be placed on effectively partitioning the processing and optimizing the classical portion. A caveat is in order - if the classical side is made to do too much work then the quantum annealing aspect becomes trivial and of little value. Further work is needed in identifying the specific strengths of annealing processors, and in leveraging the two sides appropriately.

A second observation is that using chain majority voting during post-processing can in some cases provide a benefit. Previously, such majority voting was evaluated for a different set of problems (scheduling) and was not found to provide a significant benefit [157]. In our context, there were many problems for which the zero energy ground state solution was only achieved when using this post-processing; without this form of error correction (in other words, when all solutions containing a broken chain were discarded), false negatives occurred. For instance, at $N = 12$, 53 of 83 embedded problems were solved on the first job without using chain majority voting, and an additional 12 problems were solved by applying chain majority voting (Table 5.1). Classical error correction strategies other than majority voting should be explored and assessed in future studies, and their costs quantified.

To our knowledge, the evaluated solver is the first validated, experimental implementation of a QA-based graph isomorphism solver. While it ultimately classified every embeddable problem correctly and demonstrated clear advantages over the baseline approach, it has serious limitations as a graph isomorphism solver. The problem sizes are not competitive with those handled by classical solvers, which can

handle $G(N, 0.5)$ graphs with thousands of vertices [128] and even for the hardest graph types can handle hundreds of vertices before running into difficulty [126]. Similarly, the scaling of the total annealing times (Figure 5.19) is not competitive with classical scaling [128]. Ultimately, new approaches are likely needed if quantum annealing is to contribute to graph isomorphism theory or practice. Fortunately, the case study provides some new insight into experimental quantum annealing, and contributes methods that have relevance beyond the GI problem. It is hoped that the experimental evaluation of alternative Hamiltonian formulations adds to the understanding of the factors affecting quantum annealing performance, and that the demonstration of majority voting raises new questions about the role of post-processing for a variety of problems.

5.11 Methods

Quantum annealing experiments were performed on the D-Wave Two machine housed at USC ISI and operated by the USC-Lockheed Martin Quantum Computing Center. Experiments were conducted in October and November, 2014. The working graph of the machines Vesuvius-6 quantum annealing processor consisted of 504 qubits and 1427 couplers during this period. The pattern of working qubits is shown in Figure 5.20. The qubit temperature was estimated by the manufacturer to be 16± 1 mK. Additional processor specifications include a maximum anti-ferromagnetic mutual inductance of 1.33, and $1/f$ amplitude of $7.5 \pm 1 \frac{\mu\phi_0}{\sqrt{Hz}}$.

Simple undirected N-vertex graphs were constructed according to the Erdős-

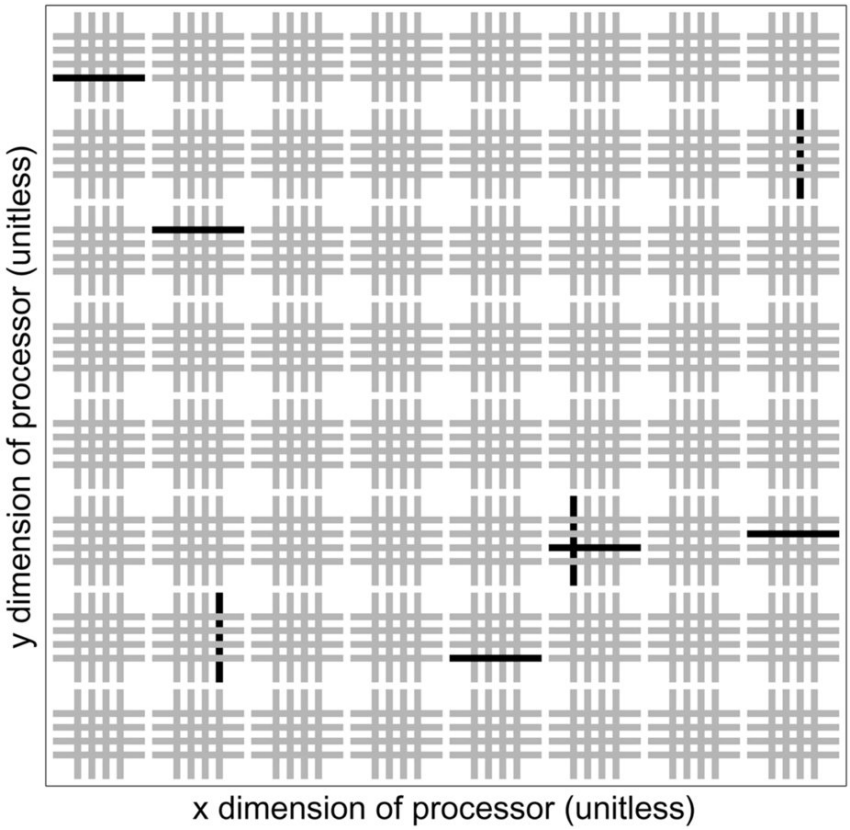


Figure 5.20: Physical layout of the working qubits in the USC-LM D-Wave Two Vesuvius-6 processor as of October 10, 2014.

Rényi $G(n, p)$ model [59] with $n = N$ and with the probability p of including each edge equal to 0.5. Non-isomorphic pairs were generated by creating two graphs as above and checking for non-isomorphism using the MATLAB `graphisomorphism()` function. Isomorphic pairs were generated by generating a single graph then applying a random permutation to arrive at the second graph. For each pair of input graphs, an Ising model was created using either H_1 or H_2. Programming was performed using MATLAB $R2014a$ win64 and the D-Wave MATLAB pack 1.5.2-beta2. The current version of the D-Wave `sapiFindEmbedding()` function cannot directly embed Ising models with more than one connected component (i.e. a set of variables that interact only with each other and not any of the remaining variables); therefore, models with this characteristic were not included in the input data. When attempting to generate 100 input pairs for each size, such disconnected models occurred no more than 4 times for each size $N \geq 14$. Similarly, the heuristic cannot accept models with fewer than two variables, so in the rare case of a trivial Ising problem with fewer than two variables (e.g. a non-isomorphic pair with no matching degrees), dummy variables were added to the problem.

The h_i values of the Ising problem were split evenly across each qubit in the associated chain in the embedded Ising problem. The J_{ij} values of the Ising problem were assigned to a single coupler connecting two variable chains in the embedded problem. The magnitudes of the embedded h' and J' were scaled together such that the maximum magnitude reached 20% of the full range supported by the processor; the range of the embedded h'_i values was $[0.4, 0.4]$ and the range of the embedded J'_{ij} values coupling different variables was $[0.2, 0.2]$. This 20% value was determined

empirically to provide good performance on the median difficulty problem at the largest sizes. Subsequently, the J'_{ij} values connecting physical qubits within a chain were set to the maximum ferromagnetic value (1). A single random spin gauge transformation [25] was then applied to each embedded problem, with a gauge factor $a_i \in \{1, 1\}$ associated with each qubit and transformation $h'_i \rightarrow a_i h'_i$; $J'_{ij} \rightarrow a_i a_j J'_{ij}$. One job was submitted to the quantum annealer per embedded problem; some Ising problems were associated with multiple embedded problems and jobs. After each programming cycle, the processor was allowed to thermalize for 10ms (the maximum supported by the machine). The annealing time was set to the minimum value of $20\mu s$. The number of annealing and readout cycles per programming cycle was 40000, which allowed the total job time to be within the limits of the machine (1s). The readout thermalization time was set to the default value of 0. Regarding error correction through majority voting of chains of physical qubits, ties were broken by choosing the spin up state. The probability of achieving the zero energy state on job k is denoted

$$P_{0,k} = \frac{\text{number of annealing cycles achieving zero energy}}{\text{number of annealing cycles}}. \tag{5.4}$$

When multiple jobs are required, I calculate the geometric mean in the style of Boixo et al. [25]:

$$\bar{P_0} = 1 - \prod_{k=1}^{K} (1 - P_{0,k})^{\frac{1}{K}} . \tag{5.5}$$

The total annealing time required to reach 0.99 probability of success was calculated by multiplying the annealing time by the expected number of annealing cycles (repetitions R) using the formula [25]:

$$R = \left\lceil \frac{\ln (1 - 0.99)}{\ln \left(1 - \bar{P_0}\right)} \right\rceil . \tag{5.6}$$

5.12 Summary

In this section I have constructed a pseudo-Boolean representation of the graph isomorphism problem. I have also shown that while being minimized on a quantum annealing computer, the worst case is the $\frac{n-1}{2}$-regular graph isomorphism problems. I have also demonstrated the first ever proof-of-concept experiment of solving graph isomorphism on a commercial quantum annealing computer.

Chapter 6

Hopfield Network and Quantum Annealing

6.1 Introduction

In this chapter, I have shown that exponential memory capacity for a Hopfield network can be achievable if the memory recall phase is implemented using quantum annealing. Associative memory models (AMM) are supervised learning models for the brain and reconstruct memories - desired configurations of quiescent and firing neurons - from input data that has only incomplete or incorrect information [96]. To this end, the Hopfield network [96, 97] is a well established paradigm for associative memory where neurons are treated as binary threshold units [123] with their interconnections described by real weights. The network can be trained to memorise patterns - called *learning* - via different algorithms [149] which evaluate the connection strengths based on the set of these fundamental memories. Once a network has learnt a certain number of patterns it should *recall* an initial (possibly imperfect) bit string configuration to a stored pattern which had maximum overlap with the initial state, and this is interpreted as successful recognition. Each distinct combination of a learning rule and recall method corresponds to a different associative memory model. Even in the canonical setting for Hopfield networks, where the neurons are considered to be classical Ising spins, succesful memory recall amounts to global minimization of an cost (energy) function over the possible collective spin

configurations [9, 97, 178]. The classical dynamical update rules however do not guarantee that this global minimum is indeed reached - often the asymptotic state is a local energy minimum [34]. While AMMs trace their origins to theoretical neuroscience, they have been widely considered in the classical setting, and an area of active current research in the quantum setting, in the context of content-addressable memories [148], machine learning [167, 2], artificial neural networks [122, 50] and neuromorphic computing [177]. This chapter is based on the following preprint [164] which was resulted from a collaboration with Drs. Siddhartha Santra and Radhakrishnan Balu of the U.S. Army Research Laboratory.

