CLASSICAL OPTIMIZATION TO IMPROVE VARIATIONAL QUANTUM ALGORITHMS

A Dissertation
Presented to
The Academic Faculty

By

Reuben Tate

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Mathematics
Algorithms, Combinatorics, and Optimization

Georgia Institute of Technology

May 2023

CLASSICAL OPTIMIZATION TO IMPROVE VARIATIONAL QUANTUM ALGORITHMS

Thesis committee:

Dr. Swati Gupta
Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Stuart Hadfield
Quantum Artificial Intelligence Laboratory
National Aeronautics and Space Administration

Dr. Gregory Mohler
Cybersecurity, Information Protection, and Hardware Evaluation Research
Georgia Tech Research Institute

Dr. David Gamarnik
Operations Research
Massachusetts Institute of Technology

Dr. Grigoriy Blekherman
Mathematics
Georgia Institute of Technology

Date approved: April 12, 2023

To everyone who believed in me,

ACKNOWLEDGMENTS

First, I would like to express my deepest thanks to my advisor Swati Gupta. I have gained an interest and appreciation for the field of quantum computing as a result of working with her. She has also been invaluable in regards to giving advice, networking, and navigating certain aspects of academia; I thank her for her patience, guidance, and for believing in me.

I would also like to thank Greg Mohler, Grigoriy Blekherman, Stuart Hadfield, and David Gamarnik for taking the time to serve on my dissertation committee. I would also like to express my gratitude to my coauthors: Creston Herold, Jai Moondra, Swati Gupta, Bryan Gard, Greg Mohler, Majid Farhadi.

I also very much appreciate all the help and support from everyone involved in the Feymman Quantum Academy Internship Program at the USRA-NASA Quantum Artificial Intelligence Laboratory; I would like to give additional special thanks to my mentor Stuart Hadfield for their advice and kind guidance during and beyond the internship.

I would also like to extend my gratitude to all the faculty and staff in the ACO program and the Mathematics department at Georgia Tech for teaching excellent classes, providing invaluable support, giving many intriguing talks, and overall creating a rich and vibrant academic program and community.

Next, I would like to give a thank you to DARPA for providing the funding necessary for me to work on the Optimization with Trapped Ion Qubits (OPTIQ) team. On the OPTIQ team, I would like to give my sincere thanks to Creston Herold and Greg Mohler in particular who were instrumental in helping me get "up-to-speed" on quantum computing; their experiences and perspectives as physicists were also very helpful.

I also have my parents, parental figures (Stepahnie Oshiro and Darryl Mizuguchi), and many others in my family to thank for their unwavering support and encouragement throughout my life. I am also eternally grateful for my partner, Jonathon Lucas Mitchell,

for their love and support, for putting up with my erratic sleep schedule and vanilla coke addiction, for surprising me with little treats, for the countless rides to the airport, and for ultimately keeping me sane; I could not have done this with without him.

I have countless friends to thank; however, I would like to specifically thank my friend Jad Salem, also an ACO student; they have had my back since Day 1 of arriving at Georgia Tech and I am happy to have gone through this academic journey with them. I would also like to acknowledge my friends who are affiliated with Georgia Tech's SOFT organization who have made my experiences in Atlanta both fun and memorable.

Lastly, there are countless others (e.g. old teachers and friends) from before my time at Georgia Tech who have previously provided support and encouragement and have molded me into the person I am today; even though, for some, our paths in life may have diverged, I am forever grateful for our time together, thank you.

TABLE OF CONTENTS

Acknowledgments iv

List of Tables x

List of Figures xiii

Summary xx

Chapter 1: Introduction 1

1.1 Contributions of the Thesis 5

1.2 Related Work 10

Chapter 2: Background and Notation 13

2.1 General Quantum Computing Background 13

2.1.1 Classical Perspective 13

2.1.2 General Qubits 19

2.1.3 Geometric Interpretation: Bloch Sphere 21

2.1.4 Quantum Operations 22

2.1.5 Properties of the Kronecker Product 25

2.1.6 Multi-Qubit Operations 30

2.2 Approximation Ratio 31

2.3 Classical Methods for Max-Cut 32

2.4 The Quantum Approximate Optimization Algorithm 37

2.5 Independent Set Problem 39

Chapter 3: Interesting Instances For Quantum Advantage 41

3.1 Small Instances from Karloff's Construction with Low GW Approximation Ratios 44

3.1.1 Review of Karloff's Construction 44

3.1.2 Identification of Small Instances Using Karloff's Approach 46

3.2 Provable Guarantees for the GW Algorithm on Strongly-Regular Graphs . . 48

3.3 Empirical Results 58

3.3.1 Instances where Classical Heuristics Perform Poorly 59

3.3.2 Extensions of Karloff's Construction 61

3.4 QAOA's Performance on Challenging Instances for the GW Algorithm . . . 64

Chapter 4: Warm-starts for QAOA using Standard Mixers 70

4.1 Framework for Constructing Initial Quantum States 70

4.1.1 Random Rotations 71

4.1.2 Mapping to Bloch Sphere 74

4.2 Initialization Schemes for Warm-Start State 75

4.2.1 k-dimsional Burer-Monteiro 75

4.2.2 Projected Goemans-Williamson 82

4.2.3 Single Cut Initializations 90

4.3 Limitations of QAOA-Warm 96

4.4 QAOA-Warm on Antipodal Structures 98
4.5 Numerical Simulations for QAOA-Warm 107
4.5.1 Experimental Setup 108
4.5.2 Optimizer Choice 111
4.5.3 Choice of Rank and Rotations 113
4.5.4 Aggregate Results 114
4.5.5 Parameter Landscapes and Trajectories 118
4.5.6 Pre-processing Time vs Parameter Search Time 122
4.5.7 QAOA-Warm with Median and Worst Vertex-At-Top Rotations . . . 124
4.6 Discussion 126
4.7 Conclusion 128

Chapter 5: Warm-Starts with Customized Mixers 131

5.1 Custom Mixer Construction 131
5.2 Proof of Convergence in the Adiabatic Limit 133
5.2.1 Eigenstates of Custom Mixers 133
5.2.2 Eigenvalue Gap for Custom Mixers 135
5.2.3 Proof of Convergence (Special Case) 139
5.2.4 Proof of Convergence (General) 140
5.3 Convergence Rate of QAOA-warmest 145
5.4 Numerical Simulations and Hardware Experiments for QAOA-Warmest . . 146
5.4.1 Comparing QAOA-warmest to Other Methods 147
5.4.2 QAOA-warm With Noise 155

5.4.3 Projected GW vs BM-MC Warm-Starts . . . 160
5.4.4 Choice of Rank and Rotation . . . 161
5.4.5 Projected GW Solutions and BM-MC Scaling . . . 161
5.4.6 Interesting Instances . . . 162
5.4.7 Comparison with Egger et al. . . . 163
5.5 Discussion . . . 164

Chapter 6: Warm-Starts from Classical Local Algorithms . . . 166
6.1 Using Ancilla Qubits to Emulate Classical Algorithms . . . 167
6.1.1 Properties of Local-Search Quantum Algorithm . . . 173
6.2 Effect of QAOA on States with Ancilla . . . 178
6.2.1 Effects on Measurement on States with Ancilla . . . 180
6.3 Discussion and Future Directions . . . 182

Chapter 7: Conclusion . . . 184
7.1 Open Questions . . . 185

Appendices . . . 188
Appendix A: Partial Geometries . . . 189

References . . . 194

Vita . . . 205

LIST OF TABLES

3.1 A listing of small (< 1000 nodes) instances using Karloff's construction. For each instance, we include the number of nodes, edges, and the theoretical instance-specific approximation ratio one would obtain if running the GW algorithm on that instance (assuming an optimal solution of Y as described earlier). 47

3.2 Instances $G = (V, E)$ of strongly-regular graphs parameterized by $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some t, where $\frac{\text{Max-Cut}(G)}{|E|} \neq \frac{2}{3}$. The last column is the instance-specific approximation ratio that the GW algorithm achieves on each of these instances. 57

3.3 The table includes all the instances in the MQLib library for which no MQLib heuristic obtained 99.9% of the optimal solution within 5% of the alloted runtime (abbreviated as R.T. throughout the table) described in Footnote 1. In the first part of the table, the columns correspond to the instance name (as given in the MQLib library), the number of nodes and (nonzero-weight) edges, the allotted runtime (in seconds), the instance-specific GW approximation ratio $\alpha_{\text{GW},G}$, the time needed to run the GW solver in seconds, whether the instance has negative-weighted edges, and the range of edge-weight magnitudes (defined as $\max_{e\in E} \log(|w_e|) - \min_{e\in E} \log(|w_e|)$). In the second part of the table, we find the best approximation ratio obtained by a MQLib heuristic that is allowed to run for only 5% of the alloted runtime; the second and third columns correspond to this ratio and the corresponding heuristic (with ties between heuristics being broken by runtime). The next two columns are obtained the same way but with the full allotted runtime being allowed. All ratios in the table are written with the denominator being the value of the Max-Cut for that instance. The GW algorithm times and the MQLib heuristic times were computed on different machines so these times are incomparable to one another. 62

4.1 Multiple tables comparing the average instance-specific approximation ratio achieved during QAOA-warm when utilizing different combinations of ranks and rotations during the preprocessing stage. For the top row of tables, these averages were computed using all the graphs in our graph library $\mathcal{G}$ (see Section 4.5.1) whereas for the bottom row, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights. Each run of standard QAOA and QAOA-warm terminates when the difference in successive values of $F_p(\gamma, \beta)$ is less than $10^{-6}\bar{W}$ where $\bar{W}$ is the sum of the absolute values of the edge weights. . . . 113

4.2 We consider 4 algorithms: Goemans-Williamson (G), 2-dimensional Burer-Monteiro with hyperplane rounding (B), QAOA-warm (W), and standard QAOA (S). There is a row for each of the $4! = 24$ ways the algorithms can perform relative to one another with the cell value indicating the percentage of instances for which that ordering occurs. As an example, the top-leftmost value indicates that for 25.08% of instances, $W \geq B \geq G \geq S$ in terms of AR with W and S being depth-1. The four largest entries in each column are bolded for emphasis. To account for numerical error for nearly solved instances, we declare QAOA-warm (W) as the best as long as it is within 0.001 AR of the best algorithm. We include columns corresponding to the entire graph library $\mathcal{G}$ as well as the subset of $\mathcal{G}$ that have positive-weighted edges. 115

4.3 The table summarizes what is known about different variants of QAOA (for Max-Cut) based on various combinations of initializations (equal superposition, BM-MC$_k$, projected GW, and single-cut initializations) and mixing Hamiltonian (standard, custom mixers (Section 5), and the mixer proposed by Egger et al. [42] for single-cut initializations). For various combinations, we state what is known regarding the convergence and worst-case approximation ratio (for depths $p \geq 0$) of the corresponding QAOA variant for graphs with non-negative edge weights. 130

5.1 For each Max-Cut algorithm (Goemans-Williamson, standard QAOA, QAOA-warmest, and QAOA-warm), we report the percentage of instances for which it did the best and second-best (in terms of approximation ratio). Both QAOA-warm and QAOA-warmest use BM-MC$_2$ warm-starts. There is a tie (last column) if the top two algorithms have approximation ratios that differ by no more than 0.01. QAOA-warmest is a part of every tie. Each instance is either accounted for in "Tie" or the other columns. For the column labeled *, we report, for each circuit depth, the percentage of instances for which QAOA-warmest was within 0.01 approximation ratio of the best algorithm. 148

5.2 The percentage of instances for which QAOA-warmest achieves an instance-specific AR of 99.0% for each combination of circuit depth and initialization method (standard initialization, BM-MC$_2$ initialization, single-cut initialization with $\theta^* = 0.1$). 151

5.3 These tables reports the approximation ratios achieved for the five instances (amongst those in our instance library $\mathcal{G}$) for which depth-8 QAOA-warmest did not obtain the best approximation ratio when compared to depth-8 QAOA-warm, depth-8 standard QAOA, and the GW algorithm. The top and bottom tables are for instances in which standard QAOA and the GW algorithm performed the best respectively. The instances in the bottom table have the property that there exists exactly one negative edge weight whose magnitude is much larger than the other edge weights. For the bottom table, in the last column, we also include the approximation ratio for QAOA-warmest in the case where a more suitable vertex-at-top rotation is used; i.e., we take one of the vertices incident to the large-magnitude negative edge and rotate it to the top. 160

5.4 Multiple tables comparing the average (instance-specific) approximation ratio achieved during QAOA-warmest when utilizing different combinations of ranks and rotations during the preprocessing stage. For the top row of tables, these averages were computed using all the graphs in our graph library $\mathcal{G}$ whereas for the bottom row, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights. 161

LIST OF FIGURES

1.1 To the left, an unweighted graph $G = (V, E)$ with five vertices. To the right, the node colors correspond to the partition of V for the max-cut; the node placements are altered to illustrate that all but one edge "goes across" the cut. Since the graph is not bipartite, the cut illustrated on the right is optimal. 4

2.1 Two coins, C_1 (red) and C_2 (blue) connected by a piece of glue so that they are always either both heads-side-up or tails-side-up. 18

2.2 A single-qubit state $|\psi\rangle$ represented on the surface of the Bloch sphere. 20

3.1 The plot shows the instances that the BiqCrunch solver solved exactly. Each instance is represented by a single point with the position dependent on the number of vertices, n, in the graph and the density of the graph. The logarithm in the plot is base-e and density is defined to be the ratio between the sum of the normalized absolute edge weights and the total number of possible edges in the graph, i.e.,the density is $\frac{\sum_{e\in E}|w_e|}{\binom{n}{2}\max_{e\in E}|w_e|}$, where w_e is the weight of edge e. Aside from some graphs with densities close to zero, the solver could only solve instances of up to 735 nodes (indicated by the black line on the figure). A time limit of 24 hours was imposed on the solver (excluding preprocessing). 60

3.2 A PCA analysis of the MQLib library with respect to 58 graph properties. Gray dots represent instances that were not solved (exactly) by the BiqCrunch solver. The remaining non-red dots are colored based on which heuristic was first to obtain 99.9% of the optimal solution with 5% of the allotted runtime. The red instances (circled) correspond to $\mathcal{G}_{\text{CC}}$, instances for which no heuristic reached 99.9% of the optimal solution with 5% of the allotted runtime. 60

3.3 The above is the edge-weight distribution of instance g003179. As the edge weights are integers, we see that there are both positive and negative edge weights spanning several orders of magnitude; a similar edge weight distribution can be found for the other MQLib instances in Table 3.3 with the exception of instances g000435 and g001349. 61

3.4 Both plots show the result of the GW algorithm when modifying the instances made via Karloff's construction. In particular, we consider modifications of $J(6,3,1)$, $J(8,4,1)$, $J(10,5,1)$ and $J(10,5,2)$. The top plot shows the estimated approximation ratio using the GW algorithm for instances modified via edge deletions (with up to 4 edges removed) where the estimated approximation ratio for an instance G' is calculated as $\frac{\text{Round}(G',Y')}{\text{Max-Cut}(G)-|F|}$ where G is the corresponding (unmodified) original instance from Karloff's construction and $|F|$ is the number of edges deleted (as seen in Equation 3.1); the actual approximation ratios could possibly be lower for each instance. The bottom plot shows the approximation ratio using the GW algorithm for instances modified via edge weight perturbations (as described in Section 3.3.2). The bottom plot also includes the GW approximation ratio for the interesting instances in the MQLib library (found in Section 3.3.1). 65

4.1 Comparison of the hyperplane rounding and quantum sampling for a 3-cycle (Max-Cut=2): figure (a) shows a local optimal BM-MC$_3$ solution, where any random hyperplane will give a cut of size 2. Both (b) and (c) show two different embeddings of the BM-MC$_3$ solution (from (a)) onto the Bloch sphere. In (b), the qubits lie on $x = 0$ plane and quantum sampling results in a expected cut of 1.875. In (c), all qubits lie on the equator of the Bloch sphere (similar to the standard start of QAOA), so each edge has a probability of $1/2$ of being cut, yielding a total expected cut of 1.5. Both (b) and (c) demonstrate that the orientation of the rotated BM-MC$_3$ solution is important when embedding it into the Bloch sphere and can result in different expected cuts. 71

4.2 We begin with the classically obtained x^*. We then apply a rotation $R \in \{R_U, R_V\}$; here we show R_U being applied (top-right). Lastly, we use Q to map this rotated solution to a quantum product state. 72

4.3 Pie charts representing best expected cut value (expectation over randomness in sampling) obtained by using (i) hyperplane rounding of the BM-MC$_k$ solution (HR), (ii) quantum sampling of the BM-MC$_k$ solution (QS1), and quantum sampling of the initial state of standard QAOA (QS2). For every instance, QS2 always yielded the worst result of the three, and for majority of the instances QS1 $\geq$ HR. For HR and QS1, the best of 5 (in terms of SDP objective) locally optimal BM-MC$_k$ solutions are used; for that solution, the best of 5 rotations is used for QS1. The regions marked in gray indicate instances for which QS1 and HR had a tie (difference in instance-specific approximation ratio of at most 0.001). 78

4.4 In red, the function $\frac{\frac{1}{2}\left(1-\frac{\cos\theta}{k}\right)}{\theta/\pi}$ that is minimized in the proof of Corollary 20 with $k = 3$. In green, a similar function $\frac{\frac{1}{2}\left(1-\frac{\cos\theta}{k}\right)}{\frac{1}{2}(1-\cos(\theta)}$ that is minimized in the proof of Theorem16 (where the BM-MC$_k$ objective is compared to the maximum cut instead of the expected cut value from hyperplane rounding) with $k = 3$. Over the interval $[0, \pi]$, both achieve a minimum value of $2/3$ at $\theta = \pi$. The corresponding plots for $k = 2$ are similar but instead both functions reach a minimum value of $3/4$ at $\theta = \pi$. 89

4.5 A schematic for projected GW warm-starts. There are three procedures to obtain a cut from an SDP solution. The first is to use Goemans-Williamson hyperplane rounding (on the top labelled I). The second (labelled II) is to do a two-step rounding through an intermediate state in $\mathbb{R}^k, k \in \{2, 3\}$. We prove that this two-step rounding procedure is equivalent to Goemans-Willimson hyperplane rounding in Theorem 19. The third procedure is our proposed warm-start of QAOA using the SDP solution (highlighted in blue, labelled III). This procedure involves rounding the SDP solution to $\mathbb{R}^k$ first, then rotating this solution using uniform or vertex-at-top rotations and mapping to the Bloch sphere to get an initial state for QAOA, and finally running QAOA on this initial state. 90

4.6 A plot of the percentage of the Max-Cut achieved with QAOA-warm (when the optimal variational parameters are chosen) with $p = 1$ for a one-edge graph G at various starting states $|s_0\rangle = Q(x)$ where one point of x has polar angle θ and azimuthal angle φ and the remaining point is diametrically opposed. The starting states that perform the worst, i.e. $|+-\rangle$ and $|-+\rangle$, are marked with a black $\times$. For each point in the figure, the optimal variational parameters were estimated by performing a dense grid-search over the variational parameter space. 98

4.7 Parameter landscapes of the instance-specific approximation ratio of the four-cycle C_4 for $p = 1$. When no warm start is used, the landscape has many peaks and valleys in the form of local maxima and minima (a). For both BM-MC$_2$ and BM-MC$_3$, a vertex-at-top rotation yields a convex landscape with a ridge-like shape (b), thereby effectively capturing the optimal solution for the 4-cycle. When a uniform rotation is used, a BM-MC$_2$ formulation (c) achieves optimality for *some* choice of parameters whereas this is not the case for a BM-MC$_3$ formulation (d). 99

4.8 To the left is the rank-1 solution x^*. To the right is the solution x' after moving the points in A and B by angle θ along the unit circle. Thale's Theorem tells us that connecting the endpoints of the diameter of a circle with another point on the circle yields a right triangle. 101

4.9 Illustration depicting the range of graph metrics for our instance library $\mathcal{G}$. When comparing unit-weight Erdős-Rényi graphs (red) with the remaining graphs in $\mathcal{G}$ (blue), there is an increase in the range of values obtained for both graph metrics. 110

4.10 Histogram of differences in instance-specific approximation ratio (AR) $\alpha_{\mathcal{A},G}$ between ADAM and other optimizers for QAOA-Warm (top) and standard QAOA (bottom). Overlapping regions are in purple. The red bin indicates instances for which optimizers performed similarly to ADAM. 112

4.11 Histogram of differences in instance-specific approximation ratios between QAOA-warm and standard QAOA. Overlapping regions are in purple. 114

4.12 The number of instances for which QAOA-warm obtained at least an $r\%$ improvement in AR as the circuit depth increases from $p = 0$ to $p = 1$ (left) and from $p = 1$ to $p = 8$ (right). For each instance, the best percent improvement (across all five vertex-at-top rotations) is used. Note that % improvements in instance-specific approximation ratios go up to 80-120% from $p = 0$ to $p = 1$, and up to 12-20% from as depth increases from $p = 1$ to $p = 8$. 118

4.13 This figure shows how standard QAOA (dotted) and QAOA-warm (solid) perform as we alter the circuit depth and the number of nodes. For QAOA-warm, we take the best of 5 vertex-at-top rotations. For the left plot, for each $n = 2, \ldots, 12$, we find the instance-specific approximation ratio achieved for both standard QAOA and QAOA-warm for each n-node instance in $\mathcal{G}$ (see Section 4.5.1), and take the median of those instance-specific approximation ratios. The right plot is constructed similarly except only instances in $\mathcal{G}$ with positive edge-weights are considered. We plot the results for circuit depths $p = 1, 2, 4, 8$. 118

4.14 Parameter landscapes for $\hat{G}$ (top-left) with corresponding SDP solution (bottom-left). For each trajectory of optimization of the variational parameters, we use a black circle to denote the beginning of the trajectory and a white $\times$ to denote the end of the trajectory. When no warm start is used, there are many peaks and valleys (top-center). When vertex 1 rotated to the top; we have a ridge-like landscape with the optimal solutions occurring on the horizontal line $\beta_1 = 0$ (bottom-center). When rotating vertex 2 at the top instead, the parameter landscape is less ridge-like and the endpoints of the trajectories are more scattered (top-right). When using a uniform rotation we have peaks and valleys similar to when no warm-start was used but with overall better solution qualities (bottom-right). . . 119

4.15 This figure shows how various statistics of the parameter landscape change with the variant of QAOA considered (standard QAOA, QAOA-warm with vertex-at-top rotations, and QAOA-warm with random rotations). For each unit weight graphs in our graph library $\mathcal{G}$ (See Section 4.5.1) and for each QAOA variant, we first generate the parameter landscape; we use a single 2-dimensional initialization for both rotation schemes considered for QAOA-warm. For each landscape, we calculate the minimum, maximum, and average across the landscape in addition to the range (the difference between the highest and lowest instance-specific approximation ratio achieved in the landscape). 121

4.16 This figure shows how the median runtime changes for both GW and BM-MC$_k$ ($k = 2, 3$) as the number of nodes increases. The extended graph library $\mathcal{G}'$ (2076 instances) was used to generate the results above; we also run plot the results for just the positive-weighted graphs in $\mathcal{G}'$ as well. The top and bottom of the colored regions corresponding to 75 and 25 percentiles respectively. 122

4.17 This figure shows how the median runtime changes for the optimization loop of both standard QAOA and QAOA-warm for various optimizers (ADAM, BFGS, and Nelder-Mead). COBYLA was not included due to technical limitations with our software; in particular, we were unable to gain direct access to the source code needed to in order to exclude the runtime of function or gradient evaluations. as the circuit depth increases. These runtimes do not include the time taken to evaluate/estimate the function values or gradients of the expected cut value $F_p(\gamma, \beta)$ (since in practice, such calls would be made on the quantum device). These plots were generated by randomly selecting 20 8-node graphs from our graph library $\mathcal{G}$ (see Section 4.5.1), with 10 of the 20 graphs having only positive edge weights. For each solid colored line (corresponding to the median), there are two dashed lines of the same color above and below representing the 75th and 25th percentiles respectively. On the right, we plot the runtimes for BFGS separately in order to more easily see the trend in runtime as p increases. 123

4.18 Histograms comparing the instance-specific approximation ratio in (depth-p) QAOA-warm and (depth-p) standard QAOA for both $p = 1$ (blue) and $p = 8$ (red) where the median vertex-at-top rotations are used. Overlapping portions of the histogram are in purple. The left plot is generated using the graphs in our graph library $\mathcal{G}$ (see Section 4.5.1) whereas for the bottom right plot, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights. 125

4.19 Histograms comparing the instance-specific approximation ratio in (depth-p) QAOA-warm and (depth-p) standard QAOA for both $p = 1$ (blue) and $p = 8$ (red) where the worst vertex-at-top rotations are used. Overlapping portions of the histogram are in purple. The left plot is generated using the graphs in our graph library $\mathcal{G}$ (see Section 4.5.1) whereas for the right plot, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights. 125

5.1 For both plots, we compare the log-error of QAOA-warmest to both QAOA-warm (right) and standard QAOA (left). BM-MC$_2$ warm-starts are used for both approaches. Each marker in the plot corresponds to a combination of instance (from our graph ensemble $\mathcal{G}$) and circuit depth (either $p = 1$ or $p = 8$) with the shape of the marker being used to denote if the instance has only positive edge weights or not. Points below the black line correspond to instances where QAOA-warmest performs better than the other algorithm being compared. 148

5.2 Trends in median log-error of standard QAOA (dotted), QAOA-warm (dashed), and QAOA-warmest (solid) as one varies the number of nodes and circuit depths; the median is taken across graphs in instance library $\mathcal{G}$. BM-MC$_2$ warm-starts are used for both QAOA-warm and QAOA-warmest. 149

5.3 Instance specific approximation ratios achieved by QAOA-warmest with various types of initializations: standard initialization (equivalent to standard QAOA), BM-MC_2 initialization, single-cut initialization, and uniform random initializations) for a randomly selected 10-node instance. For QAOA-warmest, we used a BM-MC_2 initialization with a vertex-at-top rotation; here we intentionally chose the worst vertex (i.e. the one with worst AR at depth-0) to better illustrate the convergence rate. For the single-cut initialization, we chose a cut that obtains an instance-specific approximation ratio of 0.848 and created initial quantum states using regularization angles of $\theta^* = 0.01, 0.1$, and 0.5 radians. For the uniform random initilizations, five such initializations were created and only the best one was kept (i.e. the one with best AR at depth-0). 152

5.4 Performance of QAOA-warmest (with BM-MC_2 warm-starts) and standard QAOA as a function of QAOA depth for an ideal (dashed) and noisy simulation (dotted). For the chosen 20-node graph, the GW algorithm achieves an approximation ratio of 0.912, while in the ideal case, QAOA-warmest outperforms the GW algorithm for $p \geq 2$ while standard QAOA requires $p > 4$. The noise simulation is based on calibration data from IBM-Q's Guadalupe device. 152

5.5 IBMQ Guadalupe device which shows the physical connectivity of qubits. We choose a native graph which matches this connectvity and random weights as indicated by color. . . 153

5.6 Performance of QAOA-warmest (with BM-MC_2 warm-starts) compared to standard QAOA in an ideal simulation and on IBM-Guadalupe hardware. Each subfigure is a scan of $p = 1$ parameters β vs γ, brighter regions indicating values which result in a larger cut. All figures share the same absolute color scale. 153

5.7 Performance of QAOA-warmest (with BM-MC_2 warm-starts) compared to standard QAOA in an ideal simulation (dashed), noisy simulation (dotted) and on IBM Guadalupe hardware (solid). Each subplot considers a different native hardware graph with randomly selected weights as well as a different choice of initialization procedures. (Left) For a random initialization of the classically informed QAOA-warmest start rotation. (Right) For an efficiently selected, optimal choice for the classically informed QAOA-warmest start rotation. 154

5.8 QAOA-warmest performance on Quantinuum simulators and hardware. The 20-node Karloff instance considered here is directly mapped to the fully-connected Quantinuum 20-ion hardware. In contrast to Figure 5.4, here we use a GW warmest start and find that this particular initialization outperforms the GW algorithm on average for $p \leq 2$. 158

5.9 Comparison between standard QAOA mixer to using BM-MC_2 warm-starts with custom mixers . We show the noiseless (left) and noisy (right) case. In both cases, the custom mixer significantly outperforms the standard mixer. Shaded regions indicate the distribution of results for 20 randomly chosen 8 node graphs with positive and negative weights. . 158

5.10 Difference in approximation ratio between rank-n GW hyperplane rounding and hyperplane rounding of various warm-start initializations (rank-k projected GW SDP solutions (Proj$_k$-GW) and approximate BM-MC$_k$ solutions for $k = 2, 3$) as the number of nodes varies. For each instance and each rank k, we obtained 5 random projections and 5 approximate BM-MC$_k$ solutions, and then kept the best one (of 5) in regards to the BM-MC$_k$ objective. Each circle in the figure corresponds to an instance from (a subset of) the MQLib library [38]; see Appendix 5.4.5 for details. 159

5.11 For both plots, we compare the log-error of QAOA-warmest (with BM-MC$_2$ warm-starts) to the variant of QAOA proposed by Egger et al. [42] for Max-Cut. For Egger et al.'s approach, we consider two different initializations: initializing the variational parameters to the origin (left) and initializing the parameters in a way that recovers the cut used to initialize the quantum state (right). Each marker in the plot corresponds to a combination of instance (from our graph ensemble $\mathcal{G}$) and circuit depth (either $p = 1$ or $p = 8$) with the shape of the marker being used to denote if the instance has only positive edge weights or not. Points below the black line correspond to instances where QAOA-warmest performs better than the other algorithm being compared. 159

5.12 Histogram showing the difference in (instance-specific) approximation ratio (AR) when using QAOA-warmest (on instance library $\mathcal{G}$) with various warm-start strategies: projected GW and locally optimal BM-MC warm-starts. The blue and red bars correspond to depth $p = 1$ and $p = 8$ QAOA-warmest respectively with the purple regions indicating an overlap in the histograms. For both approaches, rank-3 solutions and vertex-at-top rotations are used to produce the figure; the results are similar if one instead uses a different combination of rank and/or rotation scheme. 162

SUMMARY

There is growing interest in utilizing near-term quantum technology to solve challenging problems in combinatorial optimization. The Quantum Approximate Optimization Algorithm (QAOA) is a general framework proposed by Farhi et al. [1] in 2014 for obtaining approximate solutions to certain classes of combinatorial optimization problems (namely those that can be formulated as a Quadratic Unconstrained Binary Optimization (QUBO) problem). In this thesis, we explore ways to bridge the theories of classical and quantum optimization in an effort to develop fast hybrid solvers for the Max-Cut problem.

In Chapter 1 and 2, we discuss related work and introduce some background and notation regarding both classical and quantum methods of solving combinatorial optimization problems. In Chapter 3, in an effort to find classes instances for which QAOA can potentially demonstrate quantum advantage, we identify (weighted) Max-Cut instances which are classically hard in two different senses. In the first sense, amongst a library of Max-Cut instances and heuristics from Dunning et al., we consider instances for which no classical heuristic achieves a cut value within 0.99 of the optimal cut value (within some instance-dependent time limit). Such instances have both positive and negative edges whose weights span several orders of magnitude; smaller instances with a similar edge-weight distribution were created for the purposes of numerical simulation. These numerical simulations show that QAOA also performs poorly on these instances; in general, QAOA tends to produce low-quality solutions whenever there is a (roughly) balanced mix of both positive and negative edge weights. In the second sense of being classically hard, we identify instances for which the Goemans-Williamson (GW) algorithm, the best known classical approximation algorithm, performs poorly. In 1996, Karloff [2] constructed an sequence of graphs that proves that GW's 0.878-approximation is tight; we use this construction and the result of a conjecture to find small instances that are suitable for near-term quantum devices. Although QAOA converges to Max-Cut in the adiabatic limit, we prove that its performance

at depth-1 is limited on Karloff's sequence of graphs, only yielding 0.592 of the optimal cut value in the sequence's limit. Additionally, we also show that for a family of strongly-regular graphs, the GW algorithm yields a 0.912-approximation.

Next, in Chapter 4, to enhance the solving power of low-depth QAOA, we introduce a classically-inspired "warm-start" to initialize the QAOA, using solutions to the low-rank SDP relaxation of Max-Cut and randomized projections of solutions to Max-Cut's SDP relaxation. We call this variant QAOA-warm and show that this outperforms standard QAOA on lower circuit depths in solution quality and training time. We show that the warm-starts constructed from projected SDP solutions initialize the QAOA circuit with constant-factor approximations of 0.658 and 0.585 for rank-2 and rank-3 warm-starts for graphs with non-negative edge weights, improving upon previously known trivial (i.e., 0.5 for standard initialization) worst-case bounds. While this improvement is partly due to the classical warm-start, we find strong evidence of further improvement using QAOA circuit at small depth. We provide experimental evidence of improved performance; however, we find that the performance of QAOA-warm eventually plateaus. More precisely, we show that there exists an instance for which, at any circuit depth, QAOA-warm returns half the optimal cut value in expectation, regardless of the choice of variational parameters used in the circuit.

Afterwards, in Chapter 5, we improve the QAOA-warm algorithm by modifying the QAOA circuit so that the starting state is the most excited state of the mixing Hamiltonian. We demonstrate that this version of QAOA, which we call QAOA-warmest, converges to Max-Cut under the adiabatic limit as the circuit depth increases for any choice of initial product state whose qubits are not at the poles of the Bloch sphere; when using the warm-starts proposed in this thesis, this convergence is shown to be empirically fast compared to other initialization choices. Additionally, our numerical simulations with QAOA-warmest yield higher quality cuts (compared to standard QAOA, the classical Goemans-Williamson algorithm, and a warm-started QAOA without custom mixers). We further

show that QAOA-warmest outperforms the standard QAOA of Farhi et al. in experiments on current IBM-Q and Quantinuum hardware.

Finally, in Chapter 6 we propose possible methods for handling both unconstrained and constrained optimization problems with QAOA; in particular, we discuss a novel algorithm where ancillary qubits are used to generate feasible initial quantum states (i.e. those which are a superposition of feasible solutions) whose distributions match those obtained by classical local-search algorithms. We provide partial results on the performance of this algorithm and show that when applying the standard QAOA algorithm to such a state; the ancilla qubits prevent certain types of entanglement from occurring, thus limiting QAOA's performance.

We conclude the thesis with a discussion of research questions stemming from this work in Chapter 7.

The results of this thesis have been published in ACM Transactions on Quantum Computing [3] and under review at Quantum [4].

CHAPTER 1

INTRODUCTION

In the field of combinatorial optimization, the goal is to find an optimal object amongst a given finite collection; more precisely, a general problem in combinatorial optimization takes the form of

$$\max_{s \in S} f(s),$$

where S is a finite set and $f : S \to \mathbb{R}$ is some real-valued objective function on S. Examples of combinatorial optimization problems include the Shortest Path problem [5], the Max-Cut problem [6], Hamiltonian Cycle problem [6], the Traveling Salesman problem [7], Minimum Spanning Tree [8], Maximum Weight Matching [9], and Job Shop scheduling [10]. In the case of the shortest path problem for example, given a graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}^{\geq 0}$ and two vertices $u, v \in V$, let S be the set of all paths from u to v and, for all $s \in S$, let $f(s)$ be the sum of the edge weights along the path s; in this case, $\max_{s \in S} f(s)$ is the objective. For this problem, the set of all paths is entirely determined by the graph G which can be concisely represented; such is typically the case in the field of combinatorial optimization, i.e., the size of the set S to be optimized over is typically exponential in the size of the representation. For this reason, evaluating $f(s)$ for each $s \in S$ becomes intractable as the size of the representation grows and more sophisticated methods are required that exploit the structure of S and its relation with the objective function f.

For many problems in combinatorial optimization, there exists an algorithm (such as Dijkstra's algorithm [5] for the Shortest Path problem) there exists a combinatorial algorithm that finds the optimal solution and whose runtime is polynomial in the size of the (representation of the) input; such problems with polynomial-time algorithms are said to

belong to complexity class[1] **P**. On the other hand, there exists challenging combinatorial optimization problems in the class **NP-Hard**, meaning that no such polynomial-time algorithm exists for such problems (assuming **P** $\neq$ **NP**)[2]; one such example is the Max-Cut problem [6].

Given a graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}$, the goal of the Max-Cut problem is to partition V into two parts so that the sum of the weights of the edges between the parts is maximized (Figure 1.1), i.e., we wish to determine

$$\max_{S \subseteq V} \sum_{e \in E} w_e \mathbf{1}[e \in S \times (V \setminus S)].$$

Although no polynomial-time algorithm exists to solve Max-Cut exactly (unless **P = NP**), Max-Cut is in the complexity class **APX**, meaning that there exists a polynomial-time algorithm for which a solution is produced whose value is at least some guaranteed fraction of the optimal solution value [11]. The celebrated Goemans-Williamson (GW) algorithm [11] is one such algorithm that achieves a 0.878-approximation; more precisely, the GW algorithm relaxes Max-Cut to a semidefinite program (SDP), solves the SDP in polynomial time, and then rounds the optimal SDP solution (via a technique called hyperplane rounding) to obtain a cut $(S, V \setminus S)$ whose expected cut value is within 0.878 of the optimal cut value. The GW algorithm is thought to be the best possible approximation algorithm for Max-Cut, i.e., under the Unique Games Conjecture (UGC) and assuming **P** $\neq$ **NP**, there does not exist a polynomial-time algorithm with an approximation ratio better than the 0.878 [12–14].[3]

[1] To be more precise, for any class of combinatorial optimization problem $\max_{s \in S} f(s)$ where S is determined by an instance I from a set of possible instances $\mathcal{I}$, it is the corresponding *decision problem* "Given $k \in \mathbb{R}$ and $I \in \mathcal{I}$, does there exists $s \in S$ such that $f(s) \geq k$?" that is in **P** if the answer to the decision problem can be determined in polynomial time for all choices of k and I. If the optimization problem can be solved in polynomial-time, it is clear that the same can be said about the corresponding decision problem.

[2] The complexity class **NP** is the class of all decision problems for which a proof for a "yes" answer can be *verified* in polynomial time. It is not difficult to see that **P** $\subseteq$ **NP**; however determining if **P** is a *strict* subset of **NP** or not remains a major open problem [6]. The complexity class **NP-Hard** consists of all problems P with the property that, for all decision problems $Q \in$ **NP**, Q can be reduced to P in polynomial time.

[3] If only **P** $\neq$ **NP** is assumed, it is known that there does not exist a polynomial-time algorithm for Max-Cut that achieves an approximation ratio greater than $0.941 + \epsilon$ for any $\epsilon > 0$ [15, 16].

Up to this point, the results considered thus far in this thesis have been in terms of *classical* algorithms, i.e., algorithms performed by a classical computer or Turing machine; however, there are other models of computation, such as quantum computing, that also exist. Unfortunately, it is believed that even quantum algorithms can not break past the 0.878-inapproxamibility result under UGC; this is because such an algorithm would imply that **NP** $\subseteq$ **BQP** which is widely conjectured to be false [17–19].[4] Despite this, it may still be the case that for *some* classes of Max-Cut instances, algorithms on quantum devices can outperform the classical GW algorithm (in the worst-case on such instances). Algorithms on quantum computers have already demonstrated a (nearly) exponential speedup over their classical counterparts for some problems, e.g., Shor's algorithm [23] is a quantum algorithm that factorizes integers in polynomial time which is not the case for any (known) classical algorithms; however, in order to be of any use for modern cryptographic purposes, a suitably large, fault-tolerant quantum device would need to be constructed and such devices likely will not exist in the near-term.

For problems like Max-Cut, Farhi et al. [1] proposed the Quantum Approximate Optimization Algorithm (QAOA) which is a general framework for solving certain classes of combinatorial optimization problems. The QAOA algorithm alternates between two types quantum operations, a cost unitary (corresponding to a cost Hamiltonian H_C) and a mixing unitary (corresponding to a mixing Hamiltonian H_B), with each operation being applied a total of p times each, where p is chosen by the user; p is often referred to as the *depth* of the quantum circuit[5]. The cost and mixing unitaries are parameterized by $\gamma, \beta \in \mathbb{R}^p$ which are classically optimized. QAOA is of interest to quantum researchers since it (or some variant) has the potential to realize a *quantum advantage* (over classical machines)

[4]The complexity class **BQP** consists of decision problems solvable by a quantum computer in polynomial time with an error probability of at most $1/3$ for all instances. While it is known that P $\subseteq$ BQP [20–22], the relationship between NP and BQP is still unknown; many suspect that there are problems in NP that are not in BQP and vice-versa [18, 19].

[5]For the purposes of this thesis, a quantum circuit can be thought of as a portion of a quantum algorithm that is performed entirely on the quantum device. Additionally, circuit depth is analogous to the notion of runtime in classical computing.

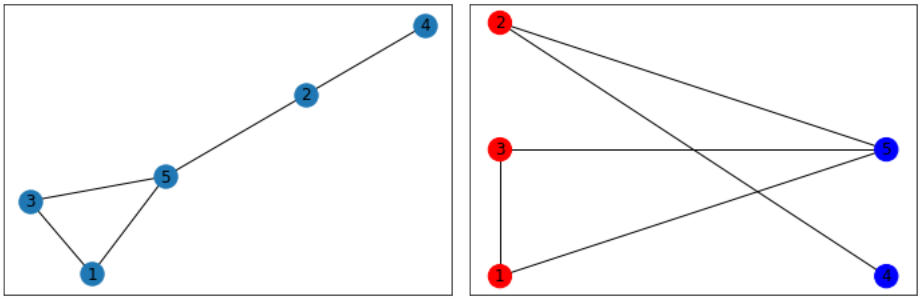

Figure 1.1: To the left, an unweighted graph $G = (V, E)$ with five vertices. To the right, the node colors correspond to the partition of V for the max-cut; the node placements are altered to illustrate that all but one edge "goes across" the cut. Since the graph is not bipartite, the cut illustrated on the right is optimal.

on current and near-term Noisy Intermediate-Scale Quantum (NISQ) [24, 25] devices. In addition to being well-known amongst classical optimizers, the Max-Cut problem has also been a problem of interest to quantum optimizers and physicists as it can be easily formulated in terms of the Ising model[6] [26]; in particular, the objective of the Max-Cut problem is easily formulated in the QAOA framework using native quantum gates/operations across various quantum devices [1].

QAOA has shown some promise; for example, in the context of the Max-Cut problem, numerical simulations have shown that, for some instances, QAOA often outperforms the Goemans-Williamson (GW) algorithm [11] for modest circuit depths [28]. On the theoretical forefront, Farhi et al. showed that depth-1 QAOA achieves a 0.6924-approximation for Max-Cut on 3-regular graphs [1]; at depth-2, Wurst and Love [29] show that the approximation ratio improves to 0.7559 for Max-Cut QAOA on 3-regular graphs. Although classically trivial to solve, Farhi et al. [1] conjectured that depth-p QAOA on an n-node even cycle achieves an approximation ratio of $\frac{2p+1}{2p+2}$ for Max-Cut whenever $n > 2p$; however, it has since been shown that the fraction $\frac{2p+1}{2p+2}$ is an upper bound on the performance of such cycles [30].

Others have also proven limitations of the standard QAOA algorithm as well: Bravyi et al. [31] show that, for all $d \geq 3$, there exists a sequence of d-regular bipartite graphs

[6]The Ising model [26] is the physicists' analog of the Quadratic Unconstrained Binary Problem (QUBO) problem studied in classical optimization. The Ising model and QUBO formulation are both used to formulate a large class of combinatorial optimization problems [27].

such that depth-p QAOA with $p < (1/3\log_2 n - 4)d^{-1}$ on such instances produces a cut (in expectation) whose value is at most $\frac{5}{6} + \frac{\sqrt{d-1}}{3d}$, meaning that, in the worst case, constant-depth QAOA for Max-Cut is inferior to the classical Goemans-Williamson algorithm as $\lim_{d\to\infty}\left(\frac{5}{6} + \frac{\sqrt{d-1}}{3d}\right) = \frac{5}{6} \approx 0.833 < 0.878$; Farhi et al. [32] show a similar result when QAOA is applied to the Max Independent Set problem[7]. Theoretically, there is still much that is not known about QAOA, especially in regards to lower bounds on its approximation ratio at higher circuit depths.

There are two key issues affecting the quality of the solutions obtained by the standard QAOA algorithm:

Issue 1: At higher circuit depths p, finding the optimal variational parameters $\gamma, \beta \in \mathbb{R}^p$ for the QAOA circuit becomes increasingly difficult.

Issue 2: At low circuit depths, QAOA fails to distinguish between certain instances due to the locality of the algorithm, i.e., no correlation is built-up between qubits that are far away from each other in the graph.

In the context of near-term NISQ technology, there is also concern regarding an increase in noise as the circuit depth of QAOA increases. Regarding the first issue, many have proposed initialization and optimization strategies for finding a good set of variational parameters [33–37]. Many have also proposed variants of QAOA that modify the QAOA circuit and the algorithm itself, some of which have the possibility of addressing the second issue. Such variants are discussed in Section 1.2; moreover, such modifications are a core aspect of this thesis as we will see momentarily.

1.1 Contributions of the Thesis

The primary goal of this thesis is to bridge the theories of both classical optimization and quantum computing in order to better understand and improve upon current quantum optimization algorithms. In this thesis, we work towards this goal by addressing two key

[7]Given a graph $G = (V, E)$, the goal of the Max Independent Set problem is to find $S \subseteq V$, with $|S|$ as large as possible, such that for all vertices $u, v \in S$, the edge $(u, v) \in E$.

questions:

Question 1: What are the classically hard instances which have a potential for demonstrating quantum advantage on quantum devices?

Question 2: Can solutions obtained by classical algorithms be exploited in order to improve quantum algorithms?

This thesis addresses the first question in Chapter 3. First, the data collected from Dunning et al.'s empirical study [38], in which they ran 38 Max-Cut heuristics on a total of 3,296 Max-Cut instances, is further analyzed; the purpose of their study was to develop a machine learning algorithm that predicts the best heuristic to use for any given Max-Cut instance. When searching for instances where no heuristic algorithm performs well, 11 such instances in Dunning et al.'s instance library were found. These 11 instances are much too large for current quantum devices; however, these instances have a similar edge-weight distribution and thus, one can produce smaller instances with a similar edge-weight distribution. On these smaller constructed instances, numerical simulations show that low-depth QAOA also produces low-quality solutions meaning that such instances are challenging for both the classical and quantum regime.

In general, the task of finding Max-Cut instances for which a quantum algorithm outperforms *every* classical heuristic is incredibly difficult. Later in Chapter 3, we relax this task to instead finding a class of instances for which some quantum algorithm outperforms the GW algorithm, the classical algorithm with the best provable approximation guarantee. Karloff [2] showed that there exists a sequence of graphs $\{G_m\}_{m=1}^{\infty}$ such that the (instance-specific) approximation ratio of Goemans-Williamson on these graphs approaches the 0.878 bound; such instances are promising in regards to the relaxed task previously described. Since the graphs in the sequence grow very quickly in size, this work also considers perturbations of the smaller instances in $\{G_m\}_{m=1}^{\infty}$; interestingly, we find that, empirically, even small changes to these instances yields a significant change in the performance of the GW algorithm. In particular, we empirically show that the GW algo-

rithm has a higher instance-specific approximation ratio on these modified graphs when compared to their unmodified counterparts. In addition to these instances found using Karloff's construction, we also prove that the GW algorithm achieves an approximation ratio of 0.912 on a particular family of strongly-regular graphs; in general, strongly-regular graphs are of interest since their vertices can be mapped to a hypersphere in a way where points corresponding to adjacent angles are equiangular, thus simplifying the analysis of the GW instance-specific approximation ratio. Lastly, we consider how these instances fare in the quantum regime; we prove (Theorem 15) that depth-1 QAOA on $\{G_m\}_{m=1}^{\infty}$ achieves an (instance-specific) approximation ratio that approaches 0.592 as $m \to \infty$, which is provably worse than the 0.878 bound of the GW algorithm.

We address Question 2 in Chapters 4 through 6 by exploiting a class of techniques in classical optimization known as *warm-starting*; such techniques initialize the combinatorial problem with a feasible, often high-quality, solution in an effort to more quickly obtain solutions of even higher quality [39–41]. In the context of quantum computing, we propose a method of mapping 2-dimensional and 3-dimensional[8] classical solutions (to the GW SDP relaxation for Max-Cut) to the quantum Bloch sphere to obtain a warm-started quantum product state. In Chapter 4, we introduce a variant of QAOA called QAOA-warm which considers applying the QAOA circuit to such warm-started quantum states. We find that, for some methods of warm-starting, this approach empirically produces better cuts (compared to standard QAOA) at extremely low circuit depths of $p = 0$ (i.e. quantum measurement of the warm-started initial quantum state) and $p = 1$. As the circuit depth increases, the standard QAOA algorithm converges to the optimal solution, in theory, due to QAOA's relationship with quantum adiabatic computing [1]; however, very high-depth quantum algorithms are not of practical interest since current and near-term NISQ devices are subject to increasing amounts of noise/error as the circuit depth grows. Unfortunately, for low-depths beyond $p > 1$, the performance of QAOA-warm plateaus; in addition to pro-

[8]Here, by k-dimensional classical solution, we mean that each vertex in the graph is associated with a unit vector in $\mathbb{R}^k$.

viding emperical evidence for this claim, we also prove that there exists an instance whose (instance-specific) approximation ratio with QAOA-warm is $1/2$, regardless of the choice of circuit depth or variational parameters (Section 4.3). The plateau in performance is due to the fact that the initial state of the QAOA algorithm was changed without making any modifications to the remainder of the QAOA circuit; thus, the relationship between QAOA and quantum adiabatic computing no longer holds.

In Chapter 5, we improve the QAOA-warm algorithm by making additional modifications to the QAOA circuit, more specifically, we alter the mixing Hamiltonians so that they incorporate the structure of the warm-start in a way that reestablishes the connection between QAOA and quantum adiabatic computing. We refer to this improved version of QAOA-warm as QAOA-warmest. We prove the convergence of QAOA-warmest (Section 5.2) and furthermore, we show that this convergence is empirically fast (Section 5.4) with a suitable choice of warm-start initialization. We additionally show that QAOA-warmest empirically yields better results across all circuit depths compared to standard QAOA, QAOA-warm, and a similar warm-start approach proposed by Egger et al. [42].

In both Chapters 4 and 5, there are two main initializations for QAOA-warm and QAOA-warmest that we consider. The first are initializations obtained via k-dimensional Burer-Monteiro solutions. The Burer-Monteiro relaxation is a relaxation of the Max-Cut objective where each vertex of the graph is represented by a point on a $(k-1)$-dimensional sphere. Note that this relaxation is equivalent to the Goemans-Williamson relaxation when $k = n$ (where n is the number of vertices); however, we consider the case where $k = 2, 3$ since the classical solutions are easily mapped to the quantum Bloch sphere. The relaxation is non-convex, meaning that globally optimal solutions are not necessarily easy to find; however, locally optimal solutions can be found rather quickly. The second type of initialization considers projections of solutions to the Goemans-Williamson relaxation (i.e. n unit vectors in $\mathbb{R}^n$) to two and three dimensions; we prove that such projections preserve the approximation ratio obtained by hyperplane rounding. For QAOA-warm and

QAOA-warmest, we prove guarantees for both initializations in the case of graphs with non-negative edge weights, in particular, at depth $p = 0$ (i.e. quantum measurement of the initial quantum state) locally optimal solutions to the Burer-Monteiro relaxations yield a $\frac{3}{4}\kappa$ and $\frac{2}{3}\kappa$ approximation for two-dimensional and three-dimensional solutions respectively, where κ is the ratio of the (Burer-Monteiro) objective value of the solution to the optimal cut in the graph (Theorem 16); by applying the results of Mei et al., which bound the performance of locally optimal solutions to Burer-Monteiro relaxations, this corresponds to approximation ratios of 0 and $1/3$ respectively (for 2 and 3-dimensional solutions) when using QAOA-warm or QAOA-warmest. However, these low approximation ratios have not been proven to be tight, i.e., it may be the case that a more careful analysis demonstrates a higher (worst-case) approximation ratio for QAOA-warm with such warm-starts. Meanwhile, for the projected GW solutions, when using a suitable projection (which we show can be easily found with high probability), we prove that depth-0 QAOA-warm and QAOA-warmest achieve an approximation ratio of $\frac{3}{4} \cdot 0.878 \approx 0.658$ and $\frac{2}{3} \cdot 0.878 \approx 0.585$ respectively (for 2 and 3-dimensional solutions) (Corollary 20); as locally optimal solutions to low-rank Burer-Monteiro relaxations are typically faster to obtain compared to such projected optimal GW solutions, we obtain a trade-off between time and theoretical solution quality when choosing between these two warm-start methods. Although these results are at depth $p = 0$, note that the performance of QAOA (and the variants we consider) monotonically increase with circuit depth (assuming a suitable choice of variational parameters are chosen).

Finally, in Chapter 6, we consider the notion of warm-starts from one more perspective; unlike Chapter 4 and 5, this method can directly be used for both unconstrained and constrained optimization problems. For many combinatorial problems, randomized local-search algorithms have been developed that utilize some notion of a local move to obtain improved solutions. We provide a general framework for constructing initial quantum states that have the same probability distribution of solutions as these randomized classical local-

search algorithms (Theorem 15); this method exploits the use of ancilla qubits to perform operations that would be considered non-unitary from the perspective of the sub-system of non-ancilla qubits. The randomized classical local-search procedure we consider is parameterized by a randomization parameter α; we show partial results regarding the performance of the (classical and quantum) local search algorithm as function of α. Lastly, we discuss the potential use of such quantum states (whose distributions match classical local-search algorithms) as a warm-start for QAOA-like algorithms; we find that standard QAOA (and many variants) have no effect on the ancilla qubits, thus causing a lack-of-interference effect between bitstrings with different ancilla tags throughout the QAOA circuit. However, Bärtschi and Eidenbenz [43] had proposed a variant of QAOA that uses a Grover-inspired mixer; such a mixer is of potential use in the context of our ancilla approach since the Grover-inspired mixer maintains feasibility of solutions for constrained optimization problems (assuming the state its applied to is a superposition of feasible solutions) while simultaneously having the ability (unlike standard QAOA) to non-trivially force interactions between bitstrings with differing ancilla tags.

1.2 Related Work

Since Farhi et al.'s [1] seminal paper in 2014, many others have researched QAOA from a theoretical perspective [17, 31, 32, 44–49] while others have took more experimental directions [33–37, 50–56]. Many have also considered modifications of the QAOA algorithm itself which we discuss more below.

For the Max-Cut problem, Egger et al. [42] construct the initial quantum state by (non-trivially) mapping a single specific cut $(S, V \setminus S)$ (obtained via the GW algorithm or possibly other means) to an initial quantum state. Egger et al. also modify the mixing Hamiltonian so that at depth $p = 1$ (with the right choice of QAOA variational parameters), the cut $(S, V \setminus S)$ is recovered; however, there is no evidence to suggest that such an approach will converge to the optimal solution for depth $p \to \infty$.

Egger et al. [42] also consider a different warm-started approach (which they refer to as continuous warm-started QAOA) for Quadratic Unconstrained Binary Optimization (QUBO) problems. For certain classes of QUBO's, the binary variables can be relaxed to be in the interval $[0, 1]$ to obtain a convex quadratic program whose optimal solution can then be mapped to the quantum Bloch sphere. The mixer used in our QAOA-warmest approach is a generalization of the mixer used in this particular approach by Egger et al.; however, the initialization scheme is quite different. In particular, their continuous warm-started QAOA approach cannot be directly applied to Max-Cut as the corresponding relaxed quadratic program is not convex; however, we do consider local optima of such relaxations in Section 4.2.3.

Recent work by Cain et al. [57] further explored convergence properties of warm-starts, when augmented with standard mixers. They showed using a perturbative approach that QAOA with single-cut warm-starts (i.e., each qubit is initialized at $|0\rangle$ or $|1\rangle$) does not show any improvement in the approximation ratio, even when the circuit depth is increased. This result is interesting when compared to our QAOA-warmest approach, since we find that (1) using custom mixers, one can guarantee convergence as long as the initialization is not at a single-cut, and (2) warm-starts that are close to a single-cut initilazation converge very slowly even with custom mixers.

Additionally, there have been other works which have explored other modifications of the QAOA algorithm. Farhi et al. [45] consider having separate variational parameters for each vertex and edge. Hadfield et al. [58] and Wang et al. [59] consider versions of QAOA that are suitable for combinatorial optimization problems with both hard constraints (that must be satisfied) and soft constraints (for which we want to minimize violations). Zhu et al. [60] modify QAOA such that the ansatz is expanded in an iterative fashion with the mixing Hamiltonian being allowed to change between iterations. Bravyi et al. [31] proposed a recursive QAOA approach that decreases the instance size at each iteration; Egger et al. [42] also consider a similar recursive version of their approaches. Bärtschi

et al. [61] and Jiang et al. [62] consider modifications of QAOA inspired by Grover's (quantum) algorithm [63] for fast database search. For the scope of this work, we do not consider these alternate approaches to modify the QAOA algorithm; however, it may serve as an interesting direction for future work the approaches discussed in this thesis can likely be used in conjunction with some of these other approaches.

CHAPTER 2
BACKGROUND AND NOTATION

In this chapter, we introduce notation that will be useful in presenting the results in this thesis. We also review some background on quantum computing and combinatorial optimization. This is by no means an exhaustive review, and we refer the reader to [64–66] for further reading. All of the results in this chapter have been previously known; however, we do provide some proofs, calculations, and explanations for the reader's convenience.

2.1 General Quantum Computing Background

In order to make this thesis understandable to a wide audience, this section will give a brief introduction to the basics of quantum computing.

2.1.1 Classical Perspective

Before diving into the aspects of what makes quantum computing truly "quantum", we first consider things from a classical perspective whilst introducing the notation and terminology used in quantum computing. As we will see in the rest of this section, quantum devices and algorithms are probabilistic in nature, so we begin by first introducing quantum computing in the context of classical probability.

Single Fair Coin

We begin with the simplest example in (classical) probability: a single fair coin. At any point in time, if we *observe* the coin, it will be in one of two states: heads or tails which we will represent with the notation $|H\rangle$ and $|T\rangle$ respectively.

