Scaling overhead of locality reduction in binary optimization problems

Elisabetta Valiante,[1] Maritza Hernandez,[1] Amin Barzegar,[2] and Helmut G. Katzgraber[3,2,4,*]

[1] *1QB Information Technologies (1QBit), 1285 W Pender St Unit 200, Vancouver, British Columbia V6E 4B1, Canada*
[2] *Microsoft Quantum, Microsoft, Redmond, Washington 98052, USA*
[3] *Professional Services, Amazon Web Services, Seattle, Washington 98101, USA*
[4] *Department of Physics and Astronomy, Texas A&M University, College Station, Texas 77843-4242, USA*

(Dated: December 18, 2020)

Recently, there has been considerable interest in solving optimization problems by mapping these onto a binary representation, sparked mostly by the use of quantum annealing machines. Such binary representation is reminiscent of a discrete physical two-state system, such as the Ising model. As such, physics-inspired techniques—commonly used in fundamental physics studies—are ideally suited to solve optimization problems in a binary format. While binary representations can be often found for paradigmatic optimization problems, these typically result in k-local higher-order unconstrained binary optimization cost functions. In this work, we discuss the effects of locality reduction needed for solvers such as the D-Wave quantum annealer, the Fujitsu Digital Annealer, or the Toshiba Simulated Bifurcation Machine that can only accommodate 2-local (quadratic) cost functions. General locality reduction approaches require the introduction of ancillary variables which cause an overhead over the native problem. Using a parallel tempering Monte Carlo solver on Azure Quantum, as well as k-local binary problems with planted solutions, we show that post reduction to a corresponding 2-local representation the problems become considerably harder to solve. We further quantify the increase in computational hardness introduced by the reduction algorithm by measuring the variation of number of variables, statistics of the coefficient values, and the entropic family size. Our results demonstrate the importance of avoiding locality reduction when solving optimization problems.

PACS numbers: 75.50.Lk, 75.40.Mg, 05.50.+q, 64.60.-i

I. INTRODUCTION

In recent years there have been many technological and algorithmic advances when solving optimization problems, in particular, in an industrial setting. Sparked by the work of D-Wave Systems Inc., a whole new field of optimization based on physical processes has emerged. Specifically, the development of hardware quantum annealers has stimulated new ways of analyzing problems previously thought to be intractable.

Despite these advances, the use of quantum annealers for large-scale industry applications remains limited if not paired with classical algorithms on CMOS hardware. Being able to tackle an application requires first having a Boolean representation of the problem. To this *mapping* step, in most cases a variable overhead is associated, which typically makes a problem harder to solve. However, due to hardware limitations, only 2-local (quadratic unconstrained binary optimization, or QUBO) cost functions can be tackled with quantum annealing hardware. This means that a higher-order binary polynomial unconstrained optimization problem requires a *locality reduction* which can result in a sizable variable overhead. In this work, we focus on the *locality reduction*, and do not discuss additional overheads due to the *embedding* of a binary problem onto the hardwired sparse quasi-two-dimensional topology of annealing hardware or the effects of analog noise.

The hardware limitations play an important role when solving problems naturally formulated as a Hamiltonian with k-local interactions with $k > 2$. There are various optimization problems both in fundamental physics and applications that are natively k-local. Examples in physics are computing the partition function of a four-dimensional pure lattice gauge theory [1, 2], measuring the fault-tolerance in topological colour codes [3], and solving k-SAT problems with $k > 2$. Examples of practical applications are circuit fault diagnosis [4, 5], molecular similarity measurement [6], molecular conformational sampling [7], and traffic light synchronization [8].

Quadratization techniques are algorithms used to reduce a higher-degree multilinear polynomial into a quadratic one [9]. The reduction process can introduce two different types of overheads. First, the quadratization itself can result in a large overhead before any solver is applied to the problem of interest. Second, quadratization requires the introduction of additional variables and terms. As such, the complexity of the problem increases and, in turn, so does the time to solution. Finally, the quadratization process might also introduce features (e.g., broader coupler distributions) that can affect the intrinsic difficulty of the problem. An extensive comparison between several quadratization methods, highlighting the pros and cons of each method, has been compiled by Dattani in Ref. [10].