Exponential capacity of associative memories under quantum annealing recall
Siddhartha Santra, Omar Shehab and Radhakrishnan Balu

Associative memory models, in theoretical neuro- and computer sciences, can generally store a sublinear number of memories. I show that using quantum annealing for recall tasks endows associative memory models with exponential storage capacities. Theoretically, I obtain the radius of attractor basins, $R(N)$, and the capacity, $C(N)$, of such a scheme and their tradeoffs. Our calculations establish that for randomly chosen memories the capacity of a model using the Hebbian learning rule with recall via quantum annealing is exponential in the size of the problem, $C(N) = \mathcal{O}(e^{C_1 N}),\ C_1 \geq 0$, and succeeds on randomly chosen memory sets with a probability of $(1 - e^{-C_2 N}),\ C_2 \geq 0$ with $C_1 + C_2 = (.5 - f)^2/(1 - f)$, where, $f = R(N)/N,\ 0 \leq f \leq .5$ is the radius of attraction in terms of Hamming distance of an input probe from a stored memory as a fraction of the problem size. I demonstrate the application of this scheme on a programmable quantum annealing device - the DWave processor.

I demonstrate the use of quantum annealing (QA) [33, 48] for recalling stored

memories in AMMs. QA is a non-universal form of adiabatic quantum computation (AQC) [66] that solves hard optimization problems [163] by encoding the solution into the lowest energy state of a problem-defined Hamiltonian. The search for the global energy minimum is assisted via quantum tunneling across barriers in the energy landscape which reduces the chances of getting trapped in local minima [143]. Leveraging the robustness [40, 165] of open-system AQC, QA has become a promising scalable quantum sub-routine for the solution of practical computational problems [163, 187] using currently available technology [103].

By casting the problem of succesful memory recall in associative memory models as one of finding the spin configuration corresponding to the global energy minimum under a Hamiltonian, determined in part by the stored memories and partially by the imperfect input memory, one can hope to use QA for recall tasks in AMM. Just as in the classical AMM case, the memories are encoded as the weights of a fully connected network of spins - qubits in my case. However, in contrast to the classical case, the probe memories are input to the system as local field biases on the qubits and not as their initial values. I show that using QA for recall tasks [145, 168] in associative memory models, with any learning rule that does not discriminate against any of the fundamental stored memories, leads to an exponential storage capacity for randomly chosen memories and succeeds, over random choices of fundamental memory sets, with a probability approaching unity exponentially in the problem size. I also demonstrate an implementation of quantum annealing recall in an associative memory model with the Hebbian learning rule on a programmable quantum annealing device - the Dwave processor [103, 25]. My results are valid for

completey general Hopfield networks and may be contrasted with purely classical schemes for AMMs that require special pattern classes or connectivity structures [92] in order to achieve super-polynomial capacity.

The paper has three following sections with the theoretical setup described in Sec. (6.2), implementation results with the Dwave processor in Sec. (6.3) and concludes with a discussion in Sec. (6.4).

6.2 Theoretical Setup

Here, the framework of Hopfield networks is first discussed in subsec. (6.2.1) along with its use in AMMs. I then describe the process of quantum annealing from the perspective of finding ground states of classical Hamiltonians in subsec. (6.2.2). Followed by a discussion of how quantum annealing may be used for recall tasks in AMM, subsec. (6.2.3). As with any update rule in the classical setting only those input probe memories which lie within a certain maximum Hamming distance from any of the stored fundamental memories - the radius of attraction - may be successfully recalled using QAR-AMM which is discussed in subsec. (6.2.4). Finally, I obtain the capacity of the total scheme where the learning is done via the Hebbian rule with quantum annealing recall in subsec. (6.2.5) and show the tradeoff between the capacity and the size of the radius of attraction. In the same section I discuss the probability of my scheme succeeding over randomly chosen fundamental memory sets.

6.2.1 Hopfield network and Associative memory

The Hopfield network (HN) is a fully connected graph K_N of interacting binary state 'neurons' $\{S_i = \pm 1\}_{i=1,2...,N}$ whose weights, W_{ij}, encode the bit strings (of size N) that form its memory $M := \{\xi^\mu\}_{\mu=1,...,p}$. Given the set of memories, M, the use of different learning rules lead to different entries W_{ij} for the weight matrix. In this paper I consider the Hebbian learning rule whose weights W_{ij} for $i \neq j$ are given by,

$$W_{ij} := \frac{1}{N}(\sum_{\mu=1}^{p} \xi_i^\mu \xi_j^\mu - p\delta_{ij}) \ \forall i, j \in [1, N] \tag{6.1}$$

The Hebbian learning rule [88] has the characteristics of being local, incremental and immediate. Locality here means that a connection weight depends only on the state of the spins across the connection. Incrementality implies that new memories can be learnt without referring to previously learnt memories and immediate means that the connection weights for any number of memories can be obtained in a finite number of steps. The absolute storage capacitiy $C(N)$ defined as the number of memories one can store in a network of size N when perfect recall accuracy is desired is $C(N) = N/2\log(N)$ for the Hebbian rule [124].

In conjunction with a learning rule a Hopfield network may be used as an associative memory model i.e. as a content addressable memory where given an initial configuration of neurons $S^I \in \{0, 1\}^N$ (considered the input or probe memory vector) the dynamics of the network ideally results in a final network configuration S^F which is some stored memory (also called fundamental memory) $\xi \in M$ that was closest in Hamming distance from the original configuration S^I. This dynamic

is implemented as an update rule for the spins,

$$S_i(t+1) \rightarrow \text{Sign}(\sum_j W_{ij} S_j(t) - \theta_i) \tag{6.2}$$

where $S_i(t+1)$ is the value of spin i in the timestep after t. One can understand the attractor dynamics as a search for a global energy minimum by attaching an energy value to a configuration of spins $S = (S_1, S_2, ..., S_N)$ in the network through the definition,

$$E(\bar{S}) := -\sum_{i<j} W_{ij} S_i S_j - \sum_i \theta_i S_i, \tag{6.3}$$

where θ_i is the threshold value for spin i in the network. The Markovian dynamics generated by the rule (6.2) ensures that $E(\bar{S})$ is non-increasing during the evolution. The asymptotic fixed point is thus (at least) a local minimum and the corresponding spin configuration is a stable attractor for the dynamics. The local threshold values θ_i can serve to bias (or even freeze) certain spin values to the ones desired.

6.2.2 Quantum annealing

Quantum annealing (QA) is a finite temperature, non-universal form of Adiabatic Quantum computation (AQC) useful for solving hard optimization problems. Given the cost function, $Cost(X) : \{0,1\}^n \rightarrow \mathbb{R}$, of an optimization problem, QA finds the configuration, X, a vector of Boolean variables obtained after a qubit-wise read out of a quantum state - that minimizes $Cost(X)$. Thus QA can also be under-

stood as the quantum couterpart of simulated annealing [110]. In general, the cost function can be encoded as a Hamiltonian operator $\hat{H}_P$ whose ground state encodes the solution to the computational problem. Of interest in QA is the final ground state of the time dependent Hamiltonian,

$$\hat{H}(t) = A(t)\hat{H}_I + B(t)\hat{H}_P, \tag{6.4}$$

which undergoes *annealing* as the classical control process takes the parameters from (effectively), $A(t=0) = 1, B(t=0) = 0$ to $A(t = T_{\text{anneal}}) = 0, B(t = T_{\text{anneal}}) = 1$, where, T_{anneal} is the duration of the annealing process. The expectation is that the ground state of the initial Hamiltonian $\hat{H}_I$ is easily-prepared and annealing takes it to the ground state of the final Hamiltonian $\hat{H}_P$ which can also be read out easily to yield the solution. QA requires that the Hamiltonian $\hat{H}_P$ be diagonal in the computational basis which means that the process of reading out the final state does not introduce any further complexity to the computational problem beyond requirements of adiabaticity [66, 165] during the anneal process. In practice, for the specific QA device I use, Sec. (6.3), QA at non-zero temperature T starts in the limit of strong transverse field terms in $\hat{H}_I$ and weak $\hat{H}_P$, i.e. $A(0) \gg \max\{k_B T, B(0)\}$, with the initial ground state close to an equal superposition of all computational basis states of the qubits in the problem. Here, k_B is the Boltzmann constant. Monotonically decreasing $A(t)$ and increasing $B(t)$ takes the system (close) to the ground state of $\hat{H}_P$ as at the final time $B(T_{\text{anneal}}) \gg A(T_{\text{anneal}})$ [25].

The QA process relies on the quantum adiabatic theorem - thus the annealing

duration T_{anneal} has to be sufficiently large and the temperature T of the system sufficiently high to prevent diabatic transitions away from the instantaneous ground state of $H(t)$. For any given problem, i.e. H_P, the initial Hamiltonian H_I and the temporal dependence of $A(t), B(t)$ - the minimum required values for T_{anneal} and the maximum allowed value of T are determined by the inverse energy gap between the instantaneous ground state and the first excited state of $H(t)$ at any $t \in [0, T_{\text{anneal}}]$.

6.2.3 Recall tasks in AMM using Quantum Annealing

The energy function (6.3) of a HN admits a physical interpretation as the Hamiltonian operator of a spin-glass problem (with local field terms) [18] where the binary-state neurons may be treated as Ising spins interacting with each other via their couplings $J_{ij} = W_{ij}$ and the threshold values may be interpreted as local fields $h_i = \theta_i$. By making the correspondence $S_i \mapsto \hat{\sigma}_i^z$ the energy function (6.3) may be identified with the Hamiltonian operator, $\hat{H}_{\text{AM}} = -\sum_{i>j} J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z + \sum_i h_i\hat{\sigma}_i^z$, for the system whose set of ground states (the set containing all the degenerate lowest energy states) encodes the configurations the network *has committed to its memory*. If an eigenvector $|\chi\rangle$ of $\hat{H}$ is an element of the set of ground states then the corresponding bit string χ with entries $\chi_i = \langle\chi|\hat{\sigma}_i^z|\chi\rangle$, is a memory state only if χ corresponds to one of the stored memories, i.e. $\chi \in M$, otherwise it is a spurious state [34, 97] and corresponds to incorrect memory recall. Since the memories are all encoded only in the coupling terms between different spins I call $\hat{H}_{\text{mem}} := -\sum_{i>j} J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z$ the memory Hamiltonian whereas, as I show below, the local field term can be used

to probe memory recall using an input state and hence is called the probe term i.e. $\hat{H}_{\text{probe}} := \sum_i h_i \hat{\sigma}_i^z$. I use the Hamiltonian $\hat{H}_{\text{AM}}$ as the final problem Hamiltonian in Eq. (6.4), i.e. $\hat{H}_P = \hat{H}_{\text{AM}}$, with the probe memory part dependent on an input string χ, of length $n := |\chi|$, $0 \leq n \leq N$, given by $\hat{H}_{\text{probe}} = -h \sum_i \chi_i \hat{\sigma}_i^z$ where $h > 0$ is some overall scale. Thus,

$$\hat{H}_P = \hat{H}_{\text{AM}} = -\sum_{i>j} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z - h \sum_i \chi_i \hat{\sigma}_i^z \tag{6.5}$$

While the memory hamiltonian $\hat{H}_{\text{mem}}$ is degenerate for all stored memories $|\xi^\mu\rangle \in \mathcal{M}$, the probe Hamiltonian $\hat{H}_{\text{probe}}$ breaks this degeneracy as follows,

$$\begin{aligned}
\hat{H}_{\text{probe}}|\xi^\mu\rangle &= -h\Big(\sum_{i|\chi_i=\xi_i^\mu} 1 - \sum_{i|\chi_i=-\xi_i^\mu} 1\Big)|\xi^\mu\rangle \\
&= -h\Big(\sum_{i=1}^{i=|\bar{\chi}|} 1 - 2\sum_{i|\chi_i=-\xi_i^\mu} 1\Big)|\xi^\mu\rangle \\
&= -h(n - 2d_\chi^\mu)|\xi^\mu\rangle
\end{aligned} \tag{6.6}$$

where n is the length of the input probe bit string and $0 \leq d_\chi^\mu \leq n$ is the Hamming distance between the input bit string χ and the bit string ξ^μ corresponding to the memory eigenvector $|\xi^\mu\rangle$. Eq. (6.6) implies that the probe Hamiltonian energetically orders the stored memories according to their Hamming distances from the input bit string.

