Quadratization in Pseudo-Boolean Optimization and Adiabatic Quantum Computing

Richard Tanburn[1, *]

[1]*Oxford University, Mathematical Institute, OX2 6GG, Oxford, UK.*

Nikesh S. Dattani[2,3†]

[2]*Oxford University, Physical and Theoretical Chemistry Laboratory, OX1 3BW, Oxford, UK,*

There is a ridiculous amount of literature in so many different places, in communities that do not talk to each other, and do not even understand each other's language.

I. INTRODUCTION

Every computation can be done by minimizing a Hamiltonian of either one of the following forms:

$$H = \sum_i^n (\alpha_i z_i + \beta_i x_i) + \sum_{ij}^n (\alpha_{ij} z_i z_j + \beta_{ij} x_i x_j), \tag{1}$$

$$H = \sum_i^n (\alpha_i z_i + \beta_i x_i) + \sum_{ij}^n (\alpha_{ij} z_i z_j + \beta_{ij} x_i z_j), \tag{2}$$

where the z and x variables denote the Pauli matrices:

$$z \equiv \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \; x \equiv \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \tag{3}$$

the α and β coefficients are complex numbers; and the Appendix will teach any unfamiliar readers how to interpret this notation.

We know that every computation can be done this way because the minimization can be done by adiabatic quantum computation (AQC), and it has been proven [1, 2] that AQC can simulate any circuit-based quantum computation with overhead that grows at

* richard.tanburn@hertford.ox.ac.uk

† dattani.nike@gmail.com

most polynomially with the size of the problem, and we know that circuit-based quantum computation can simulate any classical computation with at most polynomial overhead. If we remove all the terms in Eq. 1 or Eq. 2 that have x operators, we get an Ising Hamiltonian that is quadratic:

$$H = \sum_{i}^{n} a_i z_i + \sum_{ij}^{n} a_{ij} z_i z_j, \tag{4}$$

and we no longer know of any proof that ***any*** computation can be done by minimizing such a Hamiltonian with only polynomial overhead compared to circuit-based quantum computation or even classical computation, but plenty of very interesting problems can still be formulated as the minimization or approximate minimization of a quadratic Ising Hamiltonian, such as neural network training [3], computerized image denoising [4–7], integer factorization [8], and Ramsey number determination [9, 10], just to name a few. However unlike Eq. 2, plenty of devices that can minimize Ising Hamiltonians already exist, and with further restrictions on the a coefficients, D-Wave has made devices with $n = 2000$, and this number has roughly doubled every 14 months since the first 1-qubit demonstration in 2003, a phenomenon analogous to Moore's law of classical computer growth, which has become known as Rose's Law.

Furthermore, ***no classical computer has ever been able to minimize certain quadratic Ising Hamiltonians faster than the D-Wave machines to date*** [11, 12], so being able to turn computations into the form of Eq. 4 appears to be promising in terms of demonstrating the first useful application of quantum supremacy (which is the term used for quantum devices outperforming classical computation devices). Moreover, for problems like neural networks and computerized image denoising, the best known classical computation algorithms also work by turning the problems into minimizations of Ising Hamiltonians, so even if you are happy with continuing to only use classical computer and have no interest in using quantum annealing devices, turning a problem into a quadratic Ising Hamiltonian minimization problem can be very powerful for some computations.

For neural networks, image denoising, integer factorization, and Ramsey number determination, it is rather easy to turn the problem into the form:

$$H = \sum_{i}^{n} \alpha_i z_i + \sum_{ij}^{n} \alpha_{ij} z_i z_j + \sum_{ijk}^{n} \alpha_{ijk} z_i z_j z_k + \sum_{ijkl}^{n} \alpha_{ijkl} z_i z_j z_k z_l \ldots, \tag{5}$$