In this paper, we use Microsoft Quantum's k-local solvers based on simulated annealing and parallel tempering Monte Carlo to measure the time overhead introduced by the quadratization process to reduce an optimization problem with k-local interaction to its 2-local counterpart. We study unconstrained problems with a binary representation and planted solutions and disregard the time it takes for the quadratization algorithm to run. Our results demonstrate that the lo-

[*] The work of H. G. K. was performed before joining Amazon Web Services.

cality reduction introduces a large overhead when solving the problems. Employing a commonly used proxy metric, we demonstrate that, on average, optimization problems become much harder to solve when the locality is reduced. Könz et al. (in prep.) study the embedding overhead when using sparse hardware topologies. Both complementary studies highlight the importance of developing new optimization machines and techniques that can handle k-local cost functions natively on complete graphs.

This paper has the following structure: in Sec. II, we describe the benchmark problems used for the experiment; in Sec. III, we present the setup of the experiment and the metrics used to compare performance; in Sec. IV and Sec. V, we discuss and analyze the results of the experiment; in Sec. VI, we present our conclusions.

II. BENCHMARK PROBLEMS

In order to study the scaling overhead caused by reducing a k-local problem to a quadratic (2-local) formulation, we first generate Ising problems for $k = 3$ and $k = 4$. The k-local instances have been generated using the `Chook` package, which is publicly available on GitHub; see [11]. Using this package we are able to construct planted-solution instances, thus ensuring that the ground state and corresponding energy are known a priori. The construction of k-local problems is done by combining problems of lower-order, that is, $k \leq 2$.

In this study, 3-local instances have been generated by combining a tile planting problem with Ising spins coupled to a bimodal random field, while for 4-local instances, the problems have been generated by combining two tile planting problems. The tile planting problems are defined by four subproblem classes that correspond to unit cycles (plaquettes) with different levels of frustration. A subproblem is constructed by assigning to the couplers values equal to -1, 1, or 2, according to the class to which the subproblem belongs. The class is assigned with a certain probability, and each instance class is defined by three probability parameters. We set these parameters to the default values used in `Chook` [11]. For each locality considered, we generate instances with problem sizes N (number of variables) between 16 and 400.

The k-local instances are then reduced to their quadratic form using an iterative reduction-by-substitution algorithm [12, 13]. Here, the product of two variables is substituted by a new auxiliary variable and a penalty term is added to enforce equality in the ground state. This process is repeated until the final function becomes quadratic. Tuning the value of the energy penalty term is extremely important: a small value could return a 2-local problem not having the same optimum as the original higher-order problem. Therefore, a large value is commonly used in various implementations of this algorithm. The penalty value can grow to be very large if high values of k are being reduced. This can pose issues when attempting to solve problems on current analog quantum annealing hardware, because large coefficients amplify the effects of the analog noise. The reduction of k-local problems in this work is done via the `Hobo2Qubo` function available through 1QBit's 1Qloud Platform [14], which uses a tight bound for the penalty coefficient and sets it independently for each reduced term. The computational time required to reduce a single instance is negligible with respect to the time required by the solver. Moreover, the reduction from k-local to 2-local is known to scale in polynomial time [13], that is, it should be irrelevant in the thermodynamic limit.

The sizes and densities of the 2-local instances obtained after reduction from 3-local and 4-local instances are shown in Tables I and II, respectively. The number of variables increases considerably when reducing locality from k-local to 2-local, as can be expected for a reduction-by-substitution algorithm.

The density of a k-local instance ρ is calculated as

$$\rho = \frac{1}{k-1} \sum_{k_{\mathrm{t}}=2}^{k} \frac{(N-k_{\mathrm{t}})!k_{\mathrm{t}}!}{N!} E_{k_{\mathrm{t}}}\,, \tag{1}$$

where k is the locality of the polynomial. The sum is taken over all the degrees in the polynomial running from $k_{\mathrm{t}} = 2$ to $k_{\mathrm{t}} = k$, $E_{k_{\mathrm{t}}}$ is the number of individual terms with degree k_{t}, and N is the number of variables in the polynomial. For 2-local instances, this expression is reduced to the common graph density expression. Notice that, for all problems, the densities decrease slightly post locality reduction.