I thus have a scheme for a quantum annealing implementation of an AMM using Eqs. (6.4) and (6.5). Starting from the ground state of an arbitrary initial

Hamiltonian $\hat{H}_I$ (generally fixed by experimental limitations) if one can tune the temporal evolution of $\hat{H}(t)$ to arrive at the final Hamiltonian $\hat{H}_P = \hat{H}_{\rm AM}$ while maintaining conditions of adiabaticity then the final ground state should be the memory I hoped to recover. I call this scheme quantum annealing recall in associative memory models (QAR-AMM).

The requirement that the energy, relative to the Hamiltonian (6.5), for any input probe vector that is not one of the fundamental memories be greater than for any of the fundamental memories leads to a bound on the maximum value of the field strength h for successful recall, $E(|\xi^\mu\rangle) < E(|\chi\rangle) \implies h < (\langle\xi^\mu|W/2|\xi^\mu\rangle - \langle\chi|W/2|\chi\rangle)/2d_\chi^\mu$. This maximum field strength can be shown to depend on the number of random memories stored and the Hamming distances of the input probe memory from the stored fundamental memories in general, however when the fundamental memories are mutually orthogonal to each other i.e. every pairwise Hamming distances is $N/2$, this maximum value evaluates to, see Appendix. (A.1),

$$h < \frac{1}{4d_\chi^\mu}[N(1-p) + 4\sum_{\nu=1}^{p} d_\chi^\nu - \frac{4}{N}\sum_{\nu=1}^{p}(d_\chi^\nu)^2] =: h_{\chi,\max}^\mu \tag{6.7}$$

If I restrict ourselves to working with positive field biases then I get that for successfully recalling a memory the allowed values of h are $0 < h < \max_\mu h_{\chi,\max}^\mu$.

6.2.4 Radius of attraction using QAR-AMM.

The memory Hamiltonian, $\hat{H}_{\rm mem}$, part of $\hat{H}_{\rm AM}$ has a global spin flip symmetry which implies that for each memory eigenket $|\xi^\mu\rangle \in \mathcal{M}$ the spin-flipped state $|\tilde{\xi}^\mu\rangle =$

$\otimes_{i=1}^{N}\hat{\sigma}_i^x|\xi^\mu\rangle$ - which would be a spurious state for purposes of memory recall - is also degenerate with respect to $\hat{H}_{\rm mem}$. While the probe Hamiltonian breaks the degeneracy of memory states $|\xi^\mu\rangle \in \mathcal{M}$ in the desired manner, I find that it also shifts the energy of these spurious spin-flipped states in the reverse manner,

$$\hat{H}_{\rm probe}|\tilde{\xi}^\mu\rangle = h(n - 2d_\chi^\mu)|\tilde{\xi}^\mu\rangle, \tag{6.8}$$

i.e., spurious states that are *further* from the input memory in Hamming distance have lower energies w.r.t. $\hat{H}_{\rm probe}$. This means that for a set of p memories if a given input state $|\chi\rangle$ is nearest to $|\xi^\mu\rangle$ and furthest from $|\xi^\nu\rangle$ at Hamming distances d_χ^μ, d_χ^ν respectively (I call the shortest distance $d_\chi^s := \min_\mu d_\chi^\mu$ and the largest $d_\chi^b := \max_\mu d_\chi^\mu$) then by requiring that the energy w.r.t. $\hat{H}_{\rm probe}$ of the nearest memory state be lower than that of the lowest energy spurious state I arrive at the condition,

$$d_\chi^s + d_\chi^b \leq (n-1) = (|\chi| - 1), \tag{6.9}$$

which along with the definition $d_\chi^s \leq d_\chi^b$ results in $d_\chi^s < \lfloor n/2 \rfloor$. Thus all full length ($|\chi| = N$) input states χ within a radius,

$$\begin{aligned} R(N) &\leq (N-2)/2 \quad \text{N: Even} \\ &\leq (N-1)/2 \quad \text{N: Odd} \end{aligned} \tag{6.10}$$

are attracted to the closest fundamental memory and thus defines its basin of attraction.

Further, using the triangle inequality for Hamming distances I can also lower bound the sum $d^s_\chi + d^b_\chi$ by the maximal Hamming distance between any two vectors in the memory set for the given combination of $n = |\chi|$ bits. That is given $d(n) := \max_{\mu,\nu} d_n(\bar{\xi}^\mu, \bar{\xi}^\nu)$ where d_n is the Hamming distance between $\bar{\xi}^\mu, \bar{\xi}^\nu$ for a particular combination of n bits - I have in combination with Ineq. (6.9) [1],

$$d(n) \leq d^s_\chi + d^b_\chi \leq (n-1). \tag{6.11}$$

which describes the set of all input vectors $\bar{\chi}$ that can be successfully recalled using QAR-AMM. Clearly for smaller $d(n)$ values there are a larger number of states within the attraction basin. Intuitively, this implies that for set M of memory vectors with a smaller span in Hamming distances, but well separated within this span, the QA-AMM scheme works well. On the other hand for $n = N$ if, for example, $d(N) = N$, meaning that both a bit string and its negation are in the memory set, then the scheme fails.

6.2.5 Capacity, Attraction Basin size and tradeoffs

The capacity $C(N)$ of a model for associative memory for bit strings of length N is defined as the number of randomly chosen bit strings that may be stored in the network with the requirement that these states be stable fixed points under the

[1] Since Hamming distances can only take natural number values, the strict Ineq. (6.9) means $d^s_\chi + d^b_\chi$ can atmost equal $(n-1)$.

dynamics dictated by the update rule for the spins. Thus in the classical setting the capacity depends on the learning rule as well as the update rule for network dynamics. This is true using the QAR-AMM for memory recall as well.

In my scheme, when the memories are randomly chosen in a balanced manner, i.e. the probability of a bit being $1(-1)$ in any of the p memories is $.5(.5)$, there is a finite probability that an input probe memory χ which even though is within the basin of attraction of a fundamental memory fails to be recalled correctly because its distance from the furthest memory, d_χ^b, violates inequality (6.9) i.e. $d_\chi^b > N-1-d_\chi^s$. The probability of this happening can be made to approach zero exponentially in the size N provided the radius of attraction (6.10) is reduced by a constant fraction of N from the one given in Eq. (6.10). Allowing for this non-zero failure probability at any finite N - the capacity turns out to be exponential in the size of the problem.

To calculate the capacity I consider a set of p fundamental memory vectors, ξ^μ, of length N (even here for ease of presentation) whose pN entries are discrete random variables, $(\xi^\mu)_i = \pm 1$, that are i.i.d. with equal probability = .5. Suppose now I consider an input probe memory χ at a hamming distance $d_\chi^s = (N-2)/2 - x,\ x = 0, 1, ..., (N-2)/2$ from some memory that I call ξ^1. Then using Ineq. (6.9) QAR-AMM succeeds if any *other* fundamental memory vector is at a Hamming distance of at most $d_\chi^b \leq N/2 + x$. The probability that this happens for any one other fundamental memory $\xi^\mu,\ \mu \neq 1$ is given by [118],

$$P[d_\chi^\mu \leq (N/2+x)] = \sum_{l=0}^{(N/2+x)} P(d_\chi^\mu = l) = \sum_{l=0}^{(N/2+x)} \frac{\binom{N}{l}}{2^N}$$

$$\geq 1 - \frac{1}{2}\exp(\frac{-x^2}{N/2+x})$$
$$= 1 - .5\exp(\frac{-t^2}{.5+t}N) =: P^*. \tag{6.12}$$

where $x = tN,\ 0 \leq t < .5$. Intuitively this means that as the radius of the attraction basin is allowed to decrease by Hamming size $x = tN$, the probability of the scheme succeeding when there are only two stored memories approaches unity exponentially in N.

Since the memories are independently chosen - the probability that all the $(p-1)$ memories apart from the one which has Hamming distance d_χ^s from $\bar{\chi}$ have distances $d_\chi^\mu \leq (N/2+x)\ \forall\mu$ is lower bounded by $(P^*)^{p-1}$. If I now require that having stored p fundamental memories my scheme succeeds with probability at least γ, i.e. $(P^*)^{p-1} \geq \gamma$, then taking the logarithm of both sides I obtain a bound on the number of memories the network can possibly store,

$$p \leq (1 + \frac{\log\gamma}{\log P^*}) \tag{6.13}$$

I demand that asymptotically in N I get perfect recall and require that this approach be exponential. Then I have that,

$$\gamma = (1 - e^{-C_2 N}),\ C_2 > 0 \tag{6.14}$$

Then, using a small z approximation for $\log(1-z) \simeq -z$ and Eqs. (6.12,6.14)

in Ineq. (6.13) I get,

$$\begin{aligned} p &\leq 1 + 2\ \exp(-C_2 N + \frac{t^2}{.5+t} N) \\ &= 1 + 2\ \exp(C_1 N) = O(e^{C_1 N}), \end{aligned} \tag{6.15}$$

where $C_1 = t^2/(.5+t) - C_2$. Eq. (6.15) thus implies an exponential capacity for $0 \leq C_2 \leq t^2/(.5+t)$.