but then ***quadratizing*** Eq. 5 into the quadratic Ising form often requires the introduction of auxiliary variables, and since minimizing Eq. 5 often has cost $\mathcal{O}(2^n)$, it is often extremely desirable to do the quadratization with ***as few auxiliary variables as possible***. Apart from quadratizing with minimum number of variables, it is also often desirable to reduce the number of non-zero α coefficients, to keep the non-zero α coefficients bounded or with specific values, to reduce the number of positive (non-submodular) α coefficients, to reduce the spectral ratio of the Hamiltonian ('maximum energy minus minimum energy' divided by 'energy of the second minimum minus the energy of the global minimum'), and to reduce the number of energy levels close to the global minimum. Adjusting the energy landscape in order to accomplish all of these things apart from reducing the number of auxiliary variables needed has been termed Energy Landscape Manipulation (ELM) which is an entirely separate topic, sometimes even more important than quadratizing with as few auxiliary variables as possible, nonetheless this Review simply focuses on methods for the latter aim.

It has been shown in [7] that a general function of the form Eq. 5 with degree k, can be quadratized using $O(n^{k/2})$ auxiliary variables. However, it is often possible to quadratize with far fewer auxiliary variables (for example when the auxiliary variables for quadratizing one term are reused to quadratize other terms that have some of the same variables in them) or to quadratize without any auxiliaries at all (ex. if a Groebner basis can be found with all basis functions being quadratic, or if some symmetries in the Hamiltonian can be found that make it possible to quadratize without auxiliaries).

Part I

Hamiltonians with only z terms (pseudo-Boolean functions)

II. METHODS THAT INTRODUCE ZERO AUXILIARY VARIABLES

A. ELC Reduction

Summary

An Excludable Local Configuration (ELC) is a partial assignment of variables that cannot possibly give the global minimum. We can then add on a term that corresponds to the energy of this ELC. In practice we can eliminate all monomials with a variable in which a variable is set to 0, and reduce any variable set to 1. Given a general Hamiltonian we can try to find ELCs by enumerating solutions of a small subset of the problem [6].

Cost

- 0 auxiliary variables needed.
- $\mathcal{O}\left(2^m\binom{n}{m}\right)$ operations required to enumerate all possible cases for all possible subsets of size $m \leq n$ variables.
- Approximate methods exist which have been shown to be much faster and give good approximations to the global minimum [6].

Pros

- No auxiliary variables needed.

Cons

- ELCs need to be found by 'brute force', which scales exponentially with respect to m.
- ELCs do not always exist.

Example

Consider the Hamiltonian:

$$H_{3\text{–local}} = z_1z_2 + z_2z_3 + z_3z_4 - 4z_1z_2z_3. \tag{6}$$

If $z_1z_2z_3 = 0$, no assignment of our variables will we be able to reach a lower energy than if $z_1z_2z_3 = 1$. Hence $(z_1, z_2, z_3) = (1, 0, 0)$ is a Local Excludable Configuration (ELC) and we can form the polynomial:

$$H_{2\text{–local}} = H_{3\text{–local}} + 4z_1(1 - z_2)(1 - z_3) = z_1z_2 + z_2z_3 + z_3z_4 + 4z_1 - 4z_1z_2 - 4z_1z_3. \tag{7}$$

In both cases, the only global minima occur when $z_1z_2z_3 = 1$.

Bibliography

- Original paper and application to computerized image denoising: [6].

B. Deduction Reduction (Deduc-reduc)

Summary

We look for *deductions* (e.g. $z_1z_2 = 0$) that hold at the global minimum. These can be found by *a priori* knowledge of the given problem, or by enumerating solutions of a small subset of the variables. We can then substitute high-order terms using the low-order terms of the deduction, and add on a penalty term to preserve the ground states [13].

Cost

- 0 auxiliary variables needed.
- The computational cost of the search for deductions is difficult to estimate. The approximate worst-case complexity is $\mathcal{O}(n^{d+1}2^m)$ where m is the number of variables in a 'small' problem, n is the total number of qubits and d is the maximum degree of deductions we are searching for. We suggest $10 \leq m \leq 20$, so that a small problem involves checking roughly $1,000$ to $1,000,000$ states, and $d = 2$. See the appendix for more details.