TABLE I: Reduction of 3-local problems to 2-local problems. Densities for each instance are calculated as per Eq. (1). The mean values (denoted by an overbar) are calculated over the 30 instances that have been generated. The number of variables of the reduced problems increases by a factor ~ 3.

3-local		2-local reduction	
N	$\bar{\rho}$	$\bar{N}$	$\bar{\rho}$
16	0.568 ± 0.020	46.73 ± 0.573	0.329 ± 0.009
64	0.398 ± 0.014	192.0	0.295 ± 0.003
144	0.364 ± 0.008	432.0	0.294 ± 0.002
256	0.352 ± 0.007	768.0	0.294 ± 0.002
400	0.345 ± 0.005	1200.0	0.294 ± 0.001

TABLE II: Reduction of 4-local problems to 2-local problems. Densities for each instance are calculated as per Eq. 1. The mean values (denoted by an overbar) are calculated over the 30 instances that have been generated. The number of variables of the reduced problems increases by a factor ~ 6.

4-local		2-local reduction	
N	$\bar{\rho}$	$\bar{N}$	$\bar{\rho}$
16	0.615 ± 0.023	76.5 ± 2.0	0.301 ± 0.013
64	0.295 ± 0.017	448.5 ± 4.0	0.167 ± 0.004
144	0.210 ± 0.008	887.6 ± 2.2	0.176 ± 0.002
256	0.179 ± 0.007	1501.3 ± 3.6	0.173 ± 0.002
400	0.163 ± 0.004	2248.3 ± 3.6	0.174 ± 0.001

III. EXPERIMENT SETUP

The simulations are performed with Azure Quantum's solvers, which can handle k-local terms natively. There are two variants of the solvers. The parameter-free version requires the user to enter only a timeout and automatically optimizes the parameters to find solutions to binary cost functions to high probabilities. The standard solvers require parameter optimization to obtain the optimal scaling.

A. Setup

For the experiments, we use the parameter-free `ParallelTempering` (v1.0) solver. The best values for temperatures, number of sweeps, and number of replicas are calculated internally and are customized for each submitted problem individually. The only parameter to set is `timeout`, which is the time spent in the core solver loop (in seconds). It is worth specifying that `timeout` does not include the time spent by the solver to calculate the parameters that are used during the annealing process. The total time the solver needs to solve the problem is referred to as `runtime`. The advantage of using a parameter-free solver is that no tuning experiment is necessary. The disadvantage is that the `runtime` we measure includes both the time to calculate the parameters and the time to solve the problem. At the time of running the experiment, the parameters calculated by the solver are not returned to the user in the current implementation. As such, we cannot list them in this work.

The benchmark experiment consists of solving 30 random instances for each system size and locality, as well as their respective 2-local reduction (see Tables I and II for details). For each of these instances, we perform 30 runs to gather statistics. We set `timeout` $= 100$. In cases when 100 is not enough time to find the ground-state energy, we increase `timeout` to 500.

B. Metrics

The primary objective of our benchmark experiment is to quantify how the computational effort in solving a problem scales as the size of the problem input increases. The common approach is to measure the time to solution (TTS). We calculate the TTS following the approach defined in Ref. [15]:

$$\mathrm{TTS} = \tau \mathrm{R}_{99}, \tag{2}$$

where R_{99} is the number of runs required to find the ground-state energy with a probability of 99% and τ is the time it takes to run the algorithm once (i.e., the solver output `runtime`).

Measuring the TTS requires the algorithm to find the ground-state energy of each problem for at least 50% of the successive runs performed. When it is not possible to measure the TTS, because the ground-state energy cannot be determined sufficiently often, we measure other performance metrics, such as the fraction of solved problems and the residual energies—both defined below.

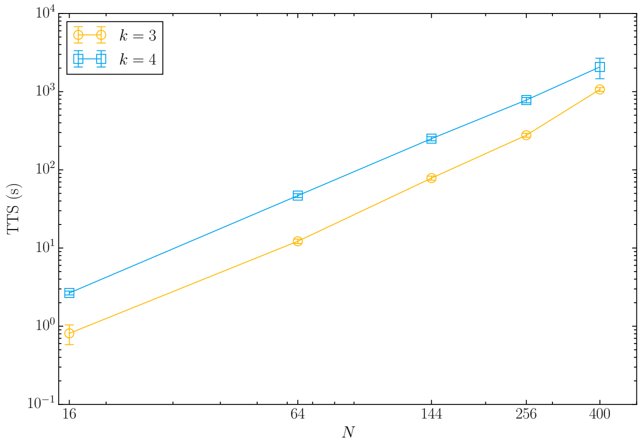

FIG. 1: TTS mean value and standard deviation for k-local problems with $k = 3$ and $k = 4$ using the parallel tempering solver.