There is also a tradeoff between the size of the attraction basin and the capacity, just as in the classical setting [176], which can be seen by obtaining the relationship between the constants C_1, C_2 in terms of the radius of the basins of attraction, $f = R(N)/N = (N/2 - 1 - x)/N = (N/2 - 1 - tN)/N \rightarrow_{N\rightarrow\infty} (.5 - t),\ 0 \leq f < .5$, resulting in,

$$C_1 + C_2 = \frac{(.5-f)^2}{(1-f)}. \tag{6.16}$$

Note that ideally one would want both C_1, C_2 to be as large as possible because, respectively, they represent the exponent for the exponential capacity and the probability for QAR-AMM to succeed. However, the R.H.S. of Eq. (6.16) is a positive valued monotonically decreasing function of f. Thus smaller values of f would imply higher capacity and scheme success probability but smaller radii of attraction basins and vice versa.

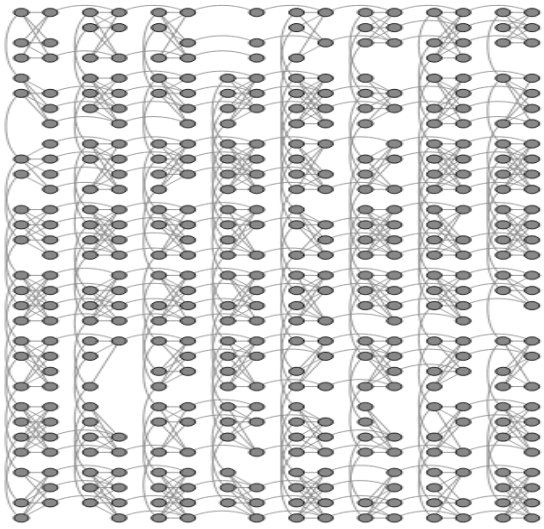


Figure 6.1: 'Chimera' graph showing the connectivity of qubits on the DW2 processor chip at Burnaby, BC that I use. Not all qubits are usable in the graph - missing qubits - which are rejected at the calibration stage. There are 64, $K_{4,4}$-connected blocks of qubits laid out as a matrix of 8×8 blocks. Each block has 2 columns (vertical) and 4 rows (horizontal). Fully connected problems such as Hopfield networks have to be embedded onto the native graph structure keeping in mind the missing qubits.

6.3 Quantum annealing recall with a programmable quantum annealer.

In this section I present results from experimental implementations of the QAR-AMM on the Dwave quantum annealing processor. In Subsec. (6.3.1) I describe the Dwave quantum annealing processor and the settings I use, a description of the required embedding of our fully connected networks onto the native qubit connectivity on the processor chip in subsec. (6.3.2) and finally examples of memory recall using quantum annealing in subsec. (6.3.3).

6.3.1 Experimental Setup

To demonstrate associative memory recall using quantum annealing I use the second generation of the commercially available Dwave processors [103], DW2, with 512 qubits of which 476 qubits are effectively available. These qubits form the nodes of the so-called "Chimera" graph shown in Fig. (6.1). The engineering and physics of the processor chip has been extensively discussed in the literature [87, 22] and references therein. In this paragraph I briefly touch upon some features that are relevant to this problem. The DW2 chip comprises of superconducting rf SQUID flux qubits interacting with each other via Josephson junctions and is maintained at a base temperature of $T \simeq 15mK$ in a dilution refrigerator. A classical program supplies the problem Hamiltonian to the chip via the $N \times N$ matrix of coupling values, J_{ij}, and a $N \times 1$ vector of field strength, h_i, values. These final J_{ij}, h_i values are achieved on the processor at the end of the user set annealing time T_{anneal} which can be set at integer values between 20 μs to $20,000$ μs with the default value being 20 μs. The time to wait after this programming, called thermalization time, in order for it to cool back to base temperature can be between 0 to 10000 microseconds with the default value being 10000 microseconds. The coupler strengths J_{ij} can be set between $J_{ij} \in [-1, 1]$ while the local fields between $h_i \in [-2, 2]$. These values go through a non-linear 9-bit analog to digital conversion (ADC) and thus the step size for either is the extent of their range divided by 2^9 - so the $|J_{ij}|$ values can be set in multiples of 2^{-8} while $|h_i|$ in steps of 2^{-7}, however there are noise contributions which are important at low values of h, J [52]. The time to wait after each state is

read from the processor in order for it to cool back to base temperature is $0\mu s$ as the readout process is not supposed to supply any thermal noise.

For my experiments - to minimize diabatic transitions due to finite annealing times I use the maximum allowed annealing time $T_{\text{anneal}} = 20,000\ \mu s$ and the maximum allowed thermalization time of $10,000\ \mu s$. Further, I choose a problem defined on $N = 2^4$ qubits so that the values of $|J_{ij}| = O(2^{-4})$, given by the Hebbian weight matrix Eq. (6.1), are much larger compared to its resolution.

6.3.2 Embedding fully connected Hopfield networks in Chimera

A major step in solving a problem on the Dwave Two computer is mapping a generic Ising problem Hamiltonian, such as $\hat{H}_{\text{AM}}$, to the Dwave's native chimera graph which is a composition of $K_{4,4}$ graphs (complete bipartite graph with 4 vertices in each partition) - an instance of the minor embedding problem which is **NP**-complete [75]. A problem can be embedded in more than one way. The Dwave API provides the function '`find_embedding(J)`' that uses a heuristic algorithm to perform the embedding which works reliably when the number of logical qubits, N, is under 50. The input argument to function is the $N \times N$ coupling matrix, J, and the algorithm looks at the adjacency matrix derived from, J, to obtain a possible embedding. Asymptotically, roughly N completely-connected logical qubits may be embedded on a hardware chimera graph of N^2 physical qubits. On the particular 512-qubit DW2 processor I use, only 476 qubits are available to be programmed as shown in Fig. (6.1), the remaining qubits being unusable due to hardware faults. I

note that not all problems require complete graphs hence larger non-trivial problem graphs can be embedded depending on which problem is being attempted. For example, the problem graph for the graph isomorphism problem on a Dwave machine is not a complete graph [187] nor are certain restrictions [92] to the canonical Hopfield networks.

When a problem is embedded on a hardware graph, a logical qubit is represented by a ferromagnetic chain of physical qubits. Ideally, after annealing, all the physical qubits are in the same state carrying the value of the state of the logical qubit. In reality, the chain tends to break down more often when it becomes longer, i.e., some physical spins corresponding to the same logical variable do not agree. In this scenario, one needs to use gauge averaging [25], majority voting [187] or the more general quantum error-correcting schemes [151].

For the representative example discussed in the next subsection I have not implemented any error-correction strategy. This lets us discuss the raw implementation of QAR-AMM with respect to the theory in the previous section.

6.3.3 Representative example

I demonstrate the actual implementation of the QAR-AMM scheme using the Dwave quantum annealing hardware by describing a representative example with three stored fundamental memories of length $N = 16$. These are,

$$\bar{\xi^1} :(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1),$$

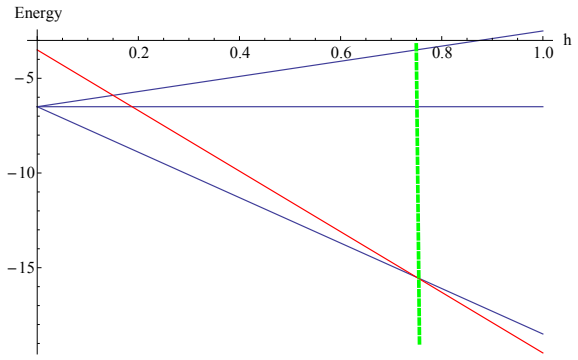


Figure 6.2: (color online) The variation of the energies of the fundamental memories and the probe memory under the Hamiltonian $\hat{H}_{\rm AM}$ with the probe vector $\bar{\chi}$ as in (6.17). The dotted vertical line (green) represents the highest ($h = .75$) allowed field strength for succesful recall of $\bar{\chi}$. Applying fields above this maximum value overbiases the Hamiltonian such that $\bar{\chi}$ itself becomes the lowest energy state. A vertical slice at any fixed value of h is the spectrum of the problem Hamiltonian $\hat{H}_P = \hat{H}_{\rm AM}$ for the p-fundamental memories plus the input probe memory.

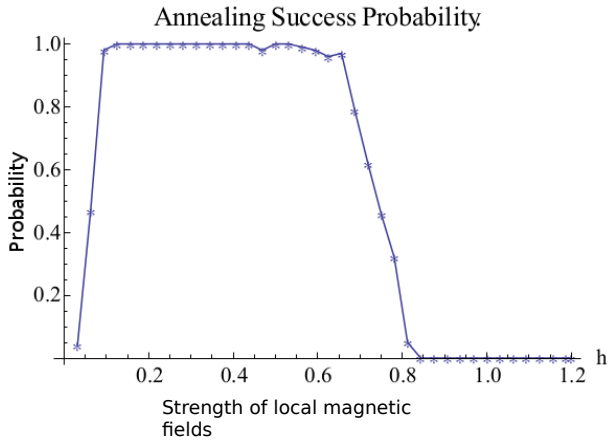


Figure 6.3: (color online) Probability of the correct recall using quantum annealing varying with respect to the applied field strength $h > 0$. This probability is high ($\simeq 1$) for the particular set of $p = 3$ memories and the input vector (6.17) for almost the entire region with $h < .75$ (green dashed line). For small values of h ($\leq .15$), the thermal noise degrades the annealing recall success significantly.

$$\bar{\xi^2} :(1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1),$$

$$\bar{\xi^3} :(1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,1,1,1,1),$$

which are stored using the Hebbian rule (6.1) in a network of 16×16 fully connected interacting qubits. The DW2 provided software tool is used to find an embedding onto the native "chimera" graph on the chip. This requires only 133 physical qubits for embedding. Each of the 16 logical qubits are encoded as ferromagnetic chains of physical qubits with the largest encoded qubit being a chain of 11 physical qubits and the smallest of size 5. The maximum value of any coupling $|J_{ij}| = W_{ij}$ is $(3/2^4)$ with the minimum being $(1/2^4)$.

