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,*

A collaborative, evolving, open review paper on k-local to 2-local transformations (quadratizations) in classical computing, quantum annealing, and universal adiabatic quantum computing.

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:

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

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 α 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:

$$H_{3\text{-local}} = b_1b_2 + b_2b_3 + b_3b_4 - 4b_1b_2b_3. \tag{6}$$

If $b_1b_2b_3 = 0$, no assignment of our variables will we be able to reach a lower energy than if $b_1b_2b_3 = 1$. Hence $(b_1, b_2, b_3) = (1, 0, 0)$ is a Local Excludable Configuration (ELC) and we can form the polynomial:

$$H_{2\text{-local}} = H_{3\text{-local}} + 4b_1(1 - b_2)(1 - b_3) = b_1b_2 + b_2b_3 + b_3b_4 + 4b_1 - 4b_1b_2 - 4b_1b_3. \tag{7}$$

In both cases, the only global minima occur when $b_1b_2b_3 = 1$.

Bibliography

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

B. Deduction Reduction (Deduc-reduc)

Summary

We look for *deductions* (e.g. $b_1b_2 = 0$) 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 XIV), 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:

$$H_{4-\text{local}} = b_1 b_2(10 + b_3 + b_3 b_4) + b_1(b_3 - 3) + b_2(b_3 - 2b_3 - b_4 - 1) + F(x_3, x_4, x_5, \ldots, x_N) \quad (8)$$

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

Since the 10 coefficient of $b_1 b_2$ is greater than the sum of all of the other coefficients involving b_1 or b_2, it must be the case that $b_1 b_2 = 0$. Specifically, for the 4 assignments of (b_3, b_4), we see that $b_1 b_2 = 0$ at every minimum of $H_{4-\text{local}} - F$.

Using deduc-reduc we have:

$$H_{2-\text{local}} = 12 b_1 b_2 + b_1(b_3 - 3) + b_2(b_3 - 2b_3 - b_4 - 1) + F(x_3, x_4, x_5, \ldots, x_N) \quad (9)$$

which has the same global minima as $H_{4-local}$ 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:

$$b_1b_2b_3b_4 + b_1b_3 + b_2b_4 - b_3 = b_1 + b_1b_2 + b_3 - 2 = 0. \tag{10}$$

Feeding these to Mathematica's `GroebnerBasis` function, along with the binarizing $b_1(b_1-1) = \ldots = b_4(b_4-1) = 0$ constraints, gives a Groebner basis:

$$\{b_4b_3 - b_4, b_2 + b_3 - 1, b_1 - 1\}. \tag{11}$$

From this we can immediately read off the solutions $b_1 = 1$, $b_2 = 1 - b_3$ and reduce the problem to $b_3b_4 - b_4 = 0$. Solving this gives a final solution set of: $(b_1, b_2, b_3, b_4) = (1,0,1,0), (1,0,1,1), (1,1,0,0)$.

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

$$H = 1 + b_1b_2b_5 + b_1b_6b_7b_8 + b_3b_4b_8 - b_1b_3b_4. \tag{12}$$

In order to quadratize H, we first have to choose a variable to split over. In this case b_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:

$$H_0 = 1 + b_3b_4b_8 \tag{13}$$

$$H_1 = 1 + b_2b_5 + b_6b_7b_8 + b_3b_4b_8 - b_3b_4. \tag{14}$$

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 b_8, resulting in problems:

$$H_{1,0} = 1 + b_2b_5 + b_6b_7 \tag{15}$$

$$H_{1,1} = 1 + b_2b_5 + b_3b_4. \tag{16}$$

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 $-b_1 b_2 ... b_k$, introduce a single auxiliary variable b_a and make the substitution:

$$-b_1 b_2 \dots b_k = \min_{b_a} \left\{ (n - 1 - \sum b_i) b_a \right\}. \tag{17}$$

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

$$H_{6-\text{local}} = -2 b_1 b_2 b_3 b_4 b_5 b_6 + b_5 b_6, \tag{18}$$

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

$$H_{2-\text{local}} = 2 \left(5 b_a - b_1 b_a - b_2 b_a - b_3 b_a - b_4 b_a - b_5 b_a - b_6 b_a \right) + b_5 b_6 \tag{19}$$

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

Bibliography

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

B. Positive Term Reduction

Summary

By considering the negated literals $\bar{b}_i = 1 - b_i$, we recursively apply the previous method to $b_1 b_2 \dots b_k = -\bar{b}_1 b_2 \dots b_k + b_2 b_3 \dots b_k$. The final identity is:

$$b_1 b_2 \dots b_k = \min_{b_a} \Big\{ \sum_{i=1}^{k-2} b_{a_i} \big(k - i - 1 + b_i - \sum_{j=i+1}^{k} b_j\big) \Big\} + b_{k-1} b_k \tag{20}$$

Cost

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

Pros

- Works for positive monomials.