The fraction of solved problems is defined as the fraction of runs for which the ground-state energy is found by the solver divided by the total number of experiments. We have performed a total of 900 runs for each problem size and locality. The energy is calculated for each problem and each run in the following way:

$$R = \frac{E_{\mathrm{GS}} - E_{\mathrm{best}}}{E_{\mathrm{GS}}}, \tag{3}$$

where E_{GS} is the known planted ground-state energy of the problem and E_{best} is the best energy found by the algorithm. The values reported here are obtained by resampling the distribution of residuals over all problems and runs.

IV. RESULTS

Figure 1 shows the TTS for planted 3- and 4-local problems with a number of variables N ranging from 16 to 400 using the parallel tempering algorithm. Both problem types show a similar scaling. We have fit an exponential function of the form $\mathrm{TTS} = 10^{\alpha+\beta N}$. The results of the fit and the estimated scaling exponent β are:

$$\begin{aligned} \beta &= 0.00737(10) \qquad (k=3) \\ \beta &= 0.00671(42) \qquad (k=4) \end{aligned}$$

The fraction of solved runs is 100% for all sizes of both 3- and 4-local problems. However, it is not possible to calculate the TTS for the 2-local reductions of either the 3- or 4-local problems. Figure 2 shows the fraction of solved problems (left panel) and the residual energies (right panel). The 2-local problems derived from the 4-local instances seem to be computationally harder to solve than the ones generated from the 3-local problems. We surmise that the higher the locality, the harder it would be to solve the 2-local reductions. The benchmark experiment has been performed with two different values of the parameter `timeout`. However, increasing the timeout does not improve the quality of the results.

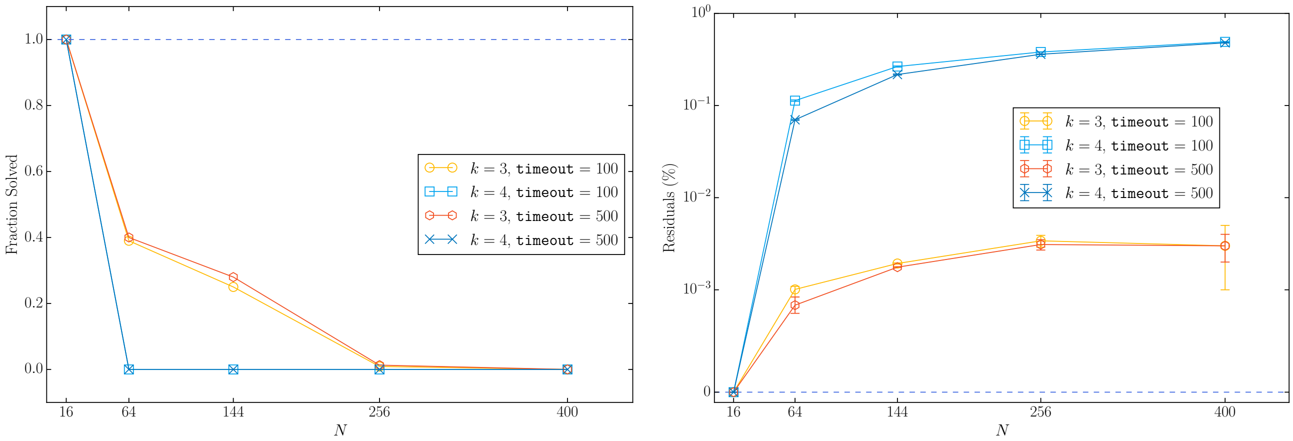

FIG. 2: Fraction solved (left) and residuals (right) of 2-local problems obtained by reducing k-local instances with $k = 3$ and $k = 4$. The dashed line represents the reference for the ideal cases. The benchmark experiment has been performed with two different values of the parameter `timeout`. The data show that solving the 2-local versions of the problems is extremely difficult. In fact, we were unable to do a scaling analysis as the majority of the problems could not be solved.