Much of quantum computing involves casting these notations of probability into the language of linear algebra; for this reason, we will make a correspondence between these

states and the standard basis vectors of $\mathbb{R}^2$:

$$|H\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |T\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

We take a moment to describe the notation $|\cdot\rangle$ which comes from the *bra-ket* notation often used by physicists. Anything of the form $|\cdot\rangle$ is referred to as a *ket* and, for the purposes of this thesis, can be thought to represent the state of the system being considered; alternatively,one can mathematically interpret kets as just something that denotes a column vector. Similarly, anything of the form $\langle\cdot|$ is called a *bra* and can be mathematically interpreted to mean denoting a row vector. Note that if a ket (i.e. column vector) $|v\rangle$ already exists, then the notation $\langle v|$ should be interpreted as meaning the corresponding bra (i.e. row vector) $\langle v| = (|v\rangle)^\dagger$ where $(\cdot)^\dagger$ denotes the conjugate transpose.[1]

Now, let us imagine that someone tosses the coin and covers the coin with their hand as it lands. If we were to then observe the coin, we know we would see heads and tails both with a 50% probability. Before the coin is revealed, we can imagine that the coin is effectively in a *superposition* of both heads $|H\rangle$ and tails $|T\rangle$ with equal probability; we thus write the state of the coin, $|C\rangle$ as follows:

$$\begin{aligned} |C\rangle &= \sqrt{1/2}\,|H\rangle + \sqrt{1/2}\,|T\rangle \\ &= \sqrt{1/2}\begin{bmatrix} 1 \\ 0 \end{bmatrix} + \sqrt{1/2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \sqrt{1/2} \\ \sqrt{1/2} \end{bmatrix}. \end{aligned}$$

Note that the *squares* of the coefficients of the kets correspond to the probabilities of the outcomes after the coin is revealed. Similarly, in the linear algebra framework, we

[1]The notations $(\cdot)^*$ and $(\cdot)^H$ are also commonly seen to denote the conjugate transpose.

see that when the state is written as a vector, that the squares of the entries correspond to the probabilities of observing the classical states $|H\rangle$ and $|T\rangle$. In classical probability, probability vectors are typically written so that they are unit-vectors with respect to the 1-norm, meanwhile, in quantum computing, we enforce that states are represented by unit-vectors with respect to the 2-norm; later in Section 2.1.4, we will discuss the importance of representing states in this way.

We now generalize and consider a biased coin such that when it is tossed, it has a probability $p \geq 0$ of landing on heads and probability $q \geq 0$ of landing on tails with $p + q = 1$. Like we previously saw, before the coin is revealed, we can imagine that the state of coin, $|C\rangle$ is in a superposition of both heads $|H\rangle$ and tails $|T\rangle$ where the squares of the coefficients/entries correspond to the probabilities of seeing the corresponding state upon revealing the coin, i.e.,

$$\begin{aligned} |C\rangle &= \sqrt{p}\,|H\rangle + \sqrt{q}\,|T\rangle \\ &= \sqrt{p}\begin{bmatrix}1\\0\end{bmatrix} + \sqrt{q}\begin{bmatrix}0\\1\end{bmatrix} \\ &= \begin{bmatrix}\sqrt{p}\\\sqrt{q}\end{bmatrix}. \end{aligned}$$

It may be the case that one is given a biased coin but does not know the exact values of p and q. However, one can estimate both quantities by repeatedly doing the coin tossing experiment and making repeated observations. In the context of quantum computing, these observations are called *measurements*. When doing quantum computing with actual quantum devices, we typically do not have direct access to the coefficients of the classical outcomes (which physicists refer to as *amplitudes*) throughout the computation; thus, such measurements are a key aspect of quantum computing.

Now, consider two biased coins, C_1 and C_2. For $j = 1, 2$, suppose the coin C_j is biased

such that it lands on heads and tails with probability p_j and q_j respectively. For each of the two coins, we can write the state of the coin, before it is revealed after a toss as:

$$|C_1\rangle = \sqrt{p_1}\,|H\rangle + \sqrt{q_1}\,|T\rangle = \begin{bmatrix} \sqrt{p_1} \\ \sqrt{q_1} \end{bmatrix},$$

$$|C_2\rangle = \sqrt{p_2}\,|H\rangle + \sqrt{q_2}\,|T\rangle = \begin{bmatrix} \sqrt{p_2} \\ \sqrt{q_2} \end{bmatrix}.$$

In the two-coin experiment, there are four possible outcomes, which we will formally write as $|H\rangle \otimes |H\rangle$, $|H\rangle \otimes |T\rangle$, $|T\rangle \otimes |H\rangle$, and $|T\rangle \otimes |T\rangle$; where, semantically, $|a\rangle \otimes |b\rangle$ means that $|a\rangle$ is the outcome of the first flip and $|b\rangle$ is the outcome of the second flip. Here, $\otimes$, is the Kronecker product (also sometimes called a tensor product). In the context of linear algebra, it is an operator on matrices which is defined as the following block matrix, given matrices A and B:

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn}B \end{bmatrix},$$

where $a_{ij} \in \mathbb{C}$ is the entry in the ith row and jth column of A. Using this definition and recalling that $|H\rangle = [1, 0]^\top$ and $|T\rangle = [0, 1]^\top$, we have that,

$$|H\rangle \otimes |T\rangle = \begin{bmatrix} 1\,|T\rangle \\ 0\,|T\rangle \end{bmatrix} = \begin{bmatrix} 1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 0 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}.$$

Similarly, we have that the other three outcomes produce the other standard basis vec-

tors of $\mathbb{R}^4$:

$$|H\rangle \otimes |H\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, |T\rangle \otimes |H\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, |T\rangle \otimes |T\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

For convenience, for these four classical outcomes, we will often drop the Kronecker product symbol and write everything in a single ket, e.g., $|HT\rangle = |H\rangle \otimes |T\rangle$.

We can also use the Kronecker product to describe the probabilities of the four classical outcomes of the two-coin experiment. Letting $|C_{1,2}\rangle$ to represent the state of the two-coin system, we have:

$$|C_{1,2}\rangle = |C_1\rangle \otimes |C_2\rangle = \begin{bmatrix} \sqrt{p_1} \\ \sqrt{q_1} \end{bmatrix} \otimes \begin{bmatrix} \sqrt{p_2} \\ \sqrt{q_2} \end{bmatrix} = \begin{bmatrix} \sqrt{p_1} \begin{bmatrix} \sqrt{p_2} \\ \sqrt{q_2} \end{bmatrix} \\ \sqrt{q_1} \begin{bmatrix} \sqrt{p_2} \\ \sqrt{q_2} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \sqrt{p_1 p_2} \\ \sqrt{p_1 q_2} \\ \sqrt{p_2 q_1} \\ \sqrt{p_2 q_2} \end{bmatrix}$$

$$= \sqrt{p_1 p_2}\,|HH\rangle + \sqrt{p_1 q_2}\,|HT\rangle + \sqrt{p_2 q_1}\,|TH\rangle + \sqrt{p_2 q_2}\,|TT\rangle\,.$$

As observed previously, the squares of the coefficients of the classical outcomes correspond to the probabilities of those outcomes occurring (recall from classical probability that if two events are independent, then the probability of them both occurring are the product of the individual probabilities). In general, if $|\psi_1\rangle$ and $|\psi_2\rangle$ represent the states of two independent subsystems, then the state of the combined system is described by $|\psi_1\rangle \otimes |\psi_2\rangle$. We discuss the interpretation of the Kronecker product on general matrices in a later section. When a state like $|C_{1,2}\rangle$ can be non-trivially decomposed as a Kronecker product of two other states[2], we say that such a state is *separable* or *factorizable*. Any state that is not

[2]More precisely, if $|\psi\rangle \in \mathbb{C}^n$ and if there exists $|\psi_1\rangle \in \mathbb{C}^{n_1}$ and $|\psi_2\rangle \in \mathbb{C}^{n_2}$ with $n_1, n_2 > 1$ and $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$, then we say $|\psi\rangle$ is separable. We will later see the role of complex numbers in quantum computing. Since it is not necessary for the understanding of this work, and to keep this background section accessible for most readers, we will not consider states in $\mathbb{C}^n$ where n is not a power of 2.

Figure 2.1: Two coins, C_1 (red) and C_2 (blue) connected by a piece of glue so that they are always either both heads-side-up or tails-side-up.

separable is called an *entangled* state which we given an example of next. If, up to a global phase (see Section 2.1.3), an n-qubit state $|\psi\rangle$ can be written as $|\psi\rangle = |\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle$ where each $|\psi_j\rangle$ (for $j \in [n]$) corresponds to a single-qubit state, then we say that $|\psi\rangle$ is a *product* state; by this definition, every product state is a separable but the converse is not necessarily true.

We now consider the scenario where there are two (unbiased) coins, but the outcome of each coin is not independent. Suppose we were to glue the two coins together as shown in Figure 2.1. In this case, if this system of two coins were to be tossed, we could describe the state of the system as:

$$|C_{\text{glued}}\rangle = \sqrt{1/2}\,|HH\rangle + \sqrt{1/2}\,|TT\rangle = \begin{bmatrix} 1/\sqrt{2} \\ 0 \\ 0 \\ 1/\sqrt{2} \end{bmatrix},$$

i.e., there is a 50% chance of observing both coins being heads and a 50% chance of observing both coins being tails. Since the coins are glued together, the observations of the two coins are dependent, and thus, we say that the state $|C_{\text{glued}}\rangle$ is an entangled state; it is impossible to non-trivially write $|C_{\text{glued}}\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$ for some $|\psi_1\rangle, |\psi_2\rangle$. As a consequence, it is impossible to find two weighted coins such that flipping them independently yields a state with the same amplitudes/coefficients as those found in $|C_{\text{glued}}\rangle$.

Another consequence of entanglement is that partial observations/measurements of one

coin can tell us something about the other coin. In the case of $|C_{\text{glued}}\rangle$, before any coin is revealed, there is a 50% chance of the 2nd coin being heads. However, if we observe just the first coin and see that it is heads, we now know with 100% certainty that the 2nd coin must also be heads.

2.1.2 General Qubits

Moving beyond the classical perspective, we now generalize a bit of information (like we saw with $|H\rangle$ and $|T\rangle$) to a quantum bit or *qubit*. Instead of $|H\rangle$ and $|T\rangle$, we now use $|0\rangle$ and $|1\rangle$ to represent the classical outcomes (and $[1,0]^\top$ and $[0,1]^\top$) respectively. The general qubit $|\psi\rangle$ can be written as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{bmatrix}\alpha\\ \beta\end{bmatrix},$$

where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$. In the context of quantum computing, an observation or measurement[3] of $|\psi\rangle$ will yield an outcome of $|0\rangle$ with probability $|\alpha|^2$ and an outcome of $|1\rangle$ with probability $|\beta|^2$. Unlike the classical setting, we now allow arbitrary complex coefficients in front of the classical outcomes; moreover, we square the magnitude of the coefficient.

Consider the following two states:

$$|\psi_1\rangle = \sqrt{1/2}\,|0\rangle + \sqrt{1/2}\,|1\rangle\,,$$

$$|\psi_2\rangle = i\sqrt{1/2}\,|0\rangle - \sqrt{1/2}\,|1\rangle\,.$$

For both states, an observation/measurement would provably yield both $|0\rangle$ and $|1\rangle$

[3]Throughout this thesis, we only consider measurements in the computational basis, meaning that a measurement of a single-qubit state yields either $|0\rangle$ or $|1\rangle$. One can generalize the notion of measurement by measuring with respect to a different basis, although not considered in this thesis, the details of such generalizations can be found in [64].

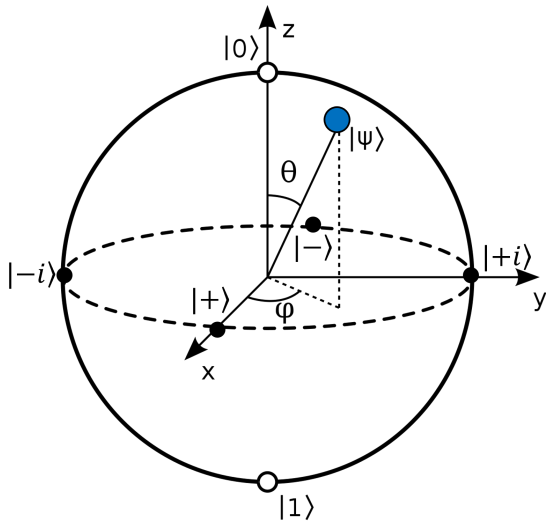

Figure 2.2: A single-qubit state $|\psi\rangle$ represented on the surface of the Bloch sphere.

with a 50% chance each; however, as we will see later, these states behave differently in the context of quantum operations.

Now, let us consider a general system of n qubits. The classical outcomes that we can observe/measure will be elements of $\{0,1\}^n$, i.e., bitstrings of length n. An arbitrary n-qubit state $|\psi\rangle$ is then given as

$$|\psi\rangle = \sum_{b\in\{0,1\}^n} \alpha_b |b\rangle = \begin{bmatrix} \alpha_{0\cdots0} \\ \vdots \\ \alpha_{1\cdots1} \end{bmatrix},$$

where $\alpha_b \in \mathbb{C}$ for all $b \in \{0,1\}^n$ and $\sum_{b\in\{0,1\}^n} |\alpha_b|^2 = 1$. Like before, this representation corresponds to a state where, for all bitstrings $b \in \{0,1\}^n$, there is a probability of $|\alpha_b|^2$ of observing b. Note that as a vector, $|\psi\rangle$ has 2^n components. Using the ket notation, a state like

$$|\psi\rangle = \sqrt{1/4}\,|01010\rangle + \sqrt{1/4}\,|11010\rangle + \sqrt{1/2}\,|10001\rangle\,,$$

is compactly written, whereas the vector representation would require us to write out $2^5 = 32$ entries (most of which would be zero).

2.1.3 Geometric Interpretation: Bloch Sphere

Consider again a single-qubit state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$. We consider how such a state could be viewed geometrically. The state could be completely described by four real numbers: $\Re(\alpha), \Im(\alpha), \Re(\beta), \Im(\beta)$, i.e., the real and imaginary parts of α and β; however, visualizing things in four dimensions is non-trivial.

Instead, physicists often view a qubit as lying along the surface of a sphere; such a sphere is called the *Bloch* sphere and is illustrated in Figure 2.2 . Using the notation for spherical coordinates[4] usually used by physicists, any point on the surface of the sphere can be described by the polar angle θ away from the top of the sphere and the azimuthal angle φ between the x-axis and the projection of the point to the xy-plane. Given a point on the surface of the sphere with polar angle θ and azimuthal angle φ, the corresponding single-qubit quantum state is given by:

$$|\psi\rangle = \cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)\,|1\rangle\,. \tag{2.1}$$

We make a few observations regarding this Bloch sphere representation. Firstly, from Equation 2.1, observe that the probability of observing/measuring either $|0\rangle$ or $|1\rangle$ is entirely determined by the polar angle θ. In general, the closer a point is to the north pole of the Bloch sphere; the more likely we are to observe $|0\rangle$; in particular, the point at the top of the Bloch sphere corresponds to the state $|\psi\rangle = |0\rangle$. Similarly, the closer a point is to the south pole of the Bloch sphere; the more likely we are to observe $|1\rangle$; in particular, the point at the bottom of the Bloch sphere corresponds to the state $|\psi\rangle = |1\rangle$. Currently, the azimuthal angle φ, which is sometimes referred to as the *phase* of $|\varphi\rangle$, has no impact on the measurement; however, this angle will have importance in the context of quantum operations which we will discuss later.

It is important to not confuse the Bloch sphere representation with the vector repre-

[4]More precisely, given the polar angle θ and azimuthal angle φ, the corresponding point in cartesian coordinates is given by $(x, y, z) = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\varphi)$.

sentation seen before. In particular, as vectors, observe that the states $|0\rangle = [1,0]^\top$ and $|1\rangle = [0,1]^\top$ are perpendicular to one another in the plane; however, on the surface of the Bloch sphere, $|0\rangle$ and $|1\rangle$ are instead antipodal.

Note that the mapping provided by Equation 2.1 results in the amplitude of $|0\rangle$ being a real-number. Thus, it appears that the Bloch sphere does not represent all single-qubit states; however, one can show that it *effectively* does, in a sense. To see this, we define an equivalence relation $\sim$ on single-qubit states where $|\psi_1\rangle \sim |\psi_2\rangle$ if there exists an angle φ such that $|\psi_1\rangle = e^{i\varphi}|\psi_2\rangle$. This equivalence relation is motivated as follows: one can show that quantum computing can not distinguish between $|\psi_1\rangle$ and $|\psi_2\rangle$, i.e., no matter what sequence of quantum operations (see Section 2.1.4) are performed on both states; the resulting probability distribution of observing $|0\rangle$ and $|1\rangle$ will be the same. Thus, for any state $|\psi_1\rangle$, there exists an equivalent state $|\psi_2\rangle$ and point x on the Bloch sphere such that Equation 2.1 maps x to $|\psi_2\rangle$.

In addition to $|0\rangle$ and $|1\rangle$, we also have the following well-known quantum states which correspond to various points along the equator of the Bloch sphere as seen in Figure 2.2 :

$$|+\rangle = \sqrt{1/2}\,|0\rangle + \sqrt{1/2}\,|1\rangle\,,$$

$$|-\rangle = \sqrt{1/2}\,|0\rangle - \sqrt{1/2}\,|1\rangle\,,$$

$$|+i\rangle = \sqrt{1/2}\,|0\rangle + i\sqrt{1/2}\,|1\rangle\,,$$

$$|-i\rangle = \sqrt{1/2}\,|0\rangle - i\sqrt{1/2}\,|1\rangle\,.$$

2.1.4 Quantum Operations

Next, we discuss quantum operations in quantum computing. These operations take as input, a quantum state, and output another quantum state. Thus, one can think of quantum operations as a method for moving between quantum states (which in turn can be thought of as moving between different probability distributions for the possible classical outcomes).

In quantum computing, quantum operations on quantum states are always linear and thus, since quantum states are represented by (finite) vectors, one can always represent a quantum operation as a matrix. Note that not any linear operator corresponds to a quantum operation; in particular, we need to ensure that the output of the quantum operation is a valid quantum state; we investigate this condition next.

Suppose there is a quantum operation (represented by the matrix) U such that $U|\psi_1\rangle = |\psi_2\rangle$ and $|\psi_1\rangle = \sum_{\{0,1\}^n} \alpha_b |b\rangle$ and $|\psi_2\rangle = \sum_{\{0,1\}^n} \beta_b |b\rangle$ with $\alpha_b, \beta_b \in \mathbb{C}$ for all $b \in \{0,1\}^n$ and $\sum_{\{0,1\}^n} |\alpha_b|^2 = \sum_{\{0,1\}^n} |\beta_b|^2 = 1$. Observe that $\{|0\cdots 0\rangle, \ldots, |1\cdots 1\rangle\}$ corresponds to the set of standard basis vectors in $\mathbb{C}^{2^n}$ and thus, the condition $\sum_{\{0,1\}^n} |\alpha_b|^2 = 1$ is equivalent to $\| |\psi_1\rangle \| = 1$; similarly the condition $\sum_{\{0,1\}^n} |\beta_b|^2 = 1$ is equivalent to $\| |\psi_2\rangle \| = 1$. This means that U must be a linear norm-preserving[5] operator, or equivalently, that U is a unitary matrix. This shows that any quantum operation must be a unitary matrix and vice-versa.

We now consider specific quantum operations. Note that applying an quantum operation corresponding to some arbitrary unitary U is not always feasible; instead, in actual quantum devices, quantum operations are constructed by applying a sequence of quantum *gates*; similar to an instruction set in low-level classical computing, these gates serve as the building blocks for constructing quantum algorithms. There are several different types of quantum devices, some of which have a different "native" set quantum gates. In what follows, we demonstrate some common quantum gates/operations, many of which will be utilized in the main body of this thesis.

We begin with the Pauli gates:

$$\sigma^x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \sigma^y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \sigma^z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Many authors also use X, Y, Z to denote the Pauli gates $\sigma^x, \sigma^y, \sigma^z$ respectively.

[5]More specifically, we have shown that if $\|x\|_2 = 1$, then $\|Ux\|_2 = 1$; however, one can easily show this implies that U is norm preserving in general.

Observe that,

$$\sigma^x |0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle ,$$

$$\sigma^x |1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle ,$$

in other words, σ^x acts as a negation or a bit-flip operator when viewed classically. Geometrically, this acts as a rotation by π around the x-axis of the Bloch sphere; a similar geometric interpretation exists for σ^y and σ^z.

We can also consider rotations by arbitrary amounts as well, in particular, we consider the quantum operations $R_x(\theta), R_y(\theta), R_z(\theta)$ corresponding to rotations by angle θ about the x, y, and z axes of the Bloch sphere respectively. The formulas for these are given below:

$$\begin{aligned} R_x(\theta) &= \cos(\theta/2)I - i\sin(\theta/2)\sigma^x = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}, \\ R_y(\theta) &= \cos(\theta/2)I - i\sin(\theta/2)\sigma^y = \begin{bmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{bmatrix}, \\ R_z(\theta) &= \cos(\theta/2)I - i\sin(\theta/2)\sigma^z = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}. \end{aligned}$$

As an example, if we rotate the point corresponding to $|+\rangle$ about the z-axis by $\pi/2$, we should expect the result to be the state $|+i\rangle$. Observe,

$$R_z(\pi/2)|+\rangle = \begin{bmatrix} e^{-i\pi/4} & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \begin{bmatrix} \sqrt{1/2} \\ \sqrt{1/2} \end{bmatrix} = \begin{bmatrix} e^{-i\pi/4} & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \begin{bmatrix} \sqrt{1/2} \\ \sqrt{1/2} \end{bmatrix} = \begin{bmatrix} e^{-i\pi/4}\sqrt{1/2} \\ e^{i\pi/4}\sqrt{1/2} \end{bmatrix}$$

$$= e^{-i\pi/4} \begin{bmatrix} \sqrt{1/2} \\ e^{i\pi/2}\sqrt{1/2} \end{bmatrix} = e^{-i\pi/4} \begin{bmatrix} \sqrt{1/2} \\ i\sqrt{1/2} \end{bmatrix} = e^{-i\pi/4}|+i\rangle ,$$

which is equivalent (under the equivalence relation $\sim$ from before) to $|+i\rangle$.

As another example, consider the Hadamard gate,

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

The operation H corresponds to a rotation by π along the axis determined by a 45 degree angle between the x and z axes. In particular, similar calculations show that $H|0\rangle = |+\rangle$ and $H|+\rangle = |0\rangle$ and that $H|1\rangle = |-\rangle$ and $H|-\rangle = |1\rangle$. In particular, note that measurement of $|+\rangle$ and $|-\rangle$ yield the same result, but if a Hadamard gate is applied to each of these gates, then measurement of such states yields differing results.

2.1.5 Properties of the Kronecker Product

Next, we consider the role that Kronecker products play in the context of multi-qubit systems. Let $|\psi_1\rangle$ and $|\psi_2\rangle$ be the state of two independent subsystems; from before, we have that $|\psi_1\rangle \otimes |\psi_2\rangle$ describe the state of the combined system. Suppose that there exists a quantum operations U_1 and U_2 for each of the two subsystems respectively; we can represent the application of both of those operations (independently to each subsystem) in the context of the overall system with the operation $U_1 \otimes U_2$, i.e.,

$$(U_1 \otimes U_2)(|\psi_1\rangle \otimes |\psi_2\rangle) = (U_1|\psi_1\rangle) \otimes (U_2|\psi_2\rangle).$$

The above is a consequence of the mixed-product property of Kronecker products as stated in the proposition below.

Proposition 1. *For matrices A, B, C, D of the appropriate size ($A \in \mathbb{C}^{m\times n}, B \in \mathbb{C}^{p\times q}, C \in \mathbb{C}^{n\times k}, D \in \mathbb{C}^{q\times r}$),*

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD).$$

Applying the mixed-product property with identity matrices of the appropriate size, we

have that, for matrices A and B that

$$(A \otimes I)(I \otimes B) = (AI) \otimes (IB) = A \otimes B,$$

and

$$(I \otimes B)(A \otimes I) = (IA) \otimes (BI) = A \otimes B;$$

in the context of quantum computing, the operation $A \otimes B$ can be interpreted as a two-step procedure where we first apply A to the first subsystem and then apply B to the second subsystem. Alternatively, we could apply B to the second subsystem first and then apply A to the first subsystem since the above shows that the order of these two steps does not matter. However, the Kronecker product itself is not necessarily commutative, i.e., typically $A \otimes B \neq B \otimes A$.

In addition to the mixed-product property, the Kronecker product has additional properties of interest, many of which are similar to properties of the standard matrix product.

Proposition 2. *Let $A \in \mathbb{C}^{m_1 \times n_1}, B \in \mathbb{C}^{m_2 \times n_2}, C \in \mathbb{C}^{m_2 \times n_2}$ be matrices and let $k \in \mathbb{C}$. Let $\mathbf{0}$ be an all-zeroes matrix of any fixed size, then,*

1. $A \otimes (B + C) = A \otimes B + A \otimes C$,
2. $(B + C) \otimes A = B \otimes A + C \otimes A$,
3. $(kA) \otimes B = A \otimes (kB) = k(A \otimes B)$,
4. $(A \otimes B) \otimes C = A \otimes (B \otimes C)$,
5. $A \otimes \mathbf{0} = \mathbf{0} \otimes A = \mathbf{0}$.
6. $I_m \otimes I_n = I_{mn}$, *where I_m and I_n are the $m \times m$ and $n \times n$ identity matrices respectively,*
7. $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$ *if A and B are both invertible,*

8. $(A \otimes B)^T = A^T \otimes B^T$

9. $(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger$

Proof. We refer the reader to [67] for detailed proofs of each of the properties above. □

By induction, properties 1 and 2 can be extended for a sum of an arbitrary number of matrices; similarly, properties 6 through 9 can be extended for a Kronecker product of an arbitrary number of matrices. We also note that the complex field $\mathbb{C}$ in the above proposition can be replaced with any arbitrary field $\mathbb{F}$ and the results will still hold. For the Kronecker product of n matrices $A_1, \ldots, A_n$, we define,

$$\bigotimes_{j=1}^{n} A_j = A_1 \otimes A_2 \otimes \cdots \otimes A_n;$$

note that no parenthesis are needed since the Kronecker product is associative by Proposition 2.

In the context of multi-qubit systems, we often want to apply a single-qubit gate to just one qubit. For example, for an n-qubit system, if one wanted to apply σ^x to the first qubit, the overall operation in the context of the full n-qubit system would be represented as $\sigma^x \otimes \underbrace{I_2 \otimes \cdots \otimes I_2}_{n-1 \text{ times}}$ where I_2 is the 2×2 identity matrix. We adopt the following notation for convenience:

$$\sigma_j^x = \underbrace{I_2 \otimes \cdots \otimes I_2}_{j-1 \text{ times}} \otimes \underbrace{\sigma^x}_{j\text{th term}} \otimes \underbrace{I_2 \otimes \cdots \otimes I_2}_{n-j \text{ times}},$$

to represent applying σ^x to the jth qubit in an n-qubit system. Both σ_j^y and σ_j^z are similarly defined to represent applying σ^y and σ^z to the jth qubit respectively.

In general, the Kronecker product can be used to describe the action of several quantum operations being applied independently to several subsystems in quantum computing. In particular, if $|\psi_1\rangle, \ldots, |\psi_k\rangle$ describe states of k independent subsystems and $U_1, \ldots, U_k$ are quantum operations on those subsystems, then in the context of the overall system,

$U_1 \otimes \cdots \otimes U_k$ represents applying each of the operations independently to each subsystem, i.e.,

$$(U_1 \otimes \cdots \otimes U_k)(|\psi_1\rangle \otimes \cdots \otimes |\psi_k\rangle) = (U_1 |\psi_1\rangle) \otimes \cdots \otimes (U_k |\psi_k\rangle);$$

this property of Kronecker products is formalized in the proposition below which is a generalization of the mixed-product property (Proposition 1).

Proposition 3. *Let $m, n \geq 2$. For each $i \in [n]$ and $j \in [m]$, let $A_i^{(j)}$ be a matrix with complex entries. Then,*

$$\prod_{j=1}^{m} \bigotimes_{i=1}^{n} A_i^{(j)} = \bigotimes_{i=1}^{n} \prod_{j=1}^{m} A_i^{(j)},$$

for matrices $A_i^{(j)}$ of the appropriate size (i.e. the matrix products above are well-defined).

Proof. For convenience, let $A^{(j)} = \bigotimes_{i=1}^{n} A_i^{(j)}$. We first prove that the proposition holds when $m = 2$ in the lemma below.

Lemma 1. *Let $n \geq 2$. Let $A = \bigotimes_{i=1}^{n} A_i$ and let $B = \bigotimes_{i=1}^{n} B_i$. Then*

$$AB = \bigotimes_{i=1}^{n} (A_i B_i).$$

Proof. We proceed by induction. When $n = 2$ the above is equivalent to the original mixed-product property (Proposition 1). Observe,

$$\begin{aligned}
AB &= \left(\bigotimes_{i=1}^{n} A_i\right)\left(\bigotimes_{i=1}^{n} B_i\right) \\
&= \left(\left(\bigotimes_{i=1}^{n-1} A_i\right) \otimes A_n\right)\left(\left(\bigotimes_{i=1}^{n-1} B_i\right) \otimes B_n\right) && \text{(by associativity)} \\
&= \left(\bigotimes_{i=1}^{n-1} A_i\right)\left(\bigotimes_{i=1}^{n-1} B_i\right) \otimes (A_n B_n) && \text{(mixed-product property)} \\
&= \left(\bigotimes_{i=1}^{n-1} (A_i B_i)\right) \otimes (A_n B_n) && \text{(by induction)} \\
&= \bigotimes_{i=1}^{n} (A_i B_i).
\end{aligned}$$

□

We now proceed with induction on m. Observe,

$$\begin{aligned}
\prod_{j=1}^{m} A^{(j)} &= \left(\prod_{j=1}^{n-1} A^{(j)}\right) A^{(m)} \\
&= \left(\bigotimes_{i=1}^{n} \prod_{j=1}^{m-1} A_i^{(j)}\right) A^{(m)} && \text{(by induction)} \\
&= \left(\bigotimes_{i=1}^{n} \prod_{j=1}^{m-1} A_i^{(j)}\right) \left(\bigotimes_{i=1}^{n} A_i^{(m)}\right) && \text{(def of } A^{(m)}\text{)} \\
&= \bigotimes_{i=1}^{n} \left(\left(\prod_{j=1}^{m-1} A_i^{(j)}\right) A_i^{(m)}\right) && \text{(by Lemma 1)} \\
&= \bigotimes_{i=1}^{n} \left(\prod_{j=1}^{m} A_i^{(j)}\right),
\end{aligned}$$

which is the desired result.

□

Up to this point, we have implicitly assumed that if A and B correspond to quantum operations (i.e. they are unitary matrices), then so does $A \otimes B$. This is formalized in the proposition below; we also show that being Hermitian is also a property that is preserved.

Proposition 4. *Let A, B be matrices over $\mathbb{C}$.*

- *If A and B are Hermitian, then $A \otimes B$ is Hermitian.*
- *If A and B are unitary, then $A \otimes B$ is unitary.*

Proof. We first prove the first statement. Let A and B be Hermitian, i.e., $A^\dagger = A$ and $B^\dagger = B$. Then, by Property 9 of Proposition 2,

$$(A \otimes B)^\dagger = A^\dagger \otimes B^\dagger = A \otimes B,$$

thus proving that $A \otimes B$ is Hermitian.

We now prove the second statement. Let A and B be unitary matrices, i.e., $A^{-1} = A^\dagger$ and $B^{-1} = B^\dagger$. Then, by Properties 7 and 9 of Proposition 2, we have that,

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1} = A^\dagger \otimes B^\dagger = (A \otimes B)^\dagger,$$

thus proving that $A \otimes B$ is unitary. □

2.1.6 Multi-Qubit Operations

Up until now, we have only considered quantum operations that directly affect a single-qubit at a time. If a quantum state is a product state, then applying single-qubit operations causes the state to remain as a (unentangled) product state. In order to cause the state to become entangled, multi-qubit operations are necessary. In this subsection, we give one such example of a multi-qubit operation: the controlled-not (CNOT) gate. There are many other multi-qubit operations of interest; however, it is well-known that any quantum state can be approximated (up to arbitrary precision) by using only single-qubit gates and CNOT gates (see Chapter 16 of [64]).

The CNOT gate operates on two qubits. When the qubits correspond to two classical bits, the CNOT gate simply flips the second bit if the first bit is $|1\rangle$, otherwise, the CNOT gate does nothing. More precisely,

$$\text{CNOT}\,|00\rangle = |00\rangle\,,$$

$$\text{CNOT}\,|01\rangle = |01\rangle\,,$$

$$\text{CNOT}\,|10\rangle = |11\rangle\,,$$

$$\text{CNOT}\,|11\rangle = |10\rangle\,.$$

Recall that the states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ correspond to vectors that form a basis for $\mathbb{C}^4$, then CNOT can be represented by the following 4×4 matrix:

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

2.2 Approximation Ratio

Before delving into the relevant algorithms in quantum and classical settings, we first define the notion of approximation ratio (AR) for Max-Cut in general weighted graphs. In the QAOA literature, many have adopted the term approximation ratio to mean the performance of a single run of an algorithm on a particular instance. This ratio is typically "normalized" to lie in the interval $[0, 1]$ and is well-defined even when the Max-Cut$(G) = 0$. Given an expected cut value of the cut obtained $\mathcal{E}_{\mathcal{A},G}$ using algorithm $\mathcal{A}$ on graph G, we call such a ratio, $\alpha_{\mathcal{A},G}$, the normalized instance-specific approximation ratio and define it as

$$\alpha_{\mathcal{A},G} = \frac{\mathcal{E}_{\mathcal{A},G} - \text{ Min-Cut(G)}}{\text{Max-Cut}(G) - \text{Min-Cut(G)}}. \tag{2.2}$$

When the weights of all the edges in the graph are non-negative, then the trivial cut $(V, \varnothing)$ is a minimum cut for G with Min-Cut$(G) = 0$; however, if the graph contains negatively-weighted edges, then it may be the case that Min-Cut$(G) < 0$. For brevity, we will simply use the term instance-specific approximation ratio (or simply the empirical approximation ratio) to gauge the performance of variants of QAOA over the family of graph instances we consider $\mathcal{G}$ (see Section 4.5.1).We also note that in the classical optimization literature, the term approximation ratio is usually reserved for a theoretical lower bound on the perfor-

mance of a particular algorithm across all problem instances[6]. We will call such a bound as the theoretical worst-case approximation ratio or simply the approximation ratio when clear from context.

For both standard QAOA and our proposed variant QAOA-warm on a particular graph G, we obtain a final quantum state $|\psi\rangle$ and $\mathcal{E}_{\mathcal{A},G}$ is defined as the expected cut value obtained from measuring $|\psi\rangle$ in the computational basis. Both the Burer-Monteiro and Goemans-Williamson algorithms yield a collection of points $x = (x_1, \ldots, x_n)$ on a hypersphere and $\mathcal{E}_{\mathcal{A},G}$ is defined to be the expected value of the cut obtained from performing randomized hyperplane rounding on x.

For our simulations in Section 4.5, we calculate $\mathcal{E}_{\mathcal{A},G}$ exactly. Since we are using a full-state vector classical simulator for QAOA (instead of an actual quantum device), we can directly calculate $\mathcal{E}_{\mathcal{A},G}$ by reading off the amplitudes of the final quantum state (as opposed to simulating several quantum measurements and approximating $\mathcal{E}_{\mathcal{A},G}$ with an emperical average). Similarly, when working with the Burer-Monteiro or the GW algorithm, $\mathcal{E}_{\mathcal{A},G}$ is computed exactly by analyzing the angles between the points on the hypersphere (as opposed to actually performing hyperplane rounding and approximating $\mathcal{E}_{\mathcal{A},G}$ with an emperical average). Precise formulas for $\mathcal{E}_{\mathcal{A},G}$ are provided in Sections 2.4 and 2.3.

2.3 Classical Methods for Max-Cut

In this section we review two classical approximation algorithms for Max-Cut. Recall that given a (weighted) graph $G = (V, E)$ with weights $w : E \to \mathbb{R}$, the Max-Cut problem is to find a partitioning of the vertices into two subsets, S and $T = V \setminus S$, that maximizes the

[6]In particular, for a randomized algorithm $\mathcal{A}$ for Max-Cut theoretical computer science literature refers to an approximation ratio of $\alpha_{\mathcal{A}}$ if

$$\alpha_{\mathcal{A}} \leq \min_{G \in \mathfrak{G}} \frac{\mathbb{E}[\text{cut}(S_{\mathcal{A},G})]}{\text{Max-Cut}(G)}, \tag{2.3}$$

where $(S_{\mathcal{A},G}, V(G) \setminus S_{\mathcal{A},G})$ is the (random) cut returned by $\mathcal{A}$ (on G), the expectation is over the randomness of algorithm $\mathcal{A}$ taken as the worst-case over all positive weighted graphs $\mathcal{G}$. For example, the 0.878 bound for Goemans-Williamson is a worst-case bound obtained where the expectation is over all positive weighted graph instances.

number of cut edges, i.e.,

$$\text{Max-Cut}(G) = \max_{S \subseteq V} \text{cut}_G(S),$$

where,

$$\text{cut}_G(S) = \sum_{e \in E} w_e \cdot \mathbf{1}[e \in S \times (V \setminus S)],$$

we often omit the subscript G above if the graph is understood from context.

Instead of maximizing over subsets of V, one can rewrite the problem as maximizing over $\{-1, 1\}^{|V|}$ instead. To do this, we associate every vertex $i \in V$ with a decision variable x_i, where $x_i = +1$ indicates that vertex $i \in S$ and $x_i = -1$ indicates that $i \notin S$. Observe that for an edge $(i, j) \in E$, we have that the edge is cut if and only if $x_i \neq x_j$. Using the above fact, one can easily check that:

$$\frac{1}{4} w_{i,j} (x_i - x_j)^2 = \begin{cases} w_{ij}, & (i,j) \text{ is cut,} \\ 0, & (i,j) \text{ is not cut.} \end{cases} \tag{2.4}$$

By adding up the contribution of each edge and letting $n = |V|$, it becomes clear that one can reformulate the Max-Cut problem as the following maximization problem:

$$\begin{aligned} \max_{x \in \{\pm 1\}^n} \text{cut}(x) &= \max_{x \in \{\pm 1\}^n} \frac{1}{4} \sum_{(i,j) \in E} w_{i,j} (x_i - x_j)^2, & (2.5) \\ &= \max_{x \in \{\pm 1\}^n} \frac{1}{2} \sum_{(i,j) \in E} w_{i,j} (1 - x_i x_j), \\ &= \frac{1}{2} W + \max_{x \in \{\pm 1\}^n} \frac{1}{4} \langle -A, x x^T \rangle, & (2.6) \end{aligned}$$

where A is the adjacency matrix of G, $\langle \cdot, \cdot \rangle$ denotes the *Frobenius* product of two matrices[7],

[7] Not to be confused with the *bra-ket* notation, the *Frobenius* product of two same-sized matrices A and B, denoted by $\langle A, B \rangle$, is equal to $\text{Tr}(A^\dagger B)$ where $\text{Tr}(\cdot)$ denotes the *trace* of a matrix and $(\cdot)^\dagger$ denotes conjugate transposition.

and $W = \sum_{(i,j)\in E} w_{ij}$.

Goemans-Williamson (GW) Algorithm. In the seminal work of Goemans and Williamson [11] in 1995, the authors pioneered the use of semi-definite programs for solving combinatorial problems. Considering $Y = xx^T \succcurlyeq 0$ from equation (2.6), Max-Cut is equivalent to maximizing $\langle -A, Y\rangle$ by matrix Y from the positive semidefinite cone, subject to having a unit diagonal, in addition to being rank-1.[8] Relaxing the last constraint gives us a semidefinite program as follows:

$$\begin{aligned} \text{maximize} \quad & \langle -A, Y\rangle \\ \text{subject to} \quad & \langle Y, e_i e_i^T\rangle = 1, \quad \forall i \in [n], \\ & Y \in \mathbb{S}_+^n \,, \end{aligned} \tag{2.7}$$

where $n = |V|$ and $\mathbb{S}_+^n$ is the set of all $n \times n$ positive semidefinite matrices. The value given by the relaxation above was first considered in 1993 by Delorme and Poljak [68] in the form of an eigenvalue maximization problem with the equivalence shown shortly after by Poljak and Rendl [69] in 1995. The above relaxation is in the form of a semi-definite program and hence since it is a convex program it can be solved in polynomial time up to arbitrary precision, e.g., by using interior point methods [70].

For a Cholesky decomposition of $Y = X^T X$ (with $X \in \mathbb{R}^{n\times n}$), one can think of the solution to the above SDP as an embedding which maps vertex i to $X_{:i}$, i.e., the ith column of X. This embedding can be viewed as a maximizer of a relaxation of equation (2.5) where x_i still has unit distance from the origin, but now in $\mathbb{R}^n$, i.e., x_i lies on the $(n-1)$-sphere.[9] To map this high dimensional solution to a cut in the graph, the GW algorithm considers a random hyperplane through the origin to partition the vertices into two sets according to which side of the hyperplane they lie on; Goemans and Williamson [11] proved that

[8] We use "$A \succcurlyeq 0$" to mean that A is a positive semidefinite matrix, i.e., A is a symmetric matrix with real, nonnegative eigenvalues.

[9] The k-sphere, denoted S^k, is defined as $S^k = \{x \in \mathbb{R}^{k+1} : \|x\|_2 = 1\}$.

this choice of rounding yields an approximation ratio of $\alpha^* \approx 0.878$ to Max-Cut, when the edge weights are non-negative. More specifically, given a fixed GW SDP solution of $Y = X^T X$, the expected value of the cut obtained via hyperplane rounding is given by $\frac{1}{\pi}\sum_{(i,j)\in E} w_{ij}\arccos(X_{:i} \cdot X_{:j})$; in the context of instance-specific approximation ratio (Equation 2.2), we define the expected cut value (on a particular run of the GW algorithm) as the previous sum, i.e., $\mathcal{E}_{\mathcal{A},G} = \frac{1}{\pi}\sum_{(i,j)\in E} w_{ij}\arccos(X_{:i} \cdot X_{:j})$. A similar definition of $\mathcal{E}_{\mathcal{A},G}$ is also used for the Burer-Monteiro method which we describe next.

Burer-Monteiro (BM) Method. Observe that changing variables as $Y = X^T X$ (with $X \in \mathbb{R}^{n\times n}$), one can eliminate the positive semi-definite constraint in (2.7) and obtain the following equivalent reformulation:

$$\begin{aligned} \text{maximize} \quad & \langle -A, X^T X\rangle & \\ \text{subject to} \quad & \|x_i\|_2 = 1, \quad \forall i \in [n], & (2.8)\\ & x_i \in \mathbb{R}^n, \qquad \forall i \in [n]\,, & (2.9) \end{aligned}$$

where x_i denotes the ith column of X. Burer and Monteiro [71] proposed relaxing x_i for each vertex to $\mathbb{R}^k$ instead of $\mathbb{R}^n$ in (2.9), i.e., use $x_i \in \mathbb{R}^k$. Unlike the relaxation used in the Goemans-Williamson relaxation, this modification yields a non-convex optimization problem. We refer to this modification as the rank-k Burer-Monteiro Max-Cut (BM-MC$_k$) relaxation. For ease of notation, given a (feasible) BM-MC$_k$ solution x, we let

$$\text{HP}(x) = \sum_{(i,j)\in E} \frac{w_{i,j}}{\pi}\arccos(x_i \cdot x_j),$$

denote the expected cut value obtained from performing hyperplane rounding on x [11] and we let,

$$\text{BM-MC}_k(x) = \sum_{(i,j)\in E} \frac{w_{i,j}}{4}\|x_i - x_j\|^2, \tag{2.10}$$

denote the BM-MC$_k$ objective at x; for a given graph G, we let

$$\text{BM-MC}_k(G) = \max_x \text{BM-MC}_k(x),$$

denote the globally optimal BM-MC$_k$ objective value for G. Lastly, we say that the solution x is κ-approximate if HP$(x) \geq \kappa$Max-Cut(G) and similarly, x is considered κ-close if BM-MC$_k(x) \geq \kappa$Max-Cut(G).

Not only is optimizing a non-convex (non-concave) optimization problem difficult, but even finding a local optimum to a non-convex optimization problem can be challenging due to saddle-points. Nevertheless, first and second-order optimization methods have showed promising performance in converging to high quality local optima for low-rank BM formulation of Max-Cut (and many other combinatorial optimization problems). Burer and Monteiro invented this heuristic method, motivated by *existence* of a low rank optimal solutions to the original (n dimensional) SDP whenever $\binom{k}{2}$ is no less than the number of constraints of the SDP, known as the Barvinok-Pataki bound [72, 73]. Their method has showed promising performance in practice, even in constant dimensions and is an active area of research in non-convex optimization theory [74–76]. Experiments by Burer, Monteiro, and Zhang [76] demonstrate that BM-MC$_2$ performs much more quickly while maintaining relatively good solutions; on one particular 20,000-node instance, the GW algorithm took over 1.5 days to complete, whereas a rank-2 approximate BM-MC solution was found in a little over a second; repeated runs of BM-MC$_2$ over the course of a couple minutes on the same graph yielded cuts that were at least as good as those obtained by the GW algorithm [76]. More details on the runtime of the GW algorithm and BM-MC$_k$ can be found in Section 4.5.6.

Recently, Mei et al. [77] showed that, for Max-Cut SDPs corresponding to graphs with non-negative edge-weights, any second-order local optimum for the BM formulation is approximately optimal with respect to the original SDP.

Theorem 2 (Mei et al. [77]). *For graphs with non-negative edge weights, the objective at a locally optimal solution, for the above non-convex, rank-k SDP formulation, is within a factor $1 - \frac{1}{k-1}$ of that of the rank-n SDP.*

To parse Mei et al.'s result in other words, for an n-node graph G with non-negative edge weights, any locally-optimal solution x to BM-MC$_k$ is also $(1 - \frac{1}{k-1})-$close since BM-MC$_n(G) \geq$ Max-Cut(G). The above theorem highlights the fact that increasing k improves performance of the BM formulation; however, for the purposes of this work (and simple mapping to the Bloch sphere), we restrict our attention to rank-2 and rank-3 solutions.

2.4 The Quantum Approximate Optimization Algorithm

In this section, we review the hybrid quantum-classical algorithm of QAOA for the Max-Cut problem. QAOA assigns a quantum spin to every binary output variable. In each of the p layers of the algorithm, a cost unitary (corresponding to Hamiltonian H_C) and a mixing unitary (corresponding to Hamiltonian $H_B = \sum_{i\in[n]} \sigma_i^x$, where σ_i^k is a Pauli matrix for qubit i with $k = x, y, z$), are alternately applied to the initial quantum processor state $|s_0\rangle$, generating a variational wavefunction

$$|\psi_p(\gamma, \beta)\rangle = e^{-i\beta_p H_B} e^{-i\gamma_p H_C} \cdots e^{-i\beta_1 H_B} e^{-i\gamma_1 H_C} |s_0\rangle \,, \tag{2.11}$$

where $|s_0\rangle = |+\rangle^{\otimes n}$ is the standard initial state. When the circuit in Equation 2.11 is used, we will often say that we are running *depth-p QAOA*; this notion of depth should not be confused with the notion of circuit depth for general quantum circuits. Sampling from the final variational state will yield a cut with an expected cut value of:

$$F_p(\gamma, \beta) = \langle \psi_p(\gamma, \beta)| \, H_C \, |\psi_p(\gamma, \beta)\rangle \,.$$

In general, for any cost function $f : \{0,1\}^n \to \mathbb{R}$, the corresponding cost Hamiltonian H_C is defined so that $H_C |b\rangle = f(b) |b\rangle$ for all $b \in \{0,1\}^n$.

In the specific case of the maximization problem Max-Cut, it can be shown [1] that the cost Hamiltonian for a graph $G = (V, E)$ (with weights $w : E \to \mathbb{R}$) can be written as

$$H_C = \frac{1}{2} \sum_{(i,j)\in E} w_{ij}(1 - \sigma_i^z \sigma_j^z) \,.$$

The (near) optimal parameters of the algorithm, γ, β, are found by a classical algorithm to maximize the performance of the QAOA algorithm, with $F_p(\gamma, \beta)$ viewed as a multi-dimensional non-convex function. We let M_p denote the expected cut value with optimal choice of γ, β parameters, i.e.,

$$M_p = \max_{\gamma,\beta} F_p(\gamma, \beta).$$

It is not difficult to see that M_p is a non-decreasing function in p; moreover, as previously mentioned, $M_p \to$ Max-Cut(G) as $p \to \infty$ [32]. For graphs with non-negative edge-weights, the ratio $M_p /$ Max-Cut$(G) \geq 0.5$ for all $p \geq 0$ due to the 0.5-approximation ratio achieved at $p = 0$ for the standard initialization.

To find the optimal variational parameters, one can simply perform a dense grid search for $\gamma, \beta \in [-\pi, \pi]^{2p}$, but this would be feasible only for small circuit depths. For scalability, one can instead treat $F_p(\gamma, \beta)$ as a black-box[10] and utilize a classical optimizer to (iteratively) update and find suitable values of γ and β in an effort to the maximize the expected cut value. For any classical optimization algorithm $\mathcal{A}$, it will eventually terminate at some $(\gamma, \beta) = (\hat{\gamma}, \hat{\beta})$; in the context of instance-specific approximation ratio (Equation 2.2), the expected cut value is $\mathcal{E}_{\mathcal{A},G} = F_p(\hat{\gamma}, \hat{\beta})$.

To optimize the variational parameters, we consider four choices of the optimizer:

[10]For actual quantum devices, the value of $F_p(\gamma, \beta)$ and its gradients can be estimated by taking multiples measurements of $\psi_p(\gamma, \beta)$ in the computational basis.

ADAM [78], COBYLA [79], Nelder-Mead [80], and BFGS [81]. Since $F_p(\gamma, \beta)$ is non-convex, classical optimizers are not guaranteed to stop at a globally optimal choice of γ and β, i.e., the expected result of QAOA will not always be equal to M_p (i.e. the expected result of QAOA had we initialized γ and β optimally). ADAM and BFGS operate with the first-order information (i.e., using gradient estimates), whereas COBYLA and Nelder-Mead operate with the zeroth-order information (i.e., function value estimates). On quantum devices, gradients are estimated using multiple evaluations of the function $F_p(\gamma, \beta)$ at various (γ, β); these function evaluations are noisy since $F_p(\gamma, \beta)$ itself is estimated by taking an average of multiple quantum measurements. For this reason, gradient-free optimizers are typically more robust against quantum noise and are recommended in practice over gradient-based methods [82]. Application of machine learning techniques for optimizing the variational parameters (a technique known as meta-learning) has also shown promise in the noisy quantum setting [83]. Recent results regarding the concentration of the (standard) QAOA landscape can also be used to speed up optimization of the variational parameters [84]. Even though the runtimes for various optimizers may significantly differ, we find that the choice of the optimizer has much smaller impact on the instance-specific approximation ratio achieved for QAOA-warm (discussed in Section 4.5).

2.5 Independent Set Problem

In Chapter 6, we consider an example of a combinatorial optimization problem with *hard* constraints called the Independent Set problem. Given a graph $G = (V, E)$, the goal of the Independent Set problem is to find largest independent set $S \subseteq V$ where an independent set is defined to be any subset of vertices in the graph such that no edges exists between the vertices of S, more precisely, we wish to solve $\max_{x_i \in \{0,1\}} x_i$ subject to $x_i + x_j \leq 1$

[10] One can calculate (or approximate) the gradient using a variety of methods. Our implementation approximates the gradient using an analytic forward difference method implemented in TENSORFLOW QUANTUM (with default parameters `error_order=1` and `grid_spacing=0.001`). By *analytic*, we mean that any expectations computed in the calculation are computed *exactly* (instead of using a sampling-based approximation).

for all $(i, j) \in E$ where $x_i = 1$ denotes that vertex i is included in the independent (and $x_i = 0$ indicates otherwise). It is known that the Independent Set problem is NP-Hard [6]. While constant-factor approximation ratios for the Independent Set problem exists for certain special classes of graphs [85–87], there does not exist such a constant-factor approximation ratio for general graphs [88].

CHAPTER 3

INTERESTING INSTANCES FOR QUANTUM ADVANTAGE

As previously discussed in Chapter 1, demonstrating quantum advantage[1], i.e. the ability for physical quantum computers to complete a given task that no classical computer can complete in a reasonable amount of time, has been of particular interest to quantum researchers. In 2019, Google had announced that they had demonstrated quantum advantage on a 53 qubit quantum device; however, the problem they solved was specifically designed for the sole purpose of illustrating quantum advantage and thus is not of practical use [89]. On the other hand, there exists algorithms, such as Shor's algorithm for integer factorization, which are theoretically faster than any currently known classical algorithm; however, to be of practical use, Shor's algorithm requires a large number of qubits which is currently not feasible for current quantum devices. [23].

Many believe that the Quantum Approximate Optimization Algorithm (QAOA) applied to the Max-Cut problem has the potential for demonstrating a quantum advantage over classical algorithms [35, 46, 47, 90–92]. This is in part due to a recent result by Farhi et al. [1] which shows that under the adiabatic limit (i.e., as the circuit depth goes to infinity), the QAOA algorithm would converge to the Max-Cut. It may be the case that QAOA (and its variants) only have an advantage over classical algorithms on certain classes of graphs for finite circuit depths. To identify such classes of graphs, one can take a two-pronged approach: (i) find graph families which are challenging for classical optimization algorithms, (ii) show that the QAOA performs well on such graph families. From the perspective of

[1] We note that there is not universal agreement regarding the definition of quantum advantage. Many have used the term *quantum supremacy* to refer to what we call quantum advantage in this thesis; meanwhile, many will instead use the term quantum advantage to mean a situation where, for some given task, a physical quantum devices has at least some *slight* edge in runtime compared to any classical algorithm on any classical machine. Regardless of the definition of quantum advantage that one uses, we believe that the instances presented in this chapter should be of interest to the reader.

showing quantum advantage for certain families of graphs on near-term NISQ devices, it is further of interest to find such relatively small graphs.

In this chapter[2], we further our knowledge of challenging families for the celebrated Goemans-Williamson algorithm [11]. The Goemans-Williamson (GW) algorithm attains the best-possible approximation ratio of 0.878 in polynomial time for graphs with non-negative edge-weights, assuming that the Unique Games Conjecture is true [12–14].

Regarding related work, Herrman et al. [90] run numerical simulations for Max-Cut QAOA, up to depth $p = 3$, for all non-isomorphic graphs with 8 or fewer nodes, in order to find correlations between various graph properties and performance metrics (including the approximation ratio and the probability of observing the maximum cut); in particular, they find that QAOA performs empirically better on graphs with higher amounts of symmetry, thus further motivating the investigation of the classes of instances considered in this work, which we will now discuss.

In Section 3.1, one such class that we consider is a sequence of graphs proposed by Karloff [2] whose GW instance-specific approximation ratio approaches the GW bound of 0.878. The smallest possible graph using Karloff's construction has 924 nodes and the remaining instances in the sequence quickly blow up in size making them not suitable for current or near-term quantum devices; however, by carefully analyzing Karloff's proof and one of his conjectures (later proven by Brouwer et al. [93]), we show (Table 3.1) that there are other instances under 1000 nodes that can be constructed using Karloff's construction with the smallest having 20 nodes. In Section 3.2, we prove that the GW algorithm also achieves a low instance-specific approximation ratio for a special class of strongly-regular graphs. In particular, for all strongly-regular graphs $G = (V, E)$ parameterized by (n, k, λ, μ) with $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some integer t, we prove (Theorem 12) that the GW algorithm yields a 0.912-approximation whenever Max-Cut$(G) = \frac{2}{3}m$; we computationally verify that this condition (Max-Cut$(G) = \frac{2}{3}m$) holds for all but 13

[2]The results presented in this chapter is joint work with Swati Gupta and is in preparation for submission to Operation Research Letters.

instances amongst all strongly-regular graphs with the parameters described above.

Many other classical algorithms exist for the Max-Cut problem including Trevisan's algorithm, which yields a 0.614-approximation [94, 95], and numerous heuristics such as those found in the MQLib library [96]. The task of identifying classes of instances that demonstrate quantum advantage over *all* classical algorithms is very difficult; for this reason, we first consider the simpler task of demonstrating an advantage over the best-known classical algorithm, i.e., the GW algorithm. Additionally, we remark that in the context of Sum of Squares (SOS) hierarchies [97, 98], the GW relaxation is equivalent to a degree-2 SOS relaxation; higher-degree relaxations can potentially be used, thus obtaining stronger relaxations, but this requires a prohibitively higher computational cost.

Next, in Section 3.3, we provide empirical results for classes of instances which may be of interest from a benchmarking perspective. First, using the instances and heuristics found in the MQLib library [38], we identify classically-hard instances for which no classical heuristic obtains an instance-specific approximation ratio of 0.999 or more within a given (instance-dependent) time-frame; such instances have edges of both positive and negative weight whose magnitudes span several orders of magnitude as seen in Figure 3.3. Looking forward in Chapter 5, we empirically show that QAOA also achieves low-quality solutions for similarly-constructed instances with mixed edge weights. Afterwards, in order to generate more suitably-sized instances for near-term quantum benchmarking, we further extend the small instances generated by Karloff's construction by applying perturbations (e.g. edge deletions and edge-weight modifications) to such instances; we find that, empirically, the GW algorithm is sensitive to such changes, causing the instance-specific approximation ratio to rise (compared to the unperturbed instance), yet these approximation ratios are still bounded considerably away from 1.

Finally, in Section 3.4, we discuss QAOA's performance on the instances in Section 3.1 where the GW algorithm provably performs poorly as a result of Karloff's construction. We prove (Theorem 15) that depth-1 QAOA achieves an instance-specific approximation ratio

that approaches 0.592 for the sequence of graphs constructed by Karloff; thus, a higher-depth or some modification of the QAOA algorithm is required in order to demonstrate some form of quantum advantage for these instances. For strongly-regular graphs, the proof of Theorem 15 can not directly be extended as our proof uses a result by Wang et al. [99] that is only applicable for triangle-free graphs and strongly-regular graphs are not triangle-free in general.

3.1 Small Instances from Karloff's Construction with Low GW Approximation Ratios

We review a known graph family constructed by Karloff in [2] where the GW algorithm attains the tight approximation ratio of 0.878 as the graph size increases. The smallest instance using Karloff's construction has 924 nodes, which is infeasible for numerically testing or simulating on current quantum hardware. Instead, we studied the construction to increase the size of the graph family by showing that a conjecture, regarding the eigenvalues of the adjacency matrix of the graphs in Karloff's construction, is in fact true [93]. This allowed us to expand the set of graphs where the GW algorithm has low approximation ratio to include graphs of 20 nodes and above (see Table 3.1).

3.1.1 Review of Karloff's Construction

We encourage the reader to first review Section 2.3 regarding the Goemans-Williamson algorithm. Using the notation from Section 2.2, we let $\alpha_{\text{GW},G}$ denote the approximation ratio achieved by the GW algorithm on graph G.