Pros

- No auxiliary variables needed.

Cons

- When deductions cannot be determined naturally (as in the Ramsey number determination problem, see Example XII), deductions need to be found by 'brute force', which scales exponentially with respect to m. For highly connected systems (systems with a large number of non-zero a coefficients), the value of m required to find even one deduction can be prohibitively large.

Example

Consider the Hamiltonian:

$$H_{4\text{–local}} = z_1 z_2(10 + z_3 + z_3 z_4) + z_1(z_3 - 3) + z_2(z_3 - 2z_3 - z_4 - 1) + F(x_3, x_4, x_5, \ldots, x_N) \quad (8)$$

where F is any polynomial in z_i for $3 \leq i$.

Since the 10 coefficient of $z_1 z_2$ is greater than the sum of all of the other coefficients involving z_1 or z_2, it must be the case that $z_1 z_2 = 0$. Specifically, for the 4 assignments of (z_3, z_4), we see that $z_1 z_2 = 0$ at every minimum of $H_{4\text{–local}} - F$.

Using deduc-reduc we have:

$$H_{2\text{–local}} = 12 z_1 z_2 + z_1(z_3 - 3) + z_2(z_3 - 2z_3 - z_4 - 1) + F(x_3, x_4, x_5, \ldots, x_N) \quad (9)$$

which has the same global minima as $H_{4\text{–}local}$ but one fewer quartic and one fewer cubic term.

Bibliography

- Original paper, with application to integer factorization: [13].

C. Groebner Bases

Summary

Given a set of polynomials, a Groebner basis is another set of polynomials that have exactly the same zeros. The advantage of a Groebner basis is it has nicer algebraic properties than the original equations, in particular they tend to have smaller degree polynomials. The algorithms for calculating Groebner bases are generalisations of Euclid's algorithm for the polynomial greatest common divisor.

Work has been done in the field of 'Boolean Groebner bases', but while the variables are Boolean the coefficients of the functions are in $\mathbb{F}_2$ rather than $\mathbb{Q}$.

Cost

- 0 auxiliary variables needed.

- $\mathcal{O}\left(2^{2^n}\right)$ in general, $\mathcal{O}(d^{n^2})$ if the zeros of the equations form a set of discrete points, where d is the degree of the polynomial and n is the number of variables [14].

Pros

- No auxiliary variables needed.

- General method, which can be used for other rings, fields or types of variables.

Cons

- Best algorithms for finding Groebner bases scale double exponentially in n.

- Only works for Hamiltonians whose minimization corresponds to solving systems of discrete equations.

Example

Consider the following pair of equations:

$$xyzw + xz + yw - z = x + xy + z - 2 = 0. \tag{10}$$

Feeding these to Mathematica's `GroebnerBasis` function, along with the binarizing $x(x-1) = \ldots = w(w-1) = 0$ constraints, gives a Groebner basis:

$$\{wz - w, y + z - 1, x - 1\}. \tag{11}$$

From this we can immediately read off the solutions $x = 1$, $y = 1 - z$ and reduce the problem to $wz - w = 0$. Solving this gives a final solution set of: $(x, y, z, w) = (1, 0, 1, 0), (1, 0, 1, 1), (1, 1, 0, 0)$.

Bibliography

- Reduction and embedding of factorizations of all bi-primes up to $200,000$: [15].

D. Split Reduction

Summary

It has been shown in [16] that, if multiple runs of a minimization algorithm is permitted, it is possible to reduce a lot of the problem by conditioning on the most connected variables. We call each of these operations a *split*.

Cost

Exponential in the number of splits, as the number of problems to solve doubles with every split.

Pros

- This method can be applied to any problem and can be very effective on problems with a few very connected variables.

Cons

- Exponential cost in the worst case.