Note that these fundamental memories are mutually orthogonal, i.e. $\sum_i \xi_i^\mu \xi_i^\nu = \delta_{\mu,\nu}$, since their pairwise Hamming distances are all equal to $N/2 = 8$. The probe input vector I use is,

$$\bar{\chi} = \{-1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,1,1,1,-1\} \tag{6.17}$$

whose Hamming distances from the fundamental memories are $d_\chi^1 = 10$, $d_\chi^2 = 8$, $d_\chi^3 = 2$. The energies of the fundamental memories $\bar{\xi^1}, \bar{\xi^2}, \bar{\xi^3}$ and input memory $\bar{\chi}$ w.r.t. the final problem Hamiltonian, $\hat{H}_P$, using $\bar{\chi}$ from Expression (6.17) to determine the probe Hamiltonian part, $\hat{H}_{\text{probe}}$ in Eq. (6.6), are shown in Fig. (6.2). I expect the bound on the maximum field strength (6.7) to be $h^\mu_{\chi,\max} = .75$ obtained for $\mu = 3$ (closest memory in Hamming distance), but test the success probability of recall using quantum annealing at increasing values of h starting from $h = 2^{-5}$

to, well beyond $h^{\mu}_{\chi,\max}$, upto $h = 1.2$ in linear steps of 2^{-5}. At each value of h I anneal a 100 times and the number of times the machine returns the closest memory $\bar{\xi^3}$ - expressed as a percentage - gives us the success probability P_{success}, Fig. (6.3). I find that the annealing success probability is very close to unity, and can be essentially made 1 if one imposes a majority vote on the percentage, i.e. a percentage value greater than 50 % is understood as success probability of 1 for that particular memory, for most of the allowed region except for very small h values. To understand why this may be so I consider the different sources of error on DW2 in the next paragraph.

How close to perfect recall this empirically determined quantity P_{success} is, depends on several factors [163, 25]. First and foremost, DW2 operates at a non-zero, albeit small, temperature of $T \simeq 15mK$. This means that the quantum states representing two bit strings at a Hamming distance of $d \leq k_B T/h$, with k_B - the Boltzmann's constant, from each other should be considered as degenerate w.r.t. the probe Hamiltonian $\hat{H}_{\text{probe}}$. This counts as thermal error [5] which is the hardest to mitigate. Secondly, the encoding of logical qubits into ferromagnetic chains of physical qubits, the longer the worse, introduces errors at the emdedding stage - encoding error - that may be reduced by adopting embedding algorithms that minimize qubit chain lengths. Next, the physical implementation of the flux qubits favors an individual spin to align in one direction compared to the opposite direction which introduces the so-called gauge error - which may be suppressed via gauge-averaging [8, 25]. For the specific class of recall tasks in AMMs, gauge-error implies that the same pair, of fundamental memory set and probe vector,

may have a different success probability if they are encoded with the signs of all their spins flipped. Finally, short annealing times can also cause diabatic errors [5] that causes higher energy eigenstates to be populated - which I have tried to reduce in my own experiments by using the maximum possible annealing time on the machine, $T_{\text{anneal}} = 20,000\ \mu S$, in each run in order to ensure that such transitions are minimized. However, I have not analyzed the energy gap Δ of our problems to determine whether $T_{\text{anneal}} >> \Delta^{-1}$.

For the small $h \leq .15$ region in Fig. (6.3), which although is well within the bound $h^{\mu}_{\chi,\text{max}} = .75$ given by Ineq. (6.7), I observe that the annealing success probability is severely degraded. This is caused by, I suspect, a combination of one or more factors discussed in the previous paragraph. However, the strongest reason might be the thermal noise that dominates at small h values. Note that the actual physical energy that the field strength h represents is obtained by multiplying it with $B(t)$ appearing in the time dependent annealing Hamiltonian (6.4). The maximum value of $B(t)$ is $\simeq 30\ GHz$ at the end of the annealing process starting from close to zero at $t = 0$. An order of magnitude calculation shows that $hB(t)$ with $h = .15$ is of the same order of magnitude as $k_B T$ with $T = 15\ mK$ - for at least half the annealing time. The reduced annealing success probability in this region of h might thus be attributed to thermally caused leakage of population to other energy eigenstates.

6.4 Discussion and conclusion

I have shown that using quantum annealing for recall tasks in Associative memory models can lead to an exponential capacity for storage and this scheme works with probability 1 for sets of randomly chosen memories in the large network size limit. The positive exponents for capacity and that for the scheme success probability have to sum to a decreasing function of the radius of the attraction basins - hence the tradeoff between the radius and the capacity and scheme success probability. Implementation of our scheme on a physical quantum annealing device may suffer thermal, encoding, gauge and diabatic errors that can lead to imperfect recall even when all theoretical conditions for successful recall are met. The effective experimental success probability is determined empirically and depends on several factors of the physical implementation.

QAR-AMM should work for every learning rule where the memory Hamiltonian is degenerate on all fundamental memory vectors. Even so, one needs to consider certain inherent theoretical limitations that I now point out. The energy degeneracy of the stored memories is lifted by an amount proportional to their Hamming distance from the input memory times the uniform field strength h. The maximum field strength value such that the system does not get overbiased is inversely proportional to the size of the problem, i.e., $h^{\mu}_{\xi,\max} \sim 1/N$ [145]. This automatically sets the upper bound for the adiabatic energy gap of the problem because for two fundamental memories differing by a Hamming distance of d - the energy difference w.r.t. the problem Hamiltonian is proportional to $d \times h^{\mu}_{\xi,\max}$. This means

that, at least, for storing a linear number of memories one can expect an efficient adiabatic (annealing) run time, $T_{\text{anneal}} = O(N^{\alpha}),\ \alpha =$ small positive integer, for the recall task. However, as the number of stored memories approaches, the theoretically achievable, exponential limit - the number of stored memories at the same Hamming distance from the input can grow exponentially - the number of possible stored memories at any distance d is given by the binomial coefficient $\binom{N}{d}$. To break the degeneracy of these equidistant memories one can choose, instead of a uniform field h, a position dependent non-uniform field $h_i,\ i = 1, 2, ..., N$. However, for any finite range of h_i-values, i.e. $\delta h = \max_i(h_i) - \min_i(h_i) < \infty$, the degeneracy would be broken by an amount proportional to $\delta h / \binom{N}{d}$. This means that the adiabatic energy gap in the exponential storage limit would close, as an inverse exponential, making recall tasks inefficient - time complexity wise. Nevertheless, even a polynomial storage capacity, with a concomitant polynomial QAR-AMM run-time, is a significant improvement (see [92] and references therein) for the case of completely general Hopfield networks considered here.

Going forward, I would like to explore equivalent recall schemes for forgetful learning rules that favor recently added fundamental memories in the learning set to the ones before [176]. There one would like to understand the minimal requirements on the additional types of terms in the problem Hamiltonian those recall schemes would need. Further, I note that the recall process may be considered as an adiabatic quantum error correction operation - a fundamental memory may be understood as a codeword and its basin of attraction as the codespace [146]. Each input memory within the basin of attraction corresponds to a distinct error. The

conditional error-correction operation is the unitary obtained as a result of the evolution under the time-dependent annealing Hamiltonian which depends on the input state. Clearly, the scheme of QAR-AMM does not *detect* errors but only corrects them - but in doing so it greatly enhances the capacity of the classical Hopfield network models. This is yet another instance of a hybrid protocol where partitioning of a computational job into quantum and classical subtasks leads to distinct advantages. However, the question of optimality of such partitioning is still open [187]. Finally, I comment that the QAR-AMM scheme also requires classical preprocessing to minor-embed the problem graph on the Dwave annealing architecture - a step that may be obviated through the use of fully connected quantum annealers as recently proposed in [117].

Appendix A

Supplementary Material for Chapter 6

A.1 Bound on field strength

We require the energies of any state that is not a fundamental memory to be greater than that of the later w.r.t. the Hamiltonian (6.6):

$$\begin{aligned} &- \langle \xi^\mu | W/2 | \xi^\mu \rangle - h \sum_i \chi_i \xi_i^\mu < - \langle \chi | W/2 | \chi \rangle - h \sum_i \chi_i \chi_i \\ &- \langle \xi^\mu | W/2 | \xi^\mu \rangle - h(n - 2d_\chi^\mu) < - \langle \chi | W/2 | \chi \rangle - hn \end{aligned} \tag{A.1}$$

For the case of orthogonal fundamental memory vectors their energy is given by,

$$\begin{aligned} \langle \xi^\mu | W/2 | \xi^\mu \rangle &= \frac{1}{2} \sum_{i,j} \xi_i^\mu W_{ij} \xi_j^\mu \\ &= \frac{1}{2N} \left\{ \sum_\nu \sum_{ij} \xi_i^\mu (\xi_i^\nu \xi_j^\nu) \xi_j^\mu - p \sum_{ij} \delta_{ij} \xi_i^\mu \xi_j^\mu \right\} \\ &= \frac{1}{2N} \left\{ \sum_\nu (\sum_i \xi_i^\mu \xi_i^\nu)(\sum_j \xi_j^\mu \xi_j^\nu) - p \sum_i \xi_i^\mu \xi_i^\mu \right\} \\ &= \frac{1}{2N} \left\{ \sum_\nu (N \delta_{\mu,\nu})(N \delta_{\mu,\nu}) - pN \right\} \\ &= (N - p)/2, \end{aligned} \tag{A.2}$$

whereas for any arbitary input vector $\bar{\chi}$,

$$\begin{aligned}
\langle\chi|W/2|\chi\rangle &= \frac{1}{2N}\left\{\sum_{\nu}\sum_{ij}\chi_i(\xi_i^{\nu}\xi_j^{\nu})\chi_j - p\sum_{ij}\delta_{ij}\chi_i\chi_j\right\} \\
&= \frac{1}{2N}\left\{\sum_{\nu}(\sum_{i}\chi_i\xi_i^{\nu})(\sum_{j}\chi_j\xi_j^{\nu}) - p\sum_{i}\chi_i\chi_i\right\} \\
&= \frac{1}{2N}\left\{\sum_{\nu}(\sum_{i}\chi_i\xi_i^{\nu})(\sum_{j}\chi_j\xi_j^{\nu}) - p\sum_{i}\chi_i\chi_i\right\} \\
&= \frac{1}{2N}\left\{\sum_{\nu=1}^{p}(N-2d_{\chi}^{\nu})^2 - pN\right\} \\
&= \frac{1}{2}\left\{\frac{1}{N}\sum_{\nu=1}^{p}(N-2d_{\chi}^{\nu})^2 - p\right\}, \qquad \text{(A.3)}
\end{aligned}$$

where we have used $\sum_i \chi_i\xi_i^{\nu} = (N - 2d_{\chi}^{\nu})$.

A.2 Annealing Schedule.

Fig. (A.1) shows the hard-coded classical controls $A(t)$, $B(t)$ evolving as a function of the scaled annealing time t/T_{anneal} where t is the physical time lapsed during the annealing process and T_{anneal} is the user-set length of the annealing process. T_{anneal} can be set at any integer value between a minimum of 20 μs to a maximum of $20,000$ μs.