Goemans and Williamson [11] proved that for a graph $G = (V, E)$ with non-negative edge weights,

$$\alpha_{\text{GW},G} \geq \alpha^*, \text{ where } \alpha^* := \frac{2}{\pi} \min_{\theta \in [0,\pi]} \frac{\theta}{1 - \cos\theta};$$

calculating α one finds that $0.878 < \alpha^* < 0.879$.

However, the typical analysis of Goemans-Williamson does not show that this is tight; that is, could there perhaps be a constant $\beta^* > \alpha^*$ such that $\alpha_{\text{GW},G} \geq \beta^*$ for all graphs G? Karloff [2] proved that the GW algorithm can indeed guarantee no better than an $\alpha^* \approx 0.878$ approximate solution, by constructing simple graphs for which the expected performance of the algorithm is arbitrary close to α^*.

These challenging instances for the Goemans-Williamson algorithm are explained as follows. For non-negative integers $b \leq t \leq m$, let $J(m, t, b)$ denote the graph with vertex set $\binom{[m]}{t}$, i.e., the vertices are all t-element subsets of $[m]$; two distinct vertices/subsets S and T of $J(m, t, b)$ are adjacent if and only if they have exactly b elements in common, i.e. $|S \cap T| = b$. Of particular interest are graphs $J(m, t, b)$ where m is even and $t = m/2$.

By choosing m, t, b appropriately as seen in Theorem 3, Karloff proves that this construction yields instances that are arbitrarily close to the $\alpha^* \approx 0.878$ bound.

Theorem 3 (Karloff [2])**.** *There exists an optimal solution Y of the GW SDP relaxation such that for each $\varepsilon > 0$, there are b and m, m even and positive and $0 \leq b \leq m/12$, such that $\alpha_{GW,G} \leq \alpha^* + \varepsilon$ where $G = J(m, m/2, b)$.*

The construction of the matrix Y in Theorem 3, given by Karloff [2], is as follows. For each vertex/subset S of $J(m, m/2, b)$, the vector $w_S \in \mathbb{R}^m$ is constructed where the ith entry of w_S is $+1$ if $i \in S$ and -1 if $i \notin S$. Then, each w_S is rescaled by $1/\sqrt{m}$ so that the w_S's are of unit length. Now, an $m \times n$ matrix W is built by letting the column indexed by S be the vector x_S. Finally, Y is set to $Y = W^T W$. The feasibility of Y (with respect to the GW relaxation) can be easily verified; Karloff proves that Y is also an optimal solution provided that $0 \leq b \leq m/12$ (a condition in Theorem 3).

In order to better understand the instances that satisfy the conditions of Theorem 3 and the GW algorithm's performance on such instances, we next review the specific approximation ratio that is achieved as a function of the parameters m and b used in Karloff's construction [2].

Theorem 4 (Karloff [2]). *Let m be an even positive integer and $G = J(m, m/2, b)$. If $0 \leq b \leq m/12$, then, using the optimal solution $\hat{Y}_G$,*

$$\alpha_{GW,G} = \frac{\frac{1}{\pi}\arccos(\frac{4b}{m} - 1)}{1 - \frac{2b}{m}} = \frac{2}{\pi}\frac{\theta}{1 - \cos(\theta)} \geq \min_{\theta' \in [0,\pi]} \frac{2}{\pi}\frac{\theta'}{1 - \cos(\theta')} = \alpha^*,$$

where $\theta = \arccos\left(\frac{4b}{m} - 1\right)$.

Numerically calculating the minimizer θ^* in the inequality above yields $\theta^* \approx 2.33112$. Thus, the closer $\theta = \arccos(4b/m - 1)$ is to $\theta^* \approx 2.33112$, the lower the instance-specific approximation ratio. Equivalently, in order to minimize the instance-specific approximation ratio, the ratio b/m should be chosen to be as close to $\frac{1}{4}(\cos(\theta^*) + 1) \approx 0.0777$ as possible; note that this can be achieved by picking m large enough and then picking a suitable b. Furthermore, if one picks $b/m \approx 0.0777$, then the conditions of Theorem 4 still hold as $b \approx 0.0777m \leq m/12$. Since it is clear that b and m can be chosen to make the approximation ratio arbitrarily close to α^*, Theorem 3 follows.

3.1.2 Identification of Small Instances Using Karloff's Approach

Theorem 4 seems to yield a promising approach for finding instances G where $\alpha_{\text{GW},G}$ is small; however, Theorem 4 is not sufficient for finding small instances that are feasible for near-term quantum computers. To illustrate why this is the case, we first consider the case where $b = 0$. In this case, $\theta = \arccos(-1) = \pi$ and thus $\alpha_{\text{GW},J(m,m/2,0)} = \frac{2}{\pi}\frac{\pi}{1-\cos(\pi)} = 1$ which is not very interesting. If $b > 0$, then the condition $b \leq m/12$ in Theorem 4 implies that $m \geq 12$ in which case the graph $J(m, m/2, b)$ would have $n \geq \binom{12}{6} = 924$ nodes which is much more than the number of qubits in most modern quantum computers.

One way to obtain smaller interesting instances is to find a way to relax the condition $b \leq m/12$ in Theorem 4. This condition is needed since Theorem 4 invokes the following theorem.

Theorem 5 (Karloff [2]). *Let m be an even positive integer and let $0 \leq b \leq m/12$. The*

Table 3.1: A listing of small (< 1000 nodes) instances using Karloff's construction. For each instance, we include the number of nodes, edges, and the theoretical instance-specific approximation ratio one would obtain if running the GW algorithm on that instance (assuming an optimal solution of Y as described earlier).

Instance G	Number of Nodes	Number of Edges	$\alpha_{\text{GW},G}$
$J(6,3,1)$	20	90	0.912260171954089
$J(8,4,1)$	70	560	0.8888888888888888
$J(10,5,1)$	252	3150	0.881040955873917
$J(10,5,2)$	252	12600	0.940157028081625
$J(12,6,1)$	924	16632	0.878735432638524
$J(12,6,2)$	924	103950	0.912260171954089

smallest eigenvalue of the adjacency matrix of $J(m, m/2, b)$ *is*

$$\binom{m/2}{b}^2 \left[\frac{4b}{m} - 1\right].$$

Karloff conjectured that the condition $0 \leq b \leq m/12$ in Theorem 5 could be relaxed to the condition $0 \leq b < m/4$. Fortunately, in 2018, this conjecture was proven by Brouwer et al. (see remark after Theorem 3.10 in [93]). We had gone through Karloff's proof of Theorem 4 and observed that the only time the condition $0 \leq b < m/12$ is used is in the invocation of Theorem 5. Thus, the proven conjecture also implies that the condition $0 \leq b \leq m/12$ in Theorem 4 can *also* be relaxed to $0 \leq b < m/4$. For convenience, we restate Theorem 4 with the relaxed inequality below.

Theorem 6 (Karloff [2])**.** *Let* m *be an even positive integer and* $G = J(m, m/2, b)$*. If* $0 \leq b < m/4$*, then*

$$\alpha_{GW,G} = \frac{\frac{1}{\pi}\arccos(\frac{4b}{m} - 1)}{1 - \frac{2b}{m}} = \frac{2}{\pi}\frac{\theta}{1 - \cos(\theta)} \geq \min_{\theta' \in [0,\pi]} \frac{2}{\pi}\frac{\theta'}{1 - \cos(\theta')} = \alpha^*$$

where $\theta = \arccos\left(\frac{4b}{m} - 1\right)$.

From Theorem 6, we generate instances $J(m, m/2, b)$ where m is as small as $m = 6$ and whose GW instance-specific approximation ratios are at most 0.940; these instances and approximation ratios are displayed in Table 3.1.

3.2 Provable Guarantees for the GW Algorithm on Strongly-Regular Graphs

In Karloff's proof of Theorem 4, he exploits the fact that, for any fixed instance $J(m, m/2, b)$ with $0 \leq b \leq m/12$, angles between adjacent vertices in the optimal GW SDP solution are all equal (with angle $\theta = \arccos\left(\frac{4b}{m} - 1\right)$). This raises an important question: *are there other graphs whose optimal GW SDP solution has equal angles between adjacent vertices, and if so, what is the GW approximation ratio for such graphs*? This subsection answers this question in the affirmative: we prove that such a property holds for nearly all instances in a particular family of strongly-regular graphs (i.e. those parameterized by $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$ for some integer t) and that the GW algorithm yields an instance-specific approximation ratio of 0.912 amongst all instances in this family (Theorem 12).

This choice of parameters for this family of strongly-regular graphs was inspired by particular instances in the MQLib library [38] and ultimately found due to the connection that strongly-regular graphs have with partial geometries; we refer the reader to Appendix A for more details. Additionally, this family of strongly-regular graphs has several instances under 100 nodes, making them suitable for current and near-term quantum devices.

We divide the proof of Theorem 12 into multiple parts. First, in Proposition 5 , we generalize a portion of Karloff's proof to prove that hyperplane rounding of the GW algorithm yields a cut with $\frac{\theta}{\pi}|E|$ edges in expectation for any unit-weight graph whose optimal GW SDP solution has equal angles θ between adjacent vertices. Then, we define the notion of strongly-regular graphs (Definition 7) and exploit the well-known fact that the vertices of such graphs can be mapped onto a unit-hypersphere so that the angles between adjacent vertices are all equal (Theorem 8) [100]. For a particular family of strongly regular graphs, we prove (Theorem 11) this mapping corresponds to an optimal GW SDP solution by finding a feasible solution to the dual SDP with the same objective value. Finally, we computationally verify that for all but 13 instances in this family of graphs, the maximum

cut contains exactly two-thirds of the edges; this last step then allows us to conclude in Theorem 12 that, with the exception of the 13 instances, the GW algorithm yields a 0.912 instance-specific approximation ratio for instances in this family.

We begin the proof with Proposition 5 which relates the angles in the SDP solution to the GW approximation ratio in the case that graph has unit-weight edges and equal angles between all adjacent vertices in the optimal GW SDP solution.

Proposition 5. *If $G = (V, E)$ is a unit-weight graph with optimal GW SDP solution $Y = x^T x$ and if there exists $\theta \in \mathbb{R}$ such that $\arccos(x_u \cdot x_v) = \theta$ for all $(u, v) \in E$, then the GW algorithm (with hyperplane rounding on x) yields exactly $\frac{\theta}{\pi}|E|$ edges in expectation and obtains an instance-specific approximation ratio of at least $\frac{\theta}{\pi}$.*

Proof. Let $G = (V, E)$ be a unit-weight graph with optimal GW SDP solution $Y = x^T x$ such that, for some $\theta \in \mathbb{R}$, $\arccos(x_u \cdot x_v) = \theta$ for all $(u, v) \in E$. Recall that HP(x) denotes the expected number of edges obtained from applying hyperplane rounding to x. Thus,

$$\begin{aligned}
\text{HP}(x) &= \sum_{(i,j)\in E} \frac{w_{i,j}}{\pi} \arccos(x_i \cdot x_j) \\
&= \sum_{(i,j)\in E} \frac{1}{\pi} \arccos(x_i \cdot x_j) \\
&= \sum_{(i,j)\in E} \frac{1}{\pi}\theta \\
&= \frac{\theta}{\pi}|E|.
\end{aligned}$$

For the last part of the proof, observe that for any unit-weight graph, Max-Cut$(G) \leq$

$|E|$, and thus,

$$\begin{aligned}\frac{\text{HP}(x)}{\text{Max-Cut}(G)} =& \frac{\frac{\theta}{\pi}|E|}{\text{Max-Cut}(G)} \\ \geq& \frac{\frac{\theta}{\pi}|E|}{|E|} \\ =& \frac{\theta}{\pi},\end{aligned}$$

which completes the proof. □

In 1963, Bose introduced the notion of strongly-regular graphs [101] defined below. These strongly graphs, parameterized by[3] n, k, λ, and μ, are highly symmetric; from this symmetry, as seen by Theorem 8, Seidel [100] proves that the vertices of an n-node strongly-regular graphs can be mapped to an $(r-1)$-dimensional hypersphere with $r \leq n$ so that, points corresponding to adjacent vertices all have the same instance-dependent angle θ.

Definition 7. *A unit-weight graph G is a SRG(n, k, λ, μ), i.e., G is a strongly-regular graph with parameters n, k, λ, μ, if G has n nodes, G is a k-regular graph, every pair of adjacent vertices in G have λ common neighbors, every pair of non-adjacent vertices in G have μ common neighbors, and G is neither a complete graph nor the complement of a complete graph. A strongly-regular graph G is said to be* primitive *if both G and its complement are connected.*

Theorem 8 (Seidel [100])**.** *Let $G = (V, E)$ be a primitive strongly regular graph SRG(n, k, λ, μ). Let ξ_2 be the smallest eigenvalue of the adjacency matrix of G. Then there exists an integer $r \leq n$ and a function $f : V \to \mathbb{S}^{r-1}$ so that for all $(u, v) \in E$,*

[3] In the literature for strongly-regular graphs, one typically uses the parameter v instead of n to represent the number of vertices; this was done to avoid confusion with the other notation used throughout this thesis. It should also be noted that for any fixed set of parameters (v, k, λ, μ), there may be more than one non-isomorphic strongly-regular graph with those parameters.

$f(u) \cdot f(v) = \xi_2/k$.

Proposition 5 cannot immediately be applied to strongly-regular graphs as Theorem 8 alone does not provide any guarantees on the optimality of the spherical mapping (with respect to the GW SDP); however, we do prove such optimality for a specific family of strongly-regular graphs, namely those with parameters of the form $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = \frac{k}{3}$. In order to demonstrate such a proof, we first utilize a closed-form analytical expression for the eigenvalues of the adjacency matrices of strongly-regular graphs (Theorem 9) to the family above in Proposition 6; this allows us to determine (in Corollary 10) the angles between adjacent vertices in the spherical mapping used in Theorem 8 for the specific family of strongly-regular graphs mentioned above.

Theorem 9 (Seidel [100])**.** *Let G be an SRG(n, k, λ, μ). Then the eigenvalues of the adjacency matrix of G are:*

- *Eigenvalue k with multiplicity 1.*
- *Eigenvalue*

$$\xi_1 = \frac{1}{2}\left[(\lambda - \mu) + \sqrt{(\lambda-\mu)^2 + 4(k-\mu)}\right]$$

with multiplicity

$$\frac{1}{2}\left[(n-1) - \frac{2k + (n-1)(\lambda-\mu)}{\sqrt{(\lambda-\mu)^2 + 4(k-\mu)}}\right].$$

- *Eigenvalue*

$$\xi_2 = \frac{1}{2}\left[(\lambda - \mu) - \sqrt{(\lambda-\mu)^2 + 4(k-\mu)}\right]$$

with multiplicity

$$\frac{1}{2}\left[(n-1) + \frac{2k + (n-1)(\lambda-\mu)}{\sqrt{(\lambda-\mu)^2 + 4(k-\mu)}}\right].$$

Additionally, ξ_2 is the smallest eigenvalue.

Theorem 9 implies that the eigenvalues of a strongly-regular graph are entirely determined by the parameters (n, k, λ, μ). By choosing the parameters appropriately as shown

in Proposition 6 below (i.e. $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = \frac{k}{3}$), this induces a family of graphs whose 2nd smallest eigenvalue is given by a simple expression ($\xi_2 = -k/3$); by Corollary 10, this implies that, for this family of graphs, there exists a spherical mapping f of the vertices such that the angles between adjacent vertices are equal (with angle $\arccos(-1/3)$). We remark that this angle is identical to the angles formed by the vertices of a regular tetrahedron with its center.[4]

Proposition 6. *Let G be a SRG(n, k, λ, μ) where $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Let ξ_2 be the smallest eigenvalue of the adjacency matrix of G. Then $\xi_2 = -\frac{k}{3}$.*

Proof. Let G be an SRG$(4(3t + 1), 3(t + 1), 2, t + 1)$ for some non-negative integer t. Observe that the SRG parameter μ is equal to $\mu = t + 1 = \frac{1}{3} \cdot 3(t + 1) = \frac{k}{3}$.

From Proposition 9, substitution of the above parameter values, and some algebraic manipulation, we have that

$$\xi_2 = \frac{1}{2}\left[(\lambda - \mu) - \sqrt{(\lambda - \mu)^2 + 4(k - \mu)}\right]$$

$$= \frac{1}{2}\left[(2 - k/3) - \sqrt{(2 - k/3)^2 + 4(k - k/3)}\right] = -\frac{k}{3}.$$

□

Corollary 10. *Let $G = (V, E)$ be a SRG(n, k, λ, μ) where $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Then there exists an integer $r \leq n$ and a function $f : V \to \mathbb{S}^{r-1}$ so that for all $(u, v) \in E$, $f(u) \cdot f(v) = -1/3$.*

Proof. Let $G = (V, E)$ be a SRG(n, k, λ, μ) where $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Let ξ_2 be the smallest eigenvalue of the adjacency matrix of G. By Proposition 10, the smallest eigenvalue of the adjacency matrix of G is given by $\xi_2 = -k/3$. It is straightforward to see that any non-primitive strongly-regular

[4] See [102] for a simple, yet succinct proof that such angles in a tetrahedron have a measure of $109.47°$.

graph must either satisfy $\mu = 0$ or $\mu = k$; since this is not the case for G, then G must be a primitive strongly-regular graph. Thus, by Theorem 8, there exists an integer $r \leq n$ and a function $f : V \to \mathbb{S}^{r-1}$ so that for all $(u, v) \in E$, $f(u) \cdot f(v) = \frac{\xi_2}{k} = \frac{-k/3}{k} = -\frac{1}{3}$. $\square$

In order to apply Proposition 5 to the family of graphs and the SDP solution corresponding to the mapping f obtained from Corollary 10 above, we must prove that such a solution is optimal (with respect to the GW SDP). We prove this optimality using a standard technique in optimization: find a feasible solution to the dual optimization problem with the same objective value. In particular, we prove (Propositions 7 and 8) that the primal GW SDP and its dual have an optimal objective value of $\frac{2}{3}|E|$.

For ease of notation, for feasible solutions Y of the primal GW SDP, we let $z_P(Y)$ denote the objective value of the primal GW SDP at Y. Similarly, for feasible solutions ζ of the dual to the GW SDP, we let $z_D(\zeta)$ denote the objective value of the dual at ζ. Additionally, we let $z_P^* = \max_Y z_P(Y)$ and $z_D^* = \min_\zeta z_D(\zeta)$ represent the optimal objective values of the GW SDP and its dual, where the max and min are taken over all feasible solutions of the SDP and its dual (respectively).

Proposition 7. *Let G be a SRG(n, k, λ, μ) where $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Then there exists a feasible Y for the GW SDP so that $z_P(Y) = \frac{2}{3}|E|$.*

Proof. Let $G = (V, E) \in \mathcal{Q}$ with $|V| = n$. By Corollary 10, there exists an integer $r \leq n$ and a function $f : V \to \mathbb{S}^{r-1}$ so that for all $(u, v) \in E$, we have $f(u) \cdot f(v) = -1/3$. We can embed the solution onto a sphere that lies in $\mathbb{R}^n$ by extending the function $f : V \to \mathbb{S}^{r-1}$ to the function $\hat{f} : V \to \mathbb{S}^{n-1}$ where, for all $i, j \in [n]$,

$$\hat{f}(i)_j = \begin{cases} f(i)_j, & 1 \leq j \leq r \\ 0, & r + 1 \leq j \leq n, \end{cases}$$

where $\hat{f}(i)_j$ denotes the jth entry of $\hat{f}(i)$. It is straightforward to see that angles are preserved, i.e., for all $u, v \in V$, $\hat{f}(u) \cdot \hat{f}(v) = f(u) \cdot f(v)$.

Let x be an matrix where the ith column is given by $\hat{f}(i)$ for all $i \in [n]$ and let $Y = x^T x$. We can now calculate $z_P(Y)$:

$$\begin{aligned} z_P(Y) &= \sum_{(i,j)\in E} w_{ij} \frac{1 - Y_{ij}}{2} \\ &= \sum_{(i,j)\in E} \frac{1 - Y_{ij}}{2} && (w_{ij} = 1) \\ &= \sum_{(i,j)\in E} \frac{1 - \hat{f}(i) \cdot \hat{f}(j)}{2} \\ &= \sum_{(i,j)\in E} \frac{1 - f(i) \cdot f(j)}{2} \\ &= \sum_{(i,j)\in E} \frac{1 - (-1/3)}{2} \\ &= \frac{2}{3}|E|, \end{aligned}$$

as desired. □

Proposition 8. *Let G be a SRG(n, k, λ, μ) where $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Then there exists a feasible ζ for the dual of the GW SDP so that $z_D(\zeta) = \frac{2}{3}|E|$.*

Proof. Let G be a SRG(n, k, λ, μ) where $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Let ξ_2 be the smallest eigenvalue of the adjacency matrix of G; from Proposition 9, we have that $\xi_2 < 0$.

The dual of the GW SDP relaxation is given as follows: find $\zeta = (\zeta_1, \ldots, \zeta_n) \in \mathbb{R}^n$ to minimize

$$z_D(\zeta) = \frac{1}{2}W + \frac{1}{4}\sum_{i=1}^{n} \zeta_i,$$

subject to,

$$A + \text{diag}(\zeta_1, \ldots, \zeta_n),$$

being positive semidefinite, where W is the sum of the weights in G and $\text{diag}(\zeta_1, \ldots, \zeta_n)$ is a diagonal matrix whose ith entry on the diagonal is ζ_i [2].

Let $\zeta = (-\xi_2, \ldots, -\xi_2)$. Then,

$$A + \text{diag}(\zeta_1, \ldots, \zeta_n) = A + \text{diag}(-\xi_2, \ldots, -\xi_2) = A - \xi_2 I.$$

It is a standard result of linear algebra that for any square matrix M and constant c, if λ is an eigenvalue of M, then $\lambda + c$ is an eigenvalue of $M + cI$; thus letting $\lambda_{\text{min}}(M)$ denote the smallest eigenvalue of a matrix M, we have,

$$\lambda_{\text{min}}(A + \text{diag}(\zeta_1, \ldots, \zeta_n)) = \lambda_{\text{min}}(A - \xi_2 I) = \lambda_{\text{min}}(A) - \xi_2 = \xi_2 - \xi_2 = 0.$$

Since $\lambda_{\text{min}}(A+\text{diag}(\zeta_1, \ldots, \zeta_n)) \geq 0$, then $A+\text{diag}(\zeta_1, \ldots, \zeta_n)$ is positive-semidefinite, meaning that ζ is a feasible solution.

We now calculate $z_D(\zeta)$:

$$
\begin{aligned}
z_D(\zeta) &= \frac{1}{2}W_{\text{tot}} + \frac{1}{4}\sum_{i=1}^{n}\zeta_i \\
&= \frac{|E|}{2} + \frac{1}{4}\sum_{i=1}^{n}\zeta_i && (w_{ij} = 1 \text{ for } (i,j) \in E) \\
&= \frac{|E|}{2} + \frac{1}{4}\sum_{i=1}^{n} -\xi_2 \\
&= \frac{|E|}{2} - \frac{\xi_2 n}{4} \\
&= \frac{|E|}{2} + k\frac{n}{12} && (\xi_2 = -k/3 \text{ from proof of Prop. 10}) \\
&= \frac{|E|}{2} + \frac{2|E|}{n}\frac{n}{12} && (|E| = nk/2 \text{ as } G \text{ is } k\text{-regular}) \\
&= \frac{2}{3}|E|,
\end{aligned}
$$

as desired. □

Theorem 11. *Let G be a SRG(n, k, λ, μ) where $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$ for some non-negative integer t. be a graph. Then $z_P^* = \frac{2}{3}|E|$ (and similarly, $z_D^* = \frac{2}{3}|E|$). Moreover, the mapping f obtained by Corollary 10 corresponds to an optimal GW SDP solution Y.*

Proof. Let G be a SRG(n, k, λ, μ) where $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some non-negative integer t. Let Y be the feasible GW SDP solution corresponding to the mapping f obtained in Corollary 10. From Propositions 7 and 8, there exists feasible ζ such that $z_P(Y) = \frac{2}{3}|E| = z_D(\zeta)$. Thus,

$$
\frac{2}{3}|E| = z_P(Y) \leq z_P^* \leq z_D^* = z_D(\zeta) = \frac{2}{3}|E|,
$$

implying that $\frac{2}{3}|E| \leq z_P^* \leq \frac{2}{3}|E|$ and $\frac{2}{3}|E| \leq z_D^* \leq \frac{2}{3}|E|$, and hence $z_P^* = \frac{2}{3}|E| = z_P(Y)$ and $z_D^* = \frac{2}{3}|E| = z_D(\zeta)$. Note that in the inequalities above, $z_P^* \leq z_D^*$ follows by weak

Table 3.2: Instances $G = (V, E)$ of strongly-regular graphs parameterized by $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t+1$ for some t, where $\frac{\text{Max-Cut}(G)}{|E|} \neq \frac{2}{3}$. The last column is the instance-specific approximation ratio that the GW algorithm achieves on each of these instances.

ID #	n	$\lvert E\rvert$	Max-Cut(G)	$\frac{\text{Max-Cut}(G)}{\lvert E\rvert}$	$\alpha_{G,\text{GW}}$
11	40	240	156	0.6500	0.9357
12	40	240	156	0.6500	0.9357
13	40	240	156	0.6500	0.9357
14	40	240	156	0.6500	0.9357
15	40	240	156	0.6500	0.9357
16	40	240	158	0.6583	0.9238
17	40	240	158	0.6583	0.9238
18	40	240	158	0.6583	0.9238
19	40	240	158	0.6583	0.9238
20	40	240	158	0.6583	0.9238
21	40	240	158	0.6583	0.9238
22	40	240	158	0.6583	0.9238
23	40	240	158	0.6583	0.9238

duality. [5] □

Next, we discuss the Max-Cut value that is achieved for the family of strongly-regular graphs parametrized by $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$ for some non-negative integer t. It is known that that there are a finite number of strongly-regular graphs with such parameters; these details can be found in Appendix A. A straightforward computer verification shows that for all but 13 instances in this family of strongly-regular graphs, the maximum cut contains exactly two-thirds of the number of edges. A list of exceptions can be found in Table 3.2; we remark that all the exceptions have 40 nodes (corresponding to $t = 3$) and that the maximum cut value is still close to (but not equal to) two-thirds of the number of edges. The verification code can be found online in the CI-QuBe library [104].

For the instances for which Max-Cut $= \frac{2}{3}|E|$ holds, a non-computer proof of this equality would be more satisfying; however, we were unable to find such a proof as of the writing of this thesis. We believe that such a non-computer proof and the other proofs in this subsection have the potential to be generalized to produce theorems similar to Theorem 12

[5] Unlike linear programs, strong duality (i.e. $z_P^* = z_D^*$) does not hold generally for SDP's; however, for problems that satisfy certian criteria known as Slater's conditions, strong-duality does hold [103].

below for other families of strongly-regular graphs.

Theorem 12. *Let G be a SRG(n, k, λ, μ) where $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$ for some non-negative integer t. If Max-Cut$(G) = \frac{2}{3}|E|$, then the GW algorithm achieves an instance-specific approximation ratio of 0.912.*

Proof. Let $G = (V, E)$ be a SRG(n, k, λ, μ) where $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$ for some non-negative integer t with Max-Cut$(G) = \frac{2}{3}|E|$. By Corollary 10, there exists an $r \leq n$ and $f : V \to \mathbb{S}^{r-1}$ so that $\arccos(f(u) \cdot f(v)) = \arccos(-1/3)$ for all $(u, v) \in E$; let $\theta = \arccos(-1/3)$. By Theorem 11, this mapping corresponds to an optimal solution Y to the GW SDP relaxation. Since Y is optimal with respect to the GW SDP, then by Proposition 5, the instance-specific GW approximation ratio for G is given by:

$$\frac{\frac{\theta}{\pi}|E|}{\text{Max-Cut}(G)} = \frac{\frac{\theta}{\pi}|E|}{\frac{2}{3}|E|} = \frac{3}{2} \cdot \frac{\theta}{\pi} = \frac{3}{2} \cdot \frac{\arccos(-1/3)}{\pi} \approx 0.912.$$

□

3.3 Empirical Results

In this section, we review classes of instances which we show are classically challenging (in some form) by empirical means. First, we show that empirically, mixed-weight graphs with roughly equal amounts of positively and negatively-weighted edges are the most difficult instances for classical heuristics; moreover, results in Chapter 5 show that QAOA also performs empirically poorly on similarly-constructed instances with mixed-weights. Next, we extend the small instances from Karloff's construction from Section 3.1 by considering perturbations of such instances by means of edge deletions and edge-weight perturbations. We show that, empirically, such perturbations cause an increase in the GW instance-specific approximation ratio; however, for many of these perturbed instances, the instance-specific approximation ratio is still bounded considerably away from 1.

3.3.1 Instances where Classical Heuristics Perform Poorly

In this subsection, we consider instances for which no classical heuristic quickly finds a near-optimal solution. More specifically, we analyze the data collected by Dunning et al. consisting of 38 Max-Cut heuristics and 3,296 instances [38]. In their work, machine learning was used in order to predict which heuristic would perform best on which instances depending on certain graph properties.

The data in their library (called MQLib) included the value of the cut obtained on every instance by each heuristic; however, it did not include the value of the Max-Cut for each instance which is needed in order to calculate the instance-specific approximation ratio for each graph. To rectify this, we used the semidefinite-based exact solver BiqCrunch to find the optimal solution for as many instances as possible [105]. For each instance, BiqCrunch was run for 24 hours; however, BiqCrunch could only find and verify the optimal cut for a subset of instances as seen in Figure 3.1.

In Dunning et al.'s work, every graph instance has an instance-specific allotted runtime and every heuristic is allowed to run up to that allotted runtime on that instance.[6] For nearly every instance (for which BiqCrunch found the optimal solution), there was at least one heuristic that would obtain at least 99.9% of the solution in 5% of the allotted runtime; there are 11 instances for which this was not the case. Details of these classically challenging instances, which we denote by $\mathcal{G}_{\text{CC}}$, can be found in Table 3.3. A Principle Component Analysis (PCA) was performed in order to see where $\mathcal{G}_{\text{CC}}$ lies in the larger *instance landscape*, see Figure 3.2. In the figure, the 11 instances of $\mathcal{G}_{\text{CC}}$ form two different clusters of sizes 2 and 9 instances.

The cluster of 9 instances have certain similar characteristics. In particular, these 9 instances have between 200 and 300 nodes and have both positive and negative edge weights

[6] In order to be meaningful across machines of varying performance, the (instance-dependent) allotted runtime was originally determined by the amount of time it took to generate 1500 random solutions and run the local search heuristic `callAllFirst1Swap` to local optimality. In the data provided by Dunning et al., the final allotted runtime (for the machines they used) is determined via a linear regression on the original allotted runtime and the number of nodes [38].

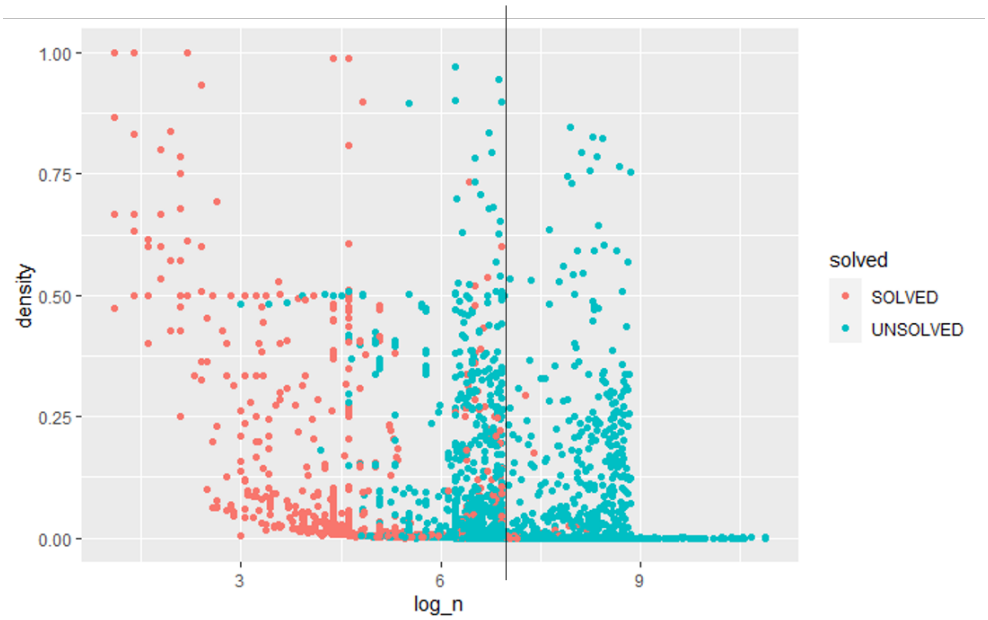

Figure 3.1: The plot shows the instances that the BiqCrunch solver solved exactly. Each instance is represented by a single point with the position dependent on the number of vertices, n, in the graph and the density of the graph. The logarithm in the plot is base-e and density is defined to be the ratio between the sum of the normalized absolute edge weights and the total number of possible edges in the graph, i.e.,the density is $\frac{\sum_{e \in E} |w_e|}{\binom{n}{2} \max_{e \in E} |w_e|}$, where w_e is the weight of edge e. Aside from some graphs with densities close to zero, the solver could only solve instances of up to 735 nodes (indicated by the black line on the figure). A time limit of 24 hours was imposed on the solver (excluding preprocessing).

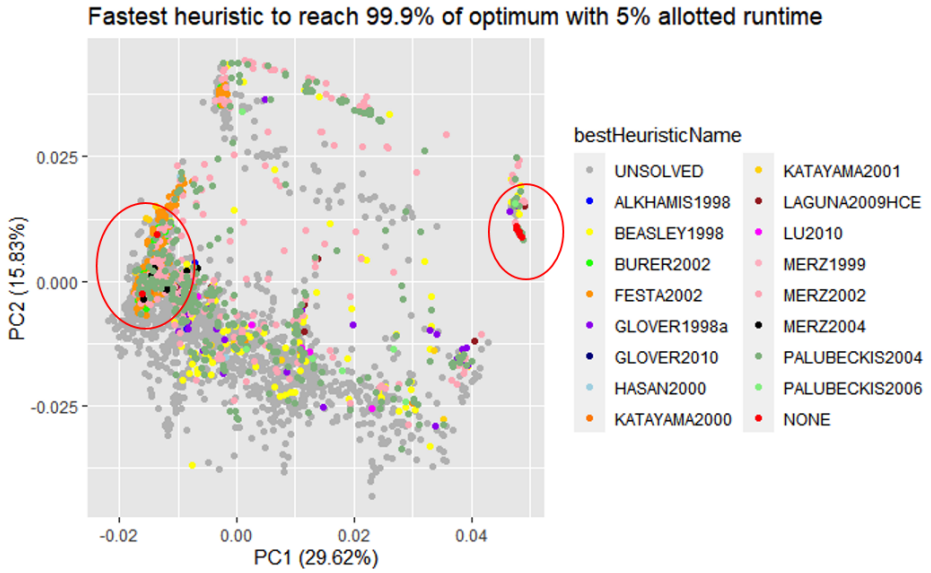

Figure 3.2: A PCA analysis of the MQLib library with respect to 58 graph properties. Gray dots represent instances that were not solved (exactly) by the BiqCrunch solver. The remaining non-red dots are colored based on which heuristic was first to obtain 99.9% of the optimal solution with 5% of the allotted runtime. The red instances (circled) correspond to $\mathcal{G}_{\text{CC}}$, instances for which no heuristic reached 99.9% of the optimal solution with 5% of the allotted runtime.

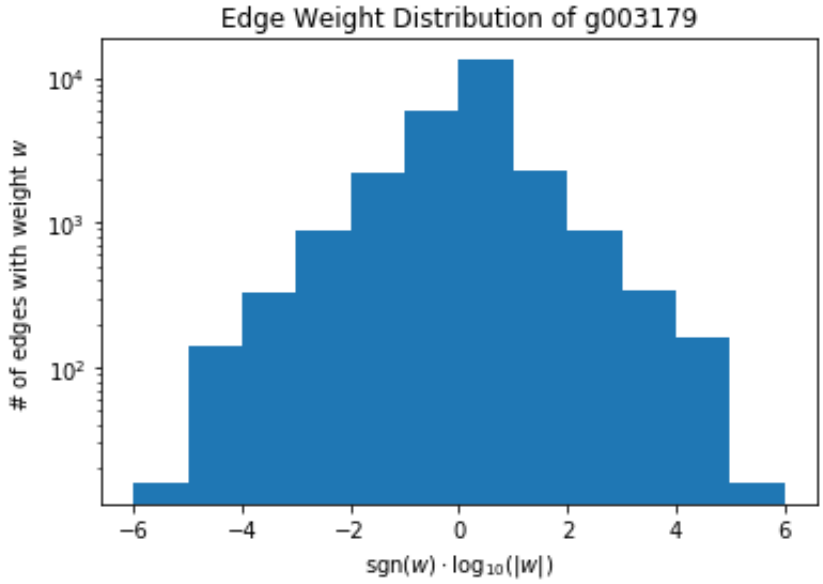

Figure 3.3: The above is the edge-weight distribution of instance g003179. As the edge weights are integers, we see that there are both positive and negative edge weights spanning several orders of magnitude; a similar edge weight distribution can be found for the other MQLib instances in Table 3.3 with the exception of instances g000435 and g001349.

spanning several orders of magnitude as seen in Figure 3.3. In regards to simulation of QAOA on a classical computer, these 9 instances are much too large; thus, in Chapters 4 and 5, we consider smaller instances ($|V| \leq 12$) with a similar edge weight distribution obtained by having weights of the form $w_e = \pm 2^k$ where $\Pr(w_e = 2^k) = \Pr(w_e = -2^k) = 2^{-k-2}$ for each non-negative integer k. In these future chapters, we find that such instances are also challenging for QAOA and its variants.

3.3.2 Extensions of Karloff's Construction

Even with the proven conjecture by Brouwer et al. [93], the construction by Karloff only yields 6 small instances under 1000 nodes where the GW algorithm performs poorly. In order to obtain more such instances, we propose perturbing the instances constructed by Karloff in one of two ways: (1) removing edges and (2) perturbing edge weights.

For the first method, let $G = J(m, m/2, b)$ be an instance using Karloff's construction (with m even and $b < m/4$) and let $G' := G - F$ for some subset $F \subseteq E$ of edges in

Table 3.3: The table includes all the instances in the MQLib library for which no MQLib heuristic obtained 99.9% of the optimal solution within 5% of the alloted runtime (abbreviated as R.T. throughout the table) described in Footnote 1. In the first part of the table, the columns correspond to the instance name (as given in the MQLib library), the number of nodes and (nonzero-weight) edges, the allotted runtime (in seconds), the instance-specific GW approximation ratio $\alpha_{\mathrm{GW},G}$, the time needed to run the GW solver in seconds, whether the instance has negative-weighted edges, and the range of edge-weight magnitudes (defined as $\max_{e\in E}\log(|w_e|) - \min_{e\in E}\log(|w_e|)$). In the second part of the table, we find the best approximation ratio obtained by a MQLib heuristic that is allowed to run for only 5% of the alloted runtime; the second and third columns correspond to this ratio and the corresponding heuristic (with ties between heuristics being broken by runtime). The next two columns are obtained the same way but with the full allotted runtime being allowed. All ratios in the table are written with the denominator being the value of the Max-Cut for that instance. The GW algorithm times and the MQLib heuristic times were computed on different machines so these times are incomparable to one another.

Instance G	Nodes	Edges	R.T. (s)	$\alpha_{\mathrm{GW},G}$	GW Time (s)	Neg. Weights	Magnitude Range
g000330	300	14011	176	$\frac{8334779.65}{8493173} = 98.14\%$	359.34525442123413	Yes	5.308
g000417	300	34753	176	$\frac{8828162.87}{9102033} = 96.99\%$	321.5829267501831	Yes	5.388
g000435	420	619	247	$\frac{56964.35}{58537} = 97.31\%$	880.8003544807434	No	2.581
g000572	200	18246	120	$\frac{6574563.33}{6795365} = 96.75\%$	250.95755791664124	Yes	5.365
g000723	200	18230	120	$\frac{5307502.47}{5568272} = 95.32\%$	130.70434188842773	Yes	5.316
g000762	250	26522	147	$\frac{7636025.14}{7919449} = 96.42\%$	409.95273995399475	Yes	5.443
g001349	640	960	377	$\frac{97842.93}{101723} = 96.19\%$	838.4398355484009	No	0.566
g001361	250	12031	147	$\frac{6330851.05}{6418276} = 98.64\%$	192.36800813674927	Yes	5.267
g002581	200	18237	120	$\frac{6102967.87}{6294701} = 96.95\%$	249.02441000938416	Yes	5.380
g002935	300	13983	176	$\frac{8069149.92}{8242904} = 97.89\%$	235.74102520942688	Yes	5.342
g003179	250	26534	147	$\frac{6391628.70}{6596797} = 96.89\%$	169.3680293560028	Yes	5.263

Instance G	Best Heuristic, 5% R.T.	Best Ratio, 5% R.T.	Best Heuristic, 100% R.T.	Best Ratio, 100% R.T.
g000330	PALUBECKIS2004bMST2	$\frac{8479772}{8493173} = 99.8422\%$	PALUBECKIS2004bMST2	$\frac{8485022}{8493173} = 99.9040\%$
g000417	PALUBECKIS2004bMST2	$\frac{9079073}{9102033} = 99.7477\%$	PALUBECKIS2004bMST2	$\frac{9086298}{9102033} = 99.8271\%$
g000435	FESTA2002GPR	$\frac{58431}{58537} = 99.8189\%$	MERZ2004	$\frac{58498}{58537} = 99.9334\%$
g000572	PALUBECKIS2004bMST2	$\frac{6784784}{6795365} = 99.8443\%$	PALUBECKIS2004bMST2	$\frac{6795365}{6795365} = 100.0000\%$
g000723	PALUBECKIS2004bMST2	$\frac{5560562}{5568272} = 99.8615\%$	HASAN2000GA	$\frac{5566434}{5568272} = 99.9670\%$
g000762	PALUBECKIS2004bMST2	$\frac{7909253}{7919449} = 99.8713\%$	PALUBECKIS2004bMST2	$\frac{7917221}{7919449} = 99.9719\%$
g001349	BURER2002	$\frac{101579}{101723} = 99.8584\%$	MERZ2004	$\frac{101723}{101723} = 100.0000\%$
g001361	MERZ1999GLS	$\frac{6411845}{6418276} = 99.8998\%$	PALUBECKIS2004bMST2	$\frac{6413509}{6418276} = 99.9257\%$
g002581	PALUBECKIS2004bMST2	$\frac{6287070}{6294701} = 99.8788\%$	PALUBECKIS2004bMST2	$\frac{6294701}{6294701} = 100.0000\%$
g002935	PALUBECKIS2004bMST2	$\frac{8231686}{8242904} = 99.8639\%$	PALUBECKIS2004bMST2	$\frac{8239573}{8242904} = 99.9596\%$
g003179	MERZ1999GLS	$\frac{6583762}{6596797} = 99.8024\%$	PALUBECKIS2004bMST2	$\frac{6595245}{6596797} = 99.9765\%$

G. Observe that by removing the edges in F from the optimal cut in G, then this yields a cut for G' and the number of edges across the cut has decreased by at most $|F|$; a Max-Cut for G' could possibly have even more edges across. In other words, Max-Cut$(G') \geq$ Max-Cut$(G) - |F|$. We can thus bound the approximation ratio for G' from above in terms of G as follows:

$$\alpha_{\text{GW},G'} = \frac{\text{GW}(G')}{\text{Max-Cut}(G')} \leq \frac{\text{GW}(G')}{\text{Max-Cut}(G) - |F|}. \tag{3.1}$$

The numerator of the bound $\frac{\text{GW}(G')}{\text{Max-Cut}(G)-|F|}$ can be quickly computed by equations in Section 2.3 and the denominator can also be quickly computed since Karloff found a closed-form analytical expression for the optimal cut of the instances he constructed:

$$\text{Max-Cut}(J(m, m/2, b)) = \frac{n}{2}\binom{m/2}{b}^2 \left[1 - \frac{2b}{m}\right], \tag{3.2}$$

where m is even and $b < m/4$. In Figure 3.4, we show the results of modifying $J(6, 3, 1)$, $J(8, 4, 1)$, $J(10, 5, 1)$, and $J(10, 5, 2)$ via edge deletions (up to 4 edges).

For the second method of finding instances similar to those using Karloff's construction, we consider perturbations of the edge weights. Again, let $G = J(m, m/2, b)$ for m even and $b < m/4$. In the original construction by Karloff, the edges of G are of unit weight; the modified instances of G have the same edges but the weights are sampled from a normal distribution with mean 1 and standard deviation σ. These modified instances were made with varying values of σ (in particular, $\sigma = 0.01, 0.1$, and 1). The results of modifying $J(6, 3, 1)$, $J(8, 4, 1)$, $J(10, 5, 1)$, and $J(10, 5, 2)$ in this way are shown in Figure 3.4.

We have also calculated the GW approximation ratio of the 11 interesting MQLib instances discussed in Section 3.3.1. For each of these MQLib instances, the GW algorithm achieves between 95% and 99% of the optimal cut which is much higher compared to the approximation ratio of the instances made via Karloff's constructions (including some of the modified instances).

Although we found interesting instances by modifying instances made by Karloff's

construction, it is interesting to note that there were also several modified instances whose approximation ratio is close to 1, even when the modifications are small. In other words, the result of the GW algorithm can sometimes be very sensitive to small changes in the input graph.

3.4 QAOA's Performance on Challenging Instances for the GW Algorithm

Due to their construction, the GW algorithm performs poorly on the instances found using Karloff's construction; one may hope that quantum computing may have some advantage over these instances which are, in a sense, classically difficult. We prove in Theorem 15, that depth-1 QAOA achieves a lower instance-specific approximation ratio for the instances that arise from Karloff's construction; this implies that, for these instances, either a higher circuit depth or some modification of the QAOA algorithm (such as those presented in the later chapters of thesis) would be needed in order to demonstrate some form of quantum advantage (over the GW algorithm).

To first make the sequence of graphs considered more precise, we consider the family of Karloff instances $G_m = J(m, m/2, b)$ where m is even and $b = \lceil \frac{1}{4}(\cos(\theta^*) + 1)m \rceil$ and $\theta^* = \text{argmin}_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1-\cos\theta} \approx 2.33112$; recall from Section 3.1 that this choice of b is (near) optimal in the sense that, for fixed m, the corresponding graph will have the worst possible approximation ratio with respect to the GW algorithm. Some calculations give us that $b \approx \lceil 0.077m \rceil$. We will assume that $m \geq 12$, and hence one can prove that $0 \leq b < \frac{m}{6}$. It follows from Karloff [2] that the expected cut value of G_m approaches $\alpha^* = 0.878$ as $m \to \infty$.

We first prove two helpful lemmas. In the first lemma, we prove that the degree of G_m is $\binom{m/2}{b}^2$ and in the next lemma, we prove that for m large enough, the graphs G_m are triangle free.

Lemma 13. *The degree of each vertex in G_m is $\binom{m/2}{b}^2$.*

Proof. Let us now determine the degree of G_m. Let S be a vertex of G_m. To construct a

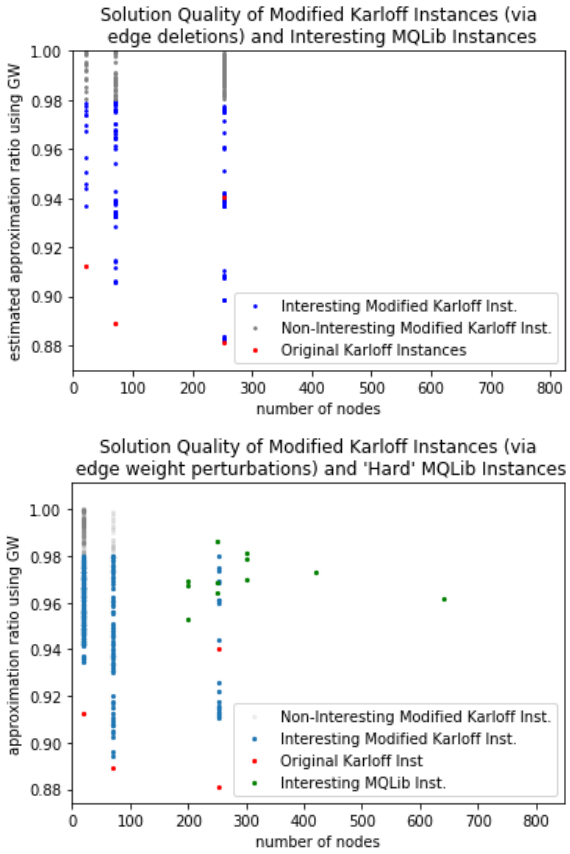

Figure 3.4: Both plots show the result of the GW algorithm when modifying the instances made via Karloff's construction. In particular, we consider modifications of $J(6,3,1), J(8,4,1), J(10,5,1)$ and $J(10,5,2)$. The top plot shows the estimated approximation ratio using the GW algorithm for instances modified via edge deletions (with up to 4 edges removed) where the estimated approximation ratio for an instance G' is calculated as $\frac{\text{Round}(G',Y')}{\text{Max-Cut}(G)-|F|}$ where G is the corresponding (unmodified) original instance from Karloff's construction and $|F|$ is the number of edges deleted (as seen in Equation 3.1); the actual approximation ratios could possibly be lower for each instance. The bottom plot shows the approximation ratio using the GW algorithm for instances modified via edge weight perturbations (as described in Section 3.3.2). The bottom plot also includes the GW approximation ratio for the interesting instances in the MQLib library (found in Section 3.3.1).

neighbor T, we must pick b elements from the $m/2$ elements in S for T for the overlap, then from the remaining $m/2$ elements in $[m] \setminus S$, we must pick $m/2 - b$ elements to determine the remaining elements in T. Thus, the neighbor of ways to construct a neighbor for S are,

$$\deg(S) = \binom{m/2}{b}\binom{m/2}{m/2-b} = \binom{m/2}{b}^2.$$

□

Lemma 14. *For all $m \geq 12$, G_m is triangle-free.*

Proof. As $m \geq 12$, we have that $0 \leq b < m/6$ as determined previously.

Now, for a given edge (S, T) in G_m, let us count how many ways we can construct a mutual neighbor U of both S and T. Let k be the number of elements in U that are also contained in $S \cap T$. Since $|S \cap T| = b$, there are $\binom{b}{k}$ ways to pick such k elements for U. Now, $|S \cap U| = b$, we need to pick $b - k$ more elements from the remaining $m/2 - b$ elements of S to add to U; there are $\binom{m/2-b}{b-k}$ ways to do this. Similarly, there are $\binom{m/2-b}{b-k}$ ways to choose the elements of T (not in $S \cap T$) that also belong in U. Lastly, there are $m/2 - (k + (b-k) + (b-k)) = m/2 - 2b + k$ remaining elements we need to add to U to ensure it has a total of $m/2$ elements and these can be selected from the $|[m] \setminus (S \cup T)| = m - (m/2 + m/2 - b) = b$ elements outside of $S \cup T$; there are $\binom{b}{m-2b+k}$ ways to do this. Summing over the possible choices of $k = 0, \ldots, b$, we have that the total number of mutual neighbors between S and T are

$$\sum_{k=0}^{b} \binom{b}{k}\binom{m/2-b}{b-k}^2\binom{b}{m/2-2b+k}.$$

We claim that the above sum is zero. To see this, consider the term $\binom{b}{m/2-2b+k}$ in the sum above. Observe that, for $k \geq 0$ and since $0 \leq b < m/6$, we have

$$m/2 - 2b + k \geq m/2 - 2b > (6b)/2 - 2b = b$$

and hence the $\binom{b}{m/2-2b+k}$ term is zero, making the whole sum zero.

□

Finally, we prove that as m increases, the approximation ratio that depth-1 (standard) QAOA achieves on G_m approaches the constant 0.592.

Theorem 15. *Let $F_p(G)$ denote the expected cut value obtained from depth-p (standard) QAOA when the optimal choice of variational parameters (γ, β) are used. Then*

$$\lim_{m\to\infty} \frac{F_1(G_m)}{\textit{Max-Cut}(G)} = 0.592.$$

Proof. By [99], since G_m is triangle-free (for $m \geq 12$) due to Lemma 14, we have that depth-1 QAOA achieves the following expected cut value when the optimal parameters, i.e.,

$$(\gamma, \beta) = (\arctan(1/\sqrt{d-1}), \pi/8),$$

are used:

$$F^*(G_m) = \frac{|E|}{2}\left(1 + \frac{1}{\sqrt{d}}\left(\frac{d-1}{d}\right)^{(d-1)/2}\right);$$

here, d is the degree of the (regular) graph G_m, which is $d = \binom{m/2}{b}^2$ (Lemma 13).

From Equation 3.2, we know that since $0 \leq b < m/6 < m/4$, we have that the maximum cut is given by

$$\text{Max-Cut}(G_m) = \frac{n}{2}\binom{m/2}{b}^2\left(1 - \frac{2b}{m}\right).$$

Thus, the approximation ratio for depth-1 standard QAOA on G_m (with $m \geq 12$) goes to,

$$
\begin{aligned}
&\lim_{m\to\infty} F^*(G_m)/\text{Max-Cut}(G_m) \\
&= \lim_{m\to\infty} \frac{\frac{|E|}{2}\left(1+\frac{1}{\sqrt{d}}\left(\frac{d-1}{d}\right)^{(d-1)/2}\right)}{\frac{n}{2}\binom{m/2}{b}^2\left(1-\frac{2b}{m}\right)} \\
&= \lim_{m\to\infty} \frac{\frac{nd/2}{2}\left(1+\frac{1}{\sqrt{d}}\left(\frac{d-1}{d}\right)^{(d-1)/2}\right)}{\frac{n}{2}\binom{m/2}{b}^2\left(1-\frac{2b}{m}\right)} && \text{(as } G_m \text{ is } d\text{-regular)} \\
&= \lim_{m\to\infty} \frac{\frac{nd/2}{2}}{\frac{n}{2}\binom{m/2}{b}^2\left(1-\frac{2b}{m}\right)} && \left(\left(\tfrac{d-1}{d}\right)^{(d-1)/2} \to e^{-1/2} \text{ as } d\to\infty\right) \\
&= \lim_{m\to\infty} \frac{\frac{nd/2}{2}}{\frac{n}{2}d\left(1-\frac{2b}{m}\right)} && \left(\text{as } d=\tbinom{m/2}{b}^2\right) \\
&= \lim_{m\to\infty} \frac{1}{1+1-\frac{4b}{m}} && \text{(simplify)} \\
&= \frac{1}{1-\cos(\theta^*)} && \left(\text{as } \cos(\theta^*)=\tfrac{4b}{m}-1 \text{ as } m\to\infty\right) \\
&= \frac{\pi}{2}\frac{\alpha^*}{\theta^*} && \left(\text{as } \alpha^*=\tfrac{2}{\pi}\tfrac{\theta^*}{1-\cos\theta^*}\right) \\
&\approx 0.592.
\end{aligned}
$$

In the above calculations we use that as $m \to \infty$, then $b \to \infty$ and we have that (for m large enough)[7], $d = \binom{m/2}{b} \geq \binom{m/2}{1} = m/2 \to \infty$. Note that all of the calculations above also hold true (including the triangle-free result) if we instead consider $b = \lfloor 0.077m \rfloor$ instead of $b = \lceil 0.077m \rceil$ in the construction of G_m. □

The theorem above indicates that a higher circuit depth and/or a modification to QAOA is needed in order to demonstrate quantum advantage (over the GW algorithm) for the sequence of graphs G_m. In the case of the family of strongly-regular graphs considered in Section 3.2 (parameterized by $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$ for some non-negative integer t), the proof used in Theorem 15 can not be used the proof depends on the graphs being triangle-free and the family of strongly-regular graphs that we consider

[7] One can prove that b is strictly positive when $m \geq 12$. If we instead take $b = \lfloor 0.077m \rfloor$, then b is strictly positive whenever $m \geq 14$.

are not triangle-free (as $\lambda = 2 \neq 0$).

CHAPTER 4

WARM-STARTS FOR QAOA USING STANDARD MIXERS

In this chapter[1], we will discuss the methods for generating warm-starts for the QAOA algorithm; when the standard mixer is used with such warm-starts, we refer to such a variant of QAOA as QAOA-warm. As we will see, the performance of QAOA-warm quickly plateaus as the circuit depth is increased; in the next chapter, we explore a way of modifying the mixing Hamiltonian in a way that depends on the warm-started state to avoid such plateaus.

In what follows, we provide a general framework for constructing initial quantum states, we then provide specific examples of initialization schemes and discuss their properties (including guarantees they obtain at depth-0 QAOA-warm). Afterwards, we discuss the limitations of QAOA-warm and discuss theoretical properties in the case where the relaxation itself effectively solves the problem. Experimental results are provided in the last two sections; these results are consistent with the theoretical results found in the chapter.

Although our results in this chapter for primarily for QAOA-warm; we do discuss some warm-start initializations schemes in the context of other mixers (including the custom mixer of QAOA-warmest discussed in the next chapter). For this reason, Table 4.3 is provided to aid in summarizing what results are known for different combinations of QAOA mixers and warm-start initializations; the table can be found towards the end of the chapter.

4.1 Framework for Constructing Initial Quantum States

In this section, we discuss the general framework for creating a initial product state using classical methods. We summarize the process in three steps:

1. Obtain $\mathbf{x}^* : \mathbf{V} \to \mathbb{S}^{\mathbf{k-1}}$ for $k \in \{2, 3\}$ via solving some classical problem,

[1]The results presented in this chapter were published in ACM Transactions on Quantum Computing [3].

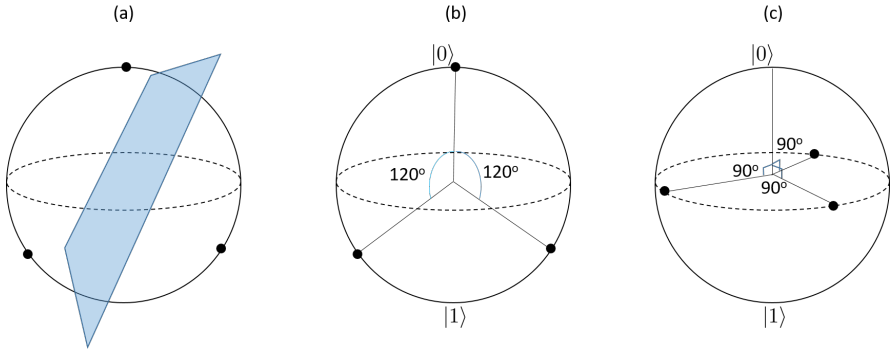

Figure 4.1: Comparison of the hyperplane rounding and quantum sampling for a 3-cycle (Max-Cut=2): figure (a) shows a local optimal BM-MC$_3$ solution, where any random hyperplane will give a cut of size 2. Both (b) and (c) show two different embeddings of the BM-MC$_3$ solution (from (a)) onto the Bloch sphere. In (b), the qubits lie on $x = 0$ plane and quantum sampling results in a expected cut of 1.875. In (c), all qubits lie on the equator of the Bloch sphere (similar to the standard start of QAOA), so each edge has a probability of $1/2$ of being cut, yielding a total expected cut of 1.5. Both (b) and (c) demonstrate that the orientation of the rotated BM-MC$_3$ solution is important when embedding it into the Bloch sphere and can result in different expected cuts.