Cons

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

Example

$$b_1 b_2 b_3 b_4 = \min_{b_a} b_{a_1} (2 + b_1 - b_2 - b_3 - b_4) + b_{a_2} (1 + b_2 - b_3 - b_4) + b_3 b_4 \tag{21}$$

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:

$$b_1 \dots b_k = \min_{b_{a_1}, \dots, b_{a_{n_k}}} \left\{ \sum_{i=1}^{n_k} b_{a_i} (c_{i,d} (-\sum_{j=1}^{k} b_j + 2i) - 1) + \sum_{i<j} b_i b_j \right\} \tag{22}$$

where $n_k = \lfloor \frac{k-1}{2} \rfloor$ 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

$$b_1b_2b_3b_4 = \min_{b_a}(3 - 2b_1 - 2b_2 - 2b_3 - 2b_4)b_a + b_1b_2 + b_1b_3 + b_1b_4 + b_2b_3 + b_2b_4 + b_3b_4 \tag{23}$$

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:

$$b_1b_2b_3 = \min_{b_a} \{b_a - b_2b_a - b_3b_a + b_1b_a + b_2b_3\} \tag{24}$$

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 b, we can consider the negation $\bar{b} = 1 - b$. The process of exchanging b for $\bar{b}$ 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 $\alpha\bar{b}_1\bar{b}_2...\bar{b}_k$ 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 b_i and $\bar{b}_i$ in the same Hamiltonian by adding on a sufficiently large penalty term: $\lambda(b_i + \bar{b}_i - 1)^2 = \lambda(1 + 2b_i\bar{b}_i - b_i - \bar{b}_i)$. 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 b_i with it's negation $\bar{b}_i$ 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 b and $\bar{b}$ we need an auxiliary variable.

Example

By bit-flipping b_2 and b_4, i.e. substituting $b_2 = 1 - \bar{b}_2$ and $b_4 = 1 - \bar{b}_4$, we see that:

$$H = 3b_1b_2 + b_2b_3 + 2b_1b_4 - 4b_2b_4 \tag{25}$$

$$= -3b_1\bar{b}_2 - \bar{b}_2b_3 - 2b_1\bar{b}_4 - \bar{b}_2\bar{b}_4 + 5b_1 + b_3 + 4\bar{b}_2 + 4\bar{b}_4 - 4. \tag{26}$$

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 b_i, b_j and substitute b_ib_j for new auxiliary variable $b_{a_{i,j}}$. Enforce equality in the ground states by adding some scalar multiple of $b_ib_j - 2b_ib_{a_{i,j}} - 2b_ib_{a_{i,j}} + 3b_{a_{i,j}}$ 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 (b_1, b_2) and combine.

$$b_1b_2b_3 + b_1b_2b_4 \mapsto b_3b_a + b_4b_a + b_1b_2 - 2b_1b_a - 2b_1b_a + 3b_a \tag{27}$$

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. $\alpha_H > 0$

$$\sum_{H\in\mathcal{H}} \alpha_H \prod_{j\in H} b_j = \min_{b_a}\Big(\sum_{H\in\mathcal{H}} \alpha_H\Big) b_a \prod_{j\in C} b_j + \sum_{H\in\mathcal{H}} \alpha_H (1 - b_a) \prod_{j\in H\setminus C} b_j \tag{28}$$

2. $\alpha_H < 0$

$$\sum_{H\in\mathcal{H}} \alpha_H \prod_{j\in H} b_j = \min_{b_a} \sum_{H\in\mathcal{H}} \alpha_H \Big(1 - \prod_{j\in C} b_j - \prod_{j\in H\setminus C} b_j\Big) b_a \tag{29}$$