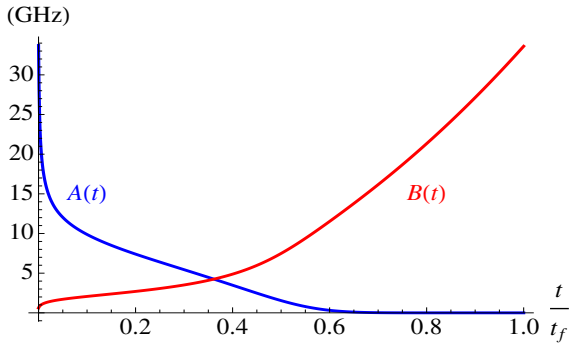


Figure A.1: (color online) Temporal evolution of the classical control functions $A(t)$, $B(t)$ in the time-dependent annealing Hamiltonian $H(t)$ in Eq. (6.4).

Appendix B

Supplementary materials for Chapter 5

B.1 Few pieces of information

If d_1 is the maximum degree of a random graph $|d_1 - pn - (2pqn\log n)^{\frac{1}{2}} + \left(\frac{pqn}{8\log n}\right)^{\frac{1}{2}} \log\log n| \leq \left(\frac{n}{\log n}\right)^{\frac{1}{2}} \log\log\log n$ [28].

- Let $0 < p < 1$, $q = 1 - p$ and b be fixed. For G a random graph from $\mathcal{G}(n,p)$, let $p_n = \mathbb{P}\left(d^{max}(G) \leq np + b\sqrt{npq}\right)$. The n $p_n^{\frac{1}{n}} \to c$ as $n \to \infty$, where c is a constant depending only on b given by $c = \max_{\theta}\left\{f(b+\theta)\, e^{-\frac{\theta^2}{2}}\right\}$ [158].

B.2 Worst case scenario of Algorithm 6

We list the following facts which are asymptotically equally applicable for both models, $\mathbb{G}(n,M)$ and $\mathbb{G}(n,p)$.

- The number of possible simple undirected graphs with n vertices is $2^{\binom{n}{2}}$ [150].
- Random graphs
  - Number/probability
    - In [58], Erdős et al., proved that the probability of a random graph with all vertices belonging to the same connected component is $e^{-e^{-2c}}$

where c is a constant when the number of verices is n and the number of edges is $N_c = \left[\frac{1}{2}n\log n + cn\right]$.

– Maximum degree

* If d_1 is the maximum degree of a random graph $|d_1 - pn - (2pqn\log n)^{\frac{1}{2}} + \left(\frac{pqn}{8\log n}\right)^{\frac{1}{2}}\log\log n| \leq \left(\frac{n}{\log n}\right)^{\frac{1}{2}}\log\log\log n$ [28].

* Let $0 < p < 1$, $q = 1 - p$ and b be fixed. For G a random graph from $\mathcal{G}(n,p)$, let $p_n = \mathbb{P}\left(d^{max}(G) \leq np + b\sqrt{npq}\right)$. The n $p_n^{\frac{1}{n}} \to c$ as $n \to \infty$, where c is a constant depending only on b given by $c = \max_{\theta}\left\{f(b+\theta)e^{-\frac{\theta^2}{2}}\right\}$ [158].

Bibliography

[1] D-wave developer guide for matlab v 1.5.2. Internal Qubist Website for D-Wave Computer. Accessed: 11/26/2014.

[2] J. Adcock, E. Allen, M. Day, S. Frick, J. Hinchliff, M. Johnson, S. Morley-Short, S. Pallister, A. Price, and S. Stanisic. Advances in quantum machine learning. *arXiv:1512.02900*, 2015.

[3] Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008.

[4] Gorjan Alagic, Cristopher Moore, and Alexander Russell. Strong fourier sampling fails over g^n. *arXiv preprint quant-ph/0511054*, 2005.

[5] Tameem Albash, Sergio Boixo, Daniel A Lidar, and Paolo Zanardi. Quantum adiabatic markovian master equations. *New Journal of Physics*, 14(12):123016, 2012.

[6] Tameem Albash, Troels F Rønnow, Matthias Troyer, and Daniel A Lidar. Reexamining classical and quantum models for the d-wave one processor. *The European Physical Journal Special Topics*, 224(1):111–129, 2015.

[7] Andris Ambainis and Oded Regev. An elementary proof of the quantum adiabatic theorem. Technical report, 2004.

[8] MohammadH.S. Amin, NeilG. Dickson, and Peter Smith. Adiabatic quantum optimization with qudits. *Quantum Information Processing*, 12(4):1819–1829, 2013.

[9] Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Spin-glass models of neural networks. *Phys. Rev. A*, 32:1007–1018, Aug 1985.

[10] John Armstrong. Characters of induced representations, 2010. Accessed: 2016-05-25.

[11] László Babai. Automorphism groups, isomorphism, reconstruction. In *Handbook of combinatorics (vol. 2)*, pages 1447–1540. MIT Press, 1996.

[12] László Babai. Graph isomorphism in quasipolynomial time. *arXiv preprint arXiv:1512.03547*, 2015.

[13] László Babai, Xi Chen, Xiaorui Sun, Shang-Hua Teng, and John Wilmes. Faster canonical forms for strongly regular graphs. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 157–166. IEEE, 2013.

[14] László Babai, Paul Erdős, and Stanley M Selkow. Random graph isomorphism. *SIAM Journal on Computing*, 9(3):628–635, 1980.

[15] László Babai, D Yu Grigoryev, and David M Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 310–324. ACM, 1982.

[16] László Babai and Eugene M Luks. Canonical labeling of graphs. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 171–183. ACM, 1983.

[17] Dave Bacon, Andrew M Childs, and Wim van Dam. Optimal measurements for the dihedral hidden subgroup problem. *arXiv preprint quant-ph/0501044*, 2005.

[18] Francisco Barahona. On the computational complexity of ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241, 1982.

[19] András Békéssy, P Bekessy, and János Komlós. Asymptotic enumeration of regular matrices. *Stud. Sci. Math. Hungar*, 7:343–353, 1972.

[20] Edward A Bender. The asymptotic number of non-negative integer matrices with given row and column sums. *Discrete Mathematics*, 10(2):217–223, 1974.

[21] Edward A Bender and E Rodney Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3):296–307, 1978.

[22] A J Berkley, M W Johnson, P Bunyk, R Harris, J Johansson, T Lanting, E Ladizinsky, E Tolkacheva, M H S Amin, and G Rose. A scalable readout system for a superconducting adiabatic quantum optimization system. *Superconductor Science and Technology*, 23(10):105014, 2010.

[23] N Biggs. Aspects of symmetry in graphs, algebraic methods in graph theory, vol. i, ii (szeged, 1978). In *Colloq. Math. Soc. János Bolyai*, volume 25, pages 27–35.

[24] Garrett Birkhoff. Sobre los grupos de automorfismos. *Rev. Unión Mat. Argent*, 11(4):155–157, 1946.

[25] Sergio Boixo, Troels F Rønnow, Sergei V Isakov, Zhihui Wang, David Wecker, Daniel A Lidar, John M Martinis, and Matthias Troyer. Evidence for quantum annealing with more than one hundred qubits. *Nature Physics*, 10(3):218–224, 2014.

[26] Sergio Boixo, Vadim N Smelyanskiy, Alireza Shabani, Sergei V Isakov, Mark Dykman, Vasil S Denchev, Mohammad Amin, Anatoly Smirnov, Masoud Mohseni, and Hartmut Neven. Computational role of collective tunneling in a quantum annealer. *arXiv preprint arXiv:1411.4036*, 2014.

[27] Béla Bollobás. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European Journal of Combinatorics*, 1(4):311–316, 1980.

[28] Béla Bollobás. Degree sequences of random graphs. *Discrete Mathematics*, 33(1):1–19, 1981.

[29] Béla Bollobás. *Random graphs*. Springer, 1998.

[30] Béla Bollobás. *Modern graph theory*, volume 184. Springer Science & Business Media, 2013.

[31] Max Born and Vladimir Fock. Beweis des adiabatensatzes. *Zeitschrift für Physik*, 51(3-4):165–180, 1928.

[32] Endre Boros and Peter L Hammer. Pseudo-boolean optimization. *Discrete applied mathematics*, 123(1):155–225, 2002.

[33] J Brooke, D Bitko, G Aeppli, et al. Quantum annealing of a disordered magnet. *Science*, 284(5415):779–781, 1999.

[34] J. Bruck and V.P. Roychowdhury. On the number of spurious memories in the hopfield model [neural network]. *Information Theory, IEEE Transactions on*, 36(2):393–397, Mar 1990.

[35] Jun Cai, William G Macready, and Aidan Roy. A practical heuristic for finding graph minors. *arXiv preprint arXiv:1406.2741*, 2014.

[36] Peter J Cameron. *Permutation groups*, volume 45. Cambridge University Press, 1999.

[37] Peter J Cameron et al. Automorphisms of graphs. *Topics in algebraic graph theory*, 102:137–155, 2004.

[38] Yair Caro, Ilia Krasikov, and Yehuda Roditty. On the largest tree of given maximum degree in a connected graph. *Journal of Graph Theory*, 15(1):7–13, 1991.

[39] Andrew M Childs. Lecture notes on quantum algorithms. 2016.

[40] Andrew M. Childs, Edward Farhi, and John Preskill. Robustness of adiabatic quantum computation. *Phys. Rev. A*, 65:012322, Dec 2001.

[41] Andrew M. Childs and Wim van Dam. Quantum algorithms for algebraic problems. *Rev. Mod. Phys.*, 82:1–52, Jan 2010.

[42] KEITH Conrad. Dihedral groups ii. *Internet Online Book*, pages 3–6, 2009.

[43] JH Conway. Talk given at the second british combinatorial conference at royal holloway college, 1971.

[44] Yves Crama and Peter L Hammer. *Boolean functions: Theory, algorithms, and applications*. Cambridge University Press, 2011.

[45] Yves Crama, Pierre Hansen, and Brigitte Jaumard. The basic algorithm for pseudo-boolean programming revisited. *Discrete Applied Mathematics*, 29(2):171–185, 1990.

[46] Charles W Curtis and Irving Reiner. *Representation theory of finite groups and associative algebras*, volume 356. American Mathematical Soc., 1966.

[47] Tomek Czajka and Gopal Pandurangan. Improved random graph isomorphism. *Journal of Discrete Algorithms*, 6(1):85–92, 2008.

[48] Arnab Das and Bikas K. Chakrabarti. *Colloquium* : Quantum annealing and analog quantum computation. *Rev. Mod. Phys.*, 80:1061–1081, Sep 2008.