2. Perform a global randomized rotation on $\mathbf{x}^*$,

3. Map the rotated solution to the Bloch sphere to obtain a quantum product state .

In the above, we refer to k as the dimension of the solution and say that the solution is k-dimensional. There are numerous ways of accomplishing the first step; we discuss various possibilities in Section 4.2. We discuss the details of the random rotation and quantum mapping in Sections 4.1.1 and 4.1.2 respectively.

4.1.1 Random Rotations

Classical hyperplane rounding of $\mathbf{x}^* : \mathbf{V} \to \mathbb{S}^{\mathbf{k-1}}$ is invariant under a global rotation of the entire solution, however quantum sampling is not as it has a fixed axis along which measurements are performed. For example, in Figure 4.1 we consider 3 rotations of a particular BM-MC initialization (to be discussed in Section 4.2.1) of 3 qubits on the Bloch sphere, and though hyperplane rounding is agnostic to a rotation of the Bloch sphere, quantum sampling depends on the choice of the measurement axis. The difference in instance-specific approximation ratio attained by quantum sampling in two different orientations of the same

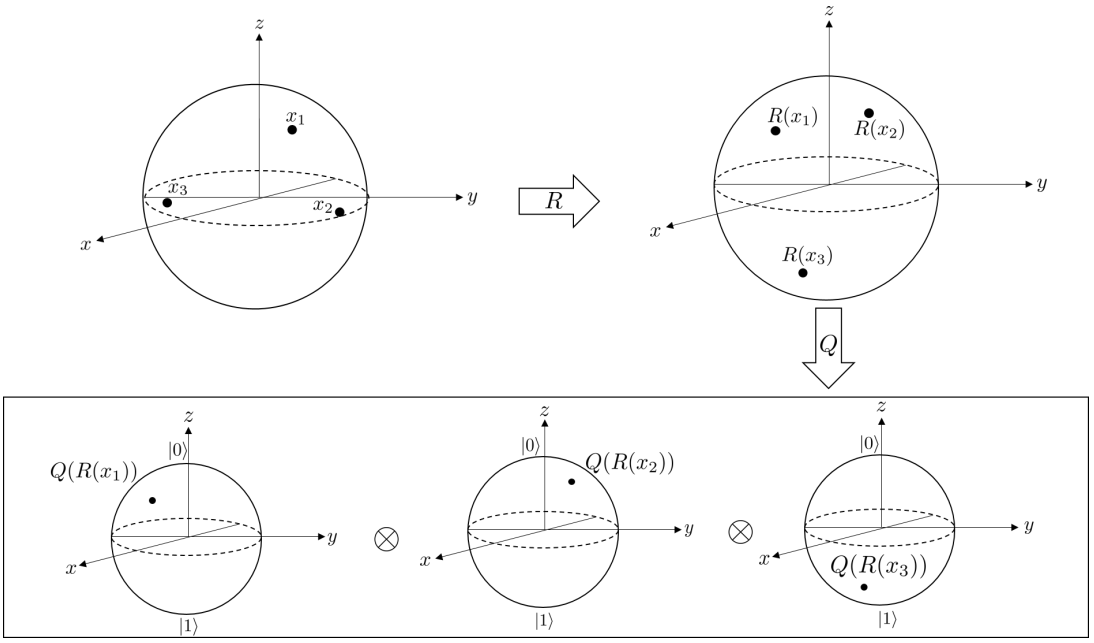

Figure 4.2: We begin with the classically obtained x^*. We then apply a rotation $R \in \{R_U, R_V\}$; here we show R_U being applied (top-right). Lastly, we use Q to map this rotated solution to a quantum product state.

solution on Bloch sphere demonstrates the importance of choosing a suitable rotation when embedding the classically obtained solution x^* to the Bloch sphere. Thus, before mapping x^* to the Bloch sphere, a global rotation is performed to mitigate unfavorable orientations due to warm-starts.

We consider two types of random rotation schemes: uniform rotation in $\mathbb{R}^k$ (for all the vertices), and random "vertex-at-top" rotations where a vertex is sampled uniformly and mapped to the $(0, 0, 1)^T$ vector for rank-3 and $(1, 0)^T$ in 2-dimensional solutions. Uniform random rotations can provably recover a significant fraction of the BM-MC$_k$ objective (see Section 4.2.1) whereas vertex-at-top rotations serve as a useful heuristic. We use the shorthand $R_V(\mathbf{x}^*)$ and $R_U(\mathbf{x}^*)$ to denote the rotations of the approximate solution $\mathbf{x}^*$ by a random vertex-at-top rotation R_V and a random uniform rotation R_U respectively. We now discuss the details of each rotation scheme below.

Uniform Rotation

In this case, we uniformly pick a rotation of the $(k-1)$-dimensional sphere and apply it to the k-dimensional solution x^*. For 3-dimensional solutions, one way to accomplish this is by picking a point $\hat{x}$ uniformly at random from the surface of the sphere, rotating $\hat{x}$ to the top of the sphere (in a way similar to the vertex-at-top rotations), and then performing a uniform random rotation in $[0, 2\pi]$ around the z-axis. Such an $\hat{x}$ can be generated by picking α, β uniformly at random from the interval $[0, 1]$ and then setting $\varphi = 2\pi\alpha$ and $\theta = \arccos(2\beta - 1)$. The pair (θ, φ) will then correspond to the polar and azimuthal angles of the randomly chosen point $\hat{x}$ on the surface of the sphere [106].

For 2-dimensional solutions, we can simply shift all the angles by some random angle. More precisely, set $\theta_j = \theta_j + \hat{\theta}$ where θ_j denotes the angle of the point corresponding to the jth vertex in the 2-dimensional solution and $\hat{\theta}$ is chosen uniformly at random in $[0, 2\pi]$.

It should be noted that the angle of the final rotation about the z-axis does influence the distribution of cuts (and expected cut value) obtained via QAOA-warm; however, we will see in the next chapter that QAOA-warm's successor, QAOA-warmest, is *not* influenced by this rotation (Theorem 14).

Vertex-at-Top Rotation

We first describe the rotation in 3-dimensions for vertex

$$v_i = (\sin\theta_i \cos\varphi_i, \sin\theta_i \sin\varphi_i, \cos\theta_i)^T,$$

which is sampled uniformly at random (for $i \in [n]$). The rotation that maps $v_i \in \mathbb{R}^3$ to $(0, 0, 1)^T$ is obtained by first rotating clockwise along the z-axis by φ_i, followed by a clockwise rotation along the y-axis by θ_i, followed by a uniform at random rotation μ in $[0, 2\pi]$ around the z-axis.

Indeed, one can check that $R_V(\mathbf{x}^*(v_i)) = (0, 0, 1)^T$ (which will correspond to the quan-

tum state $|0\rangle$ on the Bloch Sphere). For 2-dimensional solutions, with a uniform at random vertex $v_i = (\cos(\theta_i), \sin(\theta_i))^T$ sampled from $i \in [n]$, we can simply work with polar coordinates and shift all polar angles by θ_i to obtain the random vertex-at-top rotation. To be precise, we set $\theta_j = \theta_j - \theta_i$ where θ_j denotes the angle of the point corresponding to the jth vertex in the 2-dimensional solution.

4.1.2 Mapping to Bloch Sphere

To map the rotated solutions $R(\mathbf{x}^*) = ((\theta_1, \varphi_1), \ldots, (\theta_n, \varphi_n))$ (with $R \in \{R_U, R_V\}$), we can simply map the 3-dimensional solutions for each vertex to the Bloch sphere (see Figure 4.2) using a tensorizable state for each qubit, i.e., the "quantum mapping" Q is given by:

$$Q(\mathbf{x}^*) = Q_3(\theta_1, \varphi_1) \otimes \cdots \otimes Q_3(\theta_n, \varphi_n),$$

where

$$Q_3(\theta, \varphi) = \cos(\theta/2)\,|0\rangle + e^{i\varphi} \sin(\theta/2)\,|1\rangle\,.$$

For 2-dimensional solutions, let $R(\mathbf{x}^*) = (\theta_1, \ldots, \theta_n)$ be the rotated approximate solution in polar coordinates where $\theta_i \in [0, 2\pi)$ for $i = 1, \ldots, n$. We embed the solution into the yz-plane of the Bloch sphere with the following quantum mapping:

$$Q(\mathbf{x}^*) = Q_2(\theta_1) \otimes \cdots \otimes Q_2(\theta_n),$$

where $Q_2(\theta)$ is given by:

$$Q_2(\theta) = \begin{cases} \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{-i\pi/2} \sin\left(\frac{\theta}{2}\right)|1\rangle\,, & \theta \in [0, \pi), \\ \cos\left(\pi - \frac{\theta}{2}\right)|0\rangle + e^{i\pi/2} \sin\left(\pi - \frac{\theta}{2}\right)|1\rangle\,, & \theta \in [\pi, 2\pi). \end{cases}$$

The quantum mapping for 2-dimensional solutions is motivated by the fact that for 3-dimensional solutions, certain initializations along the x-axis cause QAOA-warm to per-

form poorly (see Section 4.3)); mapping to the yz-plane of the Bloch sphere allows us to avoid these problematic states.

4.2 Initialization Schemes for Warm-Start State

4.2.1 k-dimsional Burer-Monteiro

Recall from Section 2.3 that k-dimensional Burer-Monteiro (BM-MC$_k$) solutions are the result of optimizing a relaxation of the Max-Cut that relaxes each vertex variable to a k-dimensional unit vector. Unlike the GW SDP, such a relaxation does not yield a convex program and thus, it is intractable to find a globally optimal solution and so, for the purposes of warm-starting the QAOA algorithm, we consider locally optimal solutions of the BM-MC$_k$ objective.

Implementation Details of BM-MC$_k$ Optimization

We begin with n points chosen uniformly at random on the unit circle (for $k = 2$) or unit sphere (for $k = 3$). We represent these points in polar coordinates (for $k = 2$) or spherical coordinates (for $k = 3$); that is, we keep track of the polar (θ) angles (for $k = 2, 3$) and azimuthal (ϕ) angles (for $k = 3$) of each point. To find locally optimal solutions, we perform stochastic coordinate ascent[2] by making small random perturbations to these angles (thus maintaining feasibility) and update our solution if the objective increases.

In Algorithm 1, we describe our implementation for obtaining the semidefinite programming (SDP) solution for BM-MC$_k$ for $k = 2, 3$ using coordinate ascent. In the algorithms below, we write $U(a, b)$ to denote the uniform distribution on the interval $[a, b]$ (where $a, b \in \mathbb{R}$ with $a < b$). We set $\eta = 1/20$ for experiments in this work. We normalize the angles output by BM-MC$_3$ to enforce the standard range of angles for spherical coordinates without changing the objective value.

[2] *Stochastic coordinate ascent* works well in practice in finding a local optimum, see e.g., [77]. Nevertheless, for guaranteed convergence one can use other methods such as (fast) Riemannian Trust-Region methods.

Algorithm 1: Obtain Solution for BM-MC$_k$

Input: Weighted graph $G = (V, E), w : E \to \mathbb{R}, k \in \{2, 3\}$

If $k = 2$, let $\theta_1, \ldots, \theta_n \in \mathbb{R}$ be the angles of n points chosen uniformly at random on the 2-dimensional unit circle. If $k = 3$, let $(\theta_1, \phi_1), \ldots, (\theta_n, \phi_n)$ be the spherical coordinates of n points chosen uniformly at random on the 3-dimensional sphere.

repeat

  for $i = 1$ *through* n **do** `// coordinate ascent`

    Sample the perturbation value(s) $\Delta\theta$ (and $\Delta\phi$ if $k = 3$) from $U(-\eta, \eta)$ for small $\eta > 0$.

    Update $\theta_i = \theta_i + \Delta\theta$ (and $\phi_i = \phi_i + \Delta\phi$ if $k = 3$) and compute the BM objective.

    If the objective improves, keep the perturbation.

  end

until *no improvement in objective by* $\geq 10^{-5} \sum_{e \in E} |w_e|$ *within 100 evaluations.*

The algorithm above is repeated 5 times to obtain 5 local optima; we then keep the best one (with respect to the BM-MC$_k$ objective) and let t $\mathbf{x}^* : V \to S^{k-1}$ denote this solution.

Motivation for BM-MC$_k$ Warm-Starts

To motivate such an approach for creating warm-starts for QAOA, we highlight two key observations. First, since the objective of BM-MC$_k$ can be written as $\max_{x_i, x_j \in S_{k-1}} \sum_{(i,j) \in E} w_{ij} \|x_i - x_j\|_2^2$, the classical solutions are incentivized to move the adjacent vertices as far apart as possible, ideally, to opposite ends of the sphere. This helps increase the probability of an edge being in a cut obtained not only by hyperplane rounding but also quantum sampling (as long as the corresponding qubits are aligned with the measurement axis as much as possible, i.e. at the north and south poles of the Bloch sphere). In general, if there is a cluster of vertices at both the poles of the sphere, then the probability of capturing the weight of the edges that go across these clusters is increased for both classical and quantum approaches.

Next, we find a reduction to the quantum sampling objective from the BM-MC objective for an edge. Consider an edge e, such that one of the vertices is located at the top of Bloch sphere. Then the probability of that edge being cut via quantum sampling and the

contribution that edge makes to the BM-MC$_3$ objective coincides. Consider an edge $e = (i, j)$ such that $x_i = (0, 0, 1)^T$, and $x_j = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)^T$ (where θ and ϕ are the polar and azimuthal angle respectively). The expected contribution of e to the Max-Cut from quantum sampling is equal to $w_{i,j}$ multiplied by the probability that the edge e is cut, i.e., $w_{i,j}\sin^2(\theta/2)$. The contribution to the BM-MC$_3$ objective from edge e can be written as $\frac{1}{2}w_{i,j}(1 - x_i \cdot x_j)$. By definition, $\cos(\theta) = x_i \cdot x_j$, and thus, the contribution to the BM-MC$_3$ objective is $\frac{1}{2}w_{i,j}(1 - \cos(\theta)) = w_{i,j}\sin^2(\theta/2)$, which is equivalent to the expected contribution of e from quantum sampling[3].

Performance: Hyperplane Rounding vs Quantum Measurement

A natural question at this point is if there is any improvement in cut quality when applying QAOA-warm to the warm-start initialization compared to simply performing hyperplane rounding on said warm-start, and if quantum sampling of a classical solution is even competitive compared to a hyperplane rounding of the same. We show in Figure 4.3 that quantum sampling of the warm-start (QS1) initialization empirically outperforms the expected cut obtained using the standard initial state for QAOA (QS2). Moreover, with an appropriate initial rotation of the warm-start (Section 4.1.1), QS1 outperforms hyperplane rounding (HR) for the majority of instances. By interpreting QS1 and QS2 as being the result of depth-0 QAOA-warm and standard QAOA respectively, the results motivate our approach in the hopes that QAOA-warm outperforms standard QAOA (and the GW algorithm) for circuit depths $p > 0$ as well.

Depth-0 Performance Guarantees for BM-MC$_k$

We had just seen that quantum sampling a mapped BM-MC$_k$ solution has the potential to yield solutions that, often, are as good or better than that obtained from hyperplane rounding of the same solution. We next show that we can in fact provably guarantee that at

[3]This may not be true in general for the Max-Cut over the entire graph, due to alignment with the measurement axis.

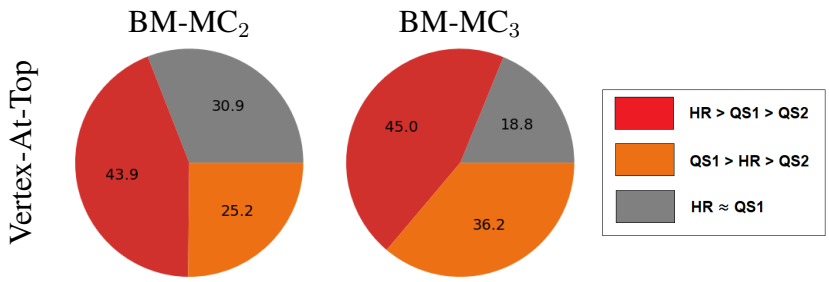

Figure 4.3: Pie charts representing best expected cut value (expectation over randomness in sampling) obtained by using (i) hyperplane rounding of the BM-MC$_k$ solution (HR), (ii) quantum sampling of the BM-MC$_k$ solution (QS1), and quantum sampling of the initial state of standard QAOA (QS2). For every instance, QS2 always yielded the worst result of the three, and for majority of the instances QS1 $\geq$ HR. For HR and QS1, the best of 5 (in terms of SDP objective) locally optimal BM-MC$_k$ solutions are used; for that solution, the best of 5 rotations is used for QS1. The regions marked in gray indicate instances for which QS1 and HR had a tie (difference in instance-specific approximation ratio of at most 0.001).

least a certain fraction of the optimal cut value is preserved when moving from the classical to the quantum regime. More precisely, we prove that, at depth $p = 0$ (i.e. before any gates are applied beyond initialization), QAOA-warm on graphs with non-negative edge-weights achieves at least 0.75κ and 0.66κ approximations for the Max-Cut when using a κ-close (i.e., BM-MC$_k(x^*) \geq \kappa$Max-Cut(G)) BM-MC$_2$ and BM-MC$_3$ solution respectively; for $\kappa > 2/3$ and $\kappa > 3/4$ (for rank-2 and rank-3 respectively), this results in an improvement from the $1/2$-approximation provided by standard QAOA at $p = 0$.

Theorem 16. *Let G be a graph with non-negative edge weights. If $\mathbf{x}$ is a κ-close solution to* BM-MC$_3$ *(for G) in 3-dimensions, (randomized) initialization of QAOA with $R_U(\mathbf{x})$ has a (worst-case) approximation ratio of 0.66κ at $p = 0$, i.e., only using quantum sampling with zero circuit depth for QAOA. Similarly, if $\mathbf{x}$ is a κ-close solution of* BM-MC$_2$ *(for Max-Cut of G) in 2-dimensions, initialization of QAOA with $R_U(\mathbf{x})$ is a 0.75κ-approximate solution at $p = 0$. Note that these approximation factors are lower bounds on the expected fraction of the Max-Cut obtained via random sampling.*

Proof. We start by proving the $2/3$ performance of a randomized mapping from BM-MC$_3$ to the Bloch sphere. Let $F_0' = F_0'(\gamma, \beta)$ be the expected value of MAX-CUT obtained by

quantum sampling (i.e., QAOA for $p = 0$). Then,

$$\frac{F_0'}{\text{Max-Cut}(G)} \geq \kappa \cdot \frac{F_0'}{\text{BM-MC}_3(\mathbf{x})} \qquad \text{(since BM-MC}_3(\mathbf{x}) \geq \kappa \text{ Max-Cut}(G))$$
$$\geq \kappa \min_{(i,j)\in E} \frac{\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]]}{\frac{1}{4}\|\mathbf{x}_i - \mathbf{x}_j\|^2}.$$
$$\left(\tfrac{a+c}{b+d} \geq \min(\tfrac{a}{b}, \tfrac{c}{d}) \text{ for } a, b, c, d \geq 0;\ w_{ij}\text{'s cancel}\right)$$

It suffices to lower bound the ratio between edge-wise contribution from quantum sampling versus edge-wise contribution to the semi-definite objective (which upper bounds the BM-MC$_k$ denominator). Instead of rotating the sphere, we can choose a random direction $w \in S^2$ to correspond to the positive spin of the Bloch sphere. Consider an edge $e = (i, j) \in E$ whose endpoints are at angles α on S^2 with respect to $\mathbf{x}$, i.e., $\mathbf{x}_i \cdot \mathbf{x}_j = \cos\alpha$. Let θ_1 and θ_2 correspond to angles from x_i and x_j to the positive spin w of the (rotated) sphere. We can write

$$\min_{(i,j)\in E} \frac{\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]]}{\frac{1}{4}\|\mathbf{x}_i - \mathbf{x}_j\|^2} \geq \min_{\alpha\in[0,\pi]} \frac{f(\theta_1, \theta_2)}{\sin^2(\alpha/2)}, \tag{4.1}$$

where

$$f(\theta_1, \theta_2) = \cos^2\frac{\theta_1}{2}\sin^2\frac{\theta_2}{2} + \cos^2\frac{\theta_2}{2}\sin^2\frac{\theta_1}{2},$$

where we replaced $\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]]$ by a sum of probabilities of the two cases corresponding to assignment of different spins to i and j, formulated considering the state is a product state and observing that $\|x_i - x_j\|^2 = 2 - 2\cos(\alpha) = 4\sin^2(\theta/2)$. We can rewrite the above as

$$\min_{\alpha\in[0,\pi]} \frac{\mathbb{E}_{\theta_1,\theta_2|\alpha}[g_1(\theta_1)g_2(\theta_2) + g_2(\theta_1)g_1(\theta_2)]}{2(1-\cos(\alpha))} = \min_{\alpha\in[0,\pi]} \frac{\mathbb{E}_{\theta_1,\theta_2|\alpha}[1 - \cos(\theta_1)\cos(\theta_2)]}{(1-\cos(\alpha))}, \tag{4.2}$$

where $g_1(\theta) = 1 + \cos(\theta)$ and $g_2(\theta) = 1 - \cos(\theta)$.

To further simplify notation of our optimization problem let us assume that in-

stead of rotating x_i and x_j and sampling with respect to a spin direction, we randomly choose the positive spin pivot w such that the z-axis is now rotated to be at $w \in S^2$. Without loss of generality, assume $x_i = (1,0,0), x_j = (\cos\alpha, \sin\alpha, 0)$ and $w = (\cos\theta, \sin\theta\cos\varphi, \sin\theta\sin\varphi) \in S^2$ is uniformly sampled from the sphere. Let

$$h(\theta, \varphi, \alpha) = \cos\theta(\cos\alpha\cos\theta + \sin\alpha\sin\theta\cos\varphi).$$

This give us the following:

$$\min_{\alpha\in[0,\pi]} \frac{\mathbb{E}_{\theta_1,\theta_2|\alpha}[1-\cos(\theta_1)\cos(\theta_2)]}{(1-\cos(\alpha))} \tag{4.3}$$

$$= \min_{\alpha\in[0,\pi]} \frac{1-\frac{1}{4\pi}\int_0^{\pi}\int_0^{2\pi} h(\theta,\varphi,\alpha)\sin\theta d\varphi d\theta}{1-\cos\alpha} \tag{4.4}$$

$$= \min_{\alpha\in[0,\pi]} \frac{1-\frac{1}{2}\cos\alpha\int_0^{\pi}\sin\theta\cos^2\theta d\theta}{1-\cos\alpha} \tag{4.5}$$

$$= \min_{\alpha\in[0,\pi]} \frac{1-\frac{\cos\alpha}{2}\left[\frac{-1}{3}\cos^3\theta\right]_0^{\pi}}{1-\cos\alpha} = \min_{\alpha\in[0,\pi]} \frac{1-\frac{\cos\alpha}{3}}{1-\cos\alpha} = \frac{2}{3}. \tag{4.6}$$

This finishes the proof for BM-MC$_3$.

Recall that for BM-MC$_2$, we perform a uniformly at random rotation along a unit circle on the Bloch sphere passing through $|0\rangle$ and $|1\rangle$. The proof is similar to the rank $k = 3$ case, and easier. It suffices to lower bound the following ratio:

$$\min_{(i,j)\in E} \frac{\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]]}{\frac{1}{4}\|\mathbf{x}_i - \mathbf{x}_j\|^2} \geq \min_{\alpha\in[0,\pi]} \frac{f(\alpha)}{\sin^2(\alpha/2)}, \tag{4.7}$$

by 0.75. Here $f(\alpha)$ denotes the probability that two unentangled qubits with (angular) distance α over the sphere/circle, are measured by opposite spins. Similar as in previous proof we can simplify the ratio as

$$\min_{\alpha\in[0,\pi]} \frac{\mathbb{E}_{\theta_1,\theta_2|\alpha}[1-\cos(\theta_1)\cos(\theta_2)]}{(1-\cos(\alpha))},$$

where θ_1 and θ_2 are the angles between two vertices and the pivot.

Again we can think of vertices to be fixed over the sphere and randomly rotate the $|1\rangle$ pivot. Without loss of generality, let $x_i = (1, 0)$ and $x_j = (\cos\alpha, \sin\alpha)$. The random pivot can be formulated as $(\cos\theta, \sin\theta)$ where θ is uniformly distributed over $[0, 2\pi)$. We can write $\theta_1 = \theta$ and $\theta_2 = \theta - \alpha$. The target ratio can be written as

$$\begin{aligned}
&\min_{\alpha\in[0,\pi]} \frac{1 - \frac{1}{2\pi}\int_0^{2\pi} \cos(\theta)\cos(\theta-\alpha)d\theta}{1-\cos(\alpha)} \\
&= \min_{\alpha\in[0,\pi]} \frac{1 - \frac{1}{2\pi}\int_0^{2\pi} \cos^2(\theta)\cos(\alpha)d\theta + 0}{1-\cos(\alpha)} \\
&= \min_{\alpha\in[0,\pi]} \frac{1 - \frac{1}{2}\cos(\alpha)}{1-\cos(\alpha)} = \frac{3}{4}.
\end{aligned}$$

□

Using the same analysis as Goemans and Williamson [11], it is straightforward to prove (for graphs with non-negative edge weights) that a κ-close solution must also be 0.878κ-approximate (i.e. HP$(x) \geq 0.878\kappa$Max-Cut(G)). For locally optimal[4] BM-MC$_k$ solutions, a theorem by Mei et al. [77] (Theorem 2) proves that such solutions are κ-close (and hence 0.878κ-approximate) with $\kappa = 1 - \frac{1}{k-1}$; thus, hyperplane rounding of such solutions yields (worst-case) approximation ratios of $0.878(0) = 0$ and $0.878\left(\frac{1}{2}\right) = 0.439$ while depth-0 QAOA-warmest can obtain (worst-case) approximation ratios of $\frac{3}{4}(0) = 0$ and $\frac{2}{3}\left(\frac{1}{2}\right) = \frac{1}{3}$ for $k = 2$ and $k = 3$ solutions respectively. Note that, in pratice, κ could be much higher; in our simulations, we observed $\kappa \geq 0.999$ for all positive-weighted instances for BM-MC$_3$, the same can be said for BM-MC$_2$ with the exception of 19 instances with the smallest κ observed being $\kappa = 0.833$.

[4]When x^* is a globally optimal BM-MC$_k$ solution, it immediately follows that x^* is (at least) 1-close and 0.878-approximate and hence (worst-case) approximation ratios of $\frac{3}{4}$ and $\frac{2}{3}$ (for $k = 2$ and $k = 3$ solutions respectively) are achieved for depth-0 QAOA-warmest; however, since the Burer-Monteiro relaxation is non-convex, finding such globally optimal solutions becomes intractable, especially as the number of nodes increases.

4.2.2 Projected Goemans-Williamson

Recall that a solution to the Goemans-Williamson (GW) SDP relaxation consists of n unit vectors $\{u_i \in \mathbb{R}^n : i \in V\}$, and their rounding algorithm uses a random hyperplane to obtain an approximation for the Max-Cut on the given graph G. One can create a warm-start to QAOA by instead rounding each of these vectors to $\mathbb{R}^k$ (with $k \in \{2, 3\}$) and then mapping the rounded vectors to the Bloch sphere.

Specifically, given $u \in \mathbb{R}^n$ for some positive integer n, and a linear subspace A of $\mathbb{R}^n$, let $\Pi_A(u)$ denote the (Euclidean) projection of u on A. Given $\Pi_A(u) \neq 0$, define

$$\Lambda_A(u) = \frac{\Pi_A(u)}{\|\Pi_A(u)\|_2},$$

as the *unit-scale* projection of u on A. This corresponds to normalizing $\Pi_A(u)$ so it is a unit vector. If $A = \text{span}(v)$ for some unit vector $v \in \mathbb{R}$, we abuse the notation and denote $\Pi_v(u) = \Pi_{\text{span}(v)}(u)$ and $\Lambda_v(u) = \Lambda_{\text{span}(v)}(u)$.

To motivate warm-starts using solutions for the Goemans-Williamson SDP, consider the following two-step process for rounding an optimal SDP solution $\{u_i : i \in [n]\}$ to a cut:

- Choose a uniformly random linear subspace A of $\mathbb{R}^n$ of dimension $k \in \{2, 3\}$, and consider the unit-scale projections $\Lambda_A(u_i) \in \mathbb{R}^k, i \in [n]$.
- Use Goemans-Williams hyperplane rounding on vectors $\Lambda_A(u_i), i \in [n]$ to get a (random) cut M'.

That is, round the vectors $u_i \in \mathbb{R}^n$, $i \in [n]$ to a random k-dimensional subspace first, and then subsequently use the Goemans-Williamson hyperplane rounding on these k-dimensional vectors.

We will later prove that this two-step rounding is equivalent to the Goemans-Williamson hyperplane rounding on vectors u_i, $i \in [n]$. To do this, we first prove a few

helpful lemmas.

Lemma 17. *Let u, v be unit vectors in $\mathbb{R}^n$ and let A denote a linear subspace of $\mathbb{R}^n$ of dimension k such that $v \in A$. If $\Pi_A(u) \neq 0$ and $\Pi_v(u) \neq 0$, then*

$$\Lambda_v(u) = \Lambda_v\big(\Lambda_A(u)\big).$$

That is, unit-scale projection of u on v is equivalent to first unit-scale projecting u to A and projecting this projection $\Lambda_A(u)$ on v.

Proof. Let $\{v_1, \ldots, v_k\}$ be an orthonormal basis for A. Let $\alpha_i = u^\top v_i$. Then

$$\Pi_A(u) = \sum_{i\in[k]} \alpha_i v_i, \quad \Lambda_A(u) = \frac{\sum_{i\in[k]} \alpha_i v_i}{\sqrt{\sum_{i\in[k]} \alpha_i^2}}$$

Since $v \in A$, write $v = \sum_{i\in[k]} \beta_i v_i$. Then we have

$$\Pi_v\big(\Lambda_A(u)\big) = \frac{\sum_{i\in[k]} \alpha_i\beta_i}{\sqrt{\sum_{i\in[k]} \alpha_i^2}}\, v,$$

so that

$$\begin{aligned}
\Lambda_v\big(\Lambda_A(u)\big) &= \frac{\Pi_v\big(\Lambda_A(u)\big)}{\|\Pi_v\big(\Lambda_A(u)\big)\|_2} \\
&= \frac{\sum_{i\in[k]} \alpha_i\beta_i}{\left|\sum_{i\in[k]} \alpha_i\beta_i\right|}\, v \\
&= \frac{\Pi_v(u)}{\|\Pi_v(u)\|_2}\, v \\
&= \Lambda_v(u).
\end{aligned}$$

□

Note that the above lemma is a deterministic statement; we have not used any randomness so far.

Let us consider what happens if we select a linear subspace A of $\mathbb{R}^n$ of dimension k uniformly randomly from $\mathbb{R}^n$ (one way to ensure it is chosen uniformly randomly is to select unit vectors $v_i \in R^n, i \in [k]$ recursively so that v_i is chosen uniformly randomly in the space orthogonal to $v_1, \ldots, v_{i-1}$). Once we have A, let us select a vector $v \in A$ uniformly randomly again. Is this equivalent to choosing a vector $v \in \mathbb{R}^n$ uniformly randomly? By symmetry, it is, since the former experiment is not biased in favor of any direction. We omit the formal proof and state it as a lemma here:

Lemma 18. *Let E denote the experiment of choosing a unit vector v chosen uniformly randomly from $\mathbb{R}^n$. Let E' denote the experiment of choosing a linear subspace A of dimension k uniformly randomly from $\mathbb{R}^n$, and then choosing a unit vector v' uniformly randomly from A. Then $E' = E$, i.e., they correspond to the same probability space.*

We now prove that the two-step rounding procedure is equivalent to the usual Goemans-Williamson rounding procedure as demonstrated in Theorem 19 below.

Theorem 19. *Suppose we are given unit vectors $u_1, \ldots, u_n \in \mathbb{R}^n$ that form an optimal solution to the SDP relaxation for Max-Cut on some graph $G = (V, E)$ with n vertices and non-negative weights on the edges. Suppose the GW random hyperplane rounding on $u_1, \ldots, u_n$ obtains a (random) cut M of value X, and the two-step rounding described above produces a (random) cut M' of value Y. Then,*

1. *The random variables $X = Y$. In particular, $\mathbb{E}(X) = \mathbb{E}(Y)$ and therefore, in expectation, M' provides a 0.878-approximation to Max-Cut on G.*

2. *Furthermore, the two-step rounding procedures produces a cut of value $(0.878 - \epsilon)$ times the Max-Cut value* with high probability *if performed independently $\frac{\log n}{\epsilon}$ times for any constant $\epsilon \in (0, 1/2)$.*

   Further, if $A_1, \ldots, A_{\frac{\log n}{\epsilon}}$ are the intermediate k-dimensional subspaces in these $\frac{\log n}{\epsilon}$ runs, there is at least some A_i (with high probability) such that performing the hyper-

plane rounding on A_i produces a (random) cut of average value at least $(0.878 - \epsilon)$ times the Max-Cut.

Proof. Let $U_n = \{v \in \mathbb{R}^n : \|v\|_2 = 1\}$ be the set of unit vectors in $\mathbb{R}^n$. Recall that for a given probability space, a random variable is a real-valued function on the sample space, or that $X, Y : U_n \to \mathbb{R}$. From Lemma 18, the two experiments correspond to the same probability space. Therefore, it is enough to prove that $X(v) = Y(v)$ for all $v \in U_n$.

One key observation is that rounding on a random hyperplane is equivalent to unit-scale projecting to a uniformly random vector v: indeed, let v be the vector normal to the uniform hyperplane, then any unit vector u is rounded to 1 if $u \cdot v > 0$ and to -1 if $u \cdot v < 0$. That is, u is rounded to $\frac{u \cdot v}{|u \cdot v|} = \Lambda_v(u)^\top v$.

Therefore,

$$X(v) = \sum_{ij \in E(G)} w_{ij} \, \mathbf{1}\Big[\Lambda_v(u_i)^\top v \cdot \Lambda_v(u_j)^\top v < 0\Big].$$

Similarly, for a given A such that $v \in A$, we have $Y(v)$ is equal to:

$$\sum_{ij \in E(G)} w_{ij} \mathbf{1}\Big[\Lambda_v\big(\Lambda_A(u_i)\big)^\top v \cdot \Lambda_v\big(\Lambda_A(u_j)\big)^\top v < 0\Big].$$

Since $\dim(A) = k$ is a constant and A is chosen uniformly randomly, $\Pi_A(u_i) \neq 0$ for each i with probability 1. From Lemma 17, we have $\Lambda_v\big(\Lambda_A(u)\big) = \Lambda_v(u)$ for all unit vectors u and for all A.

Therefore, we have

$$\begin{aligned} Y(v) &= \sum_{ij \in E(G)} w_{ij} \, \mathbf{1}\Big[\Lambda_v(u_i)^\top v \cdot \Lambda_v(u_j)^\top v < 0\Big] \\ &= X(v). \end{aligned}$$

Since X and Y have the same distribution, the same approximation guarantee holds for

X, Y. This proves part 1 of the theorem.

We prove part 2 next. Let C denote the maximum cut value on graph G, and denote $\alpha = 0.878$ for convenience. Then, part 1 shows that $\mathbb{E}Y \geq \alpha C$. We first show that $\Pr\left(Y > (1-\epsilon)\mathbb{E}Y\right) \geq \alpha\epsilon$ using Markov inequality:

$$\begin{aligned}
&\Pr\left(Y > (1-\epsilon)\mathbb{E}Y\right) \\
&= 1 - \Pr\left(Y \leq (1-\epsilon)\mathbb{E}Y\right) \\
&= 1 - \Pr\left(C - Y \geq C - (1-\epsilon)\mathbb{E}Y\right) \\
&\geq 1 - \frac{\mathbb{E}\left(C - Y\right)}{C - (1-\epsilon)\mathbb{E}Y} \\
&= \frac{\epsilon\mathbb{E}Y}{C - (1-\epsilon)\mathbb{E}Y} \\
&\geq \frac{\epsilon\mathbb{E}Y}{\frac{\mathbb{E}Y}{\alpha} - (1-\epsilon)\mathbb{E}Y} \\
&= \frac{\alpha\epsilon}{1 - \alpha(1-\epsilon)} \\
&\geq \alpha\epsilon.
\end{aligned}$$

Suppose that $\frac{\log n}{\epsilon}$ independent cuts are produced by applying the two-step rounding procedure $\log n$ times. Then the probability that all of these cuts have value less than $(1-\epsilon)\mathbb{E}Y$ is at most

$$(1-\alpha\epsilon)^{\frac{\log n}{\epsilon}} \leq \left(e^{-\alpha\epsilon}\right)^{\frac{\log n}{\epsilon}} = \frac{1}{n^{\alpha}},$$

where we have used the standard inequality $\exp(-x) \geq 1 - x$. Since $\alpha \in (0, 1)$, this probability goes to 0 as n goes to ∞.

We prove the second claim of part 2. Given a k-dimensional subspace A of $\mathbb{R}^n$, let w_A denote the average cut value after Goemans-Williamson hyperplane rounding is performed on A, i.e.

$$w_A = \frac{\int_{v \in U_A} (\text{cut value along } v)\, dv}{\int_{v \in U_A} dv},$$

where U_A is the set of all unit vectors in A. Notice that

$$\mathbb{E}Y = \frac{\int_A w_A \, dA}{\int_A \, dA}.$$

For the first step of the two-step rounding procedure (i.e., the step selecting a random subspace of dimension k), let Z denote the random variable that takes value w_A when subspace A is selected. We need to show that for $\frac{\log n}{\epsilon}$ i.i.d. random variables $Z_1, \ldots, Z_{\frac{\log n}{\epsilon}}$, there is at least some Z_i such that $Z_i \geq (1-\epsilon)\mathbb{E}Y$. Since the random subspace is selected uniformly randomly, we have that

$$\mathbb{E}Z = \frac{\int_A w_A \, dA}{\int_A \, dA} = \mathbb{E}Y.$$

A similar Markov inequality analysis on Z then gives the result.

□

The last part of Theorem 19 illustrates that we can obtain a high-quality rank-k projected GW solution in regards to hyperplane rounding; however, it is natural to ask if any kind of guarantee can be preserved when mapping the solution to a quantum state. By adapting Theorem 16, we can answer the question in the affirmative as seen in Corollary 20 below.

Corollary 20. *Let G be a graph with non-negative edge weights and let x be a corresponding κ-approximate projected GW solution in $\mathbb{R}^3$ with respect to hyperplane rounding.*[5] *Let $R_U(x)$ denote random uniform rotation applied to x, i.e., a global rotation where a uniformly selected point on the sphere gets mapped to $(0, 0, 1)$. Then initialization of QAOA with $R_U(x)$ has an (worst-case) approximation ratio of $\frac{2}{3}\kappa$ at $p = 0$, i.e., only using quantum sampling with initial state creation and no algorithmic depth for QAOA. Similarly, if x is a κ-approximate projected GW solution in $\mathbb{R}^2$, initialization of QAOA with $R_U(x)$ is a*

[5]The theorem also holds more generally for any feasible BM-MC$_2$ or BM-MC$_3$ solution.

$\frac{3}{4}\kappa$*-approximate solution at* $p = 0$*.*

If x *is chosen such that it is* κ*-approximate with* $\kappa = 0.878 - \varepsilon$ *(such an* x *is easily found via Theorem 19), then, for small* ε*, this yields (worst-case) approximation ratios (for depth-0 QAOA-warmest) of* $\frac{3}{4}(0.878 - \varepsilon) \approx 0.658$ *and* $\frac{2}{3}(0.878 - \varepsilon) \approx 0.585$ *for 2-dimensional and 3-dimensional projections respectively.*

Proof. Let $F_0' = F_0'(\gamma, \beta)$ be the expected value of MAX-CUT obtained by quantum sampling (i.e., QAOA for $p = 0$). Then,

$$\begin{aligned}
&\frac{F_0'}{\text{Max-Cut}(G)} \\
&\geq \kappa \cdot \frac{F_0'}{\text{HP}(x)} && \text{(since HP}(x) \geq \kappa \text{ Max-Cut}(G)\text{)} \\
&\geq \kappa \min_{(i,j)\in E} \frac{\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]]}{\frac{1}{\pi}\arccos(x_i \cdot x_j)}. \\
& && \left(\tfrac{a+c}{b+d} \geq \min(\tfrac{a}{b}, \tfrac{c}{d}) \text{ for } a, b, c, d \geq 0;\ w_{ij}\text{'s cancel}\right)
\end{aligned}$$

For $i, j \in [n]$, let θ_{ij} denote the angle between x_i and x_j. We can write

$$\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]] = f_k(\theta_{ij}),$$

that is, this expectation is solely a function of the angle between the adjacent vertices and the rank k considered. In particular, from the proof of Theorem 16, we have that for $k \in \{2, 3\}$ that

$$f_k(\theta_{ij}) = \frac{1}{2}\left(1 - \frac{\cos\theta_{ij}}{k}\right).$$

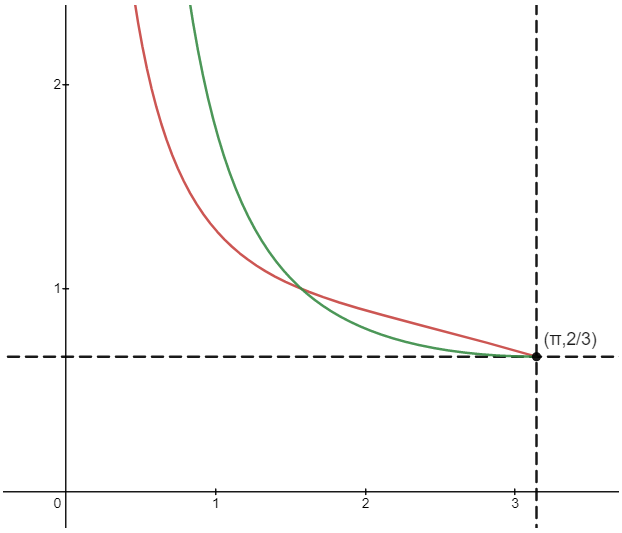

Figure 4.4: In red, the function $\frac{\frac{1}{2}\left(1-\frac{\cos\theta}{k}\right)}{\theta/\pi}$ that is minimized in the proof of Corollary 20 with $k = 3$. In green, a similar function $\frac{\frac{1}{2}\left(1-\frac{\cos\theta}{k}\right)}{\frac{1}{2}(1-\cos(\theta)}$ that is minimized in the proof of Theorem16 (where the BM-MC$_k$ objective is compared to the maximum cut instead of the expected cut value from hyperplane rounding) with $k = 3$. Over the interval $[0, \pi]$, both achieve a minimum value of $2/3$ at $\theta = \pi$. The corresponding plots for $k = 2$ are similar but instead both functions reach a minimum value of $3/4$ at $\theta = \pi$.

Additionally, $\arccos(x_i \cdot x_j) = \theta_{ij}$. Using these substitutions, we have

$$\begin{aligned}
&\frac{F_0'}{\text{Max-Cut}(G)} \\
&\geq \kappa \min_{(i,j)\in E} \frac{\mathbb{E}[\mathbf{1}[i \text{ and } j \text{ have different spins}]]}{\frac{1}{\pi}\arccos(x_i \cdot x_j)} \\
&= \kappa \min_{(i,j)\in E} \frac{f_k(\theta_{ij})}{\theta_{ij}/\pi} \\
&\geq \kappa \min_{\theta\in[0,\pi]} \frac{f_k(\theta)}{\theta/\pi} \\
&= \kappa \min_{\theta\in[0,\pi]} \frac{\frac{1}{2}\left(1-\frac{\cos\theta}{k}\right)}{\theta/\pi}.
\end{aligned}$$

For $k \in \{2, 3\}$, it is straightforward to verify that the minimum in the last line above is achieved at $\theta = \pi$ (see Figure 4.4) which leads to the ratio $\frac{F_0'}{\text{Max-Cut}(G)}$ being at least $\frac{3}{4}\kappa$ and $\frac{2}{3}\kappa$ respectively for $k = 2, 3$. □

This motivates us to use the projected vectors $\Lambda_A(u_i), i \in [n]$ to warm-start QAOA.Figure 4.5 illustrates the two-step rounding procedure and this warm-start.

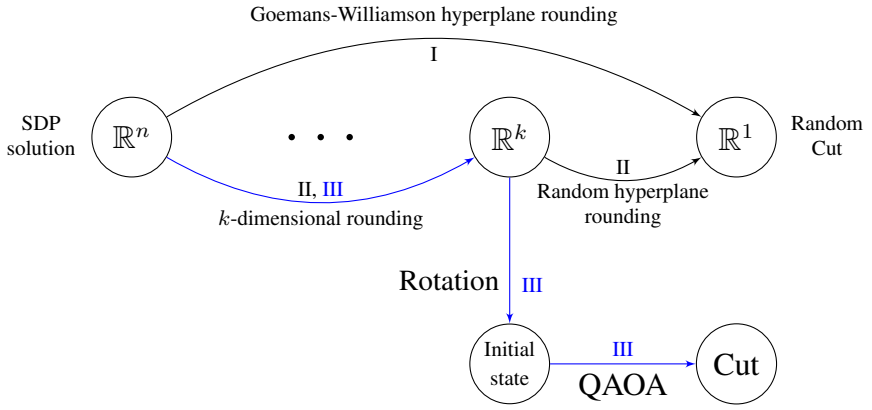

Figure 4.5: A schematic for projected GW warm-starts. There are three procedures to obtain a cut from an SDP solution. The first is to use Goemans-Williamson hyperplane rounding (on the top labelled I). The second (labelled II) is to do a two-step rounding through an intermediate state in $\mathbb{R}^k, k \in \{2, 3\}$. We prove that this two-step rounding procedure is equivalent to Goemans-Willimson hyperplane rounding in Theorem 19. The third procedure is our proposed warm-start of QAOA using the SDP solution (highlighted in blue, labelled III). This procedure involves rounding the SDP solution to $\mathbb{R}^k$ first, then rotating this solution using uniform or vertex-at-top rotations and mapping to the Bloch sphere to get an initial state for QAOA, and finally running QAOA on this initial state.

4.2.3 Single Cut Initializations

For the purposes of comparison, we consider one last warm-start technique. In previous works, initializations of the quantum state based on a single "good" cut $(S, V \setminus S)$ (obtained via the GW algorithm or possibly other classical means) have also been considered [42, 57]. Given a regularization angle $\theta^* \in [0, \pi/2]$, one can initialize the initial quantum state so that qubits lie along the xz-plane of the Bloch sphere with vertices in S and $V \setminus S$ being initialized at an angle θ away from the north and south poles of the Bloch sphere respectively; such a regularization angle aims to circumvent issues regarding reachability [42]. More specifically, the quantum state is given by,

$$|s_0\rangle = \bigotimes_{j=1}^{n} |s_{0,j}\rangle ,$$

where,

$$|s_{0,j}\rangle = \begin{cases} R_Y(\theta^*) |0\rangle , & j \in S \\ R_Y(\pi - \theta^*) |0\rangle , & j \notin S \end{cases}$$

where $R_Y(\theta)$ is a single-qubit rotation about the y-axis by angle θ.

As Proposition 9 illustrates, one can achieve a (worst-case) approximation ratio approaching 0.878 as the regularization angle θ^* approaches zero.

Proposition 9. *Let $G = (V, E)$ be graph with non-negative edge weights. Let $(S, V \setminus S)$ be a (random) cut obtained via the GW algorithm. Then, initializing $|s_0\rangle$ using regularization angle $\theta^* \in [0, \pi/2]$ and performing a quantum measurement (i.e. depth-0 QAOA) yields a (worst-case) approximation ratio of at least $0.878 \cdot \cos(\theta^*/2)^{2|V|}$.*

Proof. Let X denote the random variable corresponding to the cut value obtained from the cut obtained by quantum measurement of a single-cut initialization obtained by GW hyperplane rounding as described in Section 4.2.3. For any $S \subseteq V$, let GW(S) denote the event that the cut $(S, V \setminus S)$ was obtained from the GW hyperplane rounding step. similarly, let QM(S) denote the event that quantum measurement of the initial state resulted in the cut $(S, V \setminus S)$.

First, observe that if the cut $(S, V \setminus S)$ is used to initialize the quantum state with regularization angle θ^*, then the probability of quantum measurement getting the same cut is $\cos^{2|V|}(\theta^*/2)$; this is because each vertex independently has probability $\cos^2(\theta^*/2)$ of remaining on the same side of the cut used to initialize the quantum state. Using this fact, we find that,

$$
\begin{aligned}
&\mathbb{E}[X] \\
&= \sum_{S \subseteq V} \mathbb{E}[X \mid \textbf{GW}(S)] \Pr(\textbf{GW}(S)) \\
&= \sum_{S \subseteq V} \left(\sum_{T \subseteq V} \mathbb{E}[X \mid \textbf{GW}(S), \textbf{QM}(T)] \right. \\
&\quad \left. \cdot \Pr(\textbf{QM}(T) \mid \textbf{GW}(S)) \cdot \Pr(\textbf{GW}(S)) \right) \\
&\geq \sum_{S \subseteq V} \left(\mathbb{E}[X \mid \textbf{GW}(S), \textbf{QM}(S)] \right. \\
&\quad \left. \cdot \Pr(\textbf{QM}(S) \mid \textbf{GW}(S)) \cdot \Pr(\textbf{GW}(S)) \right) \\
&= \sum_{S \subseteq V} \left(\textbf{cut}(S) \cdot \cos^{2|V|}(\theta^*/2) \cdot \Pr(\textbf{GW}(S)) \right) \\
&= \cos^{2|V|}(\theta^*/2) \sum_{S \subseteq V} \left(\textbf{cut}(S) \Pr(\textbf{GW}(S)) \right) \\
&\geq \cos^{2|V|}(\theta^*/2) \cdot 0.878 \textbf{Max-Cut}(G),
\end{aligned}
$$

and thus $\frac{\mathbb{E}[X]}{\text{Max-Cut}(G)} \geq 0.878 \cos^{2|V|}(\theta^*/2)$ as desired. In the above formulas, we used the fact that $\mathbb{E}[X \mid \textbf{GW}(S), \textbf{QM}(S)] = \textbf{cut}(S)$ and that the sum $\sum_{S \subseteq V}(\textbf{cut}(S) \Pr(\textbf{GW}(S)))$ is simply the expected cut value of the GW algorithm, which we know is at least 0.878 of the optimal cut value for graphs with non-negative weights [11]. □

Recall (Section 4.2.2) that QAOA with a (2-dimensional) projected GW initialization has a depth-0 (worst-case) approximation ratio of 0.658. One may be led to believe that, when custom mixers are used to guarantee convergence (as described in the next chapter), that a single-cut initialization (with small regularization angle θ^*) is the better choice due to its (theoretically) better (worst-case) approximation ratio at depth-0. However, as seen

empirically in Section 5.4, this is not the case: when θ^* is small, the convergence rate of QAOA with single-cut initializations is (emperically) incredibly slow across all instances. For small θ^*, QAOA with custom mixers geometrically performs rotations around axes that are near the poles of the Bloch sphere about the qubits' initial positions; it is possible that this geometric interpretation is responsible for the slow convergence for small θ^*.

In one of their approaches for Max-Cut, Egger et al. [42] use a mixer for QAOA that is different than both the standard mixer and the custom mixers described in Section 5. They prove that their mixer has the property that, when a single-cut initialization (based on a cut $(S, V \setminus S)$) with regularization parameter $\theta^* = \pi/3$ is used, that measurement of the depth-1 QAOA with variational parameters $(\gamma_1, \beta_1) = (0, \frac{\pi}{2})$ produces exactly the cut $(S, V \setminus S)$ that was used to initialize the initial quantum state. The drawback is that, with such a mixer proposed by Egger et al., no convergence guarantees are known and experiments suggest that, unlike standard QAOA, the optimal cut value is not achieved in expectation with increased circuit depth.

Cain et al. [57] consider the case where $\theta^* = 0$ and the standard mixer is used. They find that such an approach performs very poorly; in particular, no convergence towards the optimal cut is found with increased circuit depth either.

One can also consider using a single-cut initialization together with the custom mixers proposed in Section 5; this idea was very briefly explored in the appendices of Egger et al.'s work [42]. From Proposition 9 and Theorem 14, it is clear that this approach (with non-negative weighted graphs) yields a (worst-case) approximation ratio approaching 0.878 for $\theta^* \to 0$ for depth-0 QAOA and that such an approach convergences to the optimal cut with increased circuit depth.

Next, we discuss another approach proposed by Egger et al. [42] for more general problems; we prove that in the context of Max-Cut, this approach (effectively) falls into the category of single-cut initializations as described above.

Relation to QUBO Approach

Egger et al. [42] consider a warm-start approach (which they call continuous warm-started QAOA) for QUBO's of the form

$$\min_{y \in \{0,1\}^n} y^T M y,$$

where $M \in \mathbb{R}^{n \times n}$ is a real-symmetric matrix. They then consider the relaxation

$$\min_{y \in [0,1]^n} y^T M y,$$

i.e. the binary variables are relaxed to lie in the interval $[0, 1]$. For certain matrices[6] M, this yields a convex quadratic program which can be easily solved to (global) optimality [107]; in this case, Egger et al. [42] find and use the globally optimal solution y^* of the relaxation to produce a initial quantum product state. We consider this approach in the context of Max-Cut, specifically in the case of graphs with non-negative edge weights.

One can formulate Max-Cut on a graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}$ as a QUBO problem as follows [38]. Simply construct the QUBO matrix M by setting $M_{ij} = w_{ij}$ for $i \neq j$ and $M_{ii} = -\sum_{j=1}^{n} w_{ij}$ for $i \in \{1, \ldots, n\}$. If x^* is an optimal solution to Max-Cut (using the formulation in Equation 2.10), then there is a corresponding y^* that is an optimal solution of the QUBO such that $x_i^* = 2y_i^* - 1$ for $i = 1, \ldots, n$.

Observe that $M = -L$ where L Laplacian matrix of the graph G which is known to be positive-semidefinite (for graphs with non-negative edge weights) [108]. Since L is positive-semidefinite, then the function $f(x) = x^T L x$ is convex in x (as the Hessian of f, $\nabla^2 f(x) = 2L$, is positive-semidefinite). Thus, $x^T M x = -f(x)$ is generally not convex, and hence solving $\min_{x \in [0,1]^n} x^T M x$ to global optimality (as is done by Egger et al. [42]) is non-trivial in the case of Max-Cut. However, we can still consider locally optimal solutions

[6] In particular, Egger et al. [42] consider matrices of the form $M = N + D$ where $N \in \mathbb{R}^{n \times n}$ is a (symmetric) positive-semidefinite matrix and $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix.

to the relaxation. Observe,

$$\min_{y\in[0,1]^n} y^T My = \max_{y\in[0,1]^n} y^T Ly,$$

i.e., the QUBO relaxation amounts to maximizing a convex function over a polytope, in which case, all strictly local maxima lie on the vertices of the polytope; as the theorem below illustrates.

Theorem 21. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function and let $P \subseteq \mathbb{R}^n$ be a polytope. Then, if $y^* \in \mathbb{R}^n$ is a strict local maximum of f over P, then y^* lies at a vertex of P.*

Proof. To see this, suppose by means of contradiction that y^* was a strict local maximum that did not lie at a vertex of the polytope. Then there exists $z \in \mathbb{R}^n$ such that $y^* - z$ and $y^* + z$ lie in the polytope such that $f(y^*) > f(y^* - z)$ and $f(y^*) > f(y^* + z)$. By convexity,

$$\begin{aligned} f(y^*) &= f\left(\frac{1}{2}(y^* - z) + \frac{1}{2}(y^* + z)\right) \\ &\leq \frac{1}{2}f(y^* - z) + \frac{1}{2}f(y^* + z) \\ &< \frac{1}{2}f(y^*) + \frac{1}{2}f(y^*) \\ &= f(y^*), \end{aligned}$$

a contradiction. □

The vertices of the polytope $[0, 1]^n$ correspond to cuts in the graph, thus, using the strictly locally optimal solution to the relaxation of the QUBO corresponding to Max-Cut degenerates to solutions corresponding to a single-cut; this means that, for Max-Cut (with non-negative edge-weights), this QUBO approach is (effectively) a single-cut initialization approach as described in Section 4.2.3.

4.3 Limitations of QAOA-Warm

In the case of the standard initialization for QAOA, we know that with the optimum choice of parameters γ, β, the probability of sampling the Max-Cut (with a single measurement) approaches 1 as the circuit depth p approaches infinity. This is not the case for QAOA-warm:

Theorem 22. *There exists a graph G and a warm-start initialization (of G) such that for all $p \geq 0$, depth-p QAOA-warm (with any choice of variational parameters γ and β) results in $F_p(\gamma, \beta) = \frac{1}{2}$ Max-Cut(G), or in other words, the expected cut obtained via quantum sampling is $\frac{1}{2}$ Max-Cut(G).*

Proof. We consider a graph $G = (V, E)$ on two vertices connected by an edge of unit weight initialized with $|s\rangle := |u\rangle \otimes |v\rangle$ where $|u\rangle := |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|v\rangle := |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. (Note that $|s\rangle = Q(x^*)$ where $x^* = ((1,0,0)^T, (-1,0,0)^T)$ is an optimal solution to BM-MC$_3$.)

We first consider the $p = 1$ case. For convenience, let $\gamma := \gamma_1$ and $\beta := \beta_1$ for this case. Observe,

$$|s\rangle = |u\rangle \otimes |v\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle).$$

Note that if we were to do a quantum measurement of this state, we would get each of the four states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ with equal probability, i.e., the theorem holds in the $p = 0$ case as well.