Example

Consider the simple objective function

$$H = 1 + z_1 z_2 z_5 + z_1 z_6 z_7 z_8 + z_3 z_4 z_8 - z_1 z_3 z_4. \tag{12}$$

In order to quadratize H, we first have to choose a variable to split over. In this case z_1 is the obvious choice since it is present in the most terms and contributes to the quartic term.

We then obtain two different problems:

$$H_0 = 1 + z_3 z_4 z_8 \tag{13}$$

$$H_1 = 1 + z_2 z_5 + z_6 z_7 z_8 + z_3 z_4 z_8 - z_3 z_4. \tag{14}$$

At this point, we could split H_0 again and solve it entirely, or use a qubit we saved in the previous split to quadratize our only problem.

To solve H_1, we can split again on z_8, resulting in problems:

$$H_{1,0} = 1 + z_2 z_5 + z_6 z_7 \tag{15}$$

$$H_{1,1} = 1 + z_2 z_5 + z_3 z_4. \tag{16}$$

Now both of these problems are quadratic. Hence we have reduced our original, hard problem into 3 easy problems, requiring only 2 extra runs of our minimization algorithm, and without needing any auxiliary variables.

Bibliography

- Original paper and application to Ramsey number calculation: [16].

III. METHODS THAT INTRODUCE AUXILIARY VARIABLES TO QUADRATIZE A SINGLE TERM

A. Negative Term Reduction

Summary

For a negative term $-z_1 z_2 ... z_k$, introduce a single auxiliary variable a and make the substitution:

$$-z_1 z_2 ... z_k = \min_a \left\{ (n - 1 - \sum z_i) a \right\}. \tag{17}$$

Cost

- 1 auxiliary variable for each k-local term.

Pros

- All resulting quadratic terms are submodular (have negative coefficients).
- Can reduce arbitrary order terms with only 1 auxiliary.

Cons

- Only works for negative terms.

Example

$$H_{6-\text{local}} = -2 z_1 z_2 z_3 z_4 z_5 z_6 + z_5 z_6, \tag{18}$$

has a unique minimum energy of -1 when all $z_i = 1$.

$$H_{2-\text{local}} = 2 \left(5a - z_1 a - z_2 a - z_3 a - z_4 a - z_5 a - z_6 a \right) + z_5 z_6 \tag{19}$$

has the same unique minimum energy, and it occurs at the same place (all $z_i = 1$), with $a = 1$.

Bibliography

- Original paper: [17].
- Discussion: [5], [7].

B. Positive Term Reduction

Summary

By considering the negated literals $\bar{z}_i = 1 - z_i$, we recursively apply the previous method to $z_1 z_2 \dots z_k = -\bar{z}_1 z_2 \dots z_k + z_2 z_3 \dots z_k$. The final identity is:

$$z_1 z_2 \dots z_k = \min_a \Big\{ \sum_{i=1}^{k-2} a_i \big(k - i - 1 + z_i - \sum_{j=i+1}^{k} z_j\big) \Big\} + z_{k-1} z_k \tag{20}$$

Cost

- $k-2$ auxiliary variables for each k-local term.

Pros

- Works for positive monomials.

Cons

- $k-1$ non-submodular quadratic terms.

Example

$$z_1 z_2 z_3 z_4 = \min_a a_1 \big(2 + z_1 - z_2 - z_3 - z_4\big) + a_2 \big(1 + z_2 - z_3 - z_4\big) + z_3 z_4 \tag{21}$$

Bibliography

- Summary: [18].

C. Ishikawa's Symmetric Reduction (Positive Term Reduction)

Summary

This method rewrites a positive monomial using symmetric polynomials, so all possible quadratic terms are produced and they are all non-submodular:

$$z_1 \dots z_k = \min_{a_1,\dots,a_{n_k}} \left\{ \sum_{i=1}^{n_k} a_i \big(c_{i,d} \big(-\sum_{j=1}^{k} z_j + 2i\big) - 1\big) + \sum_{i<j} z_i z_j \right\} \tag{22}$$

where $n_k = \left\lfloor \frac{k-1}{2} \right\rfloor$ and $c_{i,k} = \begin{cases} 1, & i = n_d \text{ and } k \text{ is odd,} \\ 2, & \text{else.} \end{cases}$

Cost

- $\left\lfloor \frac{k-1}{2} \right\rfloor$ auxiliary variables for each k-order term

Pros

- Works for positive monomials. About half as many auxiliary variables for each k-order term as the previous method.