V. DISCUSSION

The computational hardness of the 3- and 4-local instances is set in the planting tool `Chook` by a careful choice of the couplers from different disorder distributions with varying levels of frustration. The reduction to 2-local interactions in the Hamiltonian requires the introduction of auxiliary variables and penalty terms. The latter, in particular, change the frustration levels, and thereby the hardness of the problems. Tables I and II show the increase in the number of variables when reducing the problems to their 2-local versions. We observe an increase of a factor of approximately 3 for the 3-local problems, which increases to a factor ~ 6 when reducing the 4-local problems. Higher-order Hamiltonians will naturally require an even larger overhead.

Figure 3 compares the coupler distributions for the 3- and 4-local problems of different system sizes with their corresponding 2-local reductions. The histograms show that, while the distributions of the 3- and 4-local problem are quite similar (note that the x-axis in the plots on the left and the right sides of the figure have a different scale), the distributions of their 2-local reductions are significantly wider, in particular when the reduction occurs from a higher degree of the polynomial. A more quantitative analysis of this effect is shown in Figure 4. We have calculated the standard deviation and the kurtosis of the the coupler distributions. While the former increases by a factor of approximately 10, the latter reduces by approximately a factor of 5 when reducing the problems from k-local to 2-local. Having a large dynamic range in the coupler distributions of the reduced problems typically makes these harder to solve with physics-based solvers.

To corroborate the aforementioned observation that the problems become harder when their locality is reduced and, in turn, the coupler distributions have greater variance, we use population annealing Monte Carlo (PAMC) [16–21] to measure the *entropic family size* $\rho_{\rm s}$. Similar to simulated annealing (SA) [22], population annealing is a sequential Markov chain Monte Carlo (MCMC) algorithm in which a population of "replicas" is slowly annealed toward a target low temperature. At each temperature, the population is reconfigured via a resampling process during which some replicas are multiplied or eliminated to achieve an equilibrium Gibbs distribution of energies. In a well-thermalized PAMC simulation, a sufficient number of the original replica families must survive. This can be quantified by the *family entropy*, $S_{\rm f}$:

$$S_{\rm f} = -\sum_{i}^{R} \mathfrak{n}_i \log \mathfrak{n}_i \, , \tag{4}$$

where $\mathfrak{n}_i$ is the fraction of the replicas in the i-th family and R is the total population size. The average family size in thermal equilibrium can then be obtained from

$$\rho_{\rm s} = \lim_{R\to\infty} R/e^{S_{\rm f}}. \tag{5}$$

Note that $\rho_{\rm s}$, by definition, is an intensive quantity, and therefore independent of the population size R in the thermodynamic limit. In practice, $\rho_{\rm s}$ converges to its true value at a large but finite population. For all measurements of $\rho_{\rm s}$, we ensured that such convergence was achieved unless the simulation timed out. Figure 5 shows ρ_s averaged over all instances generated for each system size. The 2-local reduction critically increases the hardness of the problems, especially for large system sizes. In particular, measuring ρ_s for the reduced version of 4-local problems was possible only for sizes $N = 16$ and $N = 64$. The problems were so hard that the simulation converged only partially for $N = 144$, and did not converge at all during the allocated time for larger problem sizes.

Our results demonstrate the advantage of solving the optimization problems in their original k-local formulation, and we expect this result to be independent of the choice of