Since H_C is the Hamiltonian of the Max-Cut problem, $H_C |01\rangle = 1 \cdot |01\rangle$. Thus, $e^{-i\gamma H_C} |01\rangle = e^{-i\gamma \cdot 1} |01\rangle = e^{-i\gamma} |01\rangle$. Similar calculations show that $e^{-i\gamma H_C} |10\rangle = e^{-i\gamma} |10\rangle$, $e^{-i\gamma H_C} |00\rangle = |00\rangle$, and $e^{-i\gamma H_C} |11\rangle = |11\rangle$ and thus by linearity,

$$|s'\rangle := e^{-i\gamma H_C} |s\rangle = e^{-i\gamma H_C} \left(\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)\right)$$

$$= \frac{1}{2}\left(|00\rangle - |11\rangle + e^{-i\gamma}\big(|10\rangle - |01\rangle\big)\right).$$

For a 2-node graph, $H_B = \sigma_1^x + \sigma_2^x$, and thus,

$$H_B(|00\rangle - |11\rangle) = \sigma_1^x|00\rangle - \sigma_1^x|11\rangle + \sigma_2^x|00\rangle - \sigma_2^x|11\rangle = 0,$$

and similarly $H_B(|10\rangle - |01\rangle) = 0$.

By the above observations and linearity, we have that $H_B|s'\rangle = 0$, i.e., $|s'\rangle$ is an eigenvector of H_B with eigenvalue 0 and thus,

$$|\psi_1(\gamma,\beta)\rangle = e^{-i\beta H_B}|s'\rangle = e^{-i\beta\cdot 0}|s'\rangle = |s'\rangle,$$

i.e., the mixing Hamiltonian has no effect on the quantum state. Writing out $|s'\rangle$, we have

$$|\psi_1(\gamma,\beta)\rangle = \frac{1}{2}\left(|00\rangle - |11\rangle + e^{-i\gamma}\big(|10\rangle - |01\rangle\big)\right).$$

If we repeat all of these calculations in the case that $p > 1$, we get that

$$|\psi_p(\gamma,\beta)\rangle = \frac{1}{2}\left(|00\rangle - |11\rangle + e^{-i\gamma_p}\cdots e^{-i\gamma_1}\big(|10\rangle - |01\rangle\big)\right)$$

$$= \frac{1}{2}\left(|00\rangle - |11\rangle + e^{-i\sum_{i=1}^{p}\gamma_i}\big(|10\rangle - |01\rangle\big)\right),$$

in which case all four states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ are measured with equal probability meaning that the expected cut value for G is 50% of the maximum cut in G. □

The previous theorem shows that QAOA-warm may achieve poor instance-specific approximation ratios on *specific* initial states, however we next discuss that this behavior is consistent across slight perturbations around this state as well. In Figure 4.6, at any point (φ, θ) we depict the percentage of Max-Cut obtained using the optimal choice of variational parameters if the initial state of the first qubit is given by the polar and azimuthal an-

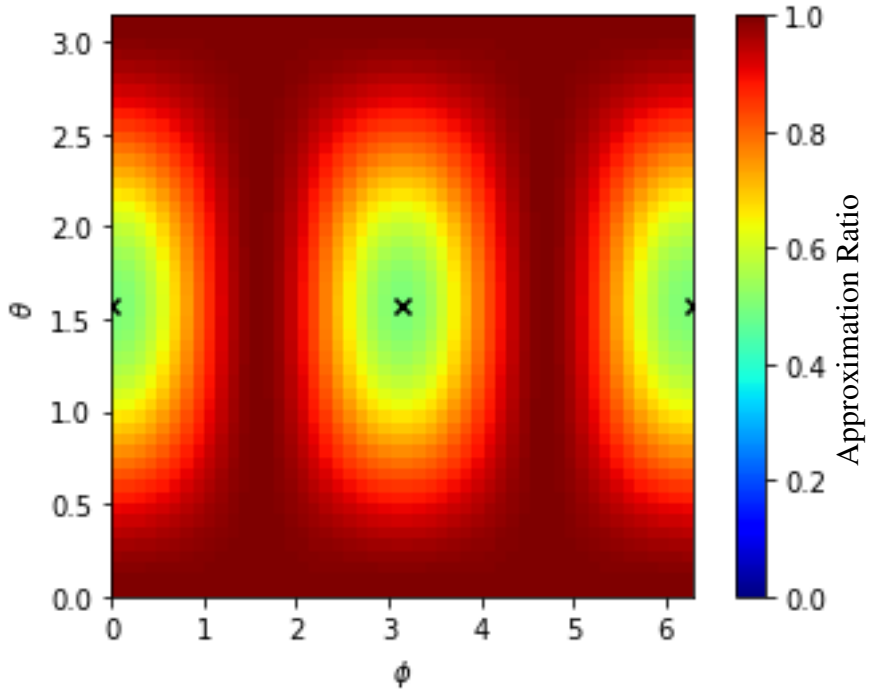

Figure 4.6: A plot of the percentage of the Max-Cut achieved with QAOA-warm (when the optimal variational parameters are chosen) with $p = 1$ for a one-edge graph G at various starting states $|s_0\rangle = Q(x)$ where one point of x has polar angle θ and azimuthal angle φ and the remaining point is diametrically opposed. The starting states that perform the worst, i.e. $|+-\rangle$ and $|-+\rangle$, are marked with a black $\times$. For each point in the figure, the optimal variational parameters were estimated by performing a dense grid-search over the variational parameter space.

gles θ and φ and the second qubit is diametrically opposed. Note that the optimal Max-Cut is achieved with probability 1 only when both vertices lie in the yz-plane. The worst case occurs when the vertices lie on the x-axis; this is consistent with Theorem 22. In general, in expectation, there is a larger gap to optimality the closer the solution $\mathbf{x}$ is to the x-axis; which suggests that it is reasonable to embed the approximate solutions of BM-MC$_2$ in the yz-plane of the Bloch sphere (as described in Section 4.1.2). Lastly, we believe that this behavior is consistent at larger circuit depths as well.

4.4 QAOA-Warm on Antipodal Structures

We illustrate a set of graph instances where QAOA-warm has a significant advantage over standard QAOA by considering BM-MC$_k$ solutions that have a special structure. For any

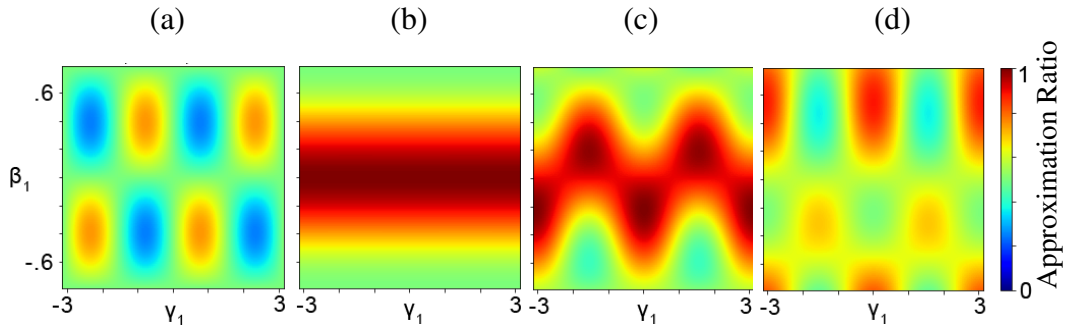

Figure 4.7: Parameter landscapes of the instance-specific approximation ratio of the four-cycle C_4 for $p = 1$. When no warm start is used, the landscape has many peaks and valleys in the form of local maxima and minima (a). For both BM-MC$_2$ and BM-MC$_3$, a vertex-at-top rotation yields a convex landscape with a ridge-like shape (b), thereby effectively capturing the optimal solution for the 4-cycle. When a uniform rotation is used, a BM-MC$_2$ formulation (c) achieves optimality for *some* choice of parameters whereas this is not the case for a BM-MC$_3$ formulation (d).

positive integer k, we say that $\mathbf{x} \in (\mathbb{R}^k)^n$ has an *optimal antipodal structure (in* $\mathbb{R}^k$*)* for a graph $G = (V, E)$ if there exists $S \subseteq V$ and a unit vector $u \in \mathbb{R}^k$ such that $(S, V \setminus S)$ is a Max-Cut of G and $x_i = u$ if $i \in S$ and $x_i = -u$ if $i \notin S$. That is, the points corresponding to each vertex lie at antipodal points on the sphere in a way determined by some Max-Cut of G. If we consider $|s_0\rangle = Q(R_V(\mathbf{x}))$ where R_V is a random vertex-at-top rotation and $\mathbf{x}$ has optimal antipodal structure, then we basically recover the Max-Cut. In this case, QAOA-warm with initial state $|s_0\rangle$ yields the Max-Cut of G for $p = 0$.

For any connected bipartite graph and any k (including $k = 2, 3$), one can prove that any globally optimal solution x of BM-MC$_k$ will have the antipodal structure described above. For the special case of even cycles, we find that the BM-MC$_3$ optimization of Max-Cut always finds the global optimum. These observations simply imply that random vertex-at-top rotations recover good solutions from the classical regime.

In Theorem 24 below, the characterization of local optima for even cycles simply implies that initializing QAOA-warm with the random vertex-at-top rotation $Q(R_V(\mathbf{x}^*))$ will also recover the Max-Cut. To prove the structure of local optima, we exploit the structure of the graph and utilize KKT conditions to first prove that any local optimum for BM-MC$_3$ has rank at most 2.[7] Further, we prove that any 2-dimensional solution can be improved

[7]In non-linear optimization, the Karush–Kuhn–Tucker (KKT) conditions are first-order necessary con-

locally, thus resulting in rank-1 local optimal (which thus corresponds to an optimal cut in the graph as a result of Lemma 23).

Lemma 23. *Let x^* be a locally-optimal solution rank-1 solution of BM-MC$_k$ on a graph $G = (V, E)$ (and edge weights $w : E \to \mathbb{R}$) for $2 \leq k \leq n$. Then the cut (S, T) corresponding to x is an optimal cut in G.*

Proof. We provide a proof for $k = 2$; the proof for higher values of k are obtained by restricting to the plane that contains both antipodal points of the rank-1 solution and running the $k = 2$ argument.

So suppose $k = 2$. By means of contradiction, suppose that the cut (S, T) that corresponds to x^* (a locally optimal rank-1 solution) is not an optimal cut. We will show there is a direction we can move in that improves the solution, thus contradicting the local optimality of x^*.

Since (S, T) is not the optimal cut in G, there is an even better cut, i.e., there are $A \subseteq S$ and $B \subseteq T$ such that $((S \setminus A) \cup B, (T \setminus B) \cup A)$ is a better cut than (S, T). For any sets $X, Y \subseteq V$, let $w(X, Y)$ denote the sum of the edges between X and Y. Note that by moving from the cut (S, T) to $((S \setminus A) \cup B, (T \setminus B) \cup A)$, the value of the cut increases by $w(A, S \setminus A) + w(B, T \setminus B)$ and decreases by $w(A, T \setminus B) + w(B, S \setminus B)$. Since this new cut is better, this net change must be strictly positive, i.e.,

$$\delta := w(A, S \setminus A) + w(B, T \setminus B) - (w(A, T \setminus B) + w(B, S \setminus A)) > 0.$$

Without loss of generality, suppose the rank-1 solution x^* is such that every vertex in S is at $(-1, 0)$ on the unit circle and every vertex in T is at $(1, 0)$. The solution x^* is depicted on the left side of Figure 4.8. We now perturb all the vertices in A and B by an arbitrarily small angle $\theta > 0$ so that all the vertices in A are now at $(-\cos\theta, \sin\theta)$ and all the vertices in B are now at $(\cos\theta, -\sin\theta)$ on the unit circle as seen on the right side of Figure 4.8.

ditions which characterize the set of optimal solutions. The usage of the KKT conditions generalizes the method of Lagrange multipliers [109][110].

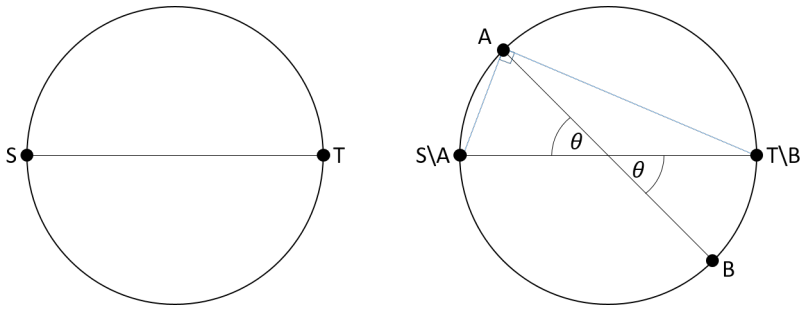

Figure 4.8: To the left is the rank-1 solution x^*. To the right is the solution x' after moving the points in A and B by angle θ along the unit circle. Thale's Theorem tells us that connecting the endpoints of the diameter of a circle with another point on the circle yields a right triangle.

We now analyze this new configuration of points x' on the unit circle and investigate the net change in the BM-MC$_k$ objective. Recall that maximizing the objective of the rank-k Burer-Monteiro relaxation is equivalent to maximizing:

$$\sum_{e=(i,j)\in E} w_{ij}\|x_i - x_j\|^2 \quad \text{s.t.} \quad \|x_i\| = 1, x_i \in \mathbb{R}^k \quad \forall i \in [n].$$

Let $d(A, S \setminus A)$ be the distance from any point in A to any point in $S \setminus A$ on the unit circle in x'. The quantities $d(B, T \setminus B), d(A, T \setminus B), d(B, S \setminus A)$ are defined similarly. Observe that the change in objective is given by:

$$\begin{aligned}\Delta :=& w(A, S \setminus A)(d(A, S \setminus A)^2 - 0) \\ &+ w(B, T \setminus B)(d(B, T \setminus B)^2 - 0) \\ &+ w(A, T \setminus B)(d(A, T \setminus B)^2 - 4) \\ &+ w(B, S \setminus A)(d(B, S \setminus A)^2 - 4).\end{aligned}$$

In the above, the 4 comes from the fact that originally (in x^*), the squared distance from A to $(T \setminus B)$ was $2^2 = 4$ (and similarly for B and $S \setminus A$). Note that points at $S \setminus A$ and $T \setminus B$ are antipodal in both x^* and x' so their distance doesn't change from x^* to x'; due to our construction, the same can be said about the points in A and B.

To complete the proof, it remains to show that $\Delta > 0$ for any choice of $\theta > 0$. By

Thale's Theorem, the triangle determined by the points at $S \setminus A, A, T \setminus B$ is a right triangle. Thus, by the Pythagorean theorem:

$$d(A, S \setminus A)^2 + d(A, T \setminus B)^2 = 4,$$

or alternatively:

$$d(A, T \setminus B)^2 - 4 = -d(A, S \setminus A)^2.$$

A similar calculation shows that:

$$d(B, S \setminus A)^2 - 4 = -d(B, T \setminus B)^2.$$

Making the above substitutions in the formula for Δ in addition to the substitution $d(A, S \setminus A) = d(B, T \setminus B)$ yields that:

$$\Delta = d(A, S \setminus A)^2 \Big(w(A, S \setminus A) + w(B, T \setminus B) - (w(A, T \setminus B) + w(B, S \setminus A)) \Big)$$

$$= d(A, S \setminus A)^2 \, \delta > 0,$$

which completes the proof. □

Theorem 24. *For a union of r even-cycles, any local optimum $\mathbf{x}^*$ for* BM-MC$_3$ *is a global optimum.*

Proof. Without loss of generality, let G be a cycle with n vertices. Let $x : V \to S^2$ be a local optimum for BM-MC$_3$. Our proof consists of two steps. First, we show rank(span($\{ x_v : v \in V \}$)) ≤ 2. Next, building upon this characterization for the local optima we show that in fact the above rank is exactly 1 and all edges are (fully) cut, i.e., global optimum is achieved.

We use first order necessary conditions, known as KKT, to derive the desired character-

ization. Let us formulate the Lagrangian for our constrained optimization problem,

$$\mathcal{L}(x, \alpha) = \sum_{(u,v)\in E} \|x_u - x_v\|^2 - \sum_{v\in V} \alpha_v(\|x_v\|^2 - 1)\,,$$

where $\alpha_v \in \mathbb{R}$ is a multiplier corresponding to the condition $\|x_v\| = 1,\ \forall v \in V$. It is easy to see the objective for our constrained optimization problem is equal to $\max_{x:V\to S^2} \min_{\alpha:V\to\mathbb{R}} \mathcal{L}(x, \alpha)$. Further using Lagrangian duality theory, we apply KKT optimality conditions that require (at any local optima) stationary condition $\frac{\partial \mathcal{L}}{\partial x_v} = 0$ is satisfied for all $v \in V$ in addition to the following feasibility and complementary slackness conditions (which are trivially satisfied):

- Primal feasibility requires $\|x_v\| = 1, \quad \forall v \in V$.
- Dual feasibility requires $\alpha_v \in \mathbb{R}, \quad \forall v \in V$.
- Complementary slackness requires $\alpha_v = 0$ whenever $\|x_v\| \neq 1$.

The stationary condition can be reformulated as

$$\frac{\partial}{\partial x_v}\Big(\sum_{(u,v)\in E} (x_v - x_u)^T(x_v - x_u) - \alpha_v(x_v^T x_v - 1)\Big) = 0 \quad \forall v \in V\,.$$

Thus, for all $v \in V$, we have the following stationary condition: $\sum_{(u,v)\in E}(x_v - x_u) = \alpha_v x_v$.

Considering our graph being a cycle, where every vertex $v \in V$ has two neighbors, the stationary condition implies a linear dependence of x_v with x_u's corresponding to its neighbors. Hence, rank(span($\{ x_v, x_w, x_u \}$)) ≤ 2 where w and u are two neighbors of v. Note that if this rank is 1, one can easily show neighbors of this vertex (and consequently for every vertex) are at antipodal points $x_w = x_u = -x_v$. Otherwise, x_w and x_u are mirrored with respect to span($\{ x_v \}$). In this case, these three vectors lie on a unique plane. Inductively, one can show all vertices of the cycle lie on the same plane.

With all the points lying on the same plane, it remains to show that the additional dimension (direction) in $\mathbb{R}^3$ allows one to (locally) increase the objective. Without loss of generality, let $x_v \in \mathbb{R}^3$ be a point on the unit sphere with polar angle $\theta = \pi/2$ and azimuthal angle φ_v. Coloring vertices of the cycle by two colors $c_v \in \{1, 2\}$, it is easy to see for $\tilde{x}_v = (1, \pi/2 + (-1)^{c_v}\varepsilon, \varphi_v)$ all edges stretch (unless they are antipodal) so the objective increases. This shows x was not a local optimum, in case of a rank 2 assignment.

Lemma 23 guarantees that locally optimal rank-1 solutions correspond to an optimal cut in G and since we have shown all local optima are rank-1, it must be the case that every local optimum x^* is also a global optimum.[8]

□

It is conjectured that the performance of standard QAOA for n-node even cycles is $(2p+1)/(2p+2)$ when $n > 2p$ [1, 99]. The above theorem is a concrete example where QAOA-warm outperforms standard QAOA, due to a warm-start with a classical optimal solution. We find that warm-starts often result in flatter parameter landscapes for QAOA-warm, e.g., see Figure 4.7 depicting the landscapes for various variants of QAOA-warm on cycle C_4 on four vertices (i.e. $C_4 = (V, E)$ with $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (2, 3), (3, 4), (1, 4)\}$). For the vertex-at-top rotation in particular, notice that the solution quality monotonically decreases in $|\beta_1|$ with the optimal parameters all lying on the line $\beta_1 = 0$. We make this precise in the following observation.

Observation 25. *Let $k \in \{2, 3\}$ and $G = (V, E)$ be a graph with weights $w : E \to \mathbb{R}$ and $S \subseteq V$ be such that $(S, V \setminus S)$ is a Max-Cut of G. Let y be a unit vector in $\mathbb{R}^k$. Let $\mathbf{x}$ be such that $x_u = y$ if $u \in S$ and $x_u = -y$ if $u \notin S$. If we initialize QAOA-warm with the initial state $|s_0\rangle = Q(R_V(\mathbf{x}))$ where R_V is a random vertex-at-top rotation, then, we recover the Max-Cut since the states are aligned with $|0\rangle$ and $|1\rangle$. The expected cut value*

[8]Note that Lemma 23 alone does not guarantee that a locally optimal rank-1 solution is also a globally optimal solution; one has to use the fact that *all* locally optimal solutions are rank-1 in Theorem 24 in order to deduce that the rank-1 solution is globally optimal.

at (γ_1, β_1) *is given by*

$$F_1(\gamma_1, \beta_1) = \frac{1}{4}\Big((2M - W)\cos(4\beta) + 2M + W)\Big),$$

where $M =$ *Max-Cut*(G) *and* $W = \sum_{e \in E} w_e$. *Observe that* $F_1(\gamma_1, 0) =$ *Max-Cut*(G) *for all* $\gamma_1 \in \mathbb{R}$ *and* $F_1(\gamma_1, \beta_1)$ *decreases as* $|\beta_1|$ *increases for all* $|\beta_1| \in [0, \pi/4]$.

Proof. Since we are working only with circuit depth $p = 1$, for simplicity, we use γ and β to denote γ_1 and β_1 respectively.

If $|s_0\rangle = Q(R_V(\mathbf{x}))$ where $R_V(\cdot)$ is a vertex-at-top rotation and $\mathbf{x}$ is as described in the statement of the observation, then it is straightforward to see that there exists a reordering of the vertices such that $|s_0\rangle = |0\rangle^{\otimes r} \otimes |1\rangle^{\otimes(n-r)}$ where $|S| = r$ (where the first r qubits correspond to vertices in S).

Let $M =$ Max-Cut(G). Since H_C is the Hamiltonian of the Max-Cut problem and $|s_0\rangle$ corresponds to an optimal cut, then $|s_0\rangle$ is a eigenvector of H_C with eigenvalue M. Thus,

$$e^{-i\gamma H_C} |s_0\rangle = e^{-i\gamma \cdot M} |s_0\rangle = \alpha |s_0\rangle . \tag{4.8}$$

where $\alpha = e^{-i\gamma M}$. Using equation (4.8), we have that

$$|\psi_1(\gamma, \beta)\rangle = e^{-i\beta H_B} e^{-i\gamma H_C} |s_0\rangle = \alpha e^{-i\beta H_B} |s_0\rangle = \alpha \left(|s_\beta\rangle^{\otimes r} \otimes |s'_\beta\rangle^{\otimes(n-r)}\right),$$

where $s_\beta = \cos(\beta)|0\rangle - i\sin(\beta)|1\rangle$ and $s'_\beta = \cos(\beta)|1\rangle - i\sin(\beta)|0\rangle$.

The expected energy is the sum of the expected energy for each edge $(u, v) \in E$ and each edge contributes a non-zero amount if and only if both endpoints have a different spin after measurement. However, since $|\psi_1(\gamma, \beta)\rangle$ is an unentangled state, then we can consider measuring each vertex independently.[9] Consider an edge $(u, v) \in E$ and suppose

[9]The α term is a global phase change that doesn't affect the measurement and can thus be ignored.

that $u \in S$ and $v \in S$. Then,

$$\begin{aligned}
&\Pr(u \text{ and } v \text{ measured to be } |0\rangle \text{ and } |1\rangle \text{ respectively}) \\
&= \Pr(u \text{ measured to be } |0\rangle) \cdot \Pr(v \text{ measured to be } |1\rangle) && (|\psi_1(\gamma,\beta)\rangle \text{ is unentangled}) \\
&= \Pr(s_\beta \text{ measured to be } |0\rangle) \cdot \Pr(s_\beta \text{ measured to be } |1\rangle) && \text{(by construction)} \\
&= \cos^2(\beta) \cdot \sin^2(\beta). && (\text{def of } s_\beta)
\end{aligned}$$

Similar calculations show that if $u \in S$ and $v \in S$, then the probability that u is measured to be $|1\rangle$ and v is measured to be $|0\rangle$ is also $\cos^2(\beta)\sin^2(\beta)$. In the case that $u \in S$ and $v \notin S$, one can similarly show that the probability of measuring both to have differing spins is given by $\cos^4(\beta) + \sin^4(\beta)$.

Combining all the calculations above, the expected energy is given by

$$F_1(\gamma,\beta) = 2(W - M)\sin^2\beta\cos^2\beta + M(\sin^4\beta + \cos^4\beta),$$

where W is the sum of all edge weights (i.e. $W = \sum_{e\in E} w_e$).

Observe that when $\beta = 0$, the above equation reduces to $F_1(\gamma,\beta) = M$ as desired. By applying various trigonometric identities and algebraic manipulations, we can rewrite the above function as

$$F_1(\gamma,\beta) = \frac{1}{4}\Big((2M - W)\cos(4\beta) + 2M + W)\Big).$$

The Max-Cut of a graph is at least half the sum of the edge weights, i.e., $M \geq W/2$ (which implies $2M - W \geq 0$).[10] Since $\cos(4\beta)$ is decreasing in $|\beta|$ for $\beta \in [-\pi/4, \pi/4]$, then it must be that F_1 is decreasing (in $|\beta|$ for $\beta \in [-\pi/4, \pi/4]$). □

[10]To see this, observe that the expected sum of the weights of the edges crossing a random cut (where one independently places each vertex on one side of the cut or the other with probability $1/2$) will be $W/2$. Since the expectation is $W/2$, then there must exist at least one cut where the sum of the weights crossing the cut is *at least* $W/2$, i.e., $M \geq W/2$.

The form of the expression for $F_1(\gamma_1, \beta_1)$ follows from the fact that the cost term of the quantum circuit has no effect and that β_1 can be interpreted as a rotation angle (about the x-axis) in the Bloch sphere that moves the state away from the measurement axis.

In contrast to the vertex-at-top rotations preserving the optimality of antipodal solutions (Observation 25), this is not always the case for uniform rotations. In Figure 4.7 for example, the *uniform* rotation does not yield the optimal cut for C_4 for any choice of parameters when 3-dimensional solutions are used. However, if we instead use the BM-MC$_2$ solution with a uniform rotation to obtain $R(\mathbf{x})$, then there *does* exists a combination of parameter values that yields the optimal cut (by choosing $\gamma_1 = 0$ and an appropriate choice of β_1, application of the quantum circuit can be interpreted as a rotation that aligns $R(\mathbf{x})$ with the measurement axis in this case). This is due to potential proximity of uniformly rotated 3-dimensional solutions to the eigenstates of the mixer, which we can avoid in 2-dimensional initializations as discussed in Section 4.3.

4.5 Numerical Simulations for QAOA-Warm

In this section, we discuss the results of our numerical simulations of QAOA-warm. We first discuss the details of the warm-start constructions and the graph instances used in Section 4.5.1. In order to compare QAOA-warm to other Max-Cut algorithms, one can use different black-box optimizers, such as ADAM, COBYLA, Nelder-Mead and BFGS. We first run computations to pick a single optimizer, then to pick the rank of the initialization, and the rotation scheme to work with in Sections 4.5.2 and 4.5.3. In Section 4.5.4, we next provide aggregate results for QAOA-warm including (i) a comparison against other Max-Cut algorithms, (ii) improvement in instance-specific approximation ratio with increased p depth, and (iii) trends in (median) instance-specific approximation ratio with varying p-depth and graph size. Lastly, to understand the behavior of QAOA-warm, we discuss the qualitative shape and numerical properties of the parameter landscape of QAOA-warm (and standard QAOA) in Section 4.5.5.

4.5.1 Experimental Setup

Graph Instances

We consider a collection of 1264 graphs, $\mathcal{G}$, generated as follows. We first generated a set of unweighted graphs, which includes all non-isomorphic graphs for $n = 2$ to $n = 6$ vertices (142 instances), and 29 random graphs for each size $n = 7$ to $n = 12$ sampled from different random graph generators in PYTHON's NETWORKX [111] package. These random graph generators include Erdös-Renyí, Barabasi Albert, Dual of Barabasi-Albert, Watts-Strogatz, and Newman-Watts-Strogatz models. Many experimental studies in the current QAOA literature only consider graphs from a single random graph model (e.g. Erdös-Renyí); however graphs from such models can have predictable behavior when it comes to Max-Cut[11] which could potentially have a large influence on the performance of QAOA. For this reason, we construct an ensemble of graphs $\mathcal{G}$ using a variety of random graph generators; this library has been made available online [104].

Construction of Unit-Weight Graphs

Below, we describe the collection of unit-weight graphs that were generated in the process described above.

The collection of non-isomorphic graphs up to 6 nodes were generated using `SageMath` [112]. The remaining instances were generated using various random graph generators found in the `NetworkX` Python package [111]; the parameter names used below are the same as those used in the corresponding `NetworkX` functions. The library consists of the following graphs:

- All non-isomorphic connected graphs up to 6 nodes (142 instances)

[11] For example, when using the Erdös-Rényi graph model, if each edge appears independently with probability q, and if we take a random cut with k vertices on one side and $n - k$ vertices on the other side, then one would observe $qk(n - k)$ edges across the cut (in expectation).

- Erdös-Renyí (42 instances): for each n from $n = 7$ to $n = 12$, 7 instances with p sampled from $[0, 1]$ uniformly were created.

- Random Regular (42 instances): for each n from $n = 7$ to $n = 12$, 7 instances with d sampled uniformly from valid degrees were created.

- Barabasi Albert (18 instances): for each n from $n = 7$ to $n = 12$ and for all m in $\{1, 2, 3\}$, 1 instance (with initial graph being star graph on $m + 1$ nodes) was created

- Dual Barabasi Albert (36 instances): for each n from $n = 7$ to $n = 12$ and for all $\{(m_1, m_2) : m_1, m_2 \in \{1, 2, 3\}, m_1 \neq m_2\}$ with $p = 0.25$, 1 instance with initial graph on star with $\max(m_1, m_2) + 1$ nodes was created

- Watts Strogatz Graphs (18 instances): for each n from $n = 7$ to $n = 12$, for all k in $\{2, 4, 6\}$, 1 instance with p sampled uniformly from $[0, 1]$ was created.

- Newman Watts Strogatz Graphs (18 instances): for each n from $n = 7$ to $n = 12$, for all k in $\{2, 4, 6\}$, 1 instance with p sampled uniformly from $[0, 1]$ was created.

Figure 4.9 demonstrates how varied our ensemble is with respect to two important graph metrics dependent on eigenvalues of the normalized Laplacian [113].

Construction of Weighted Graphs

Next, we create three weighted versions of each of the 316 unit-weighted instances constructed above, by considering independent edge-weightings drawn from (i) uniform distribution on $\{-10, -9, \cdots, 9, 10\} \setminus \{0\}$, (ii) uniform distribution on $\{1, 2, \ldots, 10\}$, and (iii) weights of form $\pm 2^k$ with $\Pr[w_e = 2^k] = \Pr[w_e = -2^k] = 2^{-k-2}$ for all non-negative integers k. The weighted and unweighted instances together give us a total of 1264 instances. The last family of weighted instances is constructed due to high variation of performance of classical heuristics on similar instances, observed in a previous study by Dunning et al. [96] and discussed in Section 3.3.1. Note that some of the ways of sampling edge-weights

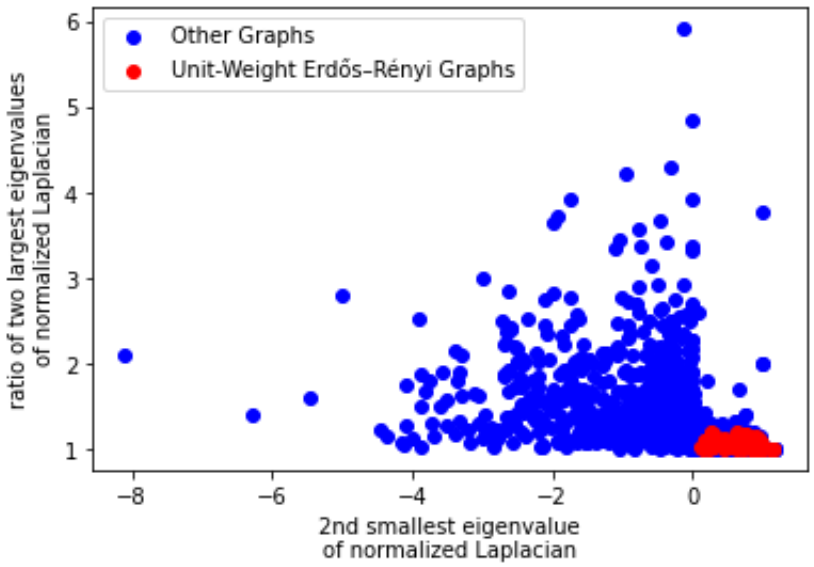

Figure 4.9: Illustration depicting the range of graph metrics for our instance library $\mathcal{G}$. When comparing unit-weight Erdős-Rényi graphs (red) with the remaining graphs in $\mathcal{G}$ (blue), there is an increase in the range of values obtained for both graph metrics.

results in only positive edge-weight graphs. We will often present results for mixed-weight graphs (positive and negative weights), and positive-only separately.

Experiment Description

We computed QAOA-warm, Goemans-Williamson and standard QAOA solutions for each of the weighted graph instances $G \in \mathcal{G}$. Both standard QAOA and QAOA-warm were run for circuit depths $p \in \{1, 2, 4, 8\}$, for each optimizer considered, and QAOA-warm for each considered rank of BM-MC$_k$ ($k = 2, 3$) and for each rotation type (vertex-at-top and uniform random). We consider the best of 5 warm-starts (in objective value) when selecting BM-MC$_k$ warm-starts, and subsequently the best of 5 random rotations, i.e., the rotation that yields the highest expected cut value at the end of the hybrid-optimization loop. In the experiments, we compute the expected cut values exactly (details in Section 2.2) rather than simulating quantum measurements or hyperplane rounding. For any two runs of standard QAOA or QAOA-warm that differ only in choice of optimizer, the initial parameter values used are the same (with γ_i and β_i sampled uniformly from the interval $[-0.0001, 0.0001]$

for all $i = 1, \ldots, p$). Our implementation of standard QAOA and QAOA-warm utilizes Google's TENSORFLOW QUANTUM library and IBM's Qiskit library. The state $|+\rangle^{\otimes n}$ is initialized by applying a Hadamard gate to each qubit in $|0\rangle^{\otimes n}$. For states initialized based on low-rank approximate solutions, we generate the initial state as discussed in Section 4.1, which is easily implemented using standard rotation gates. For each epoch of each run of standard QAOA or QAOA-warm, our implementation records the values of the variational parameters, the expected cut value at those parameters, and the probability distribution of all 2^n cuts. Each run of standard QAOA and QAOA-warm terminated when the difference in successive values of $F_p(\gamma, \beta)$ was less than $\bar{W} * 10^{-6}$, where $\bar{W}$ is the sum of the absolute values of the edge weights. We next summarize the results from these numerical simulations.

4.5.2 Optimizer Choice

We consider four different optimizers to optimize the $2p$ variational parameters: ADAM, BFGS, Nelder-Mead, and COBYLA and present comparisons between these set of optimizers. As demonstrated in Figure 4.10, when ADAM is compared to the other three optimizers, the expected cut values obtained for QAOA-warm are similar (i.e. within 0.01 difference in instance-specific AR) for at least 90% of the runs at $p = 1$; this percentage decreases at $p = 8$ but the instance-specific approximation ratios are still relatively similar for the majority of the instances. This suggests that the instance-specific approximation ratios achieved for QAOA-warm are largely independent of the optimizer used; for this reason, all remaining results involving instance-specific approximation ratios for QAOA-warm will be in terms of runs using the ADAM optimizer. It should be noted that all of the optimizers considered vary in regards to runtime, e.g., the cost per iteration and the number of iterations required to train the variational parameters (we discuss this further in Section 4.5.6)

Even though the choice of the optimizer had almost no impact on the QAOA-warm

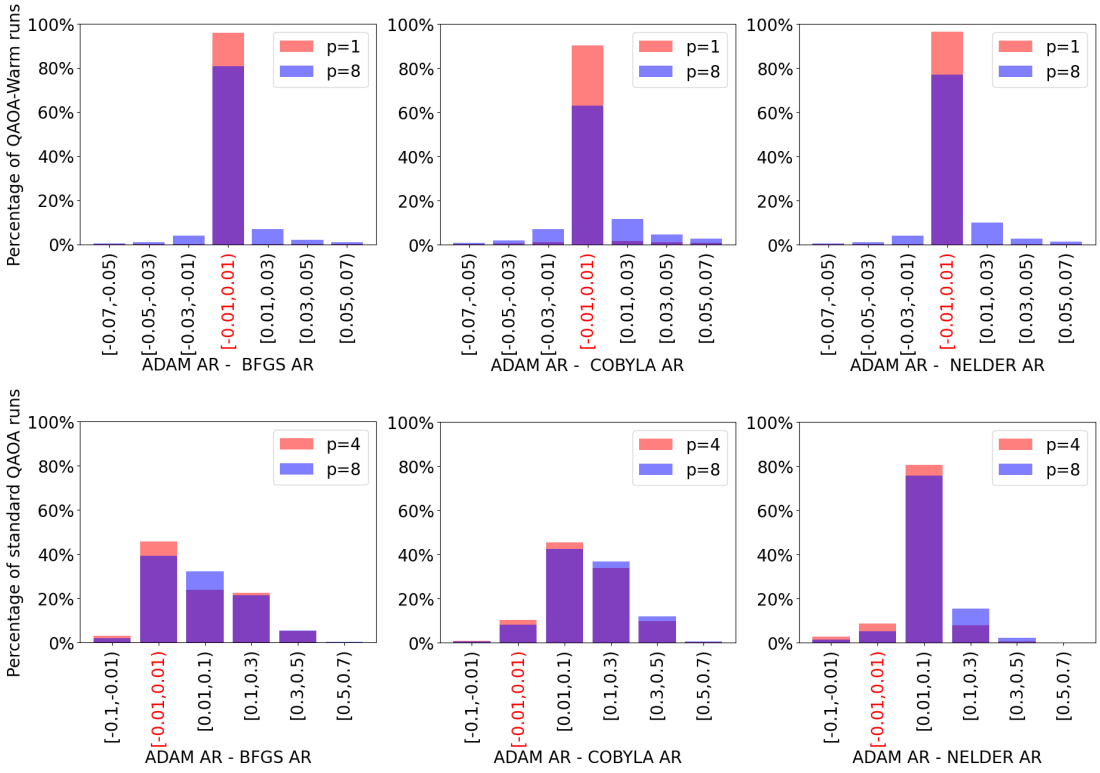

Figure 4.10: Histogram of differences in instance-specific approximation ratio (AR) $\alpha_{\mathcal{A},G}$ between ADAM and other optimizers for QAOA-Warm (top) and standard QAOA (bottom). Overlapping regions are in purple. The red bin indicates instances for which optimizers performed similarly to ADAM.

Table 4.1: Multiple tables comparing the average instance-specific approximation ratio achieved during QAOA-warm when utilizing different combinations of ranks and rotations during the preprocessing stage. For the top row of tables, these averages were computed using all the graphs in our graph library $\mathcal{G}$ (see Section 4.5.1) whereas for the bottom row, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights. Each run of standard QAOA and QAOA-warm terminates when the difference in successive values of $F_p(\gamma, \beta)$ is less than $10^{-6}\bar{W}$ where $\bar{W}$ is the sum of the absolute values of the edge weights.

	depth $p = 1$			depth $p = 8$		
all graphs		vert.	uniform		vert.	uniform
	2-dim.	**0.9581**	0.9581	2-dim.	**0.9726**	0.9718
	3-dim.	0.9576	0.9440	3-dim.	0.9688	0.9560
positive-weight graphs		vert.	uniform		vert.	uniform
	2-dim.	0.9569	**0.9569**	2-dim.	**0.9704**	0.9697
	3-dim.	0.9556	0.9441	3-dim.	0.9659	0.9548

in terms of the instance-specific approximation factors obtained, Figure 4.10 illustrates a noticeable effect on $\alpha_{\mathcal{A},G}$ achieved for standard QAOA especially at the higher circuit depths that we tested for ($p = 4$ and $p = 8$).[12] In particular, we find that runs using the ADAM optimizer tend to have better performance for standard QAOA. For this reason, the remaining results in this paper regarding standard QAOA will only include runs that utilize the ADAM optimizer in order to obtain a more simple, direct, and fair comparison with QAOA-warm.

4.5.3 Choice of Rank and Rotations

To compare QAOA-warm against standard-QAOA, the GW algorithm, and hyperplane rounding of BM-MC$_2$, we need to narrow in to the choice of the BM-MC$_k$ rank (2 or 3) and the type of rotation (vertex-at-top or uniform random) to use. We explore these two choices in this subsection.

Recall that we consider the best-of-5 warm-starts for each type of rotation[13]. Over the 1264 graph instances, for 3-dimensional initializations, we find that the vertex-at-top rotations typically have a slight increase in performance over random uniform rotations, es-

[12] We suspect this is an artefact of the parameter landscapes becoming flatter with the warm-starts.

[13] We found that restarting standard QAOA multiple times did not impact the results significantly.

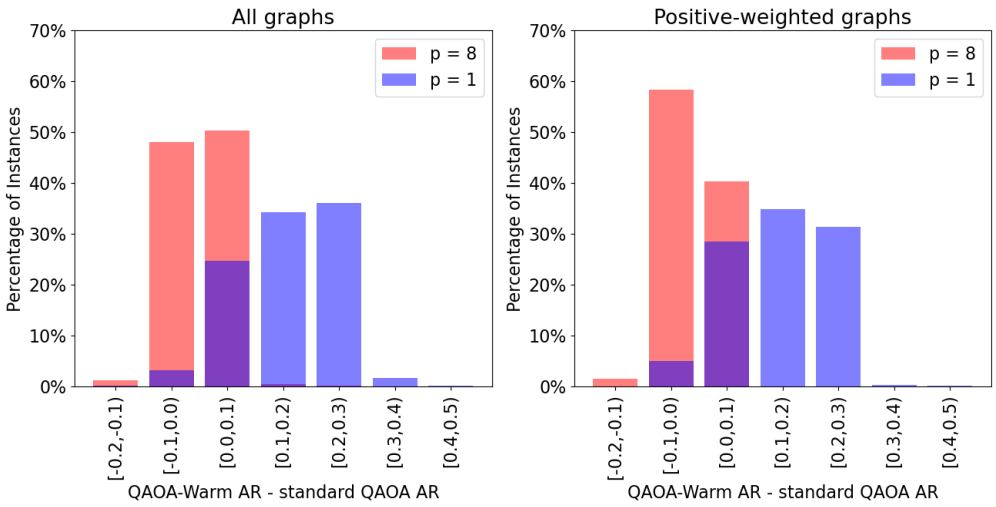

Figure 4.11: Histogram of differences in instance-specific approximation ratios between QAOA-warm and standard QAOA. Overlapping regions are in purple.

pecially when 3-dimensional solutions are used (e.g., at depth $p = 1$, 3-dimensional vertex-at-top rotations obtain 0.9576 instance-specific approximation ratio on average, whereas 3-dimensional uniform rotations obtain 0.9440). These results seem reasonable since vertex-at-top rotations rarely end up in states that plateau for warm-starts (see Section 4.3 for an example of such a warm-start). We include a summary of average instance-specific approximation ratios observed across the four choices of rank and rotations in Table 4.1.

On the other hand, when using 2-dimensional initializations, there is virtually no difference between the two rotation approaches, as 2-dimensional solutions were specifically designed to avoid bad states for warm-starts. For the ease of presentation, the remainder of the results in this paper will utilize 2-dimensional initializations with a vertex-at-top rotation scheme as this appears to be one of the most promising combinations for QAOA-warm.

4.5.4 Aggregate Results

Here we use aggregated results of QAOA-warm in order to answer three key questions: (Q1) How does QAOA-warm fare compare to standard QAOA and classical Max-Cut algorithms (BM-MC and Goemans-Williamson), (Q2) How much of QAOA-warm's instance-specific approximation ratio can be attributed to the warm-start itself v/s what is done by the

Table 4.2: We consider 4 algorithms: Goemans-Williamson (G), 2-dimensional Burer-Monteiro with hyperplane rounding (B), QAOA-warm (W), and standard QAOA (S). There is a row for each of the $4! = 24$ ways the algorithms can perform relative to one another with the cell value indicating the percentage of instances for which that ordering occurs. As an example, the top-leftmost value indicates that for 25.08% of instances, $W \geq B \geq G \geq S$ in terms of AR with W and S being depth-1. The four largest entries in each column are bolded for emphasis. To account for numerical error for nearly solved instances, we declare QAOA-warm (W) as the best as long as it is within 0.001 AR of the best algorithm. We include columns corresponding to the entire graph library $\mathcal{G}$ as well as the subset of $\mathcal{G}$ that have positive-weighted edges.

	p=1		p=2		p=4		p=8	
	all	positive	all	positive	all	positive	all	positive
WBGS	**25.08**%	**25.08**%	**26.42**%	**26.42**%	**23.97**%	**23.97**%	**16.61**%	**16.61**%
WBSG	0.00%	0.00%	0.16%	0.31%	0.32%	0.31%	0.71%	0.79%
WGBS	**30.93**%	**30.93**%	**32.28**%	**32.28**%	**29.98**%	**29.98**%	**21.84**%	**21.84**%
WGSB	0.00%	0.00%	0.24%	0.31%	0.32%	0.47%	2.06%	3.30%
WSBG	0.00%	0.00%	0.16%	0.16%	2.61%	2.52%	4.27%	4.56%
WSGB	0.08%	0.16%	0.00%	0.00%	2.29%	2.04%	4.27%	3.62%
BWGS	0.95%	1.89%	0.71%	1.10%	0.16%	0.16%	0.00%	0.00%
BWSG	0.08%	0.16%	0.24%	0.47%	0.08%	0.00%	0.00%	0.00%
BGWS	**22.39**%	**22.39**%	**17.01**%	**17.01**%	**6.25**%	6.60%	1.34%	0.47%
BGSW	1.50%	2.36%	4.83%	5.50%	**10.44**%	**10.44**%	4.03%	6.29%
BSWG	0.24%	0.47%	0.32%	0.63%	0.71%	1.26%	0.63%	1.26%
BSGW	0.32%	0.63%	0.24%	0.47%	1.34%	1.89%	1.11%	2.04%
GWBS	1.50%	1.89%	1.50%	1.89%	1.27%	1.57%	0.47%	0.31%
GWSB	0.08%	0.16%	0.08%	0.16%	0.08%	0.16%	0.55%	0.94%
GBWS	**15.66**%	**15.66**%	**10.92**%	**10.92**%	4.91%	4.87%	1.11%	0.63%
GBSW	0.71%	0.63%	3.72%	3.62%	6.09%	**6.09**%	2.69%	4.72%
GSWB	0.00%	0.00%	0.00%	0.00%	0.08%	0.16%	0.40%	0.63%
GSBW	0.00%	0.00%	0.00%	0.00%	0.08%	0.16%	0.24%	0.31%
SWBG	0.00%	0.00%	0.00%	0.00%	1.27%	1.57%	**8.86**%	**8.86**%
SWGB	0.16%	0.31%	0.32%	0.47%	1.74%	1.73%	8.23%	7.39%
SBWG	0.00%	0.00%	0.40%	0.79%	0.87%	1.57%	1.42%	2.20%
SBGW	0.16%	0.31%	0.24%	0.31%	2.69%	2.67%	**11.71**%	**11.71**%
SGWB	0.16%	0.31%	0.16%	0.31%	0.08%	0.16%	0.08%	0.16%
SGBW	0.00%	0.00%	0.08%	0.00%	2.37%	1.73%	7.36%	8.33%
Total	100%	100%	100%	100%	100%	100%	100%	100%

quantum circuit, and (Q3) What are the trends in QAOA-warm's instance-specific approximation ratio with varying depth and graph size and how does this compare with standard QAOA?

(Q1). To answer the first question, we compare standard QAOA, QAOA-warm, the GW algorithm, and hyperplane rounding of the BM-MC$_2$ solutions in Table 4.2. At depth-1, QAOA-warm is at least as good as the other three algorithms for 56.1% of the instances meanwhile standard QAOA is the best for less than 1% of the instances. However, as the circuit depth increases, standard QAOA is the best algorithm for a larger proportion of instances (37.66% of instances at depth $p = 8$); meanwhile, QAOA-warm is still at least as good as the other algorithms for 49.8%, nearly half, of the instances. These results empirically support our claim that warm-starts show improvements in performance of QAOA at low circuit depths. Since standard QAOA achieves the optimal cut in the limit as the circuit depth increases and thus, for any particular graph, there exists some (instance-dependent) circuit depth p for which standard QAOA beats the GW algorithm [1]. Current and near-term quantum devices can only reliably run QAOA for low circuit depth (due to the presense of quantum noise), and therefore we propose that QAOA-warm can be of significant use in this regime. Although our current implementation of QAOA-warm does not perform as well at higher circuit depths (compared to standard QAOA), we later extend QAOA-warm to QAOA-warmest in the next chapter by changing the mixer which we find yields better performance with increased circuit depth.

We next consider the difference in instance-specific approximation ratios obtained by QAOA-warm and standard QAOA. In Figure 4.11, we provide a detailed comparison between instance-specific approximation ratios attained by QAOA-warm and standard QAOA in the form of a histogram. We see improvements in the instance-specific approximation ratio ranging from 0.1 to 0.5 when using warm-starts, especially at low circuit depth. These results are consistent with those depicted in Table 4.2. We note that in this figure, as in the others, we take the best of 5 vertex-at-top rotations for QAOA-warm; and in Section 4.5.7,

we include results in the case where the median and worst (of 5) vertex-at-top rotations are used instead.

(Q2). We now address the second key question regarding how much of the performance of QAOA-warm can be attributed to the warm-start itself. This is an important question to address because if the improvement generated by QAOA-warm is due only to the initial quantum state at $p = 0$ having higher overlap with good solutions, then there would be no point in running the quantum device. To test this, we compare, in Figure 4.12, the improvement in instance-specific approximation ratio from depth-0 QAOA-warm (i.e. just measuring the initial state obtained from the preprocessing stage) to depth-1 QAOA-warm, as well as the improvement when we change the depth from 1 to 8. For 74 instances, we observed that the instance-specific approximation ratio from QAOA-warm improved by at least 50% when going from $p = 0$ to $p = 1$ and by at least 80% for 22 instances. This shows the promise of using QAOA on top of the warm-starts. On the other hand, the increase in instance-specific approximation ratio from depth-1 QAOA-warm to depth-8 QAOA-warm is milder, ranging upto 10% for positive-weighted instances and upto 22.3% for general graphs. These results show that empirically, running QAOA-warm does yield an increase in instance-specific approximation ratio beyond simply sampling the initial warm-start state; however, the returns diminish with higher circuit depths (this is expected because QAOA-warm can plateau for some instances, Section 4.3).

(Q3). Lastly, to address the third question, we consider how the performance of QAOA-warm varies across n (number of nodes) and p (circuit depth), which we illustrate for our graph library in Figure 4.13. While there is a significant improvement in performance for standard QAOA with increasing circuit depth, we find that QAOA-warm consistently outperforms standard QAOA (on average), except at $p = 8$. We also see that at fixed depth, the performance of both standard QAOA and QAOA-warm degrades as the number of nodes increases, while the degradation of QAOA-warm is much flatter compared to standard QAOA. We further discuss pre-processing and parameter search time for QAOA-

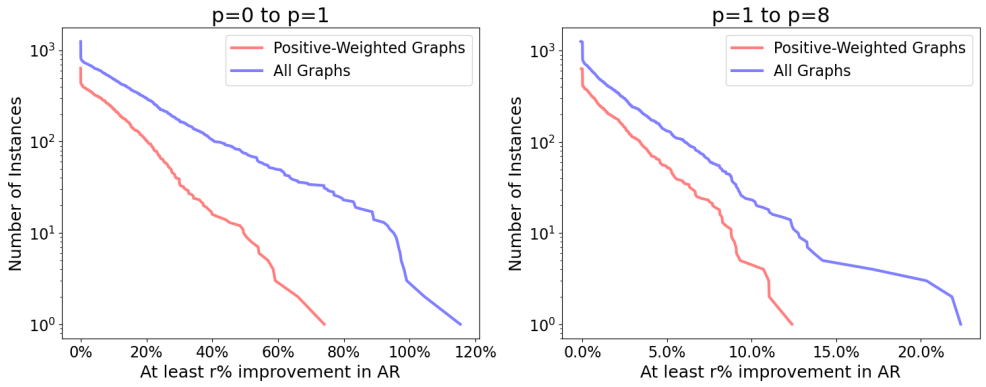

Figure 4.12: The number of instances for which QAOA-warm obtained at least an $r\%$ improvement in AR as the circuit depth increases from $p = 0$ to $p = 1$ (left) and from $p = 1$ to $p = 8$ (right). For each instance, the best percent improvement (across all five vertex-at-top rotations) is used. Note that % improvements in instance-specific approximation ratios go up to 80-120% from $p = 0$ to $p = 1$, and up to 12-20% from as depth increases from $p = 1$ to $p = 8$.

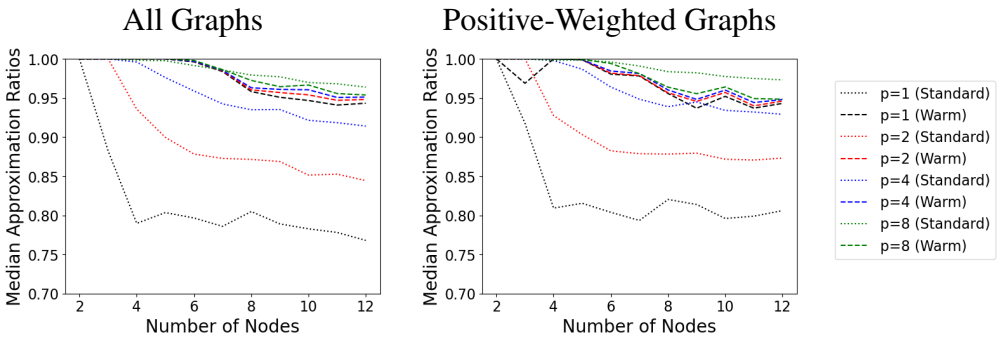

Figure 4.13: This figure shows how standard QAOA (dotted) and QAOA-warm (solid) perform as we alter the circuit depth and the number of nodes. For QAOA-warm, we take the best of 5 vertex-at-top rotations. For the left plot, for each $n = 2, \ldots, 12$, we find the instance-specific approximation ratio achieved for both standard QAOA and QAOA-warm for each n-node instance in $\mathcal{G}$ (see Section 4.5.1), and take the median of those instance-specific approximation ratios. The right plot is constructed similarly except only instances in $\mathcal{G}$ with positive edge-weights are considered. We plot the results for circuit depths $p = 1, 2, 4, 8$.

warm in Section 4.5.6.

4.5.5 Parameter Landscapes and Trajectories

We now consider looking at *all* parameter combinations for γ and β in order to obtain a better understanding of the landscape that we need to optimize over for standard QAOA and QAOA-warm. For any graph G, initial state $|s_0\rangle$, and circuit depth $p = 1$, we can plot a *parameter landscape* which allows us to visualize the solution quality as a function of the variational parameters γ_1 and β_1. In particular, each point (γ_1, β_1) in the landscape is

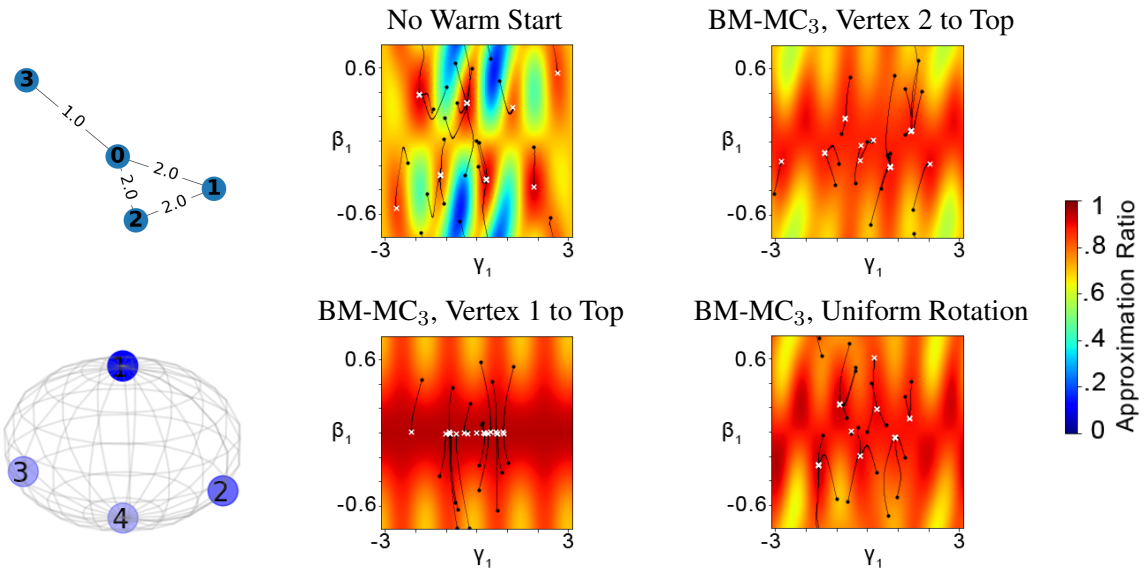

Figure 4.14: Parameter landscapes for $\hat{G}$ (top-left) with corresponding SDP solution (bottom-left). For each trajectory of optimization of the variational parameters, we use a black circle to denote the beginning of the trajectory and a white $\times$ to denote the end of the trajectory. When no warm start is used, there are many peaks and valleys (top-center). When vertex 1 rotated to the top; we have a ridge-like landscape with the optimal solutions occurring on the horizontal line $\beta_1 = 0$ (bottom-center). When rotating vertex 2 at the top instead, the parameter landscape is less ridge-like and the endpoints of the trajectories are more scattered (top-right). When using a uniform rotation we have peaks and valleys similar to when no warm-start was used but with overall better solution qualities (bottom-right).

assigned a color which corresponds to the instance-specific approximation ratio (i.e. the quantity $\frac{F_1(\gamma,\beta)-\text{MIN-CUT(G)}}{\text{Max-Cut}(G)-\text{MIN-CUT(G)}}$).

As an example, we plot the parameter landscape for graph $\hat{G}$ in Figure 4.14 without and with warm-starts (using 2 vertex-at-top rotations and one uniform rotation). For each parameter landscape, we ran the QAOA training loop twenty times with random initializations of (γ_1, β_1) and overlayed the trajectories of the parameter values throughout the training loop for the variational parameters. When no warm-start is used, the parameter landscape has many peaks and valleys and a wide range of solution qualities; using a warm-start drastically changes the landscape. However, if we rotate one of the approximate solution of BM-MC$_3$ for $\hat{G}$ using a vertex-at-top rotation, this yields a ridge-like parameter landscape where the optimal parameter values lie near the line $\beta_1 = 0$. This behavior is no longer there for a different vertex-at-top rotation for the same approximate solution. The endpoints of the optimization trajectories on the resultant are scattered, and the ridge-like

shape is not as pronounced. When performing a uniform rotation, the globally optimal solution qualities are comparable to the solution qualities when rotating vertex 1 to the top; however, the landscape retains some less symmetric peaks and valleys and some of the trajectories end at local optima that are far from optimal.