Cons

- $\mathcal{O}(k^2)$ quadratic terms are created, which may make chimerization more costly.
- $\frac{k(k-1)}{2}$ non-submodular terms.
- Worse than the previous method for quartics, with respect to submodularity.

Example

$$z_1 z_2 z_3 z_4 = \min_a (3 - 2z_1 - 2z_2 - 2z_3 - 2z_4)a + z_1 z_2 + z_1 z_3 + z_1 z_4 + z_2 z_3 + z_2 z_4 + z_3 z_4 \tag{23}$$

Bibliography

- Original paper and application to image denoising: [5].

D. Asymmetric Reduction

Summary

Similar to other methods of reducing one term, this method can reduce a positive cubic monomial into quadratic terms using only one auxiliary variable, while introducing fewer non-submodular terms than the symmetric version.

The identity is given by:

$$z_1 z_2 z_3 = \min_a \{a - z_2 a - z_3 a + z_1 a + z_2 z_3\} \tag{24}$$

Cost

1 auxiliary qubit per positive cubic term.

Pros

- Works on positive monomials.
- Fewer non-submodular terms than Ishikawa Reduction.

Cons

- Only been shown to work for cubics.

Bibliography

- Original paper and application to computer vision: [19].

E. Bit-flipping

Summary

For any variable z, we can consider the negation $\bar{z} = 1 - z$. The process of exchanging z for $\bar{z}$ is called *flipping*. Using bit-flipping, an arbitrary function in n variables can be represented using at most $2^{(n-2)}(n-3)+1$ variables, though this is a gross overestimate.

Can be used in many different ways:

1. Flipping positive terms and using III A, recursively;

2. For $\alpha < 0$, we can reduce $\alpha \bar{z}_1 \bar{z}_2 ... \bar{z}_k$ very efficiently to submodular form using III A. A generalized version exists for arbitrary combinations of flips in the monomial which makes reduction entirely submodular [5];

3. When we have quadratized we can minimize the number of non-submodular terms by flipping.

4. We can make use of both z_i and $\bar{z}_i$ in the same Hamiltonian by adding on a sufficiently large penalty term: $\lambda(z_i + \bar{z}_i - 1)^2 = \lambda(1 + 2z_i\bar{z}_i - z_i - \bar{z}_i)$. This is similar to the ideas in reduction by substitution or deduc-reduc. In this way, given a quadratic in n variables we can make sure it only has at most n nonsubmodular terms if we are willing to use the extra n negation variables as well (so we have $2n$ variables in total).

Cost

- None, as replacing z_i with it's negation $\bar{z}_i$ costs nothing except a trivial symbolic expansion.

Pros

- Cheap and effective way of improving submodularity.

- Can be used to combine terms in clever ways, making other methods more efficient.

Cons

- Unless form of the Hamiltonian is known, spotting these 'factorizations' using negations is going to be difficult.
- If we want to use z and $\bar{z}$ we need an auxiliary variable.

Example

By bit-flipping z_2 and z_4, i.e. substituting $z_2 = 1 - \bar{z}_2$ and $z_4 = 1 - \bar{z}_4$, we see that:

$$H = 3z_1z_2 + z_2z_3 + 2z_1z_4 - 4z_2z_4 \tag{25}$$

$$= -3z_1\bar{z}_2 - \bar{z}_2z_3 - 2z_1\bar{z}_4 - \bar{z}_2\bar{z}_4 + 5z_1 + z_3 + 4\bar{z}_2 + 4\bar{z}_4 - 4. \tag{26}$$

The first expression is highly non-submodular while the second is entirely submodular.