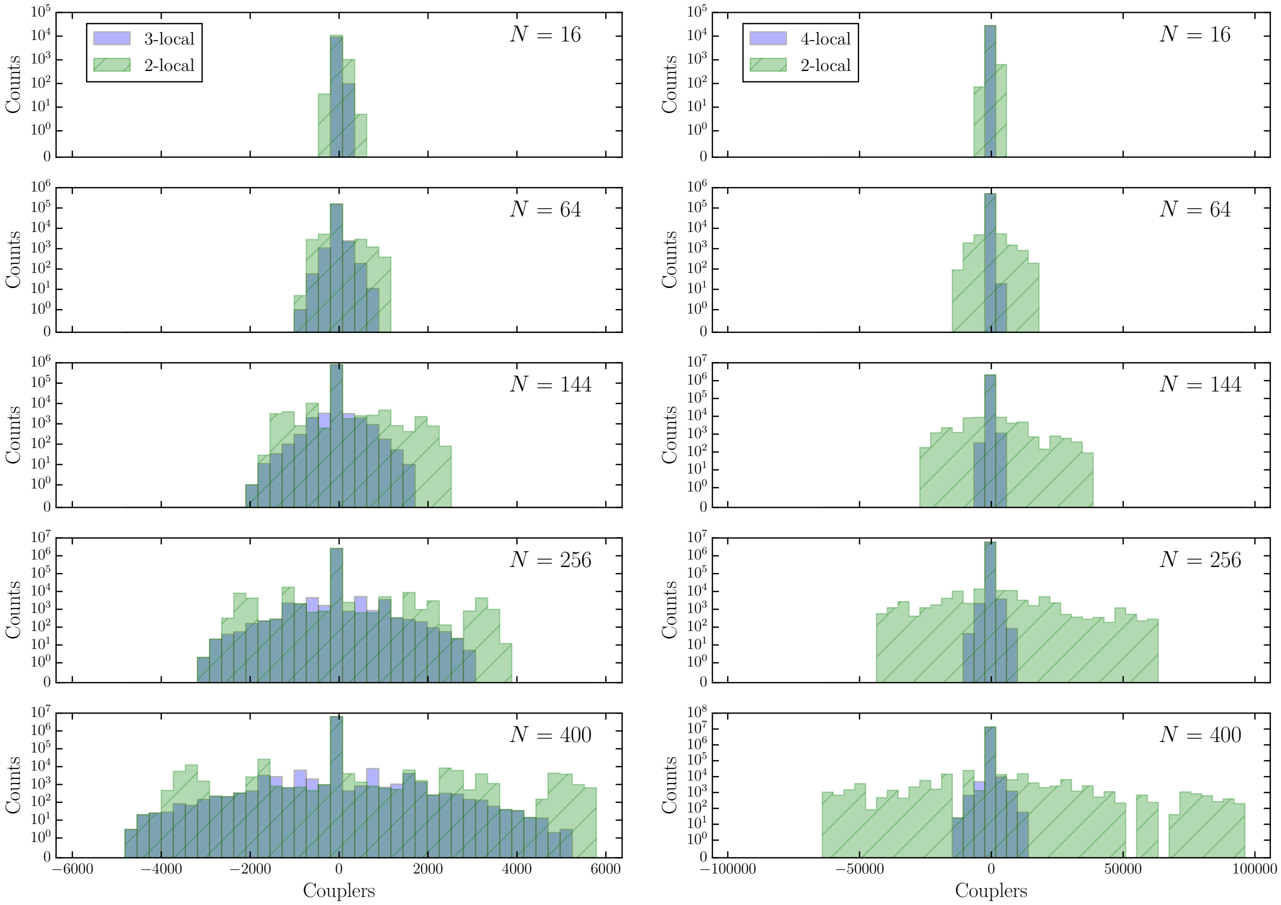

FIG. 3: Coupler distributions of k-local problems with $k = 3$ (left panel) and $k = 4$ (right panel) for different system sizes N, and their corresponding 2-local reductions. In the 3-local case the distributions are similar, however weight is redistributed to the tails. In the 4-local case there is a sizable increase in the width of the distributions post locality reduction.

solver. Reference [5] shows that a simulated quantum annealing (SQA) algorithm has no advantage in solving a k-local formulation of a problem, instead of its 2-local reduction, for $N < 20$. In particular, they claim that the tunnelling effect would pass across the large energy barriers introduced by the reduction. Nevertheless, we would expect such barriers to become wider as the size of the problem increases [23], until no finite-range tunnelling can be beneficial during the optimization.