Overall, we see that the rotation used in the preprocessing stage can have a considerable effect on both the shape of the landscape and the solution qualities. Ideally, with a good choice of rotation, the parameter landscape has a ridge-like shape with high solution qualities near the line $\beta_1 = 0$, in which case, $\gamma = \beta = \mathbf{0}$ is a natural choice of initialization when running QAOA-warm.

To quantify flatness of the parameter landscapes when using warm-starts, we consider some simple aggregate statistics of the landscapes of all unit-weight graphs[14] in $\mathcal{G}$. For each graph, we view each point in the parameter landscape as producing an instance-specific approximation ratio in [0,1]. We compute the minimum, maximum, and average instance-specific approximation ratios found across each landscape[15]. As empirically shown in Figure 4.15, QAOA-warm landscapes have lower range of instance-specific approximation ratios, e.g., 80.4% of the instances have a range of at most 0.4 in the instance-specific approximation ratios attained in the landscape. This means that any two choices of γ_1, β_1 parameters will produce solutions with a difference in instance-specific approximation ratio of at most 0.4. In contrast, only 27.5% of our graph instances have such a range of instance-specific approximation ratios for the standard QAOA. We further see that when we use warm-starts, the overall quality of approximation across the parameter landscape improves. This can be seen by observing a higher minimum, maximum, and average instance-specific approximation ratios than standard QAOA.

[14]Due to the symmetries in the QAOA circuit for unit-weight graphs, we know that it suffices to check the values of $F_p(\gamma, \beta)$ for (γ, β) in $[-\pi, \pi] \times [-\pi/4, \pi/4]$ [114].

[15]The minimum, maximum, and average are computed by considering a discretization of the landscape. In particular, we consider the values of $F_1(\gamma_1, \beta_1)$ for all $(\gamma_1, \beta_1) \in \mathcal{D} = \{(\pi \frac{i}{50}, \frac{\pi}{4} \frac{j}{50}) : i = -50, -49, \ldots, 50$ and $j = -50, -49, \ldots, 50\}$.

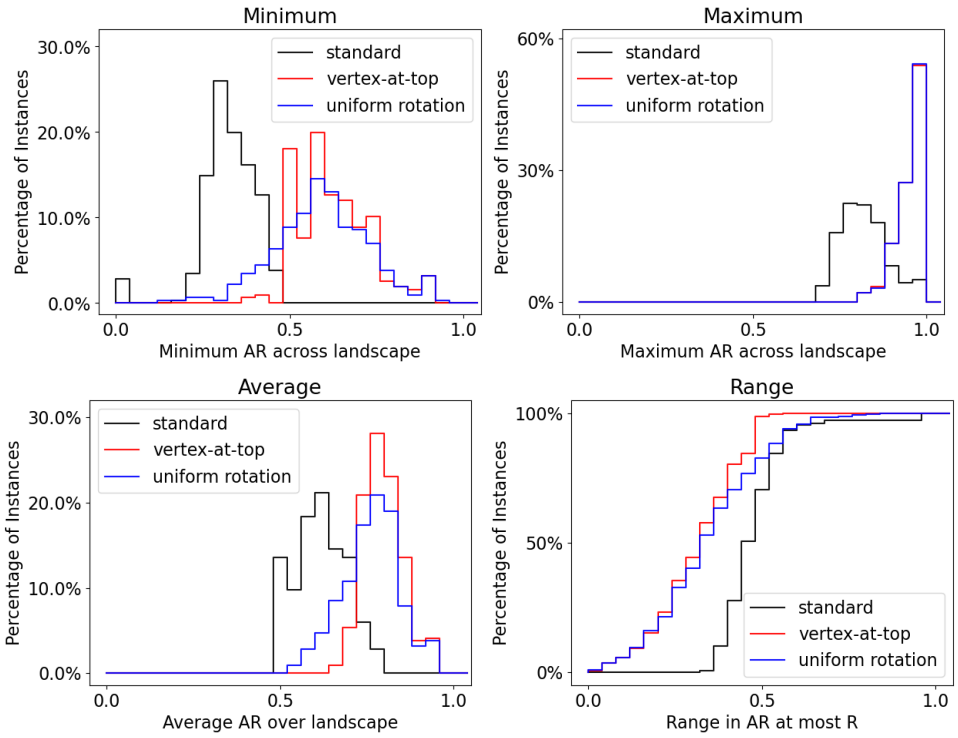

Figure 4.15: This figure shows how various statistics of the parameter landscape change with the variant of QAOA considered (standard QAOA, QAOA-warm with vertex-at-top rotations, and QAOA-warm with random rotations). For each unit weight graphs in our graph library $\mathcal{G}$ (See Section 4.5.1) and for each QAOA variant, we first generate the parameter landscape; we use a single 2-dimensional initialization for both rotation schemes considered for QAOA-warm. For each landscape, we calculate the minimum, maximum, and average across the landscape in addition to the range (the difference between the highest and lowest instance-specific approximation ratio achieved in the landscape).

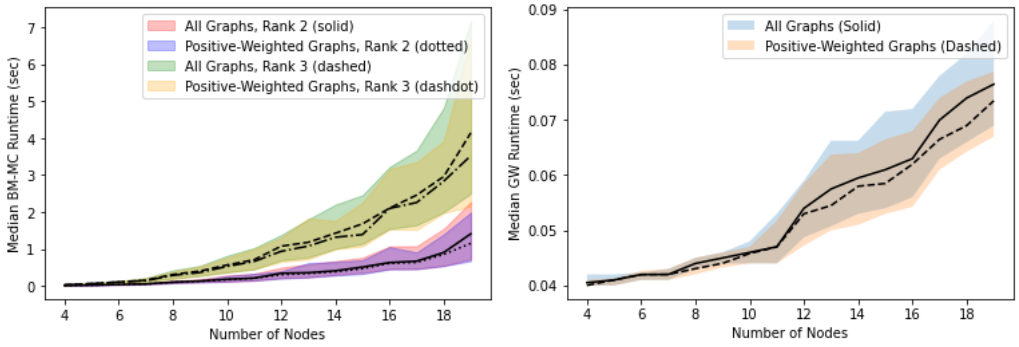

Figure 4.16: This figure shows how the median runtime changes for both GW and BM-MC$_k$ ($k = 2, 3$) as the number of nodes increases. The extended graph library $\mathcal{G}'$ (2076 instances) was used to generate the results above; we also run plot the results for just the positive-weighted graphs in $\mathcal{G}'$ as well. The top and bottom of the colored regions corresponding to 75 and 25 percentiles respectively.

4.5.6 Pre-processing Time vs Parameter Search Time

Here, we compare runtimes for various aspects of QAOA-warm to those of standard QAOA and the GW algorithm. For the preprocessing stage, finding an approximate solution for BM-MC$_k$ takes up the bulk of the time (i.e., 1-3 seconds). The rotation applied to the solution and the mapping of the rotated solution to the Bloch sphere is negligible. We plot the runtimes for BM-MC$_k$ for $k = 2, 3$ in Figure 4.16. To get a better idea of scaling, we consider an expanded graph library $\mathcal{G}'$ consisting of 2076 instances; $\mathcal{G}'$ is generated in the same way as $\mathcal{G}$ (see Section 4.5.1) but we instead consider graphs of up to 19 nodes. Finding approximate solutions to rank-2 BM-MC$_2$ is considerably faster than rank-3 BM-MC$_3$; furthermore, the runtimes are similar regardless of edge weight values. Plots of the GW algorithm's runtime for all graphs in $\mathcal{G}'$ are included in Figure 4.16; as before, we see the runtimes are similar even if restrict our attention to only positive-weighted graphs. Note that our code for BM-MC runs is not optimized, and possibly faster implementations for this might be possible.

For both classical algorithms (the GW algorithm and BM-MC$_k$), we see that the runtime increases superlinearly in the number of nodes n. In regards to theoretical results, the runtime of the GW algorithm is dominated by solving the SDP; Lee and Padmanabhan [115] develop an algorithm where one can get within factor $1 - \varepsilon$ of the optimal SDP

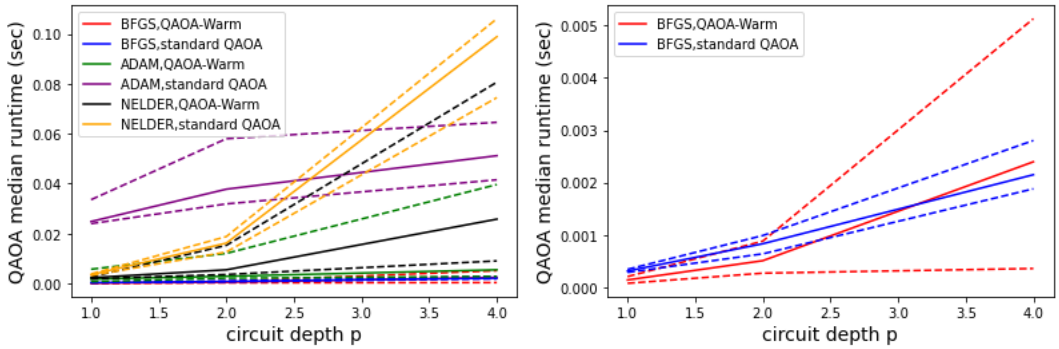

Figure 4.17: This figure shows how the median runtime changes for the optimization loop of both standard QAOA and QAOA-warm for various optimizers (ADAM, BFGS, and Nelder-Mead). COBYLA was not included due to technical limitations with our software; in particular, we were unable to gain direct access to the source code needed to in order to exclude the runtime of function or gradient evaluations. as the circuit depth increases. These runtimes do not include the time taken to evaluate/estimate the function values or gradients of the expected cut value $F_p(\gamma, \beta)$ (since in practice, such calls would be made on the quantum device). These plots were generated by randomly selecting 20 8-node graphs from our graph library $\mathcal{G}$ (see Section 4.5.1), with 10 of the 20 graphs having only positive edge weights. For each solid colored line (corresponding to the median), there are two dashed lines of the same color above and below representing the 75th and 25th percentiles respectively. On the right, we plot the runtimes for BFGS separately in order to more easily see the trend in runtime as p increases.

objective in $\widetilde{O}(m/\varepsilon^{3.5})$ time where m is the number of edges in the graph. Similarly, for BM-MC$_k$, Mei et al. [77] prove that one can use a variant of the fast Riemannian trust-region algorithm to find a locally optimal solution in $O(n^2dk^4 \log n)$ time for d-regular graphs.

We now consider the runtime of the optimization loop used in both standard QAOA and QAOA-warm as seen in Figure 4.17 for various optimizers (ADAM, BFGS, Nelder-Mead). To get an idea of the runtime of the classical portions of the optimization loop, we exclude[16] the time taken to estimate the function values or gradients of $F_p(\gamma, \beta)$. During our preliminary experiments, we found that the number of nodes did not have any noticable effect on the runtime of the optimization loop for either standard QAOA or QAOA-warm for any of the optimizers. However, for all optimizers, Figure 4.17 shows that, empirically, more time is needed to optimize γ and β as the circuit depth increases. With the exception of BFGS, for all optimizers and circuit depths, it appears that QAOA-warm converges to a

[16]We exclude such portions since including them would not be reflective of the runtime obtained on an actual quantum device; a quantum device can estimate $F_p(\gamma, \beta)$ (the expected cut value) in time polynomial in n whereas a numerical simulation would (typically) take time that is exponential in n.

set of parameters more quickly compared to standard QAOA.

We now discuss the runtime of the preprocessing stage of QAOA-warm relative to the runtime of QAOA-warm's optimization loop. A direct comparison is difficult since the former is independent of the circuit depth p and the latter is independent of the number of nodes n. However, for the p and n considered in our experiments, it appears (from Figures 4.16 and 4.17) that the preprocessing stage takes orders of magnitude longer. We remark that our current implementation for finding approximate BM-MC$_k$ solutions (Algorithm 1) was not designed to find solutions quickly; we suspect other methods can find solutions more quickly. Additionally, we remark that the runtime preprocessing stage appears to scale modestly as the number of nodes increases. The trends in Figure 4.17 also suggest that as the circuit depth p increases, that the proportion of QAOA-warm spent in the preprocessing stage diminishes. Moreover, the real runtime of the optimization loop on an actual quantum device would be longer since one needs to consider the time needed to query the quantum device in order to estimate the value or gradient of $F_p(\gamma, \beta)$ at every iteration of the optimization loop. Lastly, there is the additional benefit that if one wants to perform multiple QAOA-warm runs with different initializations of the variational parameters or different rotation schemes, then one only needs to find a solution to BM-MC$_k$ once.

4.5.7 QAOA-Warm with Median and Worst Vertex-At-Top Rotations

For our numerical simulations in Section 4.5, we use the best of either 5 vertex-at-top rotations or best of 5 uniform rotations for QAOA-warm. Performing multiple runs of QAOA-warm with different rotations and taking the best allows one to mitigate the possibility of using a warm-start with a poor rotation. We present the results with respect to the median rotation here. We plot the results below in Figure 4.18; we see that the results do not differ much from what was seen in Figure 4.11. In Figure 4.19, we also plot the results when the worst of 5 vertex-at-top rotations are used to give an idea of the worst-case performance for QAOA-warm.

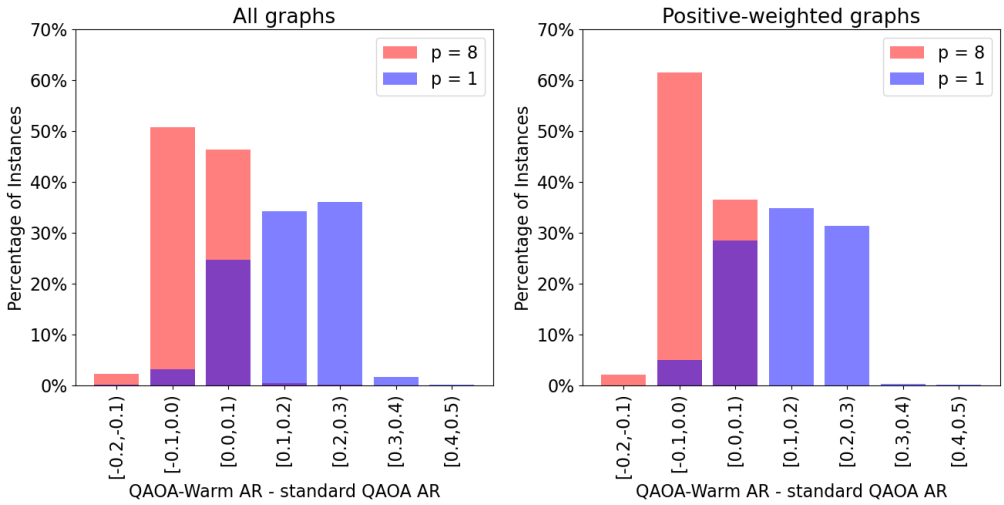

Figure 4.18: Histograms comparing the instance-specific approximation ratio in (depth-p) QAOA-warm and (depth-p) standard QAOA for both $p = 1$ (blue) and $p = 8$ (red) where the median vertex-at-top rotations are used. Overlapping portions of the histogram are in purple. The left plot is generated using the graphs in our graph library $\mathcal{G}$ (see Section 4.5.1) whereas for the bottom right plot, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights.

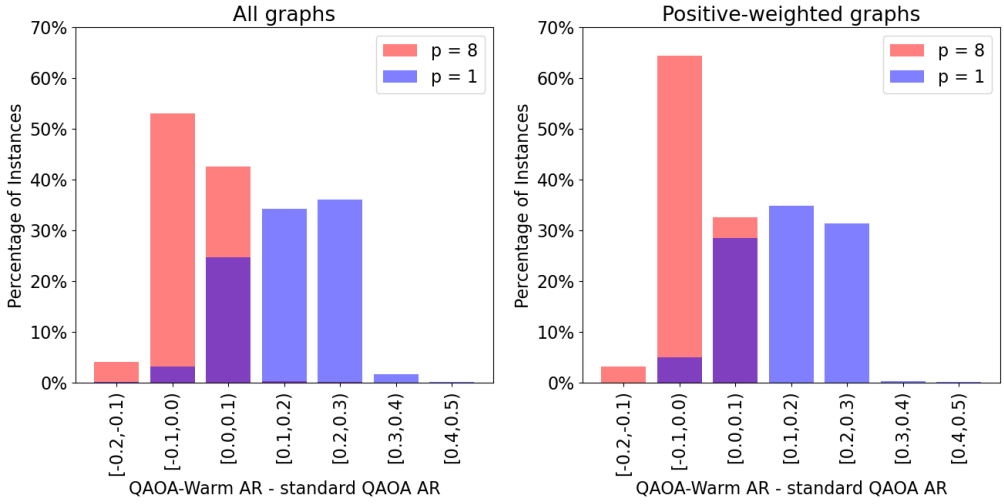

Figure 4.19: Histograms comparing the instance-specific approximation ratio in (depth-p) QAOA-warm and (depth-p) standard QAOA for both $p = 1$ (blue) and $p = 8$ (red) where the worst vertex-at-top rotations are used. Overlapping portions of the histogram are in purple. The left plot is generated using the graphs in our graph library $\mathcal{G}$ (see Section 4.5.1) whereas for the right plot, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights.

4.6 Discussion

In this chapter, we proposed using classical approximate solutions to low-rank Max-Cut formulations and low-dimensinal projections of higher-dimensional Max-Cut formulations to initialize the QAOA algorithm. There are significant differences in classical approximation algorithms for Max-Cut and quantum algorithms. For example, in the classical approach the vertices that share the same 3-dimensional representation on the sphere will always be on the same side of the cut (no matter which hyperplane is selected). In contrast, quantum sampling creates a very different distribution (with a larger support) over cuts, wherein vertices with the same state can be sampled on different sides of the cut. Despite this difference, we observe that as the angle θ of the vertices to the measurement axis approaches 0, the probability distribution of the classical solution approaches that of the quantum sampling. Intuitively, as vertices start clustering at the antipodes on the 3-dimensional sphere, quantum sampling of the corresponding qubits and hyperplane rounding of the 3-dimensional representation both give similar cuts. Moreover, SDP-based solutions spread adjacent vertices (with positive edge weights) as far as possible on the k-dimensional sphere, which can be beneficial for quantum sampling as well.

Standard QAOA is a local algorithm [1]. If the circuit depth p is not high enough, then standard QAOA may fail to achieve near-optimal solutions [31, 32]. However, when one considers the preprocessing stage used in QAOA-warm, such a locality property no longer exists. A clear example of this is BM-MC$_2$ applied to an odd cycle: the optimal solution consists of the vertices evenly spaced apart along the unit circle. However, if a single edge is deleted, the optimal solution collapses to a rank-1 solution. The edge deletion has a global effect on the positions of all the vertices, and consequently, on the probability of each edge being cut. Put another away, although the quantum operations in QAOA-warm are still local, the warm-start encodes information about the global structure of the graph, in which case, building up correlations between distant qubits (via a high circuit depth)

may not be necessary if a high-quality warm-start is used.

Warm-starts also appear to flatten the energy landscape in terms of (β, γ). In the most extreme case (for example Figure 4.7(b)), the warm start finds the optimal solution, completely decoupling the QAOA optimization loop from γ_1 and the cost Hamiltonian H_C. Even when this does not occur, warm-starting still appears to make QAOA less sensitive to initial (β, γ) values by starting off in the neighborhood of a possible solution. In particular, the role of γ is diminished, as the warm-start has already begun optimizing the cost-energy. This suggests that QAOA-warm serves as a kind of dimensional reduction, emphasizing the amplitude manipulation of the mixer over the energy weighting of the cost Hamiltonian. This is not a guarantee that the QAOA optimization will find the optimal solution in the reduced space; the reduction may hide the optimal solution for graphs that are especially challenging for SDP solvers. However, this flattening may prove important for physical implementations of QAOA. The warm-start flattened landscapes may make QAOA more robust to both classical and quantum noise that would otherwise complicate the optimal solution search.

In this chapter, we restricted our attention to rank-2 and rank-3 initializations, whereas in classical methods, one could also make an attempt at finding rank-k ($k > 3$) solutions. These solutions are easier to find, and yield provably better approximations as k increases [77]. However, increasing the number of dimensions makes the mapping to the quantum states non-trivial. Exploration of higher-rank approximations are left as a future research direction.

Another direction for future work is to apply QAOA-warm to other combinatorial problems. One path is reduction of other problems in NP to Max-Cut [6]. Alternately, Quadratic Unconstrained Binary Optimization (QUBO) problems can easily be recast as a Max-Cut problem (and vice versa) with the number of variables differing by at most 1 [96]. Many combinatorial problems are easily expressed in the form of a QUBO [116, 117] so Max-Cut is also an interesting problem to consider from a practical standpoint. However,

it may be of interest to see if our approach can be used directly for other combinatorial problems without resorting to such reductions.

Lastly, as seen in Section 4.3, we acknowledge that QAOA-warm, in its current form, has limitations; in particular, increased circuit depth does not necessarily yield optimality of Max-Cut in the limit. This performance may be extendable to higher circuit depth via modifications to the mixing Hamiltonian H_B; this idea yields positive results in Egger et al.'s work and such a modification is the subject of study in the next chapter.

4.7 Conclusion

In this chapter, we explored the idea of *warm-starts* for initializing the quantum state of the QAOA algorithm, and showed promising experimental and theoretical results for low-rank initializations using approximate SDP solution. On average, we find that QAOA-warm performs better in terms of time and quality of solutions in low depth circuits, compared to standard QAOA. Moreover, even though a portion of the instance-specific approximation ratios of QAOA-warm can be attributed to the classical warm-start itself, we find that running QAOA-warm introduces significant improvements in expected cut quality beyond simply (quantum) sampling the initial warm-start state for many instances. As the circuit depth increases, QAOA-warm is however unable to converge to the optimal solution (unlike standard QAOA). In the next chapter, this is remedied by considering further modifications to QAOA-warm (e.g. modifying the mixing Hamiltonian), although standard mixers might provide easier implementation on certain hardware. We further acknowledge that beyond the instance-specific approximation ratio, there are a variety of methods and metrics in which to measure the performance of QAOA and its possible variants. We leave such an exploration of the cut distributions (and metrics on those distributions) for potential future work; we refer the reader to a paper by Herrman et al. for such results on standard QAOA [90].

Overall, we believe that the use of the standard mixers with warm-starts allows a prin-

cipled way of bringing in information from classical solvers into quantum algorithms. The concept of warm-starts and plateauing of quality of instance-specific approximation at higher p depth could be of interest to researchers looking at reachability of solution state space and, at the limitations and strengths of the standard QAOA itself.

Table 4.3: The table summarizes what is known about different variants of QAOA (for Max-Cut) based on various combinations of initializations (equal superposition, BM-MC$_k$, projected GW, and single-cut initializations) and mixing Hamiltonian (standard, custom mixers (Section 5), and the mixer proposed by Egger et al. [42] for single-cut initializations). For various combinations, we state what is known regarding the convergence and worst-case approximation ratio (for depths $p \geq 0$) of the corresponding QAOA variant for graphs with non-negative edge weights.

Initializations	Mixers	Citation	Worst Case AR at p=0	Converges to Max-Cut?	Comment
Equal Superposition	Standard mixer	[1]	0.5	Yes	Both mixers are equivalent for this initialization
	Custom mixers				
BM-MC$_k$	Standard mixer (QAOA-warm)	[3]	0 ($k = 2$) 0.333 ($k = 3$)	No	
	Custom mixers (QAOA-warmest)	**This work**	0 ($k = 2$) 0.333 ($k = 3$)	Yes	Converges **quickly** (see Section 5.4.1)
Projected GW	Standard mixer (QAOA-warm)	**This work**	0.658 ($k = 2$), 0.585 ($k = 3$), (Corollary 20)	No	
	Custom mixers (QAOA-warmest)	**This work**	0.658 ($k = 2$), 0.585 ($k = 3$), (Corollary 20)	Yes	Converges **quickly** (see Section 5.4.1)
Single cut (with regularization angle θ^*)	Standard mixer (QAOA-warm)	[57] ($\theta^* = 0$ case)	$0.878 \cos^{2n}(\theta^*/2)$, (Proposition 9)	No	
	Custom mixers (QAOA-warmest)	Appendix I of [42]	$0.878 \cos^{2n}(\theta^*/2)$, (Proposition 9)	Yes	Converges **slowly** for small θ^* (see Section 5.4.1)
	Other custom mixers	Section 2.3 of [42]	0.878	No	AR result occurs at $(\gamma_1, \beta_1) = (0, \pi/2)$

CHAPTER 5

WARM-STARTS WITH CUSTOMIZED MIXERS

In this chapter[1], we provide a general framework that shows, for *any* initial product state, how to construct a custom mixing Hamiltonian for QAOA so that the QAOA (for Max-Cut) converges to an optimal cut with increased circuit depth; recall that the approach discussed in the previous chapter, QAOA-warm, does not have such convergence properties. When this custom mixer approach is combined with a warm-start, such as those presented in the previous chapter, we refer to such an approach as QAOA-warmest. In what follows, we describe this custom mixer framework in detail, discuss its properties in regards to convergence, and summarize the results of numerical simulations and hardware experiments.

5.1 Custom Mixer Construction

Consider any product state $|s_0\rangle$ on n qubits on the Bloch sphere, i.e., $|s_0\rangle$, up to a global phase, can be written in the form:

$$|s_0\rangle = \bigotimes_{j=1}^{n} |s_{0,j}\rangle \,,$$

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

$$|s_{0,j}\rangle = \cos(\theta_j/2)\,|0\rangle + e^{i\varphi_j}\sin(\theta_j/2)\,|1\rangle \,.$$

Here, θ_j and φ_j can be interpreted as the polar and azimuthal angle respectively of the jth qubit on the Bloch sphere. The position of the jth qubit on the Bloch sphere can also be described in Cartesian coordinates $\hat{n}_j = (x_j, y_j, z_j)$ via the following transformation from

[1]The results presented in this chapter are currently under revision for Quantum; a preprint is available on the arXiv [3].

spherical to Cartesian coordinates:

$$x_j = \sin\theta_j \cos\varphi_j,$$
$$y_j = \sin\theta_j \sin\varphi_j,$$
$$z_j = \cos\theta_j.$$

The custom mixing Hamiltonian H_B is then constructed as follows:

$$H_B = \bigoplus_{j=1}^{n} H_{B,j},$$

where $H_{B,j} = x_j\sigma^x + y_j\sigma^y + z_j\sigma^z$. To develop a geometrical understanding of the custom mixer, consider the operator $R_{\hat{n},j}(\alpha)$ that rotates the jth qubit by angle α about the $\hat{n}$-axis for some unit vector $\hat{n} = (x, y, z)$; such as operation can be written as:

$$R_{\hat{n},j}(\alpha) = \exp\left(-i\frac{\alpha}{2}(x\sigma_j^x + y\sigma_j^y + z\sigma_j^z)\right).$$

Recall that for the kth of the p stages of the QAOA circuit (where p is the circuit depth), one applies the unitary operator $e^{-i\beta_k H_B}$ with β_k being a variational parameter (to be optimized); this operator, $e^{-i\beta_k H_B}$, can be written as $\prod_{j=1}^{n} R_{\hat{n}_j,j}(2\beta_k)$, i.e., in the kth stage of the QAOA circuit, one independently rotates the jth qubit about the axis determined by its original position by angle $2\beta_k$.

The standard QAOA [1] is, therefore, a special case of our custom mixer approach as the standard starting state has each qubit on the x-axis (with Cartesian coordinates $(x_j, y_j, z_j) = (1, 0, 0)$) and with the standard mixer $H_B = \sum_{j=1}^{n} \sigma_j^x$, the unitary operator $e^{-i\beta_k H_B}$ corresponds to rotations (by $2\beta_k$) about the x-axis.

5.2 Proof of Convergence in the Adiabatic Limit

In the special case that a product state consists of qubits that lie at the intersection of the Bloch sphere and the xz-plane with positive x-coordinate, it can be proven that the corresponding custom mixer is the same as those considered by Egger et al. [42] for convex quadratic programs. It should be noted that while the construction of the mixer (for a given initial state) is the same between both approaches, our approach and Egger et al.'s differ signifcantly in regards to how the initial state is obtained (as discussed in Section 4.2.3). In their paper, Egger et al. state that QAOA with such a modified mixer converges to the optimal cut value with increased circuit depth.

While Egger et al. state such a convergence result, they do not provide a proof. While straightforward calculations prove that the initial state is the highest-energy eigenstate of the mixer (a condition needed in order to apply the adiabatic theorem and guarantee convergence), a complete proof requires careful inspection of the eigenvalues of the time-dependent Hamiltonian $H(t) = (1 - t/T)H_B + (t/T)H_C$. In Sections 5.2.1 and 5.2.2 we go over the details of the proof, before putting it all together in Section 5.2.3. We then show how this convergence result can be generalized to arbitrary product states in Section 5.2.4.

5.2.1 Eigenstates of Custom Mixers

As previously discussed, the initial state of standard QAOA is the highest energy eigenstate of the standard mixing Hamiltonian; such a property is required in order to make the adiabatic argument that standard QAOA converges to the optimal cut with increased circuit depth. Similarly, if we wish to prove convergence for QAOA-warmest; we must prove the custom mixers and warm-started initial quantum state also exhibit this property; we prove this is the case regardless of if the initial state lies in the xz-plane or not. We first prove that the result holds for a single qubit, and then generalize to the Kronecker sums of matrices.

Lemma 26. *Let* $|s\rangle = \cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)\,|1\rangle$ *be a single-qubit quantum state and*

let $\hat{n} = (x, y, z)$ be the Cartesian coordinates of that qubit on the Bloch sphere. Let $U = x\sigma^x + y\sigma^y + z\sigma^z$. Then $|s\rangle$ is the most-excited eigenstate of U.

Proof. We have the following relationship between the Cartesian and spherical coordinates: $x = \cos\varphi \sin\theta, y = \sin\varphi \sin\theta, z = \cos\theta$. Thus, the matrix $U = x\sigma^x + y\sigma^y + z\sigma^z$ is given by

$$U = \begin{bmatrix} \cos\theta & \sin\theta e^{-i\varphi} \\ \sin\theta e^{i\varphi} & -\cos\theta \end{bmatrix}.$$

One can show that the matrix can be diagonalized as $U = PDP^{-1}$ where $P = [v_1\ v_2], D = \text{diag}(1, -1), v_1 = \cos(\theta/2)\,|0\rangle + \sin(\theta/2)e^{i\varphi}\,|1\rangle\,, v_2 = -\sin(\theta/2)\,|0\rangle + \cos(\theta/2)e^{i\varphi}\,|1\rangle$. Thus, $v_1 = \cos(\theta/2)\,|0\rangle + \sin(\theta/2)e^{i\varphi}\,|1\rangle$ is the highest-energy eigenstate of U. □

We can then formulate the most-excited eigenstate of U using the following relation between eigenvalues of matrices involved in a Kronecker sum and the resultant matrix.

Theorem 27. *(Theorem 13.16 in [118]) Let $A \in \mathbb{C}^{n\times n}$ have eigenvalues $\lambda_1, \ldots, \lambda_n$ and let $B \in \mathbb{C}^{m\times m}$ have eigenvalues $\mu_1, \ldots, \mu_m$. Then the Kronecker sum $A \oplus B$ has mn eigenvalues given by $\{\lambda_i + \mu_j : i \in [n], j \in [m]\}$. Moreover, if $x_1, \ldots, x_p$ ($p \leq n$) are linearly independent eigenvectors of A corresponding to $\lambda_1, \ldots, \lambda_n$ and $z_1, \ldots, z_q$ ($q \leq m$) are linearly independent eigenvectors of B corresponding to $\mu_1, \ldots, \mu_q$, then, for all $i \in [p]$ and $j \in [q]$, we have that $x_i \otimes z_j$ are linearly independent eigenvectors of $A \oplus B$ corresponding to $\lambda_i + \mu_j$.*

By applying Theorem 27 with the summands $H_{B,j}$ of the mixing Hamiltonian H_B, we get the desired result as shown in Proposition 10.

Proposition 10. *Let $|s_0\rangle$ be any initial product state and let H_B be its corresponding custom mixer. Then $|s_0\rangle$ is the highest-energy eigenstate of H_B.*

Proof. Suppose for each $j = 1, \ldots, n$ we have a matrix A_j with real eigenvalues and suppose the largest eigenvalue of A_j is λ_j with corresponding eigenvector v_j. As a consequence of Theorem 27, we have that the largest eigenvalue of $\bigoplus_{j=1}^{n} A_j$ is $\sum_{j=1}^{n} \lambda_j$ with

one corresponding eigenvector being $\bigotimes_{j=1}^{n} v_j$. Letting $A_j = H_{B,j}$ and $v_j = |s_{0,j}\rangle$ and applying Lemma 26, we see that $|s_0\rangle = \bigotimes_{j=1}^{n} |s_{0,j}\rangle$ is a highest-energy eigenstate for $H_B = \bigoplus_{j=1}^{n} H_{B,j}$. □

In order to fully complete the adiabatic argument for proving convergence, it remains to prove that the gap between the two largest eigenvalues of the time-dependent Hamiltonian $H(t) = (1 - t/T)H_B + (t/T)H_C$ is positive for all $0 \leq t < T$; we discuss this condition in more detail in the next section.

5.2.2 Eigenvalue Gap for Custom Mixers

Let H_B and H_C be the mixing and the cost Hamiltonian for QAOA respectively. It is known that if the Quantum Adiabatic Algorithm is run for large enough time T with time-varying Hamiltonian $H(t) = (1 - t/T)H_B + (t/T)H_C$ starting with the highest-energy eigenstate of $H(0) = H_B$, then one can arrive at the highest-energy eigenstate of $H(T) = H_C$, i.e. the optimal solution, provided that the gap between the largest and second-largest eigenvalue of $H(t)$ is strictly positive for all $t < T$. This translates to finding an optimal solution when running QAOA as we let the circuit depth p tend to infinity. Farhi et al. proved that this eigenvalue gap was strictly positive for standard QAOA [1], thus guaranteeing convergence to the optimal solution. In particular, they applied the following Perron-Frobenius theorem to irreducible stoquastic[2] matrices.

Theorem 28. *[119] Let A be an irreducible matrix whose entries are all real and non-negative. Let r be the spectral radius of A, i.e., $r = \max\{|\lambda| : \lambda$ is eigenvalue of $A\}$. Then r is an eigenvalue of A and furthermore, it has algebraic multiplicity of 1.*

If the eigenvalues of an $n \times n$ matrix A are real (e.g. if A is Hermitian), then its eigenvalues (with multiplicity) can be ordered as $\lambda_1 \geq \cdots \geq \lambda_n$; if A is also irreducible

[2]Stoquastic matrices are square matrices with real entries so that all of the off-diagonal entries are non-negative. Let A be an $n \times n$ square matrix. Construct a directed graph G_A with vertex set $[n]$ where the edge (i, j) is included if and only if $A_{ij} > 0$. If G_A is strongly connected, then we say that A is irreducible. Otherwise, we say that A is reducible.

and has real, non-negative entries then Theorem 28 ensures a gap between the two largest eigenvalues (otherwise, if $\lambda_1 = \lambda_2$, then the algebraic multiplicity of λ_1 would be at least 2, contradicting the statement of the theorem.) This observation still holds if we relax the non-negativity condition to allow negative entries along the diagonal as seen in the following lemma.

Lemma 29. *Let A be an irreducible stoquastic Hermitian matrix. Then the difference between the largest and second-largest eigenvalue of A is strictly positive.*

Proof. Since A is stoquastic, then all of the off-diagonal elements are already non-negative; however, the diagonal elements may be negative. Observe that for large enough k, we have that $A + kI$ is a matrix with all non-negative entries. Note that $A + kI$ is Hermitian (since A and I are) and thus the eigenvalues of $A + kI$ are real. If we apply the Perron-Frobenius theorem to $A + kI$, one observes that the gap between the largest and second-largest eigenvalue is strictly positive.

One can show that the eigenvalues of $A + kI$ can be obtained by shifting all of the eigenvalues of A by k (i.e. of λ is an eigenvalue of A, then $\lambda + k$ is an eigenvalue of $A + kI$). Moreover, the multiplicities of these shifted eigenvalues are preserved. Thus, the gap between the largest and second-largest eigenvalue of $A + kI$ (which is strictly positive) is equal to the gap between the largest and second-largest eigenvalue of A. □

If the custom mixer H_B has the form $\sum_{j=1}^{n}(x_j\sigma_j^x + z_j\sigma_j^z)$ with $x_j \in \mathbb{R}^+$ and $z_j \in \mathbb{R}$ for $j = 1, \ldots, n$ and if we can show that $H(t)$ is an irreducible, stoquastic matrix, then, by Lemma 29, the eigenvalue gap of $H(t)$ is strictly positive meaning that one can achieve the optimal solution as the circuit depth $p \to \infty$ in QAOA-warmest. Geometrically, this special case corresponds to an initial product state whose qubits lie in the xz-plane on the Bloch sphere with $x > 0$. It remains to prove the stoquasticity and irreducibility of this special case which is done in the following two propositions; but first, we prove a useful technical lemma.

Lemma 30. *If A and B are $n \times n$ and $m \times m$ stoquastic matrices respectively, then so is $A \oplus B$.*

Proof. By definition, $A \oplus B = A \otimes I_m + I_n \otimes B$. Since we know the sum of stoquastic matrices is stoquastic, it suffices to show that $A \otimes I_m$ and $I_n \otimes B$ is stoquastic.

Observe that,

$$A \otimes I_m = \begin{bmatrix} A_{11}I_m & \cdots & A_{1n}I_m \\ \vdots & \ddots & \vdots \\ A_{n1}I_m & \cdots & A_{nn}I_m \end{bmatrix}.$$

Note that for $i \neq j$, the ijth block in the block matrix above is $A_{ij}I_m$, which contains only non-negative entries since A_{ij} is an off-diagonal element of A and A is stoquastic. Now consider the entries in the ijth block where $i = j$. Note that if there is an off-diagonal entry of $A \otimes I_m$ that is part of the iith block, then it is also an off-diagonal entry of that block but all the off-diagonal entries of the iith block ($A_{ii}I$) are zero. Thus, we have shown that every off-diagonal element is non-negative, thus $A \otimes I_m$ is stoquastic.

Next, observe that

$$I_n \otimes B = \begin{bmatrix} 1B & 0B & \cdots & 0B \\ 0B & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0B \\ 0B & \ldots & 0B & 1B \end{bmatrix},$$

which makes it clear that the off-diagonal elements of $I_n \otimes B$ are either 0 or the off-diagonal elements of B which are non-negative (by stoquasticity of B) and thus $I_n \otimes B$ is stoquastic.

□

Proposition 11. *Let n be a positive integer. For each $j = 1, \ldots, n$ let x_j be any non-negative real number and let z_j be any real number. Let $H_B = \sum_{j=1}^{n}(x_j\sigma_j^x + z_j\sigma_j^z)$ and let H_C be the problem Hamiltonian for QAOA. Then $H(t) = (1 - t/T)H_B + (t/T)H_C$ is stoquastic for all $0 \leq t \leq T$.*

Proof. By construction H_C (and thus $(t/T)H_C$) is a diagonal matrix (as $|b\rangle$ is an eigen-

vector of H_C for each n-length bitstring b). If H_B were stoquastic, then $(1 - t/T)H_B$ is stoquastic (as $1 - t/T \geq 0$ for $0 \leq t \leq T$) and thus $H(t) = (1 - t/T)H_B + (t/T)H_C$ is stoquastic (as adding a diagonal matrix to a stoquastic matrix yields a stoquastic matrix). Thus, it remains to show that H_B is stoquastic.

Let $H_{B,j} = x_j\sigma^x + z_j\sigma^z$. Expanding σ^x and σ^z, we have that

$$H_{B,j} = \begin{bmatrix} z_j & x_j \\ x_j & -z_j \end{bmatrix},$$

which is clearly stoquastic as we assumed that $x_j \geq 0$. As $H_B = \bigoplus_{j=1}^{n} H_{B,j}$, the result now follows from Lemma 30. □

Proposition 12. *Let n be a positive integer. For each $j = 1, \ldots, n$ let x_j be a positive real number and let z_j be any real real number. Let $H_B = \sum_{j=1}^{n}(x_j\sigma_j^x + z_j\sigma_j^z)$ and let H_C be the problem Hamiltonian for QAOA. Then $H(t) = (1 - t/T)H_B + (t/T)H_C$ is irreducible for all $0 \leq t < T$.*

Proof. First, we recall the definition of irreducible matrix. Let A be an $n \times n$ square matrix. Construct a directed graph G_A with vertex set $[n]$ where the edge (i, j) is included if and only if $A_{ij} > 0$. If G_A is strongly connected, then we say that A is irreducible. Otherwise, we say that A is reducible.

For any square matrix M, let G_M be the corresponding directed graph as described above. Observe that H_C (and hence $(t/T)H_C$) is a diagonal matrix, thus, by the definition of irreducibility, the irreducibility of $(1-t/T)H_B+(t/T)H_C$ is the same as $(1-t/T)H_B$. Similarly, scaling a matrix by a positive constant does not affect its irreducibility, so it suffices to prove the irreducibility of H_B.

Observe that σ^x, σ^z are symmetric and thus it is not very difficult to show that H_B is also symmetric. This means, for the purposes of showing irreducibility, G_{H_B} is effectively an undirected graph and we just need to show that it is connected. One can write H_B as

$H_B = \bigoplus_{j=1}^{n}(x_j\sigma^x + z_j\sigma^z)$ where $\bigoplus$ denotes the Kronecker sum. According to [120], this means that G_{H_B} can be written as the Cartesian graph product of the graphs $H_1, H_2, \ldots, H_n$ where $H_j = G_{A_j}$ with $A_j = x_j\sigma^x + z_j\sigma^z$. Observe, that each of the H_j's are connected if and only if $x_j \neq 0$ which is true by assumption. Since each of the H_j's are connected, then it is also the case that G_{H_B} is connected as well (see Theorem 1 of [121]) which finishes the proof. □

We can now prove the convergence for custom mixers of the special form ($\sum_{j=1}^{n}(x_j\sigma_j^x + z_j\sigma_j^z)$ with $x_j \in \mathbb{R}^+$ and $z_j \in \mathbb{R}$ for $j = 1, \ldots, n$) and their corresponding initializations as described in Proposition 13 in Section 5.

Recall that for standard QAOA, we have that $H_B = \sum_{j=1}^{n}(1 \cdot \sigma_j^x + 0 \cdot \sigma_j^z)$. Thus, the fact that standard QAOA converges to the optimal cut as $p \to \infty$ is a special case of Propositions 11 and 12.

5.2.3 Proof of Convergence (Special Case)

Using the results above, we now formally prove the convergence result for QAOA-warmest in the special case where qubits are initialized along the xz-plane of the Bloch sphere with positive x-coordinate.

Proposition 13. *Let $|s_0\rangle$ be any initial product state such that all qubits lie at the intersection of the Bloch sphere and the xz-plane with positive x-coordinate. Running QAOA with initial state $|s_0\rangle$ and its corresponding custom mixer yields that*

$$\lim_{p\to\infty} \max_{\gamma,\beta} F_p(\gamma, \beta) = \textit{Max-Cut}(G),$$

i.e., the expected cut value of QAOA-warmest with optimal variational parameters will yield the optimal cut value as the circuit depth p tends to infinity.

Proof. By construction, the corresponding custom mixers will have the form $\sum_{j=1}^{n}(x_j\sigma_j^x + z_j\sigma_j^z)$ with $x_j \in \mathbb{R}^+$ and $z_j \in \mathbb{R}$ for $j = 1, \ldots, n$. The result then follows from Proposition

10, Lemma 29 (which is applicable due to Proposition 11 and Proposition 12), and the adiabatic theorem. □

5.2.4 Proof of Convergence (General)

As seen in the previous chapter, it may be the case that we have a initial product quantum state of interest whose qubits lie outside the xz-plane. We will eventually prove (in Proposition 14), that just like in Proposition 13, convergence also holds for general initial product states as long as the qubits are not initialized at the poles[3].

The proof of convergence in the general case relies on the fact that proving general convergence can be reduced to proving convergence in the xz-plane (with $x > 0$), which we have already proven. This reduction is a consequence of Theorem 31 which proves that for any product state $|s_0\rangle$, there is another state $|s_0'\rangle$ with qubits along the xz-plane (with $x > 0$), for which our custom mixer approach achieves the same distribution of cuts.

Theorem 31. *Let $|s_0\rangle$ be any initial product state and let H_B be its corresponding custom mixer. Let $|s_0'\rangle$ be the state $|s_0\rangle$ but with each qubit's phase equaling 0 and let H_B' be the corresponding custom mixer. Then, for any fixed choice of variational parameters (γ, β), the distribution of cuts obtained from QAOA-warmest with initial state $|s_0\rangle$ and mixer H_B is the same as the distribution of cuts from obtained QAOA-warmest with initial state $|s_0'\rangle$ and mixer H_B'.*

Proof. Let $|s_0\rangle = \bigotimes_{j=1}^{n} |s_{0,j}\rangle$ with $|s_{0,j}\rangle = \cos(\theta_j/2)\,|0\rangle + e^{i\varphi_j}\sin(\theta_j/2)\,|1\rangle$ be an arbitrary initial product state. As a consequence of Proposition 13, it suffices to show that QAOA-warmest with this initial state $|s_0\rangle$ yields the same expected cut value (at the same variational parameters) as another initial product state $|s_0\rangle'$ where each qubit of $|s_0\rangle'$ lies in the xz-plane of the Bloch sphere with positive x-coordinate.

[3]For convergence in the special case, it is assumed that the x-component of each qubit's position on the Bloch sphere is strictly positive. In the reduction from the general to the special case, if a qubit is initially at the poles of the Bloch sphere, it will remain at the poles after the reduction and thus not have strictly positive x-coordinate as desired; thus, we only consider initializations where the qubits are not at the poles.

We consider the state $|s_0\rangle' = \bigotimes_{j=1}^{n} |s_{0,j}\rangle'$ where $|s_{0,j}\rangle' = \cos(\theta_j/2)\,|0\rangle + \sin(\theta_j/2)\,|1\rangle$. Geometrically, going from $|s_0\rangle$ to $|s_0\rangle'$ has the effect of dropping the phase for all qubits so that they lie in the xz-plane of the Bloch sphere with positive x-coordinate (assuming that none of the qubits are at the poles).

It suffices to show that we can drop the phase for single qubit of $|s_0\rangle$ (say qubit k) without changing the expected cut value; the argument can then be easily repeated for the remaining qubits to show that $|s_0\rangle$ and $|s_0\rangle'$ yield identical expected cut values. In this case, we consider the initial state $|\widehat{s_0}\rangle = \bigotimes_{j=1}^{n} |\widehat{s_{0,j}}\rangle$ where $|\widehat{s_{0,k}}\rangle = |s_{0,k}\rangle'$ and $|\widehat{s_{0,j}}\rangle = |s_{0,j}\rangle$ for $j \neq k$ (i.e. only the position of qubit k is modified). Letting $R_{x,k}(\theta), R_{y,k}(\theta), R_{z,k}(\theta)$ represent the standard rotation operator of the kth qubit (about axes x, y, z respectively) about the Bloch sphere by angle θ, we can also write

$$|\widehat{s_0}\rangle = R_{z,k}(-\varphi_k)\,|s_0\rangle\,, \tag{5.1}$$

i.e., $|\widehat{s_0}\rangle$ can be obtained from $|s_0\rangle$ by rotating around the z-axis (of the Bloch sphere) by the appropriate amount.

Let H_B and $\widehat{H_B}$ be the corresponding custom mixers for $|s_0\rangle$ and $|\widehat{s_0}\rangle$ respectively. Let $U_B(\beta_\ell) = \exp(-i\beta_\ell H_B)$ and $\widehat{U_B}(\beta_\ell) = \exp(-i\beta_\ell \widehat{H_B})$. For convenience, let $U_C(\gamma_\ell) = \exp(-i\gamma_\ell H_C)$. We can write $U_B(\beta_\ell) = U_{B,\neq k}(\beta_\ell) U_{B,k}(\beta_\ell)$ where $U_{B,k}(\beta_\ell)$ is the portion of $U_B(\beta_\ell)$ that acts on qubit k and $U_{B,\neq k}(\beta_\ell)$ is the portion that acts on the remaining qubits; we can similarly write $\widehat{U_B}(\beta_\ell) = U_{B,\neq k}(\beta_\ell)\widehat{U_{B,k}}(\beta_\ell)$ (the part of the mixer that does not affect the kth qubit remains the same). Geometrically, the operation $U_{B,k}(\beta_\ell)$ corresponds to rotating qubit k around its original position on the Bloch sphere by angle $2\beta_\ell$ so,

$$U_{B,k} = R_{z,k}(\varphi_k) R_{y,k}(\theta_j) R_{z,k}(2\beta_\ell) R_{y,k}(-\theta_j) R_{z,k}(-\varphi_j).$$

The equation above yields the following key relation between $U_{B,k}$ and $\widehat{U_{B,k}}$:

$$\begin{aligned} &R_{z,k}(-\varphi_k)U_{B,k}R_{z,k}(\varphi_k) \\ =&R_{y,k}(\theta_j)R_{z,k}(2\beta_\ell)R_{y,k}(-\theta_j) = \widehat{U_{B,k}}. \end{aligned} \tag{5.2}$$

For convenience, we will let

$$U(\gamma,\beta) = \prod_{\ell=1}^{p}\Big[U_B(\beta_\ell)U_C(\gamma_\ell)\Big],$$

and

$$\widehat{U}(\gamma,\beta) = \prod_{\ell=1}^{p}\Big[\widehat{U_B}(\beta_\ell)U_C(\gamma_\ell)\Big],$$

i.e., U_B and $\widehat{U_B}$ correspond to the QAOA-warmest circuit (excluding the initial state) for $|s_0\rangle$ and $|\widehat{s_0}\rangle$ respectively.

The claim amounts to showing (up to some global phase) the following:

$$\begin{aligned} &\langle s_0|\, U(\gamma,\beta)^\dagger H_C U(\gamma,\beta)\,|s_0\rangle \\ =&\langle \widehat{s_0}|\, \widehat{U}(\gamma,\beta)^\dagger H_C \widehat{U}(\gamma,\beta)\,|\widehat{s_0}\rangle\,, \end{aligned}$$

for any circuit depth p and any variational parameters $\gamma = (\gamma_1,\ldots,\gamma_p)$ and $\beta = (\beta_1,\ldots,\beta_p)$; and in particular, QAOA-warmest gives the same expected cut value for both $|s_0\rangle$ and $|\widehat{s_0}\rangle$.

First we observe that,

$$
\begin{aligned}
& U(\gamma,\beta)\,|s_0\rangle \\
=& \prod_{\ell=1}^{p}\Big[U_B(\beta_\ell)U_C(\gamma_\ell)\Big]\,|s_0\rangle \\
=& \prod_{\ell=1}^{p}\Big[U_{B,\neq k}(\beta_\ell)U_{B,k}(\beta_\ell)U_C(\gamma_\ell)\Big]\,|s_0\rangle \\
=& \prod_{\ell=1}^{p}\Big[U_{B,\neq k}(\beta_\ell)R_{z,k}(\varphi_k) \\
& R_{z,k}(-\varphi_k)U_{B,k}(\beta_\ell)R_{z,k}(\varphi_k) \\
& R_{z,k}(-\varphi_k)U_C(\gamma_\ell)\Big]\,|s_0\rangle \\
=& \prod_{\ell=1}^{p}\Big[U_{B,\neq k}(\beta_\ell)R_{z,k}(\varphi_k)\widehat{U_{B,k}}(\beta_\ell) \\
& R_{z,k}(-\varphi_k)U_C(\gamma_\ell)\Big]\,|s_0\rangle && \text{(by Equation 5.2)} \\
=& \prod_{\ell=1}^{p}\Big[R_{z,k}(\varphi_k)U_{B,\neq k}(\beta_\ell)\widehat{U_{B,k}}(\beta_\ell) \\
& U_C(\gamma_\ell)R_{z,k}(-\varphi_k)\Big]\,|s_0\rangle && \text{(commutativity)} \\
=& \prod_{\ell=1}^{p}\Big[R_{z,k}(\varphi_k)\widehat{U_B}(\beta_\ell)U_C(\gamma_\ell)R_{z,k}(-\varphi_k)\Big]\,|s_0\rangle && \text{(combine } \widehat{U_{B,k}} \text{ and } U_{B,\neq k}) \\
=& R_{z,k}(\varphi_k)\prod_{\ell=1}^{p}\Big[\widehat{U_B}(\beta_\ell)U_C(\gamma_\ell)\Big]R_{z,k}(-\varphi_k)\,|s_0\rangle && \text{(telescoping)} \\
=& R_{z,k}(\varphi_k)\prod_{\ell=1}^{p}\Big[\widehat{U_B}(\beta_\ell)U_C(\gamma_\ell)\Big]\,|\widehat{s_0}\rangle && \text{(by Equation 5.1)} \\
=& R_{z,k}(\varphi_k)\widehat{U}(\gamma,\beta)\,|\widehat{s_0}\rangle\,.
\end{aligned}
$$

We now finally show that QAOA-warmest initialized with $|s_0\rangle$ and $|\widehat{s_0}\rangle$ yield the same value; in particular the extraneous $R_z(\varphi_k)$ term from the previous calculations will not

effect the measurement due to commutativity with the cost Hamiltonian:

$$\begin{aligned}
&\langle s_0| U(\gamma,\beta)^\dagger H_C U(\gamma,\beta) |s_0\rangle \\
&= \langle \widehat{s_0}| \widehat{U}(\gamma,\beta)^\dagger R_z(\varphi_k)^\dagger H_C R_z(\varphi_k) \widehat{U}(\gamma,\beta) |\widehat{s_0}\rangle \\
&= \langle \widehat{s_0}| \widehat{U}(\gamma,\beta)^\dagger H_C \widehat{U}(\gamma,\beta) |\widehat{s_0}\rangle ,
\end{aligned}$$

where the last equality follows since H_C commutes with $R_z(\varphi_k)$. This completes the proof.

□

An interesting consequence of the above theorem is that when performing a random rotation of a classical solution in the process of getting the warm-started initial quantum state (as described in Section 4.1.1), the final rotation about the z-axis has no influence on the distribution of cuts obtained via QAOA-warmest (note this is not necessarily the case for QAOA-warm). Similarly, when mapping a 2-dimensional classical solution to the Bloch sphere, there is no benefit to mapping to the yz-plane specifically, we could have similarly mapped to the solution along the intersection of any plane with the Bloch sphere as long as that plane was parallel to the z-axis.

We are now ready to prove convergence in the general case.

Proposition 14. *Let $|s_0\rangle$ be any initial product state whose qubits do not lie at the poles of the Bloch sphere. Running QAOA with initial state $|s_0\rangle$ and its corresponding custom mixer yields that*

$$\lim_{p\to\infty} \max_{\gamma,\beta} F_p(\gamma,\beta) = \textit{Max-Cut}(G),$$

i.e., the expected cut value of QAOA-warmest with optimal variational parameters will yield the optimal cut value as the circuit depth p tends to infinity.

Proof. Let $F'_p(\gamma,\beta)$ be the expected cut value obtained from QAOA with custom mixers

with the state $|s_0'\rangle$ as described in Theorem 14. As a consequence of Theorem 14, we have for all $p \geq 0$ that $F_p'(\gamma, \beta) = F_p(\gamma, \beta)$. Thus,

$$\lim_{p\to\infty} F_p(\gamma, \beta) = \lim_{p\to\infty} F_p'(\gamma, \beta) = \text{Max-Cut}(G),$$

where the last equality holds by Proposition 13. □

5.3 Convergence Rate of QAOA-warmest

Such a convergence property for custom mixers is of great interest considering that many previous warm-started QAOA approaches lack such guarantees. The QAOA-warm approach by Tate et al. [3] considered arbitrary product states but with the standard mixer; numerical simulations showed that the solution quality with QAOA-warm quickly plateaus at higher circuit depths with some instances and initializations having (provably) zero improvement.

Cain et al. [57] consider the case where the initialization is a single bitstring with the standard mixer; this can be viewed as a special case of Tate et al.'s approach but where the initialization of the qubits lies at the poles. The latter's key result is that such an initialization may not converge to the Max-Cut.

One of the approaches by Egger et al. consider a mixer that is neither the standard mixer nor the custom mixer approach presented above. Their mixer is used in comjunction with what we call a single-cut initialization (see Section 4.2.3) which recovers a *particular* cut (obtained via the GW algorithm or other means) at depth 1. However, this initialization comes with no guarantees on convergence, and our results also do not apply to these different mixers.

While QAOA with custom mixers is guaranteed to converge, the *rate* of convergence is still an open question. Moreover, such convergence rate may be highly dependent on the initial quantum state that is used. While it may seem that custom mixers with a single-cut

initialization are superior due to having a better theoretical approximation ratio at depth-0, we empirically show in Section 5.4 that such a single-cut approach has an extremely slow convergence rate for small values of the regularization parameter ε. On the other hand, we show that our QAOA-warmest approach (with initializations obtained from BM-MC$_k$ solutions or projected GW solutions) empirically has a significantly faster rate of convergence.

In Section 4.2, we empirically show that depth-0 QAOA-warmest on positive-weighted graphs yields much better approximation ratios compared to depth-0 standard QAOA; a suitable convergence rate (for QAOA-warmest) would imply that QAOA-warmest will always fair better than standard QAOA for any finite depth p. We explore this empirically in Section 5.4.

5.4 Numerical Simulations and Hardware Experiments for QAOA-Warmest

In this section, we demonstrate the importance of using a suitable warm-start and show that with such warm-starts, QAOA-warmest empircally outperforms the Goemans-Williamson Max-Cut algorithm as well as standard QAOA [1] and QAOA-warm [3]. We also show that QAOA-warmest (with suitable warm-start) converges emperically fast, especially when compared to random initializations on the Bloch sphere or to single-cut initializations. In addition, we consider the effects of noise on QAOA and its variants. For our simulations, we use the same library of graphs from Section 4.5.1, using all graphs up to 11 nodes; we refer to this collection of graphs as $\mathcal{G}$ in this chapter. We consider comparisons with respect to a recent warm-starts approach of Egger et al. [42] in Section 5.4.7.