Bibliography

- Original paper: [5].

IV. METHODS THAT INTRODUCE AUXILIARY VARIABLES TO QUADRATIZE MULTIPLE TERMS WITH THE SAME AUXILIARIES

A. Reduction by Substitution

Summary

Pick a variable pair z_i, z_j and substitute z_iz_j for new auxiliary variable $a_{i,j}$. Enforce equality in the ground states by adding some scalar multiple of $z_iz_j - 2z_ia_{i,j} - 2z_ia_{i,j} + 3a_{i,j}$ or similar.

Cost

- 1 auxiliary variable per reduction.

Pros

- Variable can be used across the entire Hamiltonian, reducing many terms at once.
- Simple.

Cons

- Inefficient for single terms as it introduces many variables.

- Introduces quadratic terms with large positive coefficients, making them highly non-submodular.

- Determining optimal substitutions is expensive.

Example

We pick the pair (z_1, z_2) and combine.

$$z_1z_2z_3 + z_1z_2z_4 \mapsto z_3a + z_4a + z_1z_2 - 2z_1a - 2z_1a + 3a \tag{27}$$

Bibliography

- Original paper: [20]

B. FGBZ Reduction (Fix-Gruber-Boros-Zabih)

Summary

Here we consider a set C of variables which occur in multiple monomials throughout the Hamiltonian. Each application 'rips out' this common component from each term [21][18].

Let $\mathcal{H}$ be a set of monomials, where $C \subseteq H$ for each $H \in \mathcal{H}$ and each monomial H has a weight α_H. The algorithm comes in 2 parts: when all $\alpha_H > 0$ and when all $\alpha_H < 0$. Combining the 2 gives the final method:

1. $\alpha_H > 0$

$$\sum_{H\in\mathcal{H}} \alpha_H \prod_{j\in H} z_j = \min_a (\sum_{H\in\mathcal{H}} \alpha_H) a \prod_{j\in C} z_j + \sum_{H\in\mathcal{H}} \alpha_H (1-a) \prod_{j\in H\setminus C} z_j \tag{28}$$

2. $\alpha_H < 0$

$$\sum_{H\in\mathcal{H}} \alpha_H \prod_{j\in H} z_j = \min_a \sum_{H\in\mathcal{H}} \alpha_H (1 - \prod_{j\in C} z_j - \prod_{j\in H\setminus C} z_j) a \tag{29}$$

Cost

- One auxiliary variable per application.

- In combination with III A, they can be used to make an algorithm which can reduce t positive monomials of degree d in n variables using $n + t(d-1)$ auxiliary variables in the worst case.

Pros

- Can reduce the connectivity of a Hamiltonian, as it breaks interactions between qubits.

Cons

- Positive method converts positive terms into negative ones of same order rather than reducing them, though these can then be reduced more easily.
- $\alpha_H < 0$ method only works for $|C| > 1$, and cannot quadratize cubic terms.

Example

First let $C = z_1$ and use the positive weight version:

$$z_1z_2z_3 + z_1z_2z_4 \mapsto 2a_1z_1 + (1-a_1)z_2z_3 + (1-a_1)z_2z_4 \tag{30}$$

$$= 2a_1z_1 + z_2z_3 + z_2z_4 - a_1z_2z_3 - a_1z_2z_4 \tag{31}$$

now we can use III A:

$$-a_1z_2z_3 - a_1z_2z_4 \mapsto 2a_2 - a_1a_2 - a_2z_2 - a_2z_3 + 2a_2 - a_1a_2 - a_2z_2 - a_2z_4 \tag{32}$$

$$= 4a_2 - 2a_1a_2 - 2a_2z_2 - a_2z_3 - a_2z_4. \tag{33}$$

Bibliography

- Original paper and application to image denoising: [21].