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 $C = b_1$ and use the positive weight version:

$$b_1b_2b_3 + b_1b_2b_4 \mapsto 2b_{a_1}b_1 + (1 - b_{a_1})b_2b_3 + (1 - b_{a_1})b_2b_4 \tag{30}$$

$$= 2b_{a_1}b_1 + b_2b_3 + b_2b_4 - b_{a_1}b_2b_3 - b_{a_1}b_2b_4 \tag{31}$$

now we can use III A:

$$-b_{a_1}b_2b_3 - b_{a_1}b_2b_4 \mapsto 2b_{a_2} - b_{a_1}b_{a_2} - b_{a_2}b_2 - b_{a_2}b_3 + 2b_{a_2} - b_{a_1}b_{a_2} - b_{a_2}b_2 - b_{a_2}b_4 \tag{32}$$

$$= 4b_{a_2} - 2b_{a_1}b_{a_2} - 2b_{a_2}b_2 - b_{a_2}b_3 - b_{a_2}b_4. \tag{33}$$

Bibliography

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

V. METHODS THAT REPRODUCE THE FULL SPECTRUM

All known methods to map the entire spectrum of a Hamiltonian rely on the same basic principle: auxilliary variables are added, to mediate the high-order interactions, and constraints are added to restrict the values which the axilliary variables can take. In practice, constraints are implemented as one and two term equations with much stronger energy penalties than those in the original Hamiltonian. As long as the constraints are sufficiently strong, the low energy subspace of the 2-local Hamiltonian will be exactly the spectrum of the k-local Hamiltonian which has been mapped. Moreover, the location of the original 'logical' variables is known and so the resulting configuration of the mapped variables can easily be determined.

A. Recursive Order Reduction

Summary

For each k-local term an auxilliary variable and a quadratic penalty function to constrain the auxilliary variable, is used to reduce the term's order to $(k-1)$-local. This is repeated recursively until the term is 2-local.

Cost

- $k-2$ auxiliary variables to reduce each k-local terms.
- At most kt for a k-local Hamiltonian of t terms.

Pros

- Works for arbitrary k Nick, are there methods that DON'T work for arbitrary k? and simple to implement.
- In principle can be used for more types of equations than just $b_1 b_2 ... b_n$ assuming equations can be found.Nick, why is this a "pro", what is the advantage over other methods?
- Gives a 2-local Hamiltonian with sparse connectivity relative to (some?) other methods.

Cons

- Not very symmetric with respect to variables.
- Not conducive to specialized hardware design (what specialized hardware design?)

Example

Let us assume we want to reproduce the spectrum of $H_4 = b_1 b_2 b_3 b_4$, we note that $P(a_k, b_i, b_j) = 3a_k - 2a_k(b_i + b_j) + b_i b_j$ will yield an energy of $E = 0$ iff $a_k = b_i b_j$, and $E > 0$ otherwise. We therefore observe that, assuming a large enough value of λ, the low energy sector of $H_3 = b_1 b_2 b_{a,1} + \lambda(3b_{a,1} - 2b_{a,1}(b_3 + b_4) + b_3 b_4)$ will reproduce the spectrum of H_4 with terms that are k-local with $k \leq 3$. Performing the same trick again yields a 2-local Hamiltonian:

$$H_2 = b_1 a_2 + \lambda\left(3b_{a,2} - 2b_{a,2}(b_2 + b_{a,1}) + b_2 b_{a,1}\right) + \lambda\left(3b_{a,1} - 2b_{a,1}(b_3 + b_4) + b_3 b_4\right) \tag{34}$$

Bibliography

- add later, references to many D-Wave papers including Biamonte and Bian papers, as well as Perdomo-Ortiz paper

B. Chained Three Body Parity Operators

Summary

We now wish to guarantee that $b_{a,k} = b_i b_j$ can be chained together to make large product terms consisting of b. The penalty term which guarantees that $b_{a,k} = \pm b_i b_j$ is $P(b_{a,k}, b_i, z_j) = \mp b_{a,k} b_i b_j$. Using a gadget for the three local Hamiltonian, the low energy spectrum of Ising terms of any locality can be produced. This method was originally only used to reprduce the ground state of high locality terms, but states of the "wrong" parity ($q_k = \mp b_i b_j$) will all have the same energy as well, so it reproduces the full spectrum.

Cost

- The best known gadget for a three local Ising term uses one auxilliary qubit. Based on this gadget an $n \geq 4$ body Ising term can be made using $3 + 2(n-4)$ auxilliary qubits.

Pros

- Natural transmon implementation.
- Relatively low degree of connectivity.