In our simulations, for each instance, we first find five locally approximate solutions to BM-MC$_2$ and keep the best (in terms of the BM-MC$_2$ objective value). We do the same for BM-MC$_3$. Similarly, for each instance, we solve the GW SDP, perform 5 projections to random 2-dimensional subspaces, and keep the best (in terms of the BM-MC$_2$ objective); this process is repeated (using the same GW SDP solution) with projections to 3-dimensional

subspaces. Next, for both the best BM-MC_2 and best BM-MC_3 solution, and for both of the best projected GW solutions (in 2 and 3 dimensions), we perform 5 different vertex-at-top rotations and 5 different uniform rotations, yielding 40 different initial warm-started quantum states per instance. We run QAOA-warm and QAOA-warmest using all 40 of these warm-started states and, for each combination of rank and rotation scheme, record which one performed the best in terms of (instance-specific) approximation ratio (as defined in Section 2.2). Finally, we run standard QAOA on the instance.

For each run for each variant of QAOA, we initialize the variational parameters γ and β close to zero[4] and each run terminates when the difference in successive values of $F_p(\gamma, \beta)$ in the optimization loop is less than $\bar{W} \times 10^{-6}$ where $\bar{W}$ is the sum of the absolute values of the edge weights.

To simplify the results, the figures and tables in this section will only consider runs of QAOA-warmest that use BM-MC_2 initializations with vertex-at-top rotations. This choice is due to runtime considerations and to allow for easier comparisons with previous related literature [3, 42]; more details on the results and the choice of this decision can be found in Section 5.4.4.

Additionally, for conciseness, in this section we will use "approximation ratio" to mean the instance-specific approximation ratio as described in Section 2.2.

5.4.1 Comparing QAOA-warmest to Other Methods

In Table 5.1, we show the proportion of graphs where each Max-Cut algorithm (the GW algorithm and variants of QAOA) performs the best for varying values of depth $p = 1, 2, 4, 8$. We observe that for nearly all instances, QAOA-warmest beats or performs as well as every other QAOA variant considered and eventually performs at least as well as the GW algorithm as the circuit depth increases. We note that at $p = 8$, QAOA-warmest beats the GW

[4]For standard QAOA, many optimizers will immediately terminate if initialized exactly at the origin due to the presence of a saddle point. Instead, each variational parameter $(\gamma_1, \ldots, \gamma_p, \beta_1, \ldots, \beta_p)$ is initialized by sampling uniformly from the interval $[-0.0001, 0.0001]$.

Table 5.1: For each Max-Cut algorithm (Goemans-Williamson, standard QAOA, QAOA-warmest, and QAOA-warm), we report the percentage of instances for which it did the best and second-best (in terms of approximation ratio). Both QAOA-warm and QAOA-warmest use BM-MC$_2$ warm-starts. There is a tie (last column) if the top two algorithms have approximation ratios that differ by no more than 0.01. QAOA-warmest is a part of every tie. Each instance is either accounted for in "Tie" or the other columns. For the column labeled *, we report, for each circuit depth, the percentage of instances for which QAOA-warmest was within 0.01 approximation ratio of the best algorithm.

	1st Best	QAOA-warmest				Standard QAOA			GW			Tie
	2nd Best	*	Standard	Warm	GW	Warmest	Warm	GW	Warm	Warmest	Standard	
Positive Weighted Graphs	p=1	**90.3 %**	0.69%	18.85%	15.22%	0.0%	0.0%	0.0%	0.17%	9.51%	0.0%	55.53%
	p=2	**98.1%**	0.69%	20.24%	25.08%	0.0%	0.0%	0.0%	0.0%	1.9%	0.0%	52.07%
	p=4	**100%**	8.65%	17.64%	20.58%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	53.11%
	p=8	**100%**	25.77%	5.01%	2.94%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	66.26%
All Graphs	p=1	**90.6%**	0.35%	17.96%	17.43%	0.0%	0.0%	0.0%	0.53%	8.84%	0.0%	54.86%
	p=2	**98.1%**	0.44%	20.17%	24.86%	0.0%	0.0%	0.0%	0.26%	1.68%	0.0%	52.56%
	p=4	**99.6%**	9.11%	19.2%	18.93%	0.0%	0.0%	0.0%	0.08%	0.26%	0.0%	52.38%
	p=8	**99.6%**	27.16%	7.96%	3.18%	0.17%	0.0%	0.0%	0.26%	0.0%	0.0%	61.23%

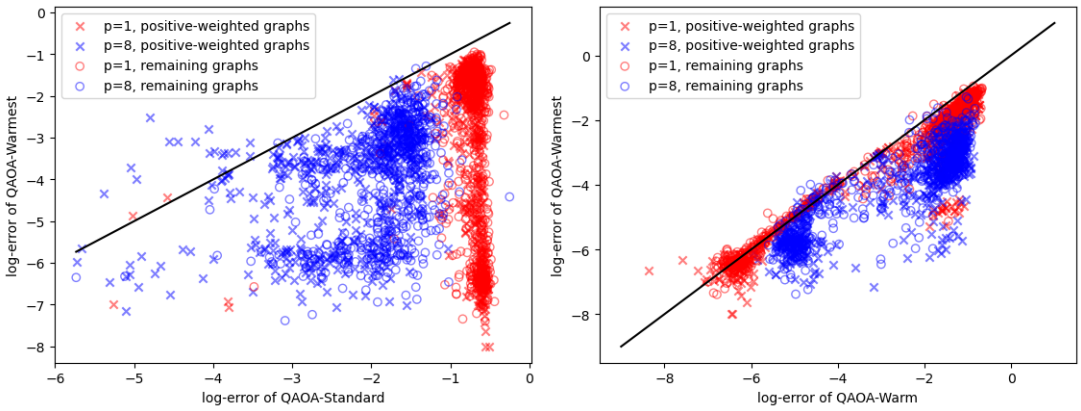

Figure 5.1: For both plots, we compare the log-error of QAOA-warmest to both QAOA-warm (right) and standard QAOA (left). BM-MC$_2$ warm-starts are used for both approaches. Each marker in the plot corresponds to a combination of instance (from our graph ensemble $\mathcal{G}$) and circuit depth (either $p = 1$ or $p = 8$) with the shape of the marker being used to denote if the instance has only positive edge weights or not. Points below the black line correspond to instances where QAOA-warmest performs better than the other algorithm being compared.

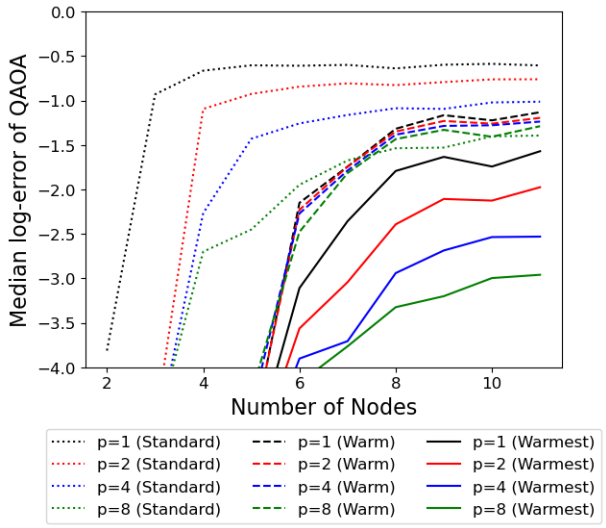

Figure 5.2: Trends in median log-error of standard QAOA (dotted), QAOA-warm (dashed), and QAOA-warmest (solid) as one varies the number of nodes and circuit depths; the median is taken across graphs in instance library $\mathcal{G}$. BM-MC$_2$ warm-starts are used for both QAOA-warm and QAOA-warmest.

algorithm on all but three instances but this is easily rectified with a suitable vertex-at-top rotation. Also at $p = 8$, QAOA-warmest outperforms standard QAOA on all but two instances but the gap in approximation ratio is less than 0.02. More information regarding these five instances can be found in Section 5.4.6.

We next report the improvement in approximation ratios when considering standard QAOA, QAOA-warm, and QAOA-warmest with circuit depths $p = 1, 8$. For convenience, for any Max-Cut algorithm, we define the approximation error (AE) by AE $= 1 -$ AR where AR is the approximation ratio. Additionally, we will refer to $\log_{10}(\text{AE})$ as the log-error. Figure 5.1 gives a comparison of log-errors achieved for various instances. All points below the $x = y$ solid line indicate instances where QAOA-warmest beats either standard QAOA or QAOA-warm. Note that due to the plots being log-scaled, being below -2 on each axis corresponds to having an approximation ratio of at least 0.99. For both plots, we see that higher approximation ratios can be achieved for positive-weighted graphs (cross-marks) and that QAOA-warmest performs significantly better for most instances. When comparing QAOA-warmest and standard QAOA at various circuit depths (red v/s blue), we see that the performance for both standard QAOA and QAOA-warmest improves at $p = 8$;

however, this phenomenon is not that apparent for QAOA-warm (which is known to plateau in performance with increased circuit depth for small instances).

Next, we show the empirical trend in approximation quality with increase in the number of nodes n and the depth of the circuit p, in Figure 5.2. We see that, across all node sizes, that circuit depth plays an important role in how good an approximation ratio one can expect to achieve using QAOA-warmest. It is clear that QAOA-warmest has superior (median) performance compared to the other algorithms for every combination of circuit depth and node-size. We remark that in contrast, an increased circuit depth resulted in only a marginal improvement in the approximation ratio for QAOA-warm, bolstering our claim that custom mixers are crucial to the improvement in performance of QAOA.

In Figure 5.3, we compare the convergence rates of standard QAOA and QAOA-warmest with various initilializations: BM-MC$_k$, single-cut initializations (as described in Section 4.2.3), and uniform random initializations[5] with custom mixers for a random 10-node instance in our graph library. Consistent with what is seen in the other figures, we see that standard QAOA, QAOA-warmest, and uniform random initializations quickly achieve high approximation ratios at relatively low circuit depths, with the BM-MC$_2$ initialization doing the best amongst the three across all circuit depths tested. On the other hand, single-cut initializations do not converge as quickly; in particular, when θ^* is small ($\theta^* \in \{0.1, 0.01\}$) hardly any improvement in the approximation ratio is observed at all. For larger regularization angles ($\theta^* = 0.5$), we do see worse performance at low depths ($p = 0, 1$) as well as a noticeable increase in performance with increased circuit depth; however, the amount of this increase is small compared to achieved by QAOA-warmest which begins to outperform the single-cut initialization (with $\theta^* = 0.5$) for $p \geq 2$. We find similar qualitative results for most other instances in our graph ensemble $\mathcal{G}$.

Table 5.2 provides a more aggregated view of the convergence of QAOA-warmest with

[5]Here, a uniform random initialization refers to a product state that is randomly created by (independently) picking a position on the surface of the Bloch sphere uniformly at random for each qubit, and then tensorizing the qubits.

Table 5.2: The percentage of instances for which QAOA-warmest achieves an instance-specific AR of 99.0% for each combination of circuit depth and initialization method (standard initialization, BM-MC$_2$ initialization, single-cut initialization with $\theta^* = 0.1$).

	p=0	p=1	p=2	p=4	p=8
BM-MC$_2$	42.3%	57.8%	75.0%	91.9%	98.1%
Standard Initialization $\|+\rangle^{\otimes n}$	0%	0.6%	2.4%	8.7%	39.4%
Single Cut Initialization with $\theta^* = 0.1$	0%	0%	0%	0%	0.7%

different choices of initializations across the entire instance library $\mathcal{G}$. For each combination of initialization method and circuit depth, the table states the percentage of instances in the library which achieved an instance-specific approximation ratio of 99.0% of higher. The data for the single-cut initializations were obtained as follows: for each instance, we obtained an optimal solution to the GW SDP relaxation, we performed 100 hyperplane roundings on the optimal SDP solution to obtain 100 cuts, we discarded all cuts whose value is more than $0.98 \cdot \text{MAX-CUT}(G)$, and we used the best remaining cut to create a single-cut initialization with regularization angle $\theta^* = 0.1$ (as described in Section 4.2.3); the discarding of cuts with very high values were done in order to ensure that, in the case of a high instance-specific approximation ratio, such a ratio can be partly attributed to the quantum circuit and not just the initial cut itself. When using the BM-MC$_2$ initializations (with vertex-at-top rotations), there are steady improvements with increased circuit depths with 42.3% of the instances achieving an instance-specific AR of 99.0% at depth-0; this percentage increases to 98.1% at depth-8. With the standard QAOA initialization, $|+\rangle^{\otimes}$, none of the instances achieve an instance-specific AR of 99.0% or more; it is not until depth $p = 8$ that we see a considerable fraction of the graphs (39.4% respectively) achieving such an AR. As for the single-cut initialization with small regularization angle, we find that nearly none of the instances achieve an instance-specific AR of 99.0% with the exception of a few instances at depth $p = 8$.

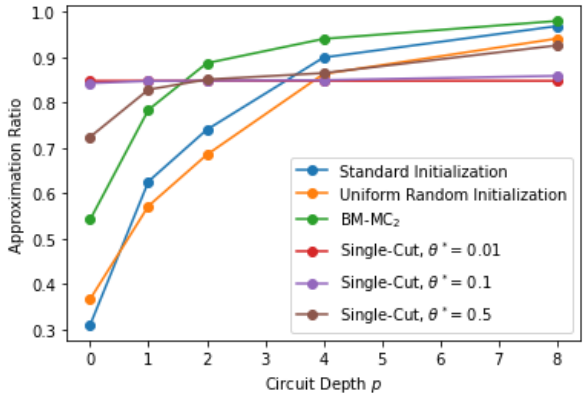

Figure 5.3: Instance specific approximation ratios achieved by QAOA-warmest with various types of initializations: standard initialization (equivalent to standard QAOA), BM-MC$_2$ initialization, single-cut initialization, and uniform random initializations) for a randomly selected 10-node instance. For QAOA-warmest, we used a BM-MC$_2$ initialization with a vertex-at-top rotation; here we intentionally chose the worst vertex (i.e. the one with worst AR at depth-0) to better illustrate the convergence rate. For the single-cut initialization, we chose a cut that obtains an instance-specific approximation ratio of 0.848 and created initial quantum states using regularization angles of $\theta^* = 0.01, 0.1$, and 0.5 radians. For the uniform random initilizations, five such initializations were created and only the best one was kept (i.e. the one with best AR at depth-0).

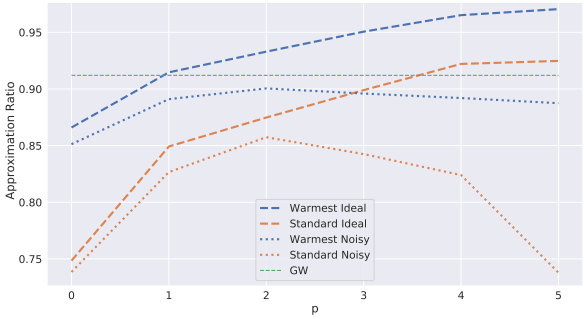

Figure 5.4: Performance of QAOA-warmest (with BM-MC$_2$ warm-starts) and standard QAOA as a function of QAOA depth for an ideal (dashed) and noisy simulation (dotted). For the chosen 20-node graph, the GW algorithm achieves an approximation ratio of 0.912, while in the ideal case, QAOA-warmest outperforms the GW algorithm for $p \geq 2$ while standard QAOA requires $p > 4$. The noise simulation is based on calibration data from IBM-Q's Guadalupe device.

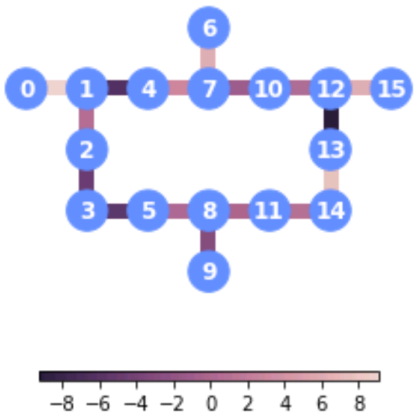

Figure 5.5: IBMQ Guadalupe device which shows the physical connectivity of qubits. We choose a native graph which matches this connectvity and random weights as indicated by color.

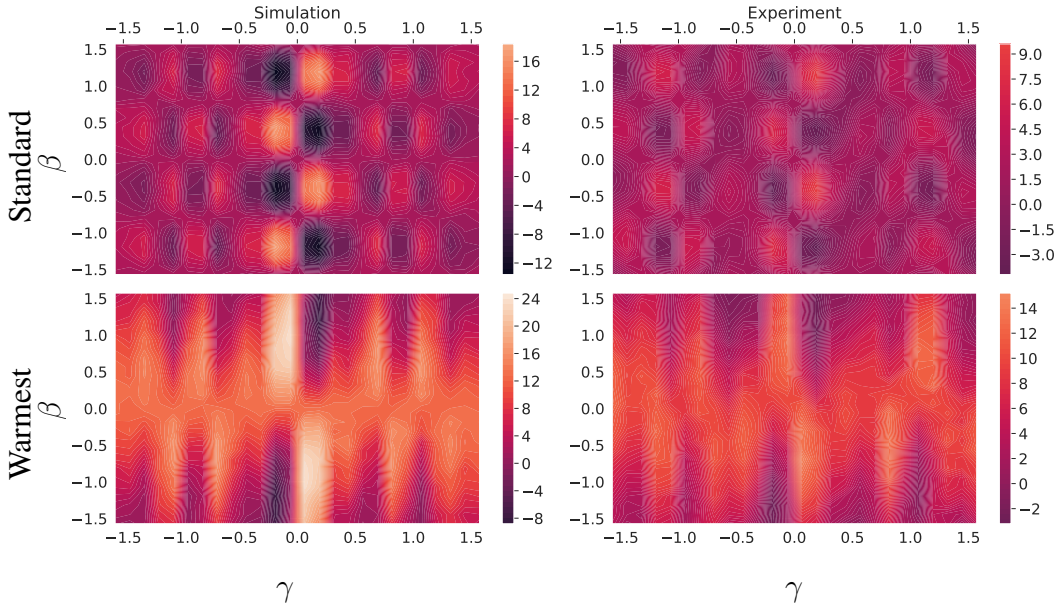

Figure 5.6: Performance of QAOA-warmest (with BM-MC_2 warm-starts) compared to standard QAOA in an ideal simulation and on IBM-Guadalupe hardware. Each subfigure is a scan of $p = 1$ parameters β vs γ, brighter regions indicating values which result in a larger cut. All figures share the same absolute color scale.

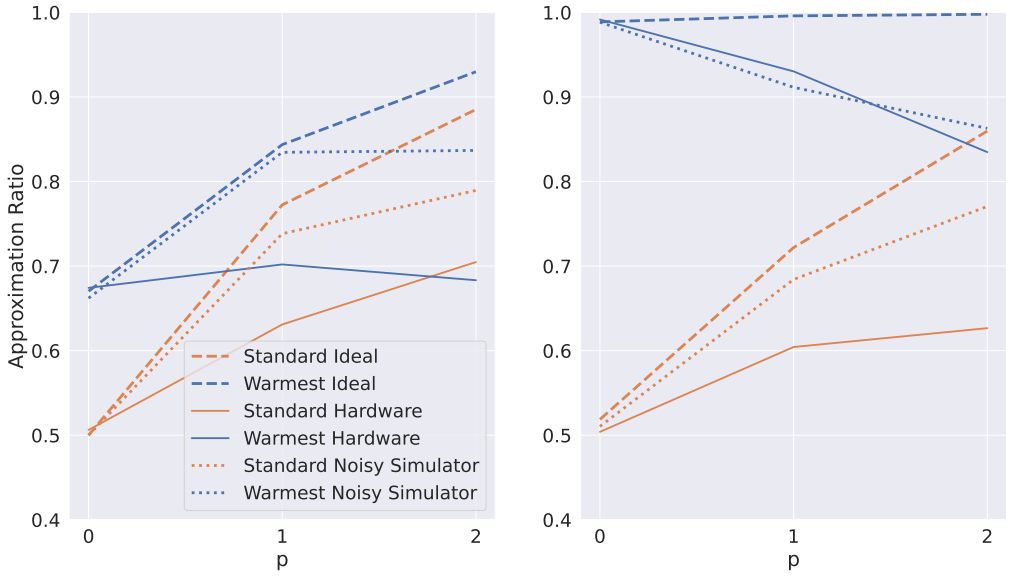

Figure 5.7: Performance of QAOA-warmest (with BM-MC_2 warm-starts) compared to standard QAOA in an ideal simulation (dashed), noisy simulation (dotted) and on IBM Guadalupe hardware (solid). Each subplot considers a different native hardware graph with randomly selected weights as well as a different choice of initialization procedures. (Left) For a random initialization of the classically informed QAOA-warmest start rotation. (Right) For an efficiently selected, optimal choice for the classically informed QAOA-warmest start rotation.

5.4.2 QAOA-warm With Noise

In addition to the theoretical (noise-less) behavior of QAOA-warmest, we also demonstrate its performance with several example cases using noise models and experiments on IBM-Q hardware. We show in Figure 5.4 the performance of QAOA-warmest and standard QAOA on an instance generated via a construction by Karloff [2]; this unweighted graph is chosen due to the fact that it is a small graph that achieves a GW approximation ratio of 0.912 (see Section 3.1) which is close to the lower bound of 0.878 provided by the GW algorithm. In contrast, both QAOA-warmest and standard QAOA outperform this approximation ratio, under ideal, noiseless conditions. However, note that QAOA-warmest outperforms standard QAOA for all QAOA depths and outperforms the GW algorithm after $p > 1$. We also consider a noise model utilizing Qiskit's built in modules [122] and use calibration data in order to simulate IBM's Guadalupe device. We note that QAOA-warmest outperforms standard QAOA for all noisy simulations, using the same fixed noise model.

In addition to this device-focused noise simulation, we also run QAOA-warmest on a native hardware graph matching IBM's Guadalupe device. In general, the connectivity of the graph and its matching to physical qubit hardware connectivity plays a key role in performance due to the overhead of inserting swap operations in order to compensate for limited connectivity [123]. Therefore, the simplest graph is a so-called native graph, which is a graph with exactly the same connectivity as the underlying physical qubit device. This graph is shown in Figure 5.5. We assign randomly chosen weights to each edge chosen from a uniform distribution $[-10, 10]$. Finding the Max-Cut solution to this graph can still be done by brute force and for a fixed choice of randomly weighted edges, we find the Max-Cut value to be approximately 33.96209.

We show the results of QAOA-warmest and standard QAOA in an ideal simulation and on hardware in Figure 5.6. The color scale is shared across all plots, showing that QAOA-warmest finds larger cut values as compared to standard QAOA, both in simulation and

on actual hardware. For hardware results, we apply the efficient SPAM noise mitigation strategy based on a CTMP strategy [124, 125].

In order to demonstrate the scaling of QAOA-warmest, we also show results for depths $p = 0, 1, 2$ in ideal simulation, noisy simulation, and on hardware, as shown in Figure 5.7. We define $p = 0$ to simply mean the preparation and measurement of the initial state[6]. In the case of QAOA-warmest, this directly demonstrates the ability of the QPU to create and measure the classically suggested cut. For both the ideal and noisy simulation, we use IBM's Qiskit software package [122]. In the case of the noisy simulation, we exercise the capability of Qiskit to pull calibration data directly from the Guadalupe device and use it to construct a noise model for use in the simulator. In principle, this combination of actual hardware calibration and noise simulation should predict the behavior of the device. However, the noise models themselves have inherent assumptions that the noise itself is uncorrelated and only directly models effects such as single and two-qubit gate errors, finite qubit lifetime and dephasing time, and readout noise. While these serve as a good starting point to model the noise in a quantum device, as shown in the Figure 5.7, there is significant disagreement between the noise simulation and the actual hardware results. This disagreement is mainly attributed to the assumptions mentioned earlier, specifically the assumption of uncorrelated noise, where physical hardware experience significant crosstalk.

In addition, Figure 5.7 shows results for two different choices of the state initialization for QAOA-warmest. The left plot shows the result of applying a uniform rotation in the classical preprocessing stage whereas the right shows the result of using the best vertex-at-top rotation amongst the 16 possible vertices, i.e., the rotation that gives the largest approximation ratio at $p = 0$. These two plots clearly show the importance of initializing the initial quantum state in an optimal way. Another important point shown in these plots is that small scale QAOA problems on 16 nodes, are nearly exactly solved when a

[6]The warm-starts come from rank-2 or rank-3 solutions whereas as the GW algorithm uses rank-n solutions. Moreover, the way cuts are determined are different (hyperplane rounding vs quantum measurement) so we should expect there to be a difference in approximation ratios.

suitable vertex-at-top rotation is chosen. When the best vertex-at-top rotation is used, the use of QAOA actually shows a decrease in solution quality on hardware. This is due to the inherent noise on the device and the fact that the solution quality is nearly optimal in the initial state. The presence of noisy two-qubit gates in further layers of the algorithm (32 CNOT gates per layer), overwhelm the small benefit of the algorithm itself for these small problems. A remaining goal then is to find native graphs on hardware for large systems, while also offering sufficiently low error rates, in order to demonstrate improved solutions with an optimally chosen initial quantum state and increased algorithmic depth ($p > 0$).

Finally, we show results for QAOA-warmest run on Quantinuum hardware in Figure 5.8. This 20-ion linear trap allows for arbitrary qubit connectivity and thus has no overhead associated with mapping a specific graph to the hardware. In this case, we again consider the 20-node Karloff instance graphs used in Figure 5.4, but here we use the GW warmest start initialization. Notably we utilize a uniform rotation which gives a large initial approximation ratio (at $p = 0$) and while hardware cannot improve on this initial state, the degradation is small considering that each p layer requires 90 two-qubit ZZ interactions (among many other single qubit operations). We also note the close agreement of the noisy simulator to the actual hardware results. In order to reduce the cost of these hardware runs, we only consider a single objective function evaluation (with 1000 shots), using noiseless simulations to find the optimal γ_i, β_i at each p depth. Even with these considerations, we see that the GW Warmest initialization outperforms the average GW performance on hardware up to $p \leq 2$. These results indicate that current quantum hardware is very close to demonstrating improvement over the GW algorithm on known hard instance graphs when using the QAOA-warmest initialization procedure and already outperforms the average performance of the GW algorithm on this particular graph.

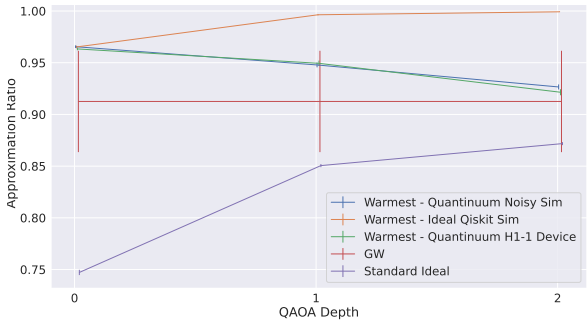

Figure 5.8: QAOA-warmest performance on Quantinuum simulators and hardware. The 20-node Karloff instance considered here is directly mapped to the fully-connected Quantinuum 20-ion hardware. In contrast to Figure 5.4, here we use a GW warmest start and find that this particular initialization outperforms the GW algorithm on average for $p \leq 2$.

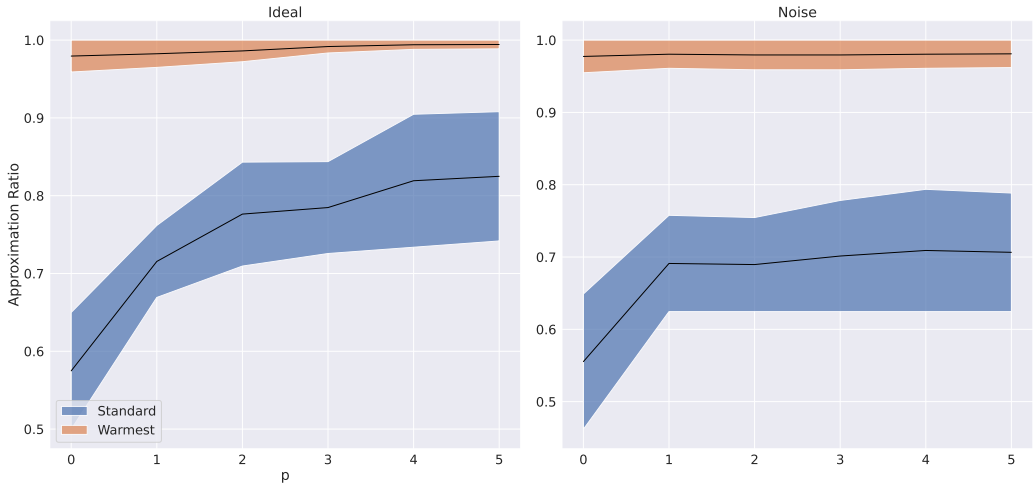

Figure 5.9: Comparison between standard QAOA mixer to using BM-MC$_2$ warm-starts with custom mixers . We show the noiseless (left) and noisy (right) case. In both cases, the custom mixer significantly outperforms the standard mixer. Shaded regions indicate the distribution of results for 20 randomly chosen 8 node graphs with positive and negative weights.

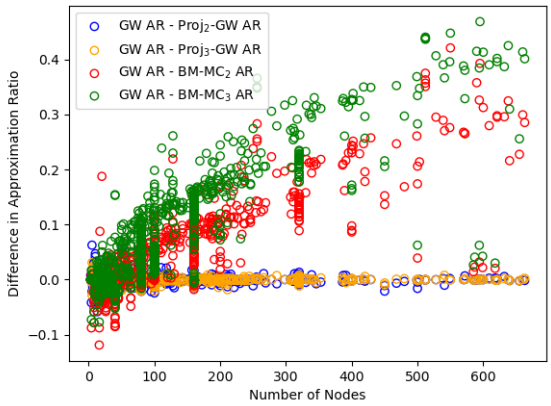

Figure 5.10: Difference in approximation ratio between rank-n GW hyperplane rounding and hyperplane rounding of various warm-start initializations (rank-k projected GW SDP solutions (Proj$_k$-GW) and approximate BM-MC$_k$ solutions for $k = 2, 3$) as the number of nodes varies. For each instance and each rank k, we obtained 5 random projections and 5 approximate BM-MC$_k$ solutions, and then kept the best one (of 5) in regards to the BM-MC$_k$ objective. Each circle in the figure corresponds to an instance from (a subset of) the MQLib library [38]; see Appendix 5.4.5 for details.

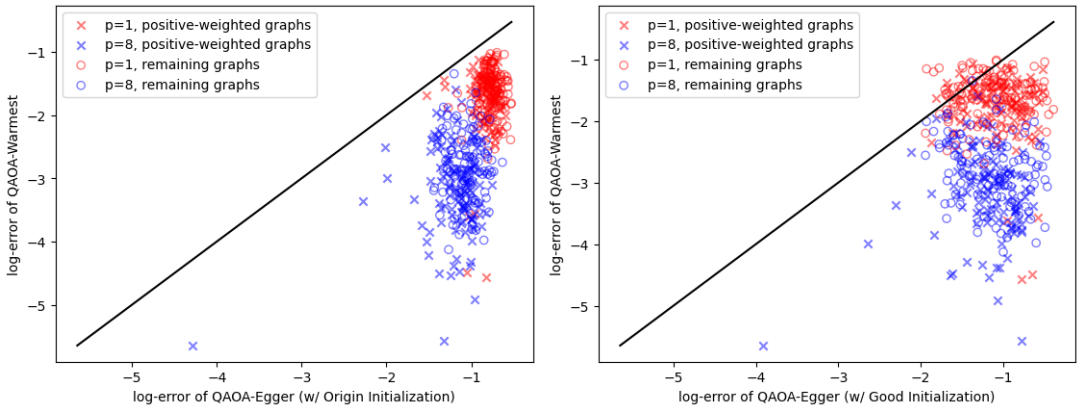

Figure 5.11: For both plots, we compare the log-error of QAOA-warmest (with BM-MC$_2$ warm-starts) to the variant of QAOA proposed by Egger et al. [42] for Max-Cut. For Egger et al.'s approach, we consider two different initializations: initializing the variational parameters to the origin (left) and initializing the parameters in a way that recovers the cut used to initialize the quantum state (right). Each marker in the plot corresponds to a combination of instance (from our graph ensemble $\mathcal{G}$) and circuit depth (either $p = 1$ or $p = 8$) with the shape of the marker being used to denote if the instance has only positive edge weights or not. Points below the black line correspond to instances where QAOA-warmest performs better than the other algorithm being compared.

Table 5.3: These tables reports the approximation ratios achieved for the five instances (amongst those in our instance library $\mathcal{G}$) for which depth-8 QAOA-warmest did not obtain the best approximation ratio when compared to depth-8 QAOA-warm, depth-8 standard QAOA, and the GW algorithm. The top and bottom tables are for instances in which standard QAOA and the GW algorithm performed the best respectively. The instances in the bottom table have the property that there exists exactly one negative edge weight whose magnitude is much larger than the other edge weights. For the bottom table, in the last column, we also include the approximation ratio for QAOA-warmest in the case where a more suitable vertex-at-top rotation is used; i.e., we take one of the vertices incident to the large-magnitude negative edge and rotate it to the top.

Instances Where Depth-8 Standard QAOA Performs Best

Instance ID	QAOA-warmest	QAOA-warm	Standard QAOA	GW
778	0.9550	0.8980	0.9635	0.9504
1820	0.9483	0.9078	0.9508	0.9429

Instances Where GW Performs Best

Instance ID	QAOA-warmest	QAOA-warm	Standard QAOA	GW	QAOA-warmest (modified)
1698	0.9899	0.9950	0.9677	0.9968	0.9998
1889	0.9867	0.9886	0.9461	0.9899	0.9995
2010	0.9762	0.9797	0.9330	0.9948	0.9992

5.4.3 Projected GW vs BM-MC Warm-Starts

As described in the previous chapter, we consider two (main) approaches for generating warm-starts: projected GW solutions and locally optimal BM-MC$_k$ solutions, with the former approach having better theoretical guarantees in regard to solution quality. However, numerical simulations displayed in Figure 5.12 show both approaches on the instance library $\mathcal{G}$ (graphs with at most 11 nodes) achieve similar expected cut values at depth $p = 1$ QAOA-warmest; in particular, the difference in (instance-specific) approximation ratio is less than 0.04 for nearly all instances. This similarity in solution quality is even more pronounced at depth $p = 8$.

Since both warm-start approaches yield similar results (in numerical simulations) and since the Burer-Monteiro approach scales better in regards to runtime (Section 4.2.1), the results in Section 5.4 assumes that locally optimal BM-MC$_k$ solutions are used to produce the warm-starts for QAOA-warm and QAOA-warmest.

Table 5.4: Multiple tables comparing the average (instance-specific) approximation ratio achieved during QAOA-warmest when utilizing different combinations of ranks and rotations during the preprocessing stage. For the top row of tables, these averages were computed using all the graphs in our graph library $\mathcal{G}$ whereas for the bottom row, we restrict our attention to only those graphs in $\mathcal{G}$ with positive edge weights.

	depth $p = 1$			depth $p = 8$		
all graphs		vert.	uniform		vert.	uniform
	rank-2	**0.9858**	0.9758	rank-2	**0.9988**	0.9977
	rank-3	0.9854	0.9535	rank-3	0.9988	0.9960
positive-weight graphs		vert.	uniform		vert.	uniform
	rank-2	**0.9867**	0.9789	rank-2	0.9991	0.9990
	rank-3	0.9864	0.9581	rank-3	**0.9993**	0.9981

5.4.4 Choice of Rank and Rotation

Table 5.4 demonstrates the average (instance-specific) approximation ratio achieved by QAOA-warmest for various combinations of the rank used for BM-MC$_k$ and the rotation scheme applied to the BM-MC$_k$ solution. We find that vertex-at-top rotations perform better than uniform rotations, especially in the context of rank-3 solutions. The data is inconclusive in regards to if rank-2 or rank-3 solutions are better for QAOA-warmest, both are promising. Finally, we remark that for depth-8 QAOA-warmest, any choice of rank or rotation scheme gave at least a 0.996 average approximation ratio across the instances.

To give fair comparison against QAOA-warm [3] (and also for comparisons with Egger et al.'s approach [42]), the results in Section 5.4 assumes that we are using rank-2 initializations and vertex-at-top rotations unless otherwise stated, since these were the recommended setting in [3].

5.4.5 Projected GW Solutions and BM-MC Scaling

As discussed in the previous chapter, two methods of creating a warm-start initialization are discussed: projecting GW SDP solutions and finding approximate BM-MC$_k$ solutions.

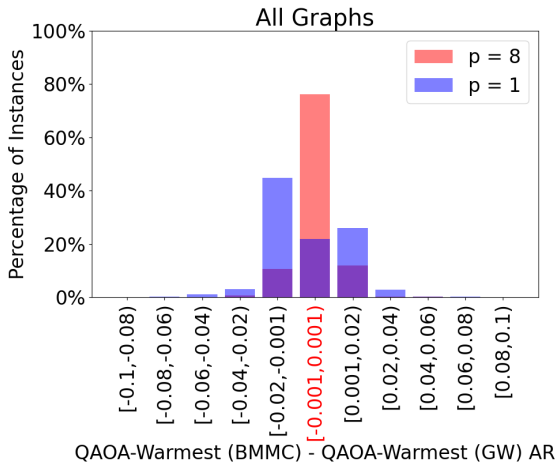

Figure 5.12: Histogram showing the difference in (instance-specific) approximation ratio (AR) when using QAOA-warmest (on instance library $\mathcal{G}$) with various warm-start strategies: projected GW and locally optimal BM-MC warm-starts. The blue and red bars correspond to depth $p = 1$ and $p = 8$ QAOA-warmest respectively with the purple regions indicating an overlap in the histograms. For both approaches, rank-3 solutions and vertex-at-top rotations are used to produce the figure; the results are similar if one instead uses a different combination of rank and/or rotation scheme.

Figure 5.10 uses instances from the MQLib [38] library[7] to compare these two methods at different ranks ($k = 2, 3$) with respect to the (instance-specific) approximation ratio they achieve with hyperplane rounding; these approximation ratios are compared against hyperplane rounding of the rank-n GW SDP solution. It is clear from the figure that the projecting of GW solutions preserves the approximation ratio (from hyperplane rounding); this is consistent with the results of Theorem 19. On the other hand, while BM-MC$_k$ solutions preserve the approximation ratio (from hyperplane rounding) for small graphs, the gap in approximation ratios (compared to rank-n GW hyperplane rounding) grows as the number of nodes increases.

5.4.6 Interesting Instances

In Section 5.4.1, Table 5.1 shows that at circuit depth $p = 8$, there are five instances (amongst the instances tested) for which QAOA-warmest did not achieve the highest

[7]MQLib [38] is a diverse library of Max-Cut instances; the results of Figure 5.10 may differ if different types/families of graphs are used (e.g. random Erdős–Rényi graphs). For each MQLib instance, we used the exact Max-Cut solver BiqCrunch [105] to try to find the optimal cut. Figure 5.10 uses all positive-weighted MQLib instances up to 663 nodes for which BiqCrunch finds the optimal cut within 24 hours.

(instance-specific) approximation ratio compared to the other algorithms considered.

Of these five instances, standard QAOA was the best algorithm for precisely two of these (instances #778 and #1820). For the remaining three instances (#1698, #1889, #2010), the GW algorithm was the best algorithm; however, all three of these instances have the property that there is a single negative edge-weight whose magnitude is much larger than the other edge weights in the graph and additional numerical simulations show that a suitable vertex-at-top rotation (selecting the vertex that is incident to the large-magnitude negative edge weight) allows QAOA-warmest to outperform the GW algorithm.

Table 5.3 gives detailed statistics for the approximation ratios achieved by each of the Max-Cut algorithms considered for these five instances.

5.4.7 Comparison with Egger et al.

Figure 5.11 compares the (instance-specific) approximation ratios achieved by QAOA-warmest and a variant of QAOA considered by Egger et al. [42]. In the context of the Max-Cut problem, Egger et al. considered an approach which takes a good starting cut $(S, V \setminus S)$ (obtained via the GW algorithm or possibly other means) and uses this cut to construct an initial quantum state $|s_0\rangle$. With this modified initial quantum state and an appropriate modification of the mixing Hamiltonian, Egger et al. prove that their variation of QAOA is recovers the cut at circuit depth $p = 1$, i.e., there is a choice of variational parameters γ_1 and β_1 such that the only cut obtained at those parameters is precisely $(S, V \setminus S)$.

To give a fair comparison for Egger et al.'s approach, we consider 10 cuts generated by the GW algorithm and take the best 5. Due to the size of the instances we consider, usually at least one of the best 5 cuts would be optimal and hence Egger et al.'s approach would essentially already start with an optimal solution which is not interesting. For this reason, in Figure 5.11, we only consider those instances (22.7% of the instance library) for which neither QAOA-warmest or Egger et al.'s approach starts with the optimal solution.

For Egger et al.'s approach, we consider two different choices for initialization of the

variational parameters: (1) near the origin and (2) the choice of parameters that recovers the value of cut used to initialize the QAOA variant (i.e. $\beta_1 = \pi/2$ with the remaining parameters being set to zero). In both cases, Figure 5.11 demonstrates that QAOA-warmest typically has the superior performance.

There are a total of 163 instances for which the approximation ratio achieved by depth-8 QAOA-warmest beats the GW algorithm (by at least 0.001) and the approximation ratio achieved by the GW algorithm beats Egger et al.'s approach (with initialization $\beta_1 = \pi/2$ with the remaining parameters being set to zero) at depth-8 (by at least 0.001). For these instances, the median gap in approximation ratio between QAOA-warmest and the GW algorithm was 0.0466 and the median gap in approximation ratio between the GW algorithm and Egger et al.'s approach is 0.0458.

5.5 Discussion

Our experimental results suggest that our QAOA-warmest method combined with initializations obtained by classical means can outperform both the standard QAOA and the Goemans-Williamson algorithm at relatively shallow circuit depths. Conversely, not all initializations on the Bloch sphere are useful; in particular random initializations underperform compared to classically obtained initializations. Moreover, adversarial initializations could be chosen if one wanted QAOA to perform poorly (i.e. by putting qubits near the poles of the Bloch sphere that correspond to the minimum cut). Overall, finding a suitable initialization is needed in order to see success in QAOA-warmest. In the case of classically-inspired initializations (e.g. Burer-Monteiro Max-Cut relaxations or projected GW SDP solutions) which are (classically) invariant under global rotations, this also includes picking a suitable rotation scheme before embedding the solution into a quantum state.

According to a paper by Farhi, Gamarnik, and Gutmann [32], QAOA needs to "see the whole graph" (i.e. have a high enough circuit depth) in order to achieve desireable results.

Their results rely on the fact that local changes in the graph (e.g. modifying an edge weight) give uncorrelated results in regards to measured qubits that are sufficiently far away from such a local change. In other words, standard QAOA cannot distinguish between graphs whose local subgraph-structure is identical. It should be noted that the circuit used in QAOA-warmest also suffers from such a locality property; however, if we consider the entirety of the QAOA-warmest procedure, including the preprocessing stage of computing warm-starts, then this procedure can possibly distinguish between graphs with identical local subgraph structure since the initial state is sensitive to the global structure of the graph (when using BM-MC$_k$ relaxations or projected GW SDP solutions). This suggests that certain negative theoretical results seen for standard QAOA may not necessarily hold for QAOA-warmest since the distinguishability arguments used would no longer apply.

While our method with custom mixers guarantees convergence for arbitrary product states, the rate of convergence for this method (and other QAOA variants) is still an open question; our simulations indicate that this convergence is heavily influenced by the initial state.

The (worst-case) approximation guarantees for our warm-starts at $p = 0$ and convergence to Max-Cut (under adiabatic limit) combined with superior empirical performance provide strong evidence for quantum advantage of this approach at low circuit depths compared to existing classical methods, especially the Goemans-Williamson approximation. An interesting open question would be to quantify the (worst-case) approximation bounds obtained by QAOA-warmest for finite circuit depth greater than zero.

CHAPTER 6

WARM-STARTS FROM CLASSICAL LOCAL ALGORITHMS

For combinatorial problems with n (binary) variables, quantum algorithms typically use exactly n qubits, one for each variable. However, such algorithms are naturally limited by the fact that, in quantum computing, any operation that is applied must be unitary (and hence also reversible). It is an interesting question to ask if some kind of advantage can be gained through the use of *additional* qubits, i.e., using $n + n'$ qubits for an n-variable combinatorial optimization problem with $n' > 0$ where the qubits are partitioned into two registers: the *system register* containing the first n qubits (which we call *system qubits*) and the *ancilla register* containing the remaining n' qubits (which we call *ancilla qubits*). With this setup, operations on the $(n + n')$-qubit device will unitarily evolve; however, by restricting one's view to just the system register, the quantum state evolves in a non-unitary, i.e. potentially irreversible, manner. The system register can then be measured at the end of such quantum algorithms to produce a (not necessarily feasible) n-length bitstring to the optimization problem.

Quantum circuits that use more qubits than variables may be of potential interest on its own; however, in this chapter[1], we focus on ways that such additional qubits can be used to construct generalized warm-starts for QAOA. In particular, by exploting ancilla qubits, we develop a framework (for both constrained and unconstrained problems) for constructing quantum states whose measurement of the system register yield the same distribution of bitstrings as those obtained from certain classes of randomized local-search algorithms. We refer to such initial states as quantum-local-search states.

Many classical local-search algorithms require one to update the state of the (classical)

[1]The results in this chapter are based on an in-progress collaboration between the thesis author and Stuart Hadfield.

system in an irreversible way, e.g. using a single bitstring to represent the best bitstring seen so far and updating this bitstring if a "neighboring" bitstring has a higher cost (and is feasible); however, by using additional memory to store the "history" of the algorithm, these local-search updates become reversible with respect to the overall state of the classical machine. In Section 6.1, with the help of ancilla qubits, we export these classical ideas to the quantum regime to construct the quantum framework described earlier.

This chapter serves as an introductory exploration of this idea of using ancilla qubits to construct warm-starts inspired by classical local-search algorithms; throughout, we provide discussion and some partial results regarding this idea that are subject to future/ongoing work.

6.1 Using Ancilla Qubits to Emulate Classical Algorithms

In this next section, we consider a general framework for emulating certain types of randomized classical local-search algorithms with quantum devices. In particular, one can create a quantum superposition of states with the property that measuring such a subsystem would yield the same distribution of states as a (possibly randomized) classical local-search algorithm. This may be interesting in its own right, but it may also be interesting from a state preparation standpoint: a classically-motivated initial quantum state can serve as a warm-start for other quantum algorithms, thus possibly giving a boost in performance. Additionally, by appropriately incorporating additional variational parameters, one is guaranteed for the performance to be at least as good as both standard QAOA and the corresponding classical algorithm.

For any optimization problem $\max_{b\in\mathcal{F}} c(b)$ (with $c : \{0,1\}^n \to \mathbb{R}$ and $\mathcal{F} \subseteq \{0,1\}$), we consider the following α-parametrized classical local-search algorithm described in Algorithm 2. For convenience, the following notation is used in the algorithm: for a bitstring b, b_j denotes the jth bit of b and $b^{(j)}$ denotes the bitstring b but with the jth bit flipped.

In words, Algorithm 6 starts with a feasible bitstring b_0, and at each step, checks if

Algorithm 2: Randomized Classical Local-Search

Input: $n, T \in \mathbb{Z}^{\geq 0}, c : \{0,1\}^n \to \mathbb{R}, \mathcal{F} \subseteq \{0,1\}, \alpha \in [0,1], \xi \in [n]^T, b_0 \in \mathcal{F}$

$b := b_0$
for $j = 1$ *through* T **do**
  if $c(b^{(\xi_j)}) > c(b)$ *and* $c(b^{(\xi_j)}) \in \mathcal{F}$ **then**
    With probability α, do nothing; else set $b := b^{(\xi_j)}$
end
return b

flipping a particular bit (e.g. the ξ_j'th bit on step j) yields a feasible bitstring with improved cost; if this check passes, then with probability α we do nothing and with probability $1 - \alpha$ we flip said particular bit. The variable $\xi = (\xi_1, \ldots, \xi_T) \in [n]^T$ represents the sequence of bit positions considered in the bitflip checks previously mentioned.

Note that when $\alpha = 0$, Algorithm 6 becomes a deterministic algorithm; on the other extreme, when $\alpha = 1$, the algorithm simply returns the initial bitstring b_0.

Now, given the same inputs as the classical Algorithm 6, we proceed to devise a quantum algorithm whose measurement of the system register yields the same distribution of strings.

Let $\overrightarrow{W}_j$ be the operator that writes from the system to the ancilla based on the improvement on flipping bit j:

$$\overrightarrow{W}_{j,k} |s\rangle |a\rangle = \begin{cases} |s\rangle |a^{(k)}\rangle , & \text{if } c(s^{(j)}) > c(s) \text{ and } s^{(j)} \text{ is feasible.} \\ |s\rangle |a\rangle , & \text{otherwise.} \end{cases}$$

For a minimization problem, one should flip the inequality in the definition above. One could also consider replacing the strict inequality ($>$) with a non-strict one ($\geq$). Moreover, if one wants to check for jumps in solution values by at least a certain amount $k > 0$, one could replace the inequality with $c(s^{(j)}) > c(s) + k$; alternatively, with $k < 0$, this would allow one to flip the ancilla even in cases where the value $s^{(j)}$ is slightly worse. The exact implementation details of $\overrightarrow{W}_{j,k}$ are outside the scope of this chapter; however, for most

combinatorial optimization problems of interest, the circuit implementation of $\overrightarrow{W}_{j,k}$ can be constructed in time polynomial in n.

Now, let $\overleftarrow{W}_{j,k}(\alpha)$ be the operator that "writes back" from the ancilla to the system based on the improvement on flipping bit j:

$$\overleftarrow{W}_{j,k}(\alpha)\left|s\right\rangle\left|a\right\rangle = \begin{cases} \sqrt{\alpha}\left|s\right\rangle\left|a\right\rangle + i\sqrt{1-\alpha}\left|s^{(j)}\right\rangle\left|a\right\rangle, & \text{if } a_k = 1 \\ \left|s\right\rangle\left|a\right\rangle, & \text{otherwise.} \end{cases}$$

In the case where α is held constant throughout, we often simply write $\overleftarrow{W}_{j,k}$ instead of $\overleftarrow{W}_{j,k}(\alpha)$. Observe that if we define $\theta = \arccos(2\sqrt{\alpha})$, then one can observe that $\overleftarrow{W}_{j,k}(\alpha)$ is simply a controlled-X rotation by angle θ on the jth system qubit (controlled by the kth ancilla qubit).

Letting $V_{j,k} = \overleftarrow{W}_j\overrightarrow{W}_j$, the overall effect is given by:

$$V_{j,k}\left|s\right\rangle\left|a\right\rangle =$$

$$\begin{cases} \sqrt{\alpha}\left|s\right\rangle\left|a^{(k)}\right\rangle + i\sqrt{1-\alpha}\left|s^{(j)}\right\rangle\left|a^{(k)}\right\rangle, & (c(s^{(j)}) > c(s) \text{ and } s^{(j)} \text{ is feas.}) \text{ and } a_k = 0 \\ \left|s\right\rangle\left|a\right\rangle, & (c(s^{(j)}) \leq c(s) \text{ or } s^{(j)} \text{ not feas.}) \text{ and } a_k = 0 \\ \left|s\right\rangle\left|a^{(k)}\right\rangle, & (c(s^{(j)}) > c(s) \text{ and } s^{(j)} \text{ is feas.}) \text{ and } a_k = 1 \\ \sqrt{\alpha}\left|s\right\rangle\left|a\right\rangle + i\sqrt{1-\alpha}\left|s^{(j)}\right\rangle\left|a\right\rangle, & (c(s^{(j)}) \leq c(s) \text{ or } s^{(j)} \text{ not feas.}) \text{ and } a_k = 1 \end{cases}.$$

Assuming that fresh ancillas are used (i.e. $a_k = 0$) at the beginning of our approach and that these ancillas are not re-used, only the first two lines of the cases described above would be performed in practice.

It should be noted that both $\overleftarrow{W}_{j,k}$ and $\overrightarrow{W}_{j,k}$ can be confirmed to be unitary operators (and hence, so is $V_{j,k}$). In particular, the i in the term $i\sqrt{1-\alpha}\left|s^{(j)}\right\rangle\left|a\right\rangle$ of $\overleftarrow{W}_{j,k}$'s definition is necessary; it's removal can be proven to remove the unitary property from the operator. Thus, although this technique is aimed at emulating a purely classical algorithm, there are

Algorithm 3: Randomized Quantum Local-Search

Input: $n, T \in \mathbb{Z}^{\geq 0}, c : \{0,1\}^n \to \mathbb{R}, \mathcal{F} \subseteq \{0,1\}, \alpha \in [0,1], \xi \in [n]^T, b_0 \in \mathcal{F}$

Set $|s_0\rangle := |b\rangle |0\rangle^{\otimes T}$

Construct and run the quantum circuit $V_{j_T,T} \cdots V_{j_1,1} |s_0\rangle$ to obtain state $|\psi_{\text{LS}}\rangle$

Measure the first n bits of $|\psi_{\text{LS}}\rangle$

return b

non-trivial phases given to some bitstrings based on the classical information. Given these $V_{j,k}$ operators, we construct the quantum local-search algorithm described in Algorithm 3 and in Proposition 15 below, we prove that both the classical and quantum local-search algorithm produce the same distribution of bitstrings. Proposition 15 implies that the quantum local-search algorithm has the same approximation factor as its corresponding classical local-search algorithm; later in Section 6.2, we explore quantum circuits that can be applied to the final state $|\psi_{\text{LS}}\rangle$ of Algorithm 3 which have the potential to further increase the instance-specific approximation ratio (beyond what is achieved by the corresponding classical local search algorithm).

Proposition 15. *The distribution of bitstrings that are output by Algorithm 2 and Algorithm Algorithm 3 are identical. Moreover, if we change the classical algorithm to start with b being a random n-length bitstring (selected uniformly as random), then we can get a similar result (regarding equality of distributions) by replacing $|s_0\rangle$ in the quantum circuit with $|s_0\rangle = |+\rangle^{\otimes n} |0\rangle^{\otimes T}$.*

Proof. We proceed by induction; the base case is trivial. Suppose that the result is true after k steps of the algorithm. Let

$$|\psi\rangle_k = \sum_{b \in \{0,1\}^n} \sum_{a \in \{0,1\}^{n'}} \beta_{b,a}^{(k)} |b\rangle |a\rangle$$

be the quantum state after k steps where $\beta_{b,a}^{(k)} \in \mathbb{C}$ for all k, a, b. For $b \in \{0,1\}^n$, let $p_k(b)$

denote the probability of obtaining bitstring b after k steps of the algorithm. We thus have,

$$p_k(b) = \sum_{a \in \{0,1\}^{n'}} |\beta_{b,a}^{(k)}|^2,$$

for all $b \in \{0,1\}^n$ by the induction hypothesis. It suffices to show that the relationship above holds after $k+1$ steps of the algorithm for all $b \in \{0,1\}^n$.

Let us fix $b \in \{0,1\}^n$, we consider different cases.

Case 1: b is not feasible. Then the result trivially holds as $p_{k+1}(b) = 0$ and $\beta_{b,a}^{(k+1)} = 0$ for all a. This is straightforward to (inductively) see since after each step of the classical algorithm, one always ends on a feasible state; similarly, by construction of the quantum algorithm, only feasible states have non-zero support in $|\psi_k\rangle$.

Case 2: b is feasible. Let $b' = b^{(\xi(k))}$. We now consider two subcases.

Cases 2a: $c(b) > c(b')$. By construction of the classical algorithm, we have

$$p_{k+1}(b) = p_k(b) + (1-\alpha)p_k(b')$$

$$= \sum_{a \in \{0,1\}^{n'}} |\beta_{b,a}^{(k)}|^2 + (1-\alpha) \sum_{a \in \{0,1\}^{n'}} |\beta_{b',a}^{(k)}|^2$$

On the quantum end, observe that $V_{\xi(k+1),k+1}$ performs the following mappings for all $a \in \{0,1\}^{n'}$:

$$|b\rangle |a\rangle \mapsto |b\rangle |a\rangle ,$$

and

$$|b'\rangle |a'\rangle \mapsto \sqrt{\alpha} |b'\rangle |a'\rangle + \sqrt{1-\alpha} |b\rangle |a\rangle ,$$

where a' denotes $a^{(k)}$. Aside from $|b\rangle |a\rangle$ and $|b'\rangle |a'\rangle$, there does not exist any other pair of system and ancilla bitstrings such that $V_{\xi(k+1),k+1}$ maps it to a superposition containing $|b\rangle |a\rangle$.

From the above, observe that:

$$\beta_{b,a}^{(k+1)} = \beta_{b,a}^{(k)} + \sqrt{1-\alpha}\beta_{b',a'}^{(k)}.$$

However, by the nature of the algorithm, we know that if $a_{k+1} = 1$, then $\beta_{b,a}^{(k)} = 0$ and similarly, if $a_{k+1} = 0$, then $\beta_{b',a'}^{(k)} = 0$ (this is due to the fact that in the first k steps of the algorithm, only the first k bits of the ancilla can possibly be 1 since we start the algorithm with the ancilla being all zeros). Thus, we really have,

$$\beta_{b,a}^{(k+1)} = \begin{cases} \beta_{b,a}^{(k)}, & a_{k+1} = 0, \\ \sqrt{1-\alpha}\beta_{b',a'}^{(k)}, & a_{k+1} = 1 \end{cases}$$

and thus,

$$\begin{aligned}
&\sum_{a\in\{0,1\}^{n'}} |\beta_{b,a}^{(k+1)}|^2 \\
=&\sum_{\substack{a\in\{0,1\}^{n'}: \\ a_{k+1}=0}} |\beta_{b,a}^{(k+1)}|^2 + \sum_{\substack{a\in\{0,1\}^{n'}: \\ a_{k+1}=1}} |\beta_{b,a}^{(k+1)}|^2 \\
=&\sum_{\substack{a\in\{0,1\}^{n'}: \\ a_{k+1}=0}} |\beta_{b,a}^{(k)}|^2 + \sum_{\substack{a\in\{0,1\}^{n'}: \\ a_{k+1}=1}} |\sqrt{1-\alpha}\beta_{b',a'}^{(k)}|^2 \\
=&\sum_{\substack{a\in\{0,1\}^{n'}: \\ a_{k+1}=0}} |\beta_{b,a}^{(k)}|^2 + \sum_{\substack{a\in\{0,1\}^{n'}: \\ a_{k+1}=0}} |\sqrt{1-\alpha}\beta_{b',a}^{(k)}|^2 \\
=&\sum_{a\in\{0,1\}^{n'}} |\beta_{b,a}^{(k)}|^2 + \sum_{a\in\{0,1\}^{n'}} |\sqrt{1-\alpha}\beta_{b',a}^{(k)}|^2 \\
=&\sum_{a\in\{0,1\}^{n'}} |\beta_{b,a}^{(k)}|^2 + (1-\alpha)\sum_{a\in\{0,1\}^{n'}} |\beta_{b',a}^{(k)}|^2 \\
=&p_{k+1}(b),
\end{aligned}$$

as desired, where the last line follows from the previous sets of calculations.