V. METHODS THAT REPRODUCE THE FULL SPECTRUM

In all the above, we take advantage of the fact that we just want the correct state and we don't care about anything else. This gave us the freedom to do all sorts of things to the Hamiltonian, as long as we keep the ground state the same. What if we want to quadratize a Hamiltonian while preserving the whole spectrum?

VI. STRATEGIES FOR COMBINING METHODS

Should look at ORI graph in [19]

Part II

Hamiltonians with both z & x terms (general quantum Hamiltonians)

In the above we only have z terms. If we have both z and x then quadratization methods have a rich history, dating back to the perturbative gadgets that were used by Kitaev and his camp for proving that the 2-local Hamiltonian problem is QMA complete and that AQC is equiavent to circuit-based quantum computing. Jacob Biamonte came up with various schemes for quadratizing general spin Hamiltonians with only 1 auxiliary qubit for each reduction (quintic to quartic, quartic to cubic, cubic to quadratic, etc.). Below we show how the ideas above such as FGBZ can be generalized to the case where there's both z and x terms. "auxiliary qubit recycling", where the auxiliary qubit used to substitute one term can be reused to reduce other terms in the Hamiltonian.

VII. RECURSIVE GADGET REDUCTION (ORIGINAL)

Kitaev. klocal -> (k-1)loal -> (k-2)local -> ... -> 2-local (1 auxiliary each time)

VIII. RECURSIVE GADGET REDUCTION (BETTER)

Oliviera & Terhal (2005), Cao et al (2013), Bravyi et al (2008), Duan et al (2011), Babbush et al (2013) klocal -> (k-1)loal -> (k-2)local -> ... -> 2-local (1 auxiliary each time)

IX. ONE-STEP GADGET REDUCTION

Jordan & Fahri

X. NUMERICAL

Panagiotis (2017) https://arxiv.org/pdf/1706.03637.pdf

XI. METHODS THAT REPRODUCE THE FULL SPECTRUM

Part III

Appendix

XII. FURTHER EXAMPLES

Example Here we show how deductions can arise naturally from the Ramsey number problem. Consider $\mathcal{R}(4,3)$ with $N = 4$ nodes. Consider a Hamiltonian:

$$H = (1-z_{12})(1-z_{13})(1-z_{23}) + \ldots + (1-z_{23})(1-z_{24})(1-z_{34}) + z_{12}z_{13}z_{14}z_{23}z_{24}z_{34}. \quad (34)$$

See [16] for full details of how we arrive at this Hamiltonian.

Since we are assuming we have no 3-independent sets, we know that $(1-z_{12})(1-z_{13})(1-z_{23}) = 0$, so $z_{12}z_{13}z_{23} = z_{12}z_{13} + z_{12}z_{23} + z_{13}z_{23} - z_{12} - z_{13} - z_{23} + 1$. This will be our deduction.

Using deduc-reduc we can substitute this into our 6-local term to get:

$$H = 2(1-z_{12})(1-z_{13})(1-z_{23}) + \ldots + (1-z_{23})(1-z_{24})(1-z_{34}) + \quad (35)$$

$$z_{14}z_{24}z_{34}(z_{12}z_{13} + z_{12}z_{23} + z_{13}z_{23} - z_{12} - z_{13} - z_{23} + 1). \quad (36)$$

We could repeat this process to remove all 5- and 4-local terms without adding any auxiliary qubits. Note in this case the error terms added by deduc-reduc already appear in our Hamiltonian.

XIII. NOTATION

The left Kronecker product by a 2×2 identity matrix is implied by Eqs. (1) and (2) in the following way for qubit operators $q \in \{z, x\}$, assuming $i < j$:

$$q_i = \mathbf{1}^{\otimes i-1} q \mathbf{1}^{\otimes (N-i+1)} \quad (37)$$

$$q_i \bar{q}_j = \mathbf{1}^{\otimes i-1} q \mathbf{1}^{\otimes (j-1+i)} \bar{q} \mathbf{1}^{\otimes (N-j+1)}. \quad (38)$$