[49] Percy Deift, Mary Beth Ruskai, and Wolfgang Spitzer. Improved gap estimates for simulating quantum circuits by adiabatic evolution. *Quantum Information Processing*, 6(2):121–125, 2007.

[50] Vasil S. Denchev, Nan Ding, S. V. N. Vishwanathan, and Hartmut Neven. Robust classification with adiabatic quantum optimization. *arXiv:1205.1148*, 2012.

[51] Persi Diaconis and Daniel Rockmore. Efficient computation of the fourier transform on finite groups. *Journal of the American Mathematical Society*, 3(2):297–332, 1990.

[52] A. Douglass and M. Thom. *private communication*.

[53] Mildred S Dresselhaus, Gene Dresselhaus, and Ado Jorio. *Group theory: application to the physics of condensed matter*. Springer Science & Business Media, 2007.

[54] David Steven Dummit and Richard M Foote. *Abstract algebra*, volume 1984. Wiley Hoboken, 2004.

[55] Paul Erdős. Some remarks on the theory of graphs. *Bulletin of the American Mathematical Society*, 53(4):292–294, 1947.

[56] Paul Erdős. Graph theory and probability. *canad. J. Math*, 11:34G38, 1959.

[57] Paul Erdös and Irving Kaplansky. The asymptotic number of latin rectangles. *American Journal of Mathematics*, 68(2):230–236, 1946.

[58] Paul Erdős and Alfréd Rényi. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[59] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci*, 5:17–61, 1960.

[60] Paul Erdős and Alfréd Rényi. On the strength of connectedness of a random graph. *Acta Mathematica Hungarica*, 12(1-2):261–267, 1961.

[61] Paul Erdős and Alfréd Rényi. Asymmetric graphs. *Acta Mathematica Hungarica*, 14(3-4):295–315, 1963.

[62] Paul Erdős and Alfréd Rényi. On random matrices. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 8(455-461):1964, 1964.

[63] Paul Erdős and Alfréd Rényi. On the existence of a factor of degree one of a connected random graph. *Acta Mathematica Hungarica*, 17(3-4):359–368, 1966.

[64] Mark Ettinger and Peter Høyer. On quantum algorithms for noncommutative hidden subgroups. *Advances in Applied Mathematics*, 25(3):239–251, 2000.

[65] Mark Ettinger, Peter Hoyer, and Emanuel Knill. Hidden subgroup states are almost orthogonal. *arXiv preprint quant-ph/9901034*, 1999.

[66] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, Joshua Lapan, Andrew Lundgren, and Daniel Preda. A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science*, 292(5516):472–475, 2001.

[67] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.

[68] Scott Fortin. The graph isomorphism problem. Technical report, Technical Report 96-20, University of Alberta, Edomonton, Alberta, Canada, 1996.

[69] Katalin Friedl, Gábor Ivanyos, Frédéric Magniez, Miklos Santha, and Pranab Sen. Hidden translation and orbit coset in quantum computing. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 1–9. ACM, 2003.

[70] Robert Frucht. Herstellung von graphen mit vorgegebener abstrakter gruppe. *Compositio Mathematica*, 6:239–250, 1939.

[71] Robert Frucht. Graphs of degree three with a given abstract group. *Canadian J. Math*, 1:365–378, 1949.

[72] William Fulton and Joe Harris. *Representation theory*, volume 129. Springer Science & Business Media, 1991.

[73] Frank Gaitan and Lane Clark. Graph isomorphism and adiabatic quantum computing. *Physical Review A*, 89(2):022342, 2014.

[74] Frank Gaitan and Lane Clark. Graph isomorphism and adiabatic quantum computing. *Phys. Rev. A*, 89:022342, Feb 2014.

[75] Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. wh freeman, 2002.

[76] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

[77] Maria Gillespie. Characters of the symmetric group. `https://mathematicalgemstones.wordpress.com/2012/05/21/characters-of-the-symmetric-group/`, 2012.

[78] Michelangelo Grigni, Leonard Schulman, Monica Vazirani, and Umesh Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 68–74. ACM, 2001.

[79] Sean Hallgren, Cristopher Moore, Martin Rötteler, Alexander Russell, and Pranab Sen. Limitations of quantum coset states for graph isomorphism. *Journal of the ACM (JACM)*, 57(6):34, 2010.

[80] Sean Hallgren, Alexander Russell, and Amnon Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 627–635. ACM, 2000.

[81] Peter L Hammer, Pierre Hansen, and Bruno Simeone. Roof duality, complementation and persistency in quadratic 0–1 optimization. *Mathematical programming*, 28(2):121–155, 1984.

[82] Peter L Hammer, I Rosenberg, and Sergiu Rudeanu. On the determination of the minima of pseudo-boolean functions. *Studii si Cercetari matematice*, 14:359–364, 1963.

[83] Peter L Hammer and Sergiu Rudeanu. *Boolean methods in operations research and related areas*, volume 7. Springer Science & Business Media, 2012.

[84] PL Hammer, I Rosenberg, and S Rudeanu. Application of discrete linear programming to the minimization of boolean functions. *Rev. Mat. Pures Appl*, 8:459–475, 1963.

[85] Frank Harary. Graph theory. 1969.

[86] Frank Harary and Edgar M Palmer. *Graphical enumeration*. Elsevier, 2014.

[87] R. Harris, J. Johansson, A. J. Berkley, M. W. Johnson, T. Lanting, Siyuan Han, P. Bunyk, E. Ladizinsky, T. Oh, I. Perminov, E. Tolkacheva, S. Uchaikin, E. M. Chapple, C. Enderud, C. Rich, M. Thom, J. Wang, B. Wilson, and G. Rose. Experimental demonstration of a robust and scalable flux qubit. *Phys. Rev. B*, 81:134510, Apr 2010.

[88] Donald Olding Hebb. The organization of behavior: A neuropsychological theory. 2005.

[89] Pinar Heggernes. Treewidth, partial k-trees, and chordal graphs. *Internet document: http://www. ii. uib. no/pinar/chordal. pdf*, 2005.

[90] Itay Hen and Marcelo S Sarandy. Driver hamiltonians for constrained optimization in quantum annealing. *arXiv preprint arXiv:1602.07942*, 2016.

[91] Itay Hen and AP Young. Solving the graph-isomorphism problem with a quantum annealer. *Physical Review A*, 86(4):042310, 2012.

[92] C. Hillar and N. M. Tran. Robust exponential memory in hopfield networks. *arXiv:1411.4625*, 2014.

[93] Judy A Holdener. Math bite: Sums of sines and cosines. *Mathematics Magazine*, 82(2):126–126, 2009.

[94] Derek F Holt, Bettina Eick, and Eamonn A O'Brien. *Handbook of computational group theory*. CRC Press, 2005.

[95] John E Hopcroft and Jin-Kue Wong. Linear time algorithm for isomorphism of planar graphs (preliminary report). In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 172–184. ACM, 1974.

[96] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[97] JJ Hopfield and DW Tank. Computing with neural circuits: a model. *Science*, 233(4764):625–633, 1986.

[98] Victoria Horan, Steve Adachi, and Stanley Bak. A comparison of approaches for finding minimum identifying codes on graphs. *Quantum Information Processing*, pages 1–22, 2016.

[99] Gábor Ivanyos, Frédéric Magniez, and Miklos Santha. Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem. *International Journal of Foundations of Computer Science*, 14(05):723–739, 2003.

[100] Robert Jajcay. The structure of automorphism groups of cayley graphs and maps. *Journal of Algebraic Combinatorics*, 12(1):73–84, 2000.

[101] Sabine Jansen, Mary-Beth Ruskai, and Ruedi Seiler. Bounds for the adiabatic approximation with applications to quantum computation. *arXiv preprint quant-ph/0603175*, 2006.

[102] Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random graphs*, volume 45. John Wiley & Sons, 2011.

[103] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi, E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. S. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, 05 2011.

[104] P Jordan and Eugene P Wigner. About the pauli exclusion principle. *Z. Phys*, 47(631):14–75, 1928.

[105] Richard Jozsa. Quantum algorithms and the fourier transform. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 454, pages 323–337. The Royal Society, 1998.

[106] Samuel J Lomonaco Jr. and Louis H Kauffman. Quantum hidden subgroup algorithms: a mathematical perspective. quantum computation and information (washington, dc, 2000). *Contemp. Math.*, 305:139202, 2002.

[107] William M Kantor. Cycles in graphs and groups. *The American Mathematical Monthly*, 115(6):559–562, 2008.

[108] Andrew D King and Catherine C McGeoch. Algorithm engineering for a quantum annealing platform. *arXiv preprint arXiv:1410.2628*, 2014.

[109] James King, Sheir Yarkoni, Mayssam M Nevisi, Jeremy P Hilton, and Catherine C McGeoch. Benchmarking a quantum annealing processor with the time-to-target metric. *arXiv preprint arXiv:1508.05087*, 2015.

[110] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[111] Michael P Knapp. Sines and cosines of angles in arithmetic progression. *Mathematics Magazine*, 82(5):371–372, 2009.

[112] Donald E Knuth. Efficient representation of perm groups. *Combinatorica*, 11(1):33–43, 1991.

[113] Johannes Kobler, Uwe Schöning, and Jacobo Torán. *The graph isomorphism problem: its structural complexity*. Springer Science & Business Media, 2012.

[114] Yokesh Kumar and Prosenjit Gupta. External memory layout vs. schematic. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 14(2):30, 2009.

[115] Aung Kyaw. A sufficient condition for a graph to have a k-tree. *Graphs and Combinatorics*, 17(1):113–121, 2001.

[116] T Lanting, AJ Przybysz, A Yu Smirnov, FM Spedalieri, MH Amin, AJ Berkley, R Harris, F Altomare, S Boixo, P Bunyk, et al. Entanglement in a quantum annealing processor. *Physical Review X*, 4(2):021041, 2014.

[117] Wolfgang Lechner, Philipp Hauke, and Peter Zoller. A quantum annealing architecture with all-to-all connectivity from local interactions. *Science advances*, 1(9):e1500838, 2015.

[118] L. Lovász, J. Pelikán, and K. Vesztergombi. *Discrete Mathematics*. Springer undergraduate texts in mathematics, 2003.

[119] Andrew Lucas. Ising formulations of many np problems. *arXiv preprint arXiv:1302.5843*, 2013.

[120] Andrew Lucas. Ising formulations of many np problems. *Frontiers in Physics*, 2(5), 2014.

[121] Eugene M Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, 25(1):42–65, 1982.

[122] A. Manju and M.J. Nigam. Applications of quantum inspired computational intelligence: a survey. *Artificial Intelligence Review*, 42(1):79–156, 2014.