VI. CONCLUSIONS

We have generated problems with planted solutions having k-local interactions and reduced them to their corresponding 2-local versions, more amenable to current physics-inspired optimization tools than the original ones. The reduction has been performed using a customized version of a classic and extensively adopted quadratization algorithm. The computational time required by the reduction algorithm is known to scale polynomially with the size of the input and thus does not affect the overall exponential scaling found in current physics-

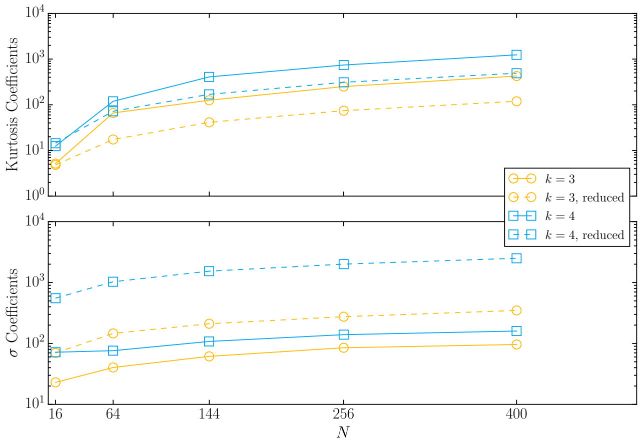

FIG. 4: Kurtosis and standard deviation calculated from the coupler distributions of k-local problems with $k = 3$ and $k = 4$, and their correspondent 2-local reductions. While the kurtosis decreases, the standard deviation of the distributions increases noticeably, thus making the problems harder to solve. Both panels have the same horizontal axis.

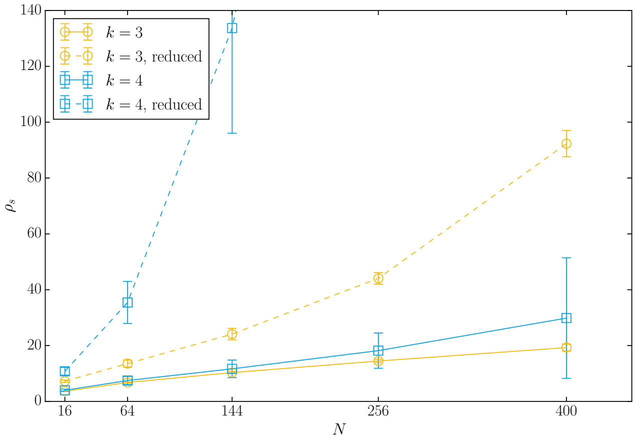

FIG. 5: Entropic family size, ρ_s, calculated using population annealing Monte Carlo for k-local problems with $k = 3$ and $k = 4$, and their corresponding 2-local reductions. The family size of the reduced version of $k = 4$ problems with $N = 144$ converged only partially, while for larger sizes the value did not converge during the allocated timeout. In all cases, a reduction in locality makes the problems computationally harder to solve.

inspired optimization methods. Using Azure Quantum's implementation of the `ParallelTempering` parameter-free algorithm, designed to handle problems of any locality, we have attempted to find optima for the native 3- and 4-local problems, as well as their 2-local reductions. All k-local problems with $k = 3$ and $k = 4$ have been solved to optimality during the allocated 100-second timeout. The TTS for 4-local problems is approximately 5 times larger than for the 3-local ones. In contrast, even after increasing the timeout to 500 seconds, the 2-local reductions could not be solved. It is common practice to apply locality reduction in order to accommodate higher-order polynomial unconstrained optimization problems to run on optimizers that natively handle only quadratic problems. Nevertheless, our results show that doing so should be, ideally, avoided. As such, investing into creating hardware and/or software to tackle higher-order problems should be prioritized.

Acknowledgments

We thank Marko Bucyk for his careful editing and reviewing of the manuscript. H. G. K. would like to thank David Poulin for inspiring discussions and dedicate this manuscript to him.

[1] G. D. las Cuevas, W. Dür, M. V. den Nest, and H. J. Briegel, *Completeness of classical spin models and universal quantum computation*, J. Stat. Mech. **2009**, P07001 (2009).

[2] G. D. las Cuevas, W. Dür, H. J. Briegel, and M. A. Martin-Delgado, *Mapping all classical spin models to a lattice gauge theory*, New J. Phys. **12**, 043014 (2010).

[3] R. S. Andrist, H. G. Katzgraber, H. Bombin, and M. A. Martin-Delgado, *Tricolored lattice gauge theory with randomness: fault tolerance in topological color codes*, New J. Phys. **13**, 083006 (2011).