Thinking in this way, with this notation might take some time to get used to, but tens of thousands of people are comfortable thinking this way (including any undergraduate student after a 1-semester quantum computing course). You can now see that H is a $2^N \times 2^N$ matrix with elements that are complex numbers. "Minimizing the Hamiltonian" just means finding the eigenvector of this matrix with lowest eigenvalue. On a classical computer, the cost of finding this eigenvector is the cost of diagonalizing the matrix: $\mathcal{O}(2^{3N})$, but undergraduate level quantum mechanics tells us that any physical system's state (wavefunction, ψ_n) is an eigenvector of a Hamiltonian and the eigenvalue E_n is just the energy associated with being in the n^{th} energy level:

$$H\psi_n = E_n\psi_n. \tag{39}$$

Eq. 39 is just the time-independent version of the Schroedinger equation, which you have at least heard of. The diagonal elements are the energies of the n levels and the off-diagonals are associated with the propensity for tunnelling from level n to level m. Any 2^N level system can be represented by N spin-1/2 particles (**qubtis**), and an example of a spin-1/2 particle is simply an electron. Some of the N electrons will have spin up and some will have spin down, hence 2^N possibilities. So instead of $\mathcal{O}(2^{3N})$ operations on a classical computer for finding the eigenvector with lowest eigenvalue (and hence doing the completely arbitrary computation), we can just (for example) put N electrons together with the appropriate H describing their energy, and then cool the system down to its ground state. This ground state is one out of 2^N possible states , and the configuration of spin up and spin down electrons encodes the desired computation.

ACKNOWLEDGMENTS

[1] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev, SIAM Review **50**, 755 (2008).

[2] J. D. Biamonte and P. J. Love, Physical Review A **78**, 012352 (2008).

[3] M. V. Altaisky, N. N. Zolnikova, N. E. Kaputkina, V. A. Krylov, Y. E. Lozovik, and N. S. Dattani, Applied Physics Letters **108**, 103108 (2016).

[4] H. Ishikawa, in *2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2009) pp. 2993–3000.

[5] H. Ishikawa, IEEE Transactions on Pattern Analysis and Machine Intelligence **33**, 1234 (2011).

[6] H. Ishikawa, in *2014 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2014) pp. 1362–1369.

[7] M. Anthony, E. Boros, Y. Crama, and A. Gruber, (2015).

[8] N. S. Dattani and N. Bryans, (2014), arXiv:1411.6758.

[9] F. Gaitan and L. Clark, (2012), arXiv:arXiv:1103.1345v3.

[10] Z. Bian, F. Chudak, W. G. Macready, L. Clark, and F. Gaitan, Physical Review Letters **111**, 130505 (2013).

[11] V. S. Denchev, S. Boixo, S. V. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J. Martinis, and H. Neven, Phys. Rev. X **6**, 31015 (2016).

[12] S. Mandrà, Z. Zhu, W. Wang, A. Perdomo-Ortiz, and H. G. Katzgraber, Phys. Rev. A **94**, 22337 (2016).

[13] R. Tanburn, E. Okada, and N. Dattani, (2015), arXiv:1508.04816.

[14] M. Bardet, Algorithms Seminar , 85 (2002).

[15] R. Dridi and H. Alghassi, (2016), arXiv:1604.05796.

[16] E. Okada, R. Tanburn, and N. S. Dattani, , 5 (2015), arXiv:1508.07190.

[17] D. Freedman and P. Drineas, in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2 (IEEE, 2005) pp. 939–946.

[18] E. Boros and A. Gruber, (2014), arXiv:1404.6538.

[19] A. C. Gallagher, D. Batra, and D. Parikh, in *CVPR 2011* (IEEE, 2011) pp. 1857–1864.

[20] I. G. Rosenberg, Cahiers du Centre d'Etudes de Recherche Operationnelle **17**, 71 (1975).

[21] A. Fix, A. Gruber, E. Boros, and R. Zabih, in *2011 International Conference on Computer Vision* (IEEE, 2011) pp. 1020–1027.