[123] WarrenS. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bul. of Math. Biophys.*, 5(4):115–133, 1943.

[124] R.J. McEliece, Edward C. Posner, Eugene R. Rodemich, and S.S. Venkatesh. The capacity of the hopfield associative memory. *Information Theory, IEEE Transactions on*, 33(4):461–482, Jul 1987.

[125] Catherine C McGeoch. Adiabatic quantum computation and quantum annealing: Theory and practice. *Synthesis Lectures on Quantum Computing*, 5(2):1–93, 2014.

[126] Brendan D. McKay. Graph isomorphism. In Ping Zhang, editor, *Handbook of Graph Theory*. Chapman and Hall/CRC, 2013.

[127] Brendan D McKay et al. *Practical graph isomorphism*. Department of Computer Science, Vanderbilt University, 1981.

[128] Brendan D McKay and Adolfo Piperno. Practical graph isomorphism, ii. *Journal of Symbolic Computation*, 60:94–112, 2014.

[129] Brendan D McKay and Nicholas C Wormald. Uniform generation of random regular graphs of moderate degree. *Journal of Algorithms*, 11(1):52–67, 1990.

[130] E Mendelsohn. Every (finite) group is the group of automorphisms of a (finite) strongly regular graph. *Ars Combinatoria*, 6:75–86, 1978.

[131] Eric Mendelsohn. On the groups of automorphisms of steiner triple and quadruple systems. *Journal of Combinatorial Theory, Series A*, 25(2):97–104, 1978.

[132] Carl D Meyer. *Matrix analysis and applied linear algebra*, volume 2. Siam, 2000.

[133] Štefko Miklavič, Primož Potočnik, and Steve Wilson. Consistent cycles in graphs and digraphs. *Graphs and Combinatorics*, 23(2):205–216, 2007.

[134] Gary Miller. Isomorphism testing for graphs of bounded genus. In *Proceedings of the twelfth annual ACM symposium on Theory of computing*, pages 225–235. ACM, 1980.

[135] Cristopher Moore, Daniel Rockmore, and Alexander Russell. Generic quantum fourier transforms. *ACM Transactions on Algorithms (TALG)*, 2(4):707–723, 2006.

[136] Cristopher Moore, Daniel Rockmore, Alexander Russell, and Leonard Schulman. The hidden subgroup problem in affine groups: Basis selection in fourier sampling. *arXiv preprint quant-ph/0211124*, 2002.

[137] Cristopher Moore, Daniel Rockmore, Alexander Russell, and Leonard J Schulman. The power of strong fourier sampling: quantum algorithms for affine groups and hidden shifts. *SIAM Journal on Computing*, 37(3):938–958, 2007.

[138] Cristopher Moore and Alexander Russell. Explicit multiregister measurements for hidden subgroup problems. *arXiv preprint quant-ph/0504067*, 2005.

[139] Cristopher Moore and Alexander Russell. For distinguishing conjugate hidden subgroups, the pretty good measurement is as good as it gets. *arXiv preprint quant-ph/0501177*, 2005.

[140] Cristopher Moore and Alexander Russell. Tight results on multiregister fourier sampling: Quantum measurements for graph isomorphism require entanglement. *arXiv preprint quant-ph/0511149*, 2005.

[141] Cristopher Moore, Alexander Russell, and Leonard J Schulman. The symmetric group defies strong fourier sampling. *SIAM Journal on Computing*, 37(6):1842–1864, 2008.

[142] Cristopher Moore, Alexander Russell, and Piotr Sniady. On the impossibility of a quantum sieve algorithm for graph isomorphism. *SIAM Journal on Computing*, 39(6):2377–2396, 2010.

[143] Satoshi Morita and Hidetoshi Nishimori. Mathematical foundation of quantum annealing. *Journal of Mathematical Physics*, 49(12), 2008.

[144] Daniel Nagaj. Local hamiltonians in quantum computation. *arXiv preprint arXiv:0808.2117*, 2008.

[145] Rodion Neigovzen, Jorge L. Neves, Rudolf Sollacher, and Steffen J. Glaser. Quantum pattern recognition with liquid-state nuclear magnetic resonance. *Phys. Rev. A*, 79:042321, Apr 2009.

[146] M.A. Nielsen and I.L. Chuang. *Quantum Computation and Quantum Information*. Cambridge Series on Information and the Natural Sciences. Cambridge University Press, 2000.

[147] Patrick Eugene ONeil et al. Asymptotics and random matrices with row-sum and column-sum restrictions. *Bull. Amer. Math. Soc*, 75(6):1276–1282, 1969.

[148] K. Pagiamtzis and A. Sheikholeslami. Content-addressable memory (cam) circuits and architectures: a tutorial and survey. *Solid-State Circuits, IEEE Journal of*, 41(3):712–727, March 2006.

[149] L. Personnaz, I. Guyon, and G. Dreyfus. Collective computational properties of neural networks: New learning mechanisms. *Phys. Rev. A*, 34:4217–4228, Nov 1986.

[150] George Pólya. Kombinatorische anzahlbestimmungen für gruppen, graphen und chemische verbindungen. *Acta mathematica*, 68(1):145–254, 1937.

[151] Kristen L. Pudenz, Tameem Albash, and Daniel A. Lidar. Quantum annealing correction for random ising problems. *Phys. Rev. A*, 91:042302, Apr 2015.

[152] Harishchandra S Ramane and Ashwini S Yalnaik. Reciprocal complementary distance spectra and reciprocal complementary distance energy of line graphs of regular graphs. *Electronic Journal of Graph Theory and Applications (EJGTA)*, 3(2):228–236, 2015.

[153] RC Read. The enumeration of locally restricted graphs (i). *Journal of the London Mathematical Society*, 1(4):417–436, 1959.

[154] RC Read. The enumeration of locally restricted graphs (ii). *Journal of the London Mathematical Society*, 1(3):344–351, 1960.

[155] Ronald C. Read and Derek G. Corneil. The graph isomorphism disease. *Journal of Graph Theory*, 1(4):339–363, 1977.

[156] Edna E Reiter and Clayton Matthew Johnson. *Limits of computation: an introduction to the undecidable and the intractable*. CRC Press, 2012.

[157] Eleanor G Rieffel, Davide Venturelli, Bryan OGorman, Minh B Do, Elicia M Prystay, and Vadim N Smelyanskiy. A case study in programming a quantum annealer for hard operational planning problems. *Quantum Information Processing*, 14(1):1–36, 2015.

[158] Oliver Riordan and Alex Selby. The maximum degree of a random graph. *Combinatorics, Probability and Computing*, 9(06):549–572, 2000.

[159] Martin Roetteler and Thomas Beth. Polynomial-time solution to the hidden subgroup problem for a class of non-abelian groups. Technical report, 1998.

[160] Andrzej Ruciński and Nicholas C Wormald. Random graph processes with degree restrictions. *Combinatorics, Probability and Computing*, 1(02):169–180, 1992.

[161] Gert Sabidussi. Graphs with given group and given graph-theoretical properties. *Canad. J. Math*, 9(515):C525, 1957.

[162] Bruce Sagan. *The symmetric group: representations, combinatorial algorithms, and symmetric functions*, volume 203. Springer Science & Business Media, 2013.

[163] Siddhartha Santra, Gregory Quiroz, Greg Ver Steeg, and Daniel A Lidar. Max 2-sat with up to 108 qubits. *New Journal of Physics*, 16(4):045006, 2014.

[164] Siddhartha Santra, Omar Shehab, and Radhakrishnan Balu. Exponential capacity of associative memories under quantum annealing recall. *arXiv preprint arXiv:1602.08149*, 2016.

[165] M. S. Sarandy and D. A. Lidar. Adiabatic quantum computation in open systems. *Phys. Rev. Lett.*, 95:250503, Dec 2005.

[166] Uwe Schöning. Graph isomorphism is in the low hierarchy. *Journal of Computer and System Sciences*, 37(3):312–323, 1988.

[167] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[168] H. Seddiqi and T. Humble. Adiabatic quantum optimization for associative memory recall. *arXiv:1407.1904*, 2014.

[169] Ákos Seress. *Permutation group algorithms*, volume 152. Cambridge University Press, 2003.

[170] Jean-Pierre Serre. *Linear representations of finite groups*, volume 42. Springer Science & Business Media, 2012.

[171] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[172] Johannes Siemons. Automorphism groups of graphs. *Archiv der Mathematik*, 41(4):379–384, 1983.

[173] Charles C Sims. Computational methods in the study of permutation groups. In *Computational problems in abstract algebra*, pages 169–183, 1970.

[174] Alistair Sinclair and Mark Jerrum. Approximate counting, uniform generation and rapidly mixing markov chains. *Information and Computation*, 82(1):93–133, 1989.

[175] Richard P Stanley. *What Is Enumerative Combinatorics?* Springer, 1986.

[176] A.J. Storkey and R. Valabregue. The basins of attraction of a new hopfield learning rule. *Neural Networks*, 12(6):869 – 876, 1999.

[177] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams. The missing memristor found. *Nature 453, 80-83*, 2008.

[178] M. Talagrand. *Mean field models for spin glasses*. Springer Modern surveys in mathematics, 2011.

[179] Dario Tamascelli and Luca Zanetti. A quantum-walk-inspired adiabatic algorithm for solving graph isomorphism problems. *Journal of Physics A: Mathematical and Theoretical*, 47(32):325302, 2014.

[180] Wim Van Dam, Michele Mosca, and Umesh Vazirani. How powerful is adiabatic quantum computation? In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 279–287. IEEE, 2001.

[181] Walter Vinci, Klas Markström, Sergio Boixo, Aidan Roy, Federico M Spedalieri, Paul A Warburton, and Simone Severini. Hearing the shape of the ising model with a programmable superconducting-flux annealer. *Scientific reports*, 4, 2014.

[182] Di Wang and Robert Kleinberg. Analyzing quadratic unconstrained binary optimization problems via multicommodity flows. *Discrete Applied Mathematics*, 157(18):3746–3753, 2009.

[183] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[184] Eugene Wigner. Einige folgerungen aus der schrödingerschen theorie für die termstrukturen. *Zeitschrift für Physik*, 43(9-10):624–652, 1927.

[185] Nicholas C Wormald. Generating random regular graphs. *Journal of Algorithms*, 5(2):247–280, 1984.

[186] Ming-Yao Xu. Automorphism groups and isomorphisms of cayley digraphs. *Discrete Mathematics*, 182(1):309–319, 1998.

[187] Kenneth M. Zick, Omar Shehab, and Matthew French. Experimental quantum annealing: case study involving the graph isomorphism problem. *Scientific Reports*, 5:11168, jun 2015.