[4] A. Feldman, G. Provan, and A. Van Gemund, *Approximate Model-Based Diagnosis Using Greedy Stochastic Search*, Journal of Artificial Intelligence Research **38**, 371–413 (2010).

[5] A. Perdomo-Ortiz, A. Feldman, A. Ozaeta, S. V. Isakov, Z. Zhu, B. O'Gorman, H. G. Katzgraber, A. Diedrich, H. Neven, J. de Kleer, et al., *Readiness of Quantum Optimization Machines for Industrial Applications*, Phys. Rev. Applied **12**, 014004 (2019).

[6] M. Hernandez, A. Zaribafiyan, M. Aramon, and M. Naghibi, *A Novel Graph-Based Approach for Determining Molecular Similarity* (2016), 1601.06693.

[7] D. J. J. Marchand, M. Noori, A. Roberts, G. Rosenberg, B. Woods, U. Yildiz, M. Coons, D. Devore, and P. Margl, *A Variable Neighbourhood Descent Heuristic for Conformational Search Using a Quantum Annealer*, Sci. Rep. **9**, 13708 (2019).

[8] Microsoft Quantum, *Jij and Toyota Tsusho: reducing carbon emissions with Azure Quantum*, `https://cloudblogs.microsoft.com/quantum/2020/08/04/jij-toyota-azure-quantum-reducing-carbon-emissions/`.

[9] E. Boros and A. Gruber, *On Quadratization of Pseudo-Boolean Functions* (2014), 1404.6538.

[10] N. Dattani, *Quadratization in discrete optimization and quantum mechanics* (2019), 1901.04405.

[11] D. Perera, I. Akpabio, F. Hamze, S. Mandrà, N. Rose, M. Aramon, and H. G. Katzgraber, *Chook – A comprehensive suite for generating binary optimization problems with planted solutions* (2020), 2005.14344.

[12] I. Rosenberg, *Reduction of bivalent maximization to the quadratic case*, Cahiers du Centre d'Etudes de Recherche Operationnelle **17**, 71 (1975).

[13] E. Boros and P. L. Hammer, *Pseudo-Boolean optimization*, Discrete Applied Mathematics **123**, 155 (2002).

[14] 1QBit, *1Qloud Documentation: Convert HOBO to QUBO*, `https://portal.1qbit-prod.com/docs/task/convert-hobo-to-a-qubo`, accessed: 2020-12-09.

[15] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, *Physics-Inspired Optimization for Quadratic Unconstrained Problems Using a Digital Annealer*, Frontiers in Physics **7**, 48 (2019), 1806.08815.

[16] K. Hukushima and Y. Iba, in *The Monte Carlo method in the physical sciences: celebrating the 50th anniversary of the Metropolis algorithm*, edited by J. E. Gubernatis (AIP, Los Alamos, New Mexico (USA), 2003), vol. 690, p. 200.

[17] J. Machta, *Population annealing with weighted averages: A Monte Carlo method for rough free-energy landscapes*, Phys. Rev. E **82**, 026704 (2010).

[18] J. Machta and R. Ellis, *Monte Carlo Methods for Rough Free Energy Landscapes: Population Annealing and Parallel Tempering*, J. Stat. Phys. **144**, 541 (2011).

[19] W. Wang, J. Machta, and H. G. Katzgraber, *Population annealing: Theory and application in spin glasses*, Phys. Rev. E **92**,

063307 (2015).

[20] C. Amey and J. Machta, *Analysis and optimization of population annealing*, Phys. Rev. E **97**, 033301 (2018).

[21] A. Barzegar, C. Pattison, W. Wang, and H. G. Katzgraber, *Optimization of population annealing Monte Carlo for large-scale spin-glass simulations*, Phys. Rev. E **98**, 053308 (2018).

[22] S. Kirkpatrick, C. D. Gelatt, Jr., and M. Vecchi, *Optimization by Simulated Annealing*, Science **220**, 671 (1983).

[23] S. Mandrà, Z. Zhu, W. Wang, A. Perdomo-Ortiz, and H. G. Katzgraber, *Strengths and weaknesses of weak-strong cluster problems: A detailed overview of state-of-the-art classical heuristics versus quantum approaches*, Phys. Rev. A **94**, 022337 (2016).