Case 2b: $c(b) \leq c(b')$. In this case, we have that

$$p_{k+1}(b) = \alpha p_k(b).$$

The result holds by an argument similar to Case 2a (moreover, the calculations are also simpler). □

6.1.1 Properties of Local-Search Quantum Algorithm

In what follows, we discuss properties of the randomized classical local-search, i.e., Algorithm 2; as a result of Theorem 15, this naturally also yields corresponding results for Algorithm 3, the quantum local-search algorithm. More specifically, we study the effect of the randomization parameter α introduced in Algorithm 2 and discuss its effect the probabilities and distributions of bitstrings that are output as a result.

First, we prove in Proposition 16 that as long as the cost function is not constant, then, for unconstrained optimization problems, if the classical Algorithm 2 is initialized with a bitstring chosen uniformly at random from $\{0,1\}^n$, then the expected cost of Algorithm 2 is strictly better than what is achieved with uniform random guessing. Because of Proposition 15, there is an analogous result for quantum Algorithm 3 where $|s_0\rangle$ is chosen to be $|+\rangle^n |0\rangle^T$ (where T is number of local-search steps, or equivalently, the number of ancilla qubits).

Proposition 16. *In Algorithm 2, let $c : \{0,1\}^n \to \mathbb{Z}$ be an cost function (with integer costs) for some unconstrained optimization problem (i.e. $\mathcal{F} = \{0,1\}$) and in Line 1 of the algorithm, set b to be a uniformly sampled bitstring from $\{0,1\}^n$. Then, if f is not a constant function, with $T = 1$ and $\alpha < 1$, there exists a choice of ξ such that the expected cost of Algorithm 2 is better than uniform random guessing, in particular, the improvement in the expected cost is at least $\frac{1-\alpha}{2^n}$.*

Proof. If f is not constant, it is straightforward to see that there exists $\hat{b} \in \{0,1\}$ and

$j \in [n]$ such that $c(\hat{b}) < c(\hat{b}^{(j)})$. With uniform random guessing, the average cost is given by $\frac{1}{2^n}\sum_{b\in\{0,1\}} c(b)$. Letting $T = 1$ and $\xi = (j)$ (here, ξ is a 1-tuple) and letting $\Pr(b)$ denote the probability that Algorithm 2 returns b, the difference in expected cost between Algorithm 2 and uniform random guessing is given by:

$$
\begin{aligned}
&\sum_{b\in\{0,1\}} c(b)\cdot\Pr(b) - \frac{1}{2^n}\sum_{b\in\{0,1\}} c(b) \\
=&\sum_{b\in\{0,1\}} c(b)\cdot\left(\Pr(b) - \frac{1}{2^n}\right) \\
=&\sum_{\substack{b\in\{0,1\}:\\ c(b)<c(b^{(j)})}} c(b)\cdot\left(\Pr(b) - \frac{1}{2^n}\right) + \sum_{\substack{b\in\{0,1\}:\\ c(b)>c(b^{(j)})}} c(b)\cdot\left(\Pr(b) - \frac{1}{2^n}\right) \\
&+\sum_{\substack{b\in\{0,1\}:\\ c(b)=c(b^{(j)})}} c(b)\cdot\left(\Pr(b) - \frac{1}{2^n}\right) \\
=&\sum_{\substack{b\in\{0,1\}:\\ c(b)<c(b^{(j)})}} c(b)\cdot\left(\frac{\alpha}{2^n} - \frac{1}{2^n}\right) + \sum_{\substack{b\in\{0,1\}:\\ c(b)>c(b^{(j)})}} c(b)\cdot\left(\left(\frac{1-\alpha}{2^n} + \frac{1}{2^n}\right) - \frac{1}{2^n}\right) \\
&+\sum_{\substack{b\in\{0,1\}:\\ c(b)=c(b^{(j)})}} c(b)\cdot\left(\frac{1}{2^n} - \frac{1}{2^n}\right) \\
=&\sum_{\substack{b\in\{0,1\}:\\ c(b)<c(b^{(j)})}} c(b)\cdot\frac{\alpha-1}{2^n} + \sum_{\substack{b\in\{0,1\}:\\ c(b)<c(b^{(j)})}} c(b^{(j)})\cdot\frac{1-\alpha}{2^n} \\
=&\sum_{\substack{b\in\{0,1\}:\\ c(b)<c(b^{(j)})}} (c(b^{(j)}) - c(b))\cdot\frac{1-\alpha}{2^n} \\
\geq&\sum_{\substack{b\in\{0,1\}:\\ c(b)<c(b^{(j)})}} \frac{1-\alpha}{2^n} \qquad\qquad \text{(as } c(b)\in\mathbb{Z},\quad \forall b\in\{0,1\}\text{)} \\
\geq&\frac{1-\alpha}{2^n}.
\end{aligned}
$$

The sum in the second-to-last line above has at least one term due to the existence of $\hat{b}$ determined earlier. □

Next, in Proposition 17, we compare the deterministic version of Algorithm 2 (at $\alpha = 0$) with the general version of the algorithm (for $\alpha \in [0, 1]$) and their probabilities of outputting the same final bitstring, which can be used to provide a bound on the expected cost of Algorithm 2 for general $\alpha \in [0, 1]$.

Proposition 17. *Let b^* be the output of Algorithm 2 when $\alpha = 0$ and all other inputs are fixed (note that the algorithm is deterministic in this case). Then the probability of Algorithm 2 returning b^* for general $\alpha \in [0, 1]$ is at least*

$$(1 - \alpha)^T.$$

Moreover, if α is chosen such that $\alpha = \frac{1}{kT}$ for some $k \in \mathbb{R}^+$, then as $T \to \infty$, the limit of the expected cost of the output of Algorithm 2 is at least $c(b^*) \cdot e^{-1/k}$.

Proof. Let T' be the number of times that the **if** condition of Algorithm 2 is satisfied; clearly $T' \leq T$. For general $\alpha \in [0, 1]$, one way that Algorithm 2 can output b^* is to perform the same sequence of bitflips as the deterministic version of the algorithm (at $\alpha = 0$) which will occur with probability $(1 - \alpha)^{T'} \geq (1 - \alpha)^T$. For general $\alpha \in [0, 1]$, Algorithm 2 may possibly output b^* in some other way as well, but this can only increase the probability of b^* being output.

Note that for small enough α, this implies that the expected cost function value will be close to the cost function of b^*, the output of the deterministic version of the classical local-search algorithm. More specifically, observe that choosing $\alpha = \frac{1}{kT}$ for some $k \in \mathbb{R}^+$, a bound on the expected cost of Algorithm 17 can be obtained as follows:

$$
\begin{aligned}
\mathbb{E}[c(b)] &\geq c(b^*) \cdot \Pr(b = b^*) \\
&\geq c(b^*)(1-\alpha)^T \\
&= c(b^*)\left(1 - \frac{1}{kT}\right)^T \\
&\to c(b^*) \cdot e^{-1/k}. && \text{(as } T \to \infty\text{)}
\end{aligned}
$$

□

For many combinatorial optimization problems, it is known that the deterministic version of Algorithm 2 (with a carefully chosen sequence ξ) can achieve a locally optimal solution (with respect to bitflips) in time polynomial in n. For example, in the case of Max-Cut on a unit-weight graph with n nodes and m edges, Algorithm 2 with $\alpha = 0$, $T = nm$ and a suitable choice of ξ is guaranteed to return a locally optimal cut and it can be proven that this cut contains at least half the edges [126]; thus, by as a result of Proposition 17 (with $\alpha = 1/(kT), T = nm, k \in \mathbb{R}^{>0}$) the randomized version of Algorithm 2 has an approximation ratio approaching $\frac{1}{2} \cdot e^{-1/k}$ for large n.

Initially, it may appear that choosing $\alpha > 0$ (making Algorithm 2 non-deterministic) may not be a wise choice, since the distribution of bitstrings that are output (with positive probability) will contain bitstrings whose cost values are *worse* than the bitstring b^* output by the deterministic version of the algorithm. However, when considering the final quantum state of the quantum version of the algorithm (Algorithm 3) as a warm-start to another quantum circuit (like QAOA), we know that warm-starts consisting of a single bitstring perform poorly [57] and our work in Chapters 4 and 5 of this thesis suggests that warm-starts with a "richer" distribution of states often perform better in the context of QAOA-like algorithms despite possibly having lower approximation ratios when just measuring the warm-start state itself.

So far, the discussion above primarily focuses on the probability of measuring a particular bitstring, namely, the one obtained as the output of the deterministic version of Algorithm 2. Below, for the Max Independent Set problem (described in Section 2.5), we provide bounds on the probability for *each* feasible solution being output in Algorithm 2.

Proposition 18. *Let c and F be chosen so as to correspond to an instance of Max Independent Set on a graph $G = (V, E)$ and consider starting the classical algorithm with the all-zeroes bitstring. Let $\alpha \in (0, 1)$. Let ξ in the classical algorithm correspond to some permutation of $[n]$. Then all valid independent sets have a positive probability of being sampled, in particular, an independent set of size k will have probability at least $f(\alpha) = \alpha^{n-k}(1-\alpha)^k$ of being sampled. For fixed k and n, $f(\alpha)$ is maximized at $\alpha = 1 - k/n$.*

Proof. Without loss of generality, let ξ be the identity permutation, i.e., $\xi(j) = j$ for all $j \in [n]$. Let $p_j(b)$ be the probability that the classical algorithm would return the bitstring b after j steps. We claim that if b is a independent set of size k and if $b_\ell = 0$ for all $\ell = j+1, \ldots, n$, then

$$p_j(b) \geq \alpha^{j-k}(1-\alpha)^k.$$

We proceed by induction. The base case, $j = 0$ is trivial since only the zero bitstring (which corresponds to an independent set of size $k = 0$) would be obtained and for $j, k = 0$, we have $\alpha^{j-k}(1-\alpha)^k = \alpha^j = \alpha^0 = 1$ for $\alpha \in (0, 1)$. Now, let b be correspond to an independent set of size k where $b_\ell = 0$ for all $\ell = j+2, \ldots, n$. If $b_{\ell+1} = 0$, then by the classical algorithm, we have

$$p_{j+1}(b) \geq \alpha p_j(b) = \alpha(\alpha^{j-k}(1-\alpha)^k) = \alpha^{j+1-k}(1-\alpha)^k.$$

Otherwise, if $b_{\ell+1} = 1$, then by the classical algorithm, we have $b^{(\ell+1)}$ corresponds to an

independent set of size $k-1$ and thus,

$$p_{j+1}(b) \geq (1-\alpha)p_j(b^{(j+1)}) = (1-\alpha)(\alpha^{j-(k-1)}(1-\alpha)^{k-1}) = \alpha^{j+1-k}(1-\alpha)^k.$$

The result then applies by the above inductive argument in the case that $j = n$.

For the last statement of proposition, observe that the derivative of f is given by,

$$f'(\alpha) = (n-k)\alpha^{n-k-1}(1-\alpha)^k - k\alpha^{n-k}(1-\alpha)^{k-1}.$$

Setting $f'(\alpha) = 0$ and dividing through by the appropriate factors, we obtain:

$$0 = (n-k)(1-\alpha) - k\alpha.$$

Solving the above for α yields that $\alpha = 1 - k/n$, as desired. (Note that the maximum does not occur at the extreme points of $\alpha = 0$ or $\alpha = 1$ as $f(0) = f(1) = 0$.)

□

6.2 Effect of QAOA on States with Ancilla

The quantum local-search algorithm (Algorithm 3) described earlier may be interesting in its own right; however, one may be interested in using such a procedure as a co-routine in a larger quantum algorithm. In particular, one may be interested in warm-starting a QAOA-like algorithm with the local-search-state $|\psi_{\text{LS}}\rangle$ obtained from Algorithm 3 as a means of potentially improving the instance-specific approximation ratio obtained from simply measuring $|\psi_{\text{LS}}\rangle$ (which, by construction, would yield an instance-specific approximation ratio that matches the corresponding classical local-search algorithm used to generate $|\psi_{\text{LS}}\rangle$).

Throughout this section, we will use $|s_0\rangle$ to denote some initial quantum state over n qubits, $U_{\text{LS}}(\alpha)$ to denote the quantum circuit in the quantum local-search algorithm (Algorithm 3) parametrized by $\alpha \in [0,1]$ (not including $|s_0\rangle$, and $U_{\text{P}}(\rho)$ to denote some general

parameterized circuit (like QAOA) parameterized by $\rho \in \mathbb{R}^p$ for some p and for convenience, we assume that $U_P(\mathbf{0})$ is the identity operator. Note that at the very least, the circuit $U_{\mathrm{P}}(\rho)U_{\mathrm{LS}}(\alpha)\left|s_0\right\rangle$ with the right choice of parameters can guarantee the same performance as simply running just $U_{\mathrm{P}}(\rho)$ or just $U_{\mathrm{LS}}(\alpha)$ on $\left|s_0\right\rangle$ as

$$U_{\mathrm{P}}(\rho)U_{\mathrm{LS}}(1)\left|s_0\right\rangle = U_{\mathrm{P}}(\rho)\left|s_0\right\rangle,$$

and

$$U_{\mathrm{P}}(\mathbf{0})U_{\mathrm{LS}}(\alpha)\left|s_0\right\rangle = U_{\mathrm{LS}}(\alpha)\left|s_0\right\rangle;$$

of course, the hope is that there exists some non-trivial choice of (ρ, α) that yield *strictly* better expected cost values compared to the trivial choices of (ρ, α) above.

Next, note that for problems in constrained optimization, if $\left|s_0\right\rangle$ is a quantum state consisting of a superposition of feasible states, then so is $\left|\psi_{\mathrm{LS}}\right\rangle = U_{\mathrm{LS}}\left|s_0\right\rangle$. Because of this, it may be desirable to choose U_{P} so that feasibility is maintained throughout the circuit $U_{\mathrm{P}}(\rho)U_{\mathrm{LS}}(\alpha)\left|s_0\right\rangle$. It is immediately clear that choosing the standard QAOA ansatz does not maintain this feasbility due to its mixer $H_B = \sum_{j\in[n]} \sigma_j^x$. Meanwhile, in the Quantum Alternating Operator Ansatz (QAOA++) [127], the mixer is adjusted in a way that maintains feasibility making it a potentially more suitable choice of U_{P} compared to standard QAOA. For standard QAOA, one can work around the issue of feasbility by introducing penalty terms in the cost Hamiltonian; however, this has been shown to yield poor results empirically [128]. In regards to the QAOA-Warmest method proposed in Chapter 5, such an ansatz can not be applied to $\left|\psi_{\mathrm{LS}}\right\rangle$ since QAOA-Warmest assumes that the warm-start state is a product state and in general, this is not the case for $\left|\psi_{\mathrm{LS}}\right\rangle$.

Finally, we note that the standard QAOA, the variants above, and many other variants of QAOA considered would only apply operations directly to the system register (and not to the ancilla register). Because of this, the phenomena of constructive and destructive interference, which is arguably one of the most important aspects of quantum computing,

is not present between basis states that have different ancilla tags; this is explored more in the next subsection.

6.2.1 Effects on Measurement on States with Ancilla

To understand the effect that the ancillas have on measurement in the context of other algorithms, let us consider the simple case of a single ancilla qubit, i.e., $n' = 1$.

Suppose, before doing any QAOA, we run one of the classically-inspired algorithms to obtain some state which can be generically written as,

$$|\psi\rangle = A_0 + A_1$$

where

$$A_0 = \sum_{x \in \{0,1\}^n} \gamma_{x,0} |x\rangle |0\rangle ,$$

$$A_1 = \sum_{x \in \{0,1\}^n} \gamma_{x,1} |x\rangle |1\rangle .$$

Let U be the standard QAOA circuit (of any depth); note that such a circuit will have no effect on the ancilla qubits. The expected cut value will be:

$$\langle\psi| U^* H_C U |\psi\rangle = (A_0 + A_1)^* U^* H_C U (A_0 + A_1)$$

$$= (UA_0)^* H_C (UA_0) + (UA_1)^* H_C (UA_1) + \underbrace{(UA_0)^* H_C (UA_1) + (UA_1)^* H_C (UA_0)}_{=0}$$

$$= (UA_0)^* H_C (UA_0) + (UA_1)^* H_C (UA_1),$$

where equality to 0 is due to the states in A_0 and A_1 (and hence UA_0 and UA_1) being orthogonal to one another (since their ancilla bit is different). The above illustrates that the result of applying QAOA to the overall state $|\psi\rangle$ is the same as applying QAOA to both parts (A_1 and A_2) and simply adding the probabilities. In other words, *the states in A_0 and*

A_1 do not interfere (in the quantum sense).

To make this more clear, suppose that we fix a bitstring y and suppose that the final quantum state $U|\psi\rangle$ contained the following: $\delta_{y,0}|y\rangle|0\rangle + \delta_{y,1}|y\rangle|1\rangle$ where $\delta_{y,0}, \delta_{y,1} \in \mathbb{C}$ are arbitrary complex coefficients. Suppose we were to measure $U|\psi\rangle$ and discard the ancilla, then the probability of measuring y is

$$|\delta_{y,0}|^2 + |\delta_{y,1}|^2.$$

Now, consider the case where instead of the initial state $|\psi\rangle$, we consider a modified initial state $|\psi\rangle'$ which is exactly the same as $|\psi\rangle$ but with the ancilla tags removed. (In some cases, this may not lead to $|\psi\rangle'$ being a properly normalized state, although there are cases where it is, for example, if all the $\gamma_{x,0}$'s and $\gamma_{x,1}$'s are non-negative.) In this case, it is straightforward to prove that the probability of measuring y in the state $U|\psi\rangle'$ is,

$$|\delta_{y,0} + \delta_{y,1}|^2.$$

In the general case,

$$|\delta_{y,0}|^2 + |\delta_{y,1}|^2 \neq |\delta_{y,0} + \delta_{y,1}|^2.$$

The left-hand expression is reminiscent of classical probability (positive probabilities being added together) whereas the latter shows how the coefficients can possibly constructively or destructively combine (as is usual in quantum computing). One direction of research to pursue is how we can alter the standard QAOA circuit to possibly get around the "lack of interference" issue (or possibly, this lack of interference can be exploited in a positive way).

6.3 Discussion and Future Directions

To summarize, we have provided a general framework of quantum algorithms which are capable of constructing a quantum state which has the same probability distribution as what can be obtained via a (randomized) classical local-search algorithm. For the randomized classical local-search algorithm parameterized by α, we proved that this yields a better result compared to uniform random guessing. Additionally, we provide a bound on the expected cost value of the randomized local-search algorithm in terms of the result of the deterministic version of the algorithm (at $\alpha = 0$) and in the case of Max Independent Set, we provided α-dependent bounds on the probabilities of obtaining any feasible independent set. Determining the optimal choice of α for various problems (both in regards to the classical local-search algorithm itself and in the context of quantum warm-starts) is still open; moreover, further modifications of the classical Algorithm 2 may of be interest, including choosing the sequence ξ of bitflips randomly or evenly dynamically, or considering local changes to bitstrings beyond single-bitflips.

It should also be noted that the class of randomized local-search algorithms is not all-encompassing, for example, for Max-Cut, instead of considering a *sequence* of bitflips, one may consider bit positions whose flip improves the solution by at least some threshold and then flip all such bits at once. Hirvonen et al. [129] prove that for triangle-free d-regular graphs, d-dependent approximation ratio greater than $1/2$ can be achieved with such a thresholding approach. Variations of this classical approach have been used to illustrate the limitations of low-depth standard QAOA [48, 130]. One potential research direction would be to adapt our approach to create a quantum state whose measurement of the system register yields bitstrings that match the distribution obtained from such thresholding algorithms, and to analyze the effectiveness of such a state as a warm-start state for QAOA-like algorithms.

This chapter briefly explored the possibility of utilizing the result $|\psi_{\text{LS}}\rangle$ of quantum

local-search (Algorithm 3) as a warm-start to parameterized quantum circuits such QAOA or its variants. The QAOA-like circuits we had considered in this chapter lacked constructive or destructive interference between basis states with different ancilla tags; we conjecture that a QAOA-like algorithm (applied to $|\psi_{\text{LS}}\rangle$) that non-trivially act on both the system and ancilla register (as is done in Algorithm 3) have a higher likelihood of yielding higher-quality solutions in expectation. When being applied to $|\psi_{\text{LS}}\rangle$, substituting the standard QAOA mixer with a Grover-inspired mixer $H_B = |\psi_{\text{LS}}\rangle \langle\psi_{\text{LS}}|$ might be a desirable choice as its acts on both the system and ancilla registers in addition to maintaining feasibility; however, we leave such an exploration for potential future work.

CHAPTER 7
CONCLUSION

To summarize, in Chapter 3, we first identified classes of Max-Cut instances that may be of interest in regards to quantum benchmarking or demonstrating quantum advantage; such classes, for which either the GW algorithm or classical heuristics perform theoretically or empirically poorly (respectively), contain small instances under 1000 nodes which are suitable for current and near-term NISQ devices. In order to potentially demonstrate such a quantum advantage, it may be the case that improvements and modifications to the standard QAOA algorithm are needed; to this end, in Chapters 4 and 5, we introduce the algorithms QAOA-warm and QAOA-warmest for Max-Cut which utilize classically-obtained warm-starts, the latter of which converges to the optimal solution with increased circuit depth (as a result of the adiabatic theorem) and empirically outperforms standard QAOA across all depths tested. Additionally, both QAOA-warm and QAOA-warmest, at depth-0 (i.e. quantum measurement of the initial warm-start state), achieve a approximation ratio of 0.658 which is better than the 0.5 approximation ratio achieved by uniform random guessing. Finally, in Chapter 6, we consider one last warm-start approach which exploits the use of ancilla qubits in order to construct a warm-start whose measurement (of some subset of the qubits) yields a distribution of bitstrings that matches what is obtained by a randomized local-search algorithm; this approach can be utilized for both unconstrained and constrained optimization problems. This ancilla approach may be of potential interest from a theoretical perspective; however, the implementation of the gates required for such an approach is non-trivial and is most likely not suitable for near-term NISQ devices.

7.1 Open Questions

To end, we provide the reader with a list of open questions and potential research directions in regards to the work presented in this thesis.

Chapter 3

Q1. *Can machine-learning be used to generate more instances that may be of interest from a quantum benchmarking or advantage viewpoint?* While such an approach may not easily lend itself to finding easily-definable *classes* of instances with interesting theoretical properties, the use of machine-learning approaches such as Generative Adversarial Networks [131] have the potential to generate such instances that are suitably-sized for NISQ devices.

Q2. *Do there exists other families of strongly-regular graphs for which the GW algorithm yields a low instance-specific approximation ratio?* The family of strongly-regular graphs that were considered in Chapter 3 had the property that the optimal GW SDP had angles $\arccos(-1/3)$ between adjacent vertices, which is exactly the angle between any two vertices and the center of a regular tetrahedron [102]. In higher dimensions, regular ℓ-simplices (in $\mathbb{R}^\ell$) generalize the notion of a regular tetrahedron and have an angle between $\arccos(-1/\ell)$ between vertices; a more thorough analysis may potentially reveal families of strongly-regular graphs that have such angles for various values of ℓ.

Chapters 4 and 5

Q3. *Can provably guarantees be found for depth-1 QAOA-warmest?* For standard QAOA, guarantees on the approximation ratio have been found for depths up to $p = 2$ for 3-regular graphs [1, 44]. In the case of QAOA-warmest with finite depth, we have only provided a guarantee at depth $p = 0$. One particular challenge is that when considering the neighborhood of an edge, the initial positions of the qubits in the neighborhood of that edge also play a role in the probability of that edge being flipped; thus, instead of having to consider a finite number of potential edge-neighborhoods (as is done in [1, 44]), there

are now infinitely many possibilities to consider when considering all potential ways in which qubits can be initialized on the surface of the Bloch sphere. A guarantee can also potentially be found by finding a closed-form analytical expression for the expected cost of low-depth QAOA, as is done by Wang et al. [99] at depth $p = 1$ for standard QAOA; however, the derivation of such an expression is dependent on numerous simplifications and cancellations that are not possible in the case of QAOA-warmest.

Q4. *Can QAOA-warmest be adapted to work with constrained optimization problems?* Although the ancilla approach in Chapter 6 can be used for constrained optimization, it may not be practical in the NISQ era of quantum computing. Meanwhile, for many quantum devices, QAOA-warmest is nearly as easy to implement as standard QAOA and performs empirically well, thus any way to adapt QAOA-warmest to work with constrained optimization problems (while maintaining its performance) would be of great interest. It is not immediately clear how to modify QAOA-warmest so that feasibility is maintained throughout the quantum circuit (including the initial warm-start state). Alternatively, another potential approach would be to utilize penalty terms in the cost Hamiltonian and analyze their effects.

Q5. *Can other modifications of the QAOA algorithm be used in conjunction with QAOA-warmest?* In addition to the approaches discussed in this thesis, many other variants of QAOA (such as those discussed in Section 1.2) have been proposed, many of which have the potential to be combined with the approaches suggested in this thesis. For example, although QAOA-warmest performs relatively well with our naive choice of parameter initialization, adapting the results and techniques that are known (for standard QAOA) in regards to initialization and optimization of the variational parameters could potentially yield faster and higher-quality solutions for QAOA-warmest.

Q6. *Does the use of a warm-start allow one to break past the limitations of QAOA that are due to its locality?* The standard QAOA is *local* in the sense that the probability of an edge being cut is uncorrelated with edges that are sufficiently far away and Farhi et

al. [32] use this fact to prove that a high-enough circuit depth is needed in order to "see the whole graph" (as described by Farhi et al. [32]) and avoid such a locality limitation. Although the quantum circuit for QAOA-warmest suffers from a similar locality property, the overall algorithm is non-local when warm-starts are obtained via projected GW SDP solutions since a small change in the given instance (e.g. edge deletion) can have a global impact on the optimal GW SDP solution. Thus, because of its (overall) non-locality, such negative results for standard QAOA may not necessarily apply to QAOA-warmest.

Chapter 6

Q7. *Do there exists combinatorial optimization problems of interest for which the ancilla approach in Chapter 6 can easily be realized on near-term quantum devices?* In general, a more thorough analysis of the circuit complexity of the ancilla approach would be of interest. As written, our quantum local-search algorithm, Algorithm 3, would require a comparator in order to compare the cost function before and after flipping a bit of some bitstring; however, for many problems, one may not necessarily need to compute the entire cost function to make such a comparison and such a comparison can thus be made much more efficiently. The use of such insights have the potential to simplify the implementation of the operations considered in our ancilla approach for certain problems.

Q8. *How should the QAOA algorithm be adapted if warm-started with the quantum-local-search state obtained via Algorithm 3 and what can be said about the performance and properties of such a QAOA-variant?* As discussed in Chapter 6, a key issue with standard QAOA and most of its variants is that they act separately on basis states with different ancilla tags, meaning that no constructive or destructive interference is possible. These notions of interference are arguably one of the most important key features of quantum computing, thus, it is reasonable to believe that any QAOA-like algorithm that does not act non-trivially on both the system and ancilla register throughout the circuit is likely to perform poorly. For future work, we propose adapting QAOA with a Grover-like mixer as proposed in [61] which "does not see" the partitioning of the the qubits into registers.

Appendices

APPENDIX A

PARTIAL GEOMETRIES

In this Appendix, we motivate the choice of parameters ($n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$) for the family of strongly regular graphs considered in Section 3.2. The work in Section 3.2 was initially inspired by several instances in the MQLib library that we found where the GW algorithm achieved an approximation ratio of 0.912; additionally, the comments in the graph files suggested that such graphs are strongly regular graphs (which we later verified to be true)

Many strongly-regular graphs have a corresponding *partial geometry* that they are associated with; the next few definitions and propositions below establish this connection between partial geometries and strongly-regular graphs.

Definition 32. *An incidence structure $C = (P, L, I)$ consists of points P, lines L, and incidence relation $I \subseteq P \times L$ where a point p is said to be incident with a line l if $(p, l) \in I$. We say that C is a partial geometry if there exists positive integers s, t, α such that:*

- *For distinct points $p, q \in P$, there is at most one line incident with both of them.*
- *Each line is incident with $s+1$ points.*
- *Each point is incident with $t+1$ lines.*
- *If a point p and a line l are not incident, there are exactly α pairs $(q, m) \in I$, such that p is incident with m and q is incident with l.*

We say that C is a PG(s, t, α) if it is a partial geometry with parameters s, t, α as described above.

Definition 33. *A partial geometry C is a generalized quadrangle if there exists positive integers s, t such that C is a PG$(s, t, 1)$, i.e., a partial geometry with parameter $\alpha = 1$. We*

say that C is a GQ(s,t) if it a generalized quadrangle with parameters s,t as described above.

Definition 34. *A graph $G=(V,E)$ is said to be a geometric graph if there exists a partial geometry C for which G is the point graph (also known as the collinearity graph) of C. The collinearity graph of C is a graph whose vertices are the points of C where two points are considered adjacent if and only if they determine a line in C (i.e. they are colinear).*

As seen in Proposition 19 below, every partial geometry has a corresponding strongly-connected graph; more precisely, the collinearity graph of a partial geometry is always a strongly-regular graph.

Proposition 19 (Bose [101])**.** *If G is a geometric graph with a corresponding partial geometry PG(s,t,α) (i.e. G is the point graph of C), then G is a strongly-regular graph with parameters,*

$$v=(s+1)\frac{(st+\alpha)}{\alpha},$$

$$k=s(t+1),$$

$$\lambda=s-1+t(\alpha-1),$$

$$\mu=\alpha(t+1).$$

There are many strongly-regular graphs whose form is the same as the parameter sets described in Proposition 19; however, they may not necessarily be geometric, i.e., they are not necessarily the collinearity graph of some partial geometry. Such graphs are considered to be pseudo-geometric; a more precise definition is provided below.

Definition 35. *Suppose there exists positive integers s,t,α with $\alpha\leq\min(s+1,t+1)$ and let G be a SRG(v,k,λ,μ) with the parameters (v,k,λ,μ) set as in Proposition 19, i.e., $v=(s+1)\frac{st+\alpha}{\alpha}$, $k=s(t+1)$, $\lambda=s-1+t(\alpha-1)$, $\mu=\alpha(t+1)$. We refer to such a G as being pseudo-geometric for PG(s,t,α), even if no such partial geometry PG(s,t,α) exists.*

For the strongly-regular graphs in the MQLib library where the GW algorithm achieved an approximation ratio of 0.912, it is natural to ask if these are pseudo-geometric with respect to some partial geometry. Such instances in the MQLib library were strongly-regular with parameters $\lambda = 2$ and $\mu = k/3$; Theorem 20 illustrates that if one has a strongly-regular graphs with such a property and if the graph is pseudo-geometric, then it must be pseudo-geometric with respect to generalized quadrangles of order $(3, t)$ for some t; moreover, we show that such graphs have exactly the same strongly-regular graph parameters as those considered in Section 3.2.

Proposition 20. *Let G be an SRG(n, k, λ, μ) where $\lambda = 2$ and $\mu = k/3$ and suppose G is pseudo-geometric, then G is pseudo-geometric for GQ$(3, t)$ for some positive integer t and moreover, $n = 4(3t + 1), k = 3(t + 1), \lambda = 2, \mu = t + 1$.*

Proof. Let G be an SRG(v, k, λ, μ) with $\lambda = 2$ and $\mu = \frac{k}{3}$ and suppose that G is pseudo-geometric with parameters (s, t, α).

Observe that by Proposition 19 and the fact our assumption that $\mu = \frac{k}{3}$, we have that,

$$\alpha(t + 1) = \mu = \frac{k}{3} = \frac{1}{3}s(t + 1),$$

thus implying that

$$s = 3\alpha. \tag{A.1}$$

By Proposition 19, we have that,

$$2 = \lambda = s - 1 + t(\alpha - 1),$$

or equivalently

$$s + t(\alpha - 1) = 3. \tag{A.2}$$

Since $s = 3\alpha$ and $\alpha \geq 1$ (due to the definition of a partial geometry), then $s \geq 3$. Observe that if s was strictly greater than 3, then Equation A.2 above implies that

$t(\alpha - 1) < 0$, a contradiction as $t, \alpha \geq 1$ (by the definition of a partial geometry). Thus, s must be exactly 3 meaning that (by Equation A.1) $\alpha = \frac{s}{3} = 1$.

As $s = 3$ and $\alpha = 1$, this means that G is pseudo-geometric to generalized quadrangles of order $(3, t)$ for some t.

By Proposition 19 and by using that $s = 3$ and $\alpha = 1$, we have that,

$$v = (s+1)\frac{st+\alpha}{\alpha} = 4(3t+1),$$

$$k = s(t+1) = 3(t+1),$$

$$\lambda = s - 1 + t(\alpha - 1) = 2,$$

$$\mu = \alpha(t+1) = t+1,$$

thus completing the proof.

□

A computer verification quickly verifies that the strongly-regular MQLib instances we considered were indeed strongly-regular graphs with parameters $n = 4(3t+1), k = 3(t+1), \lambda = 2, \mu = t+1$ for some positive integer t meaning that this class of strongly-regular graphs generalizes the collection of strongly-regular instances found in the library.

Next, it is natural to ask, for which values of t does there exists a strongly-regular graph that is pseudo-geometric for a generalized quadrangle of order $(3, t)$? Haemers and Spense [132] characterize all such values of t, however, there are only finitely many of them. In particular, there exists 2, 28, 167, and 1 non-isomorphic pseudo-geometric graphs with parameters $(v, k, \lambda, \mu) = (16, 6, 2, 2)$, $(40, 12, 2, 4)$, $(64, 18, 2, 6)$, and $(112, 30, 2, 10)$ for GQ$(3, 1)$, GQ$(3, 3)$, GQ$(3, 5)$, and GQ$(3, 9)$ respectively. As a result of Proposition 21, Corollary 36, and a non-existence result [133] detailed in the proofs below, there do not exist pseudo-geometric instances with other strongly-regular parameter sets for a GQ$(3, t)$ for some t.

Proposition 21 (Chapter 34 of [134])**.** *Let G be a primitive strongly-regular graph that is pseudo-geometric graph for a PG(s,t,α). Then:*

1. *The integer $\alpha(s+t+1-\alpha)$ divides $st(s+1)(t+1)$.*
2. *The inequalities $(s+1-2\alpha)t \leq (s-1)(s+1-\alpha)^2$ and $(t+1-2\alpha)s \leq (t-1)(t+1-\alpha)^2$ both hold.*

Corollary 36. *If G is a SRG(v,k,λ,μ) and if G is pseudo-geometric for GQ$(3,t)$ for some t, then $t \in \{1,3,5,9\}$.*

Proof. Let G be a SRG(v,k,λ,μ) and suppose G is pseudo-geometric for GQ$(3,t)$ for some t. Since a generalized quadrangle is a partial geometry with parameter $\alpha = 1$, then, as a result of Proposition 21, the integer $s+t$ divides $st(s+1)(t+1)$; furthermore, the inequalities $(s-1)t \leq (s-1)s^2$ and $(t-1)s \leq (t-1)t^2$ both hold; these inequalities simplify to $t \leq s^2$ and $s \leq t^2$. As $s = 3$, we have that $t \leq s^2 = 9$.

Substituting $s = 3$, the divisibility criteria simplifies to $3+t$ divides $12t(t+1)$. When substituting values of t with $1 \leq t \leq 9$, this divisibility criteria is only satisfied by $t \in \{1,3,5,6,9\}$. Haemers [133] shows that the possibility $t = 6$ can be eliminated, i.e., there is no pseudo-geometric graph for GQ$(3,6)$; this corresponds to the non-existence of a SRG$(76,21,2,7)$.

Thus, we have that G is a pseudo-geometric graph for GQ$(3,t)$ for $t \in \{1,3,5,9\}$. □

REFERENCES

[1] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm". In: *arXiv preprint arXiv:1411.4028* (2014).

[2] Howard Karloff. "How Good is the Goemans–Williamson MAX-CUT Algorithm?" In: *SIAM Journal on Computing* 29.1 (1999), pp. 336–350.

[3] Reuben Tate et al. "Bridging Classical and Quantum with SDP Initialized Warm-Starts for QAOA". In: *ACM Transactions on Quantum Computing* (June 2022).

[4] Reuben Tate et al. "Warm-Started QAOA with Custom Mixers Provably Converges and Computationally Beats Goemans-Williamson's Max-Cut at Low Circuit Depths". In: *arXiv preprint arXiv:2112.11354* (2021).

[5] Edsger W Dijkstra. "A note on two problems in connexion with graphs". In: *Edsger Wybe Dijkstra: His Life, Work, and Legacy*. 2022, pp. 287–290.

[6] Richard M. Karp. "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations*. Boston, MA: Springer US, 1972, pp. 85–103.

[7] Julia Robinson. *On the Hamiltonian game (a traveling salesman problem)*. Tech. rep. Rand project air force arlington va, 1949.

[8] Jaroslav Nešetřil, Eva Milková, and Helena Nešetřilová. "Otakar Borůvka on minimum spanning tree problem Translation of both the 1926 papers, comments, history". In: *Discrete mathematics* 233.1-3 (2001), pp. 3–36.

[9] Jack Edmonds. "Paths, trees, and flowers". In: *Canadian Journal of mathematics* 17 (1965), pp. 449–467.

[10] Ronald L Graham. "Bounds for certain multiprocessing anomalies". In: *Bell system technical journal* 45.9 (1966), pp. 1563–1581.

[11] Michel X Goemans and David P Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming". In: *Journal of the ACM (JACM)* 42.6 (1995), pp. 1115–1145.

[12] Subhash Khot. "On the power of unique 2-prover 1-round games". In: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. 2002, pp. 767–775.

[13] Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. "Noise stability of functions with low influences: invariance and optimality". In: *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. IEEE. 2005, pp. 21–30.

[14] Subhash Khot et al. "Optimal inapproximability results for MAX-CUT and other 2-variable CSPs?" In: *SIAM Journal on Computing* 37.1 (2007), pp. 319–357.

[15] Luca Trevisan et al. "Gadgets, approximation, and linear programming". In: *SIAM Journal on Computing* 29.6 (2000), pp. 2074–2097.

[16] Johan Håstad. "Some optimal inapproximability results". In: *Journal of the ACM (JACM)* 48.4 (2001), pp. 798–859.

[17] Boaz Barak and Kunal Marwaha. "Classical algorithms and quantum limitations for maximum cut on high-girth graphs". In: *arXiv preprint arXiv:2106.05900* (2021).

[18] Charles H Bennett et al. "Strengths and weaknesses of quantum computing". In: *SIAM journal on Computing* 26.5 (1997), pp. 1510–1523.

[19] Scott Aaronson. "BQP and the polynomial hierarchy". In: *Proceedings of the forty-second ACM symposium on Theory of computing*. 2010, pp. 141–150.

[20] Charles H Bennett. "Logical reversibility of computation". In: *IBM journal of Research and Development* 17.6 (1973), pp. 525–532.

[21] Paul Benioff. "Quantum mechanical Hamiltonian models of Turing machines". In: *Journal of Statistical Physics* 29 (1982), pp. 515–546.

[22] Ethan Bernstein and Umesh Vazirani. "Quantum complexity theory". In: *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*. 1993, pp. 11–20.

[23] Peter W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. Santa Fe, NM, USA, 1994, pp. 124–134.

[24] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79.

[25] Aram W. Harrow and Ashley Montanaro. "Quantum computational supremacy". In: *Nature* 549 (2017), pp. 203–209.

[26] Francisco Barahona. "On the computational complexity of Ising spin glass models". In: *Journal of Physics A: Mathematical and General* 15.10 (1982), p. 3241.

[27] Andrew Lucas. “Ising formulations of many NP problems”. In: *Frontiers in physics* 2 (2014), p. 5.

[28] Gavin E. Crooks. “Performance of the Quantum Approximate Optimization Algorithm on the Maximum Cut Problem”. In: *arXiv preprint arXiv:1811.08419* (2018).

[29] Jonathan Wurtz and Peter Love. “MaxCut quantum approximate optimization algorithm performance guarantees for p ¿ 1”. In: *Physical Review A* 103.4 (2021), p. 042612.

[30] Glen Bigan Mbeng, Rosario Fazio, and Giuseppe E. Santoro. “Quantum Annealing: a journey through Digitalization, Control, and hybrid Quantum Variational schemes”. In: *arXiv preprint arXiv:1906.08948* (2019).

[31] Sergey Bravyi et al. “Obstacles to Variational Quantum Optimization from Symmetry Protection”. In: *Physical Review Letters* 125 (26 Dec. 2020), p. 260505.

[32] Edward Farhi, David Gamarnik, and Sam Gutmann. “The quantum approximate optimization algorithm needs to see the whole graph: A typical case”. In: *arXiv preprint arXiv:2004.09002* (2020).

[33] Stefan H Sack et al. “Transition states and greedy exploration of the QAOA optimization landscape”. In: *arXiv preprint arXiv:2209.01159* (2022).

[34] Stefan H. Sack and Maksym Serbyn. “Quantum Annealing Initialization of the Quantum Approximate Optimization Algorithm”. In: *arXiv preprint arXiv:2101.05742 [quant-ph]* (2021).

[35] Leo Zhou et al. “Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices”. In: *Physical Review X* 10.2 (2020), p. 021067.

[36] Ruslan Shaydulin et al. “Parameter transfer for quantum approximate optimization of weighted maxcut”. In: *arXiv preprint arXiv:2201.11785* (2022).

[37] Alexey Galda et al. “Transferability of optimal QAOA parameters between random graphs”. In: *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE. 2021, pp. 171–180.

[38] Iain Dunning, Swati Gupta, and John Silberholz. “What Works Best When? A Systematic Evaluation of Heuristics for Max-Cut and QUBO”. In: *INFORMS Journal on Computing* 30.3 (2018).

[39] Dimitris Bertsimas, Angela King, and Rahul Mazumder. “Best subset selection via a modern optimization lens”. In: *The Annals of Statistics* 44.2 (2016), pp. 813–852.

[40] Tobia Marcucci and Russ Tedrake. “Warm start of mixed-integer programs for model predictive control of hybrid systems”. In: *IEEE Transactions on Automatic Control* (2020).

[41] Ted Ralphs and Menal Güzelsoy. “Duality and warm starting in integer programming”. In: *Proceedings of 2006 NSF Design, Service, and Manufacturing Grantees and Research Conference*. 2006.

[42] Daniel J. Egger, Jakub Mareček, and Stefan Woerner. “Warm-starting quantum optimization”. In: *Quantum* 5 (June 2021), p. 479.

[43] Andreas Bärtschi and Stephan Eidenbenz. “Grover Mixers for QAOA: Shifting Complexity from Mixer Design to State Preparation”. In: *arXiv preprint arXiv:2006.00354* (2020).

[44] Jonathan Wurtz and Peter Love. “MaxCut quantum approximate optimization algorithm performance guarantees for p¿ 1”. In: *Physical Review A* 103.4 (2021), p. 042612.

[45] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. “Quantum algorithms for fixed qubit architectures”. In: *arXiv preprint arXiv:1703.06199* (2017).

[46] Ruslan Shaydulin et al. “Classical symmetries and the quantum approximate optimization algorithm”. In: *Quantum Information Processing* 20 (2021), pp. 1–28.

[47] Colin Campbell and Edward Dahl. “QAOA of the Highest Order”. In: *2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*. IEEE. 2022, pp. 141–146.

[48] Kunal Marwaha. “Local classical MAX-CUT algorithm outperforms $p = 2$ QAOA on high-girth regular graphs”. In: *Quantum* 5 (2021), p. 437.

[49] Sergey Bravyi et al. “Hybrid quantum-classical algorithms for approximate graph coloring”. In: *Quantum* 6 (2022), p. 678.

[50] Gopal Chandra Santra et al. “Squeezing and quantum approximate optimization”. In: *arXiv preprint arXiv:2205.10383* (2022).

[51] Daniel Beaulieu and Anh Pham. “Max-cut clustering utilizing warm-start QAOA and IBM runtime”. In: *arXiv preprint arXiv:2108.13464* (2021).

[52] Felix Truger et al. “Selection and optimization of hyperparameters in warm-started quantum optimization for the MaxCut problem”. In: *Electronics* 11.7 (2022), p. 1033.

[53] Phillip C Lotshaw et al. “Scaling quantum approximate optimization on near-term hardware”. In: *Scientific Reports* 12.1 (2022), p. 12388.

[54] Johannes Weidenfeller et al. “Scaling of the quantum approximate optimization algorithm on superconducting qubit based hardware”. In: *Quantum* 6 (2022), p. 870.

[55] Mahabubul Alam, Abdullah Ash-Saki, and Swaroop Ghosh. “Accelerating quantum approximate optimization algorithm using machine learning”. In: *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2020, pp. 686–689.

[56] Gian Giacomo Guerreschi and Anne Y Matsuura. “QAOA for Max-Cut requires hundreds of qubits for quantum speed-up”. In: *Scientific reports* 9.1 (2019), pp. 1–7.

[57] Madelyn Cain et al. “The QAOA gets stuck starting from a good classical string”. In: *arXiv preprint arXiv:2207.05089* (2022).

[58] Stuart Hadfield et al. “From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz”. In: *Algorithms* 12.2 (2019).

[59] Zhihui Wang et al. “XY mixers: Analytical and numerical results for the quantum alternating operator ansatz”. In: *Phys. Rev. A* 101 (1 Jan. 2020), p. 012320.

[60] Linghua Zhu et al. “Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer”. In: *Phys. Rev. Research* 4 (3 July 2022), p. 033029.

[61] Andreas Bärtschi and Stephan Eidenbenz. “Grover mixers for QAOA: Shifting complexity from mixer design to state preparation”. In: *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE. 2020, pp. 72–82.

[62] Zhang Jiang, Eleanor G Rieffel, and Zhihui Wang. “Near-optimal quantum circuit for Grover’s unstructured search using a transverse field”. In: *Physical Review A* 95.6 (2017), p. 062317.

[63] Lov K Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, pp. 212–219.

[64] Scott Aaronson. *Introduction to Quantum Information Science Lecture Notes*. https://www.scottaaronson.com/qclec.pdf. 2021.

[65] Adetokunbo Adedoyin et al. “Quantum algorithm implementations for beginners”. In: *arXiv preprint arXiv:1804.03719* (2018).

[66] Michael A Nielsen and Isaac Chuang. *Quantum Computation and Quantum Information*. American Association of Physics Teachers, 2002.

[67] Bobbi Jo Broxson. “The Kronecker Product”. In: *UNF Graduate Theses and Dissertations* 25 (2006).

[68] C. Delorme and S. Poljak. “Laplacian eigenvalues and the maximum cut problem”. In: *Mathematical Programming* 62 (1993), pp. 557–574.

[69] Svatopluk Poljak and Franz Rendl. “Nonpolyhedral Relaxations of Graph-Bisection Problems”. In: *SIAM Journal on Optimization* 5.3 (1995), pp. 467–487.

[70] Yurii Nesterov and Arkadi Nemirovski. *Interior-point polynomial algorithms in convex programming*. Philadelphia, PA, USA: SIAM, 1994.

[71] Samuel Burer and Renato DC Monteiro. “A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization”. In: *Mathematical Programming* 95.2 (2003), pp. 329–357.

[72] Alexander I. Barvinok. “Problems of distance geometry and convex properties of quadratic maps”. In: *Discrete & Computational Geometry* 13.2 (1995), pp. 189–202.

[73] Gábor Pataki. “On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues”. In: *Mathematics of Operations Research* 23.2 (1998), pp. 339–358.

[74] Nicolas Boumal, Vladislav Voroninski, and Afonso S. Bandeira. “The non-convex Burer–Monteiro approach works on smooth semidefinite programs”. In: *arXiv preprint arXiv:1606.04970* (2018).

[75] Nicolas Boumal, Vladislav Voroninski, and Afonso S Bandeira. “Deterministic Guarantees for Burer-Monteiro Factorizations of Smooth Semidefinite Programs”. In: *Communications on Pure and Applied Mathematics* 73.3 (2020), pp. 581–608.

[76] Yin Zhang, Samuel Burer, and Renato D. C. Monteiro. “Rank-2 relaxation heuristics for Max-Cut and other binary quadratic programs”. In: *SIAM Journal on Optimization* 12 (2 2001), pp. 503–521.

[77] Song Mei et al. “Solving SDPs for synchronization and MaxCut problems via the Grothendieck inequality”. In: *arXiv preprint arXiv:1703.08729* (2017).

[78] Diederik Kingma and Jimmy Lei Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2017).

[79] M.J.D. Powell. “A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation”. In: *Advances in Optimization and Numerical Analysis* 275 (1994), pp. 51–67.

[80] Fuchang Gao and Lixing Han. “Implementing the Nelder-Mead simplex algorithm with adaptive parameters”. In: *Computational Optimization and Applications* 51 (2012), pp. 259–277.

[81] Roger Fletcher. *Practical Methods of Optimization (2nd edition)*. New York, NY, USA: John Wiley and Sons, 1987.

[82] Wim Lavrijsen et al. “Classical Optimizers for Noisy Intermediate-Scale Quantum Devices”. In: *arXiv preprint arXiv:2004.030043* (2021).

[83] Max Wilson et al. “Optimizing quantum heuristics with meta-learning”. In: *Quantum Machine Intelligence* 3.13 (2021).

[84] Fernando G.S.L. Brandao et al. “For Fixed Control Parameters the Quantum Approximate Optimization Algorithm’s Objective Function Value Concentrates for Typical Instances”. In: *arXiv preprint arXiv:1812.04170* (2018).

[85] Brenda S Baker. “Approximation algorithms for NP-complete problems on planar graphs”. In: *Journal of the ACM (JACM)* 41.1 (1994), pp. 153–180.

[86] Magnús Halldórsson and Jaikumar Radhakrishnan. “Greed is good: Approximating independent sets in sparse and bounded-degree graphs”. In: *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*. 1994, pp. 439–448.

[87] Meike Neuwohner. “An improved approximation algorithm for the maximum weight independent set problem in d-claw free graphs”. In: *arXiv preprint arXiv:2106.03545* (2021).

[88] David Zuckerman. “Linear degree extractors and the inapproximability of max clique and chromatic number”. In: *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. 2006, pp. 681–690.

[89] Frank Arute et al. “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574 (2019), pp. 505–510.

[90] Rebekah Herrman et al. “Impact of graph structures for QAOA on MaxCut”. In: *Quantum Information Processing* 20.9 (2021), pp. 1–21.

[91] Andreas Bayerstadler et al. “Industry quantum computing applications”. In: *EPJ Quantum Technology* 8.1 (2021), p. 25.

[92] Edward Farhi and Aram W Harrow. “Quantum supremacy through the quantum approximate optimization algorithm”. In: *arXiv preprint arXiv:1602.07674* (2016).

[93] Andries E. Brouwer et al. “The smallest eigenvalues of Hamming graphs, Johnson graphs and other distance-regular graphs with classical parameters”. In: *Journal of Combinatorial Theory, Series B* 133 (2018), pp. 88–121.

[94] Luca Trevisan. “Max Cut and the smallest eigenvalue”. In: *SIAM Journal of Computing* 41 (6 2012), pp. 1769–1786.

[95] José A. Soto. “Improved Analysis of a Max-Cut Algorithm Based on Spectral Partitioning”. In: *SIAM Journal on Discrete Mathematics* 29 (1 2015), pp. 259–268.

[96] Iain Dunning, Swati Gupta, and John Silberholz. “What works best when? A systematic evaluation of heuristics for Max-Cut and QUBO”. In: *INFORMS Journal on Computing* 30.3 (2018), pp. 608–624.

[97] Jean B Lasserre. “Global optimization with polynomials and the problem of moments”. In: *SIAM Journal on optimization* 11.3 (2001), pp. 796–817.

[98] Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology, 2000.

[99] Zhihui Wang et al. “Quantum approximate optimization algorithm for MaxCut: A fermionic view”. In: *Physical Review A* 97.2 (2018), p. 022304.

[100] JJ Seidel. “Strongly regular graphs”. In: *Surveys in combinatorics* 38 (1979), pp. 157–180.

[101] Raj Chandra Bose. “Strongly regular graphs, partial geometries and partially balanced designs.” In: *Pacific Journal of Mathematics* (1963), pp. 389–419.

[102] WE Brittin. “Valence angle of the tetrahedral carbon atom”. In: *Journal of Chemical Education* 22.3 (1945), p. 145.

[103] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[104] Reuben Tate and Swati Gupta. “CI-QuBe”. In: *GitHub Repository* (2021). https://github.com/swati1729/CI-QuBe.

[105] Nathan Krislock, Jérôme Malick, and Frédéric Roupin. "BiqCrunch: A Semidefinite Branch-and-Bound Method for Solving Binary Quadratic Problems". In: *ACM Transactions on Mathematical Software* 43.4 (Jan. 2017).

[106] William Feller. *An Introduction to Probability Theory and Its Applications*. 3rd. Vol. 2. Hoboken, New Jersey: Wiley, 1971.

[107] Jacek Gondzio and Andreas Grothey. "Solving nonlinear financial planning problems with 109 decision variables on massively parallel architectures". In: *WIT Transactions on Modelling and Simulation* 43 (2006).

[108] Fan RK Chung. *Spectral graph theory*. Vol. 92. American Mathematical Soc., 1997.

[109] William Karush. "Minima of functions of several variables with inequalities as side conditions". MA thesis. University of Chicago, 1939.

[110] Harold W. Kuhn and Albert W. Tucker. "Nonlinear Programming". In: *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*. Berkeley, Calif.: University of California Press, 1951, pp. 481–492.

[111] Aric Hagberg, Pieter Swart, and Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. Tech. rep. Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[112] The Sage Developers. *SageMath, the Sage Mathematics Software System*. https://www.sagemath.org. 2020.

[113] Russell Lyons and Yuval Peres. *Probability on trees and networks*. Vol. 42. Cambridge University Press, 2017.

[114] Leo Zhou et al. "Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices". In: *arXiv preprint arXiv:1812.01041* (2018).

[115] Yin Tat Lee and Swati Padmanabhan. "An $\tilde{O}(m/\epsilon^3.5)$-Cost Algorithm for Semidefinite Programs with Diagonal Constraints". In: *arXiv preprint arXiv:1903.01859* (2019).

[116] Fred Glover, Gary Kochenberger, and Yu Du. "Quantum Bridge Analytics I: A Tutorial on Formulating and Using QUBO Models". In: *arXiv preprint arxiv:1811.11538* (2019).

[117] Bas Lodewijks. "Mapping NP-hard and NP-complete Optimization Problems to Quadratic Unconstrained Binary Optimization Problems". In: *arXiv preprint arxiv:1911.08043* (2020).

[118] Alan J. Laub. *Matrix Analysis for Scientists and Engineers*. Vol. 91. Siam, 2005.

[119] Georg Frobenius. "Ueber Matrizen aus nicht negativen Elementen". In: *Sitzungsberichte der Königlich Preussischen Akademie der Wissenschaften* (1912), pp. 456–477.

[120] A. Kaveh and H. Rahami. "A unified method for eigendecomposition of graph products". In: *Communications in Numerical Methods in Engineering with Biomedical Applications* 21 (7 2005), pp. 377–388.

[121] Simon Špacapan. "Connectivity of Cartesian products of graphs". In: *Applied Mathematics Letters* 21 (7 2008), pp. 682–685.

[122] Abraham Asfaw et al. *Learn Quantum Computation Using Qiskit*. http : / / community.qiskit.org/textbook. 2020.

[123] Matthew P Harrigan et al. "Quantum approximate optimization of non-planar graph problems on a planar superconducting processor". In: *Nature Physics* 17.3 (2021), pp. 332–336.

[124] Sergey Bravyi et al. "Mitigating measurement errors in multiqubit experiments". In: *Phys. Rev. A* 103 (4 Apr. 2021), p. 042605.

[125] George S. Barron and Christopher J. Wood. "Measurement Error Mitigation for Variational Quantum Algorithms". In: *arXiv preprint arXiv:2010.08520* (2020).

[126] Sartaj Sahni and Teofilo Gonzalez. "P-Complete Approximation Problems". In: *J. ACM* 23.3 (July 1976), pp. 555–565.

[127] Stuart Hadfield et al. "From the quantum approximate optimization algorithm to a quantum alternating operator ansatz". In: *Algorithms* 12.2 (2019), p. 34.

[128] Tianyi Hao et al. "Exploiting In-Constraint Energy in Constrained Variational Quantum Optimization". In: *arXiv preprint arXiv:2211.07016* (2022).

[129] Juho Hirvonen et al. "Large cuts with local algorithms on triangle-free graphs". In: *arXiv preprint arXiv:1402.2543* (2014).

[130] Matthew B Hastings. "Classical and quantum bounded depth approximation algorithms". In: *arXiv preprint arXiv:1905.07047* (2019).

[131] Antonia Creswell et al. "Generative adversarial networks: An overview". In: *IEEE signal processing magazine* 35.1 (2018), pp. 53–65.

[132] Willem H Haemers and Edward Spence. “The pseudo-geometric graphs for generalized quadrangles of order (3, t)”. In: *European Journal of Combinatorics* 22.6 (2001), pp. 839–845.

[133] WH Haemers et al. “There exists no (76, 21, 2, 7) strongly regular graph”. In: *Finite Geometry and Combinatorics (F. De Clerck et al., eds.), LMS Lecture Notes Ser* 191 (1993), p. 143.

[134] Charles J Colbourn. *CRC handbook of combinatorial designs*. CRC press, 2010.

VITA

Reuben Tate earned a Bachelor of Science in Computer Science, a Bachelor of Arts in Mathematics, a minor in Physics, and certificate in Database Management from the University of Hawaii at Hilo. Reuben then joined the PhD program in Algorithms, Combinatorics, and Optimization program at the Georgia Institute of Technology under the advisorship of Professor Swati Gupta. Reuben was a member of the (Optimization with Trapped Ion Qubits) OPTIQ group, led by the Georgia Tech Research Institute, which is a DARPA-funded effort for performing optimization on trapped-ion qubits; during his time in the group, Reuben developed a warm-started quantum approach for problems in combinatorial optimization. Reuben was also an intern at the Feynman Quantum Academy program at the USRA-NASA Quantum Artificial Intelligence Laboratory where he worked on other modifications of quantum optimization algorithms under the mentorship of Dr. Stuart Hadfield.

Reuben has also had an interest in teaching: he has tutored for numerous schools and organizations, he TA'ed many courses at both the University of Hawaii at Hilo and Georgia Tech, he developed curriculum and co-instructed a problem-based learning summer program for the Upward Bound organization (who assists low-income and upcoming first-generation students), and most recently, he was a graduate student instructor for Differential Calculus and Applied Combinatorics at Georgia Tech.