Cons

- Does not preserve degeneracy, ground state will retain orginal degeneracy, but excited states will have degeneracy multipled by $n-3$
- Not very symmetric.

Example

Let us use this method to reproduce the spectrum of the following five local Hamiltonian $b_1 b_2 b_3 b_4 b_5$. We first observe that the following three local gadget reproduces the spectrum of the three local term $\pm b_1 b_2 b_3$

$$P_{\pm}(b_1, b_2, b_3; \lambda) = \lambda \left[b_1 b_2 + b_2 b_3 + b_3 b_1 + 2 b_a (b_1 + b_2 + b_3) \right] \mp (b_1 + b_2 + b_3 + 2 b_a). \tag{35}$$

Using thee copies of this three local gadget as a building block, and using two additional auxilliary variables, the spectrum of the five local term $b_1 b_2 b_3 b_4 b_5$ can be reproduced by the following Hamiltonian

$$H_5 = P_+(b_1, b_2, b_{a,1}; \lambda) + P_+(b_{a,1}, b_3, b_{a,2}; \lambda) + P_+(b_{a,2}, b_4, b_5; \lambda). \tag{36}$$

Bibliography

- add later, cite second Lechner paper, plus NC work on three local gadget

C. Multibody Operators in PAQC

Summary

Parity adiabatic quantum computing (PAQC) is an AQC archetecture in which logical qubits are mapped onto a highly non-local logical space of a Hamiltonian. These methods allow for high order terms to be directly implemented, but also require high locality terms to be implemented. While these archetectures are not strictly a mapping from devices with high locality to two local ones, they may be built upon any gadget which has the same ground state manifold as $\prod_i z_i$. In other words, even PAQC schemes can reprduce the entire spectrum of a Hamiltonian, even if the gadgets which they are built on top of do not.

Cost

- In the original (Lechner-Hauke-Zoller) scheme citation, an n body PAQC constraint which is capable of reproducing the full spectrum requires a $2n$ local gadget which reproduces only the ground state manifold of $\prod_i^{2n} z_i$.

- Since n logical operators must intersect for an n local gadget, at least a gadget which reproduces the ground state manifold of the ground state manifold of $\prod_i^n z_i$ is required.

Pros

- Connection to stablizers allows for novel decodingcitations.

- Natural for producing highly connected Hamiltonians.

- Some implementations are highly symmetric, which may be good for quantum simulations.

Cons

- Requires other gadgets to implement.
- Potentially difficult to embed arbitrary sparse Hamiltonians efficiently.

Example

Rather than considering a mapping which maps qubit values directly, let us consider a mapping which maps physical qubits to the relative parity of paris of logical qubits. If we consider only the relative parity pairs of qubits, we observe that there will be $n(n-1)$ such parity values, but not all of these parity values correspond to physically realizable configurations, for instance it is not mathematically possible for to have $z_1 z_2 = -1$, $z_2 z_3 = 1$ and $z_1 z_3 = 1$ simultaneously. We therefore need to add constraints to gaurantee that only mathematically consistant configurations are allowed. If we consider four qubits, these constraints are realized in the ground state manifold of the three qubit LHZ constraint Hamiltonian:

$$H_{LHZ} = -z_{1,2} z_{1,3} z_{2,3} - z_{1,3} z_{1,4} z_{2,4} z_{2,3} - z_{2,3} z_{2,4} z_{3,4}. \tag{37}$$

Two local coupling between logical qubits in this state can therefore be achieved by introducing single qubit variables corresponding to each coupling, for instance adding a term $m\, z_{1,2}$ will intrduce a coupling between logical qubits 1 and 2 of strength m. If we want to add the four body constraint $z_1 z_2 z_3 z_4$, we must add an additional physical qubit $z_{1,2,3,4}$ which is constrained to match the parity of $z_1 z_2 z_3 z_4$. To accomplish this, we note that this total parity has to match $z_{1,2} z_{3,4}$, the Hamiltonian with added four body coupling of strength m therefore takes the form:

$$H_4 = m\, z_{1,2,3,4} + \lambda \left(H_{LHZ} - z_{1,2} z_{3,4} z_{1,2,3,4} \right), \tag{38}$$

where λ is a large postive number.

If we instead wanted to create the three body coupling $z_1 z_2 z_3$ of strength m, we would have to add an additional variable z_1, since Eq. 37 only constrains the logical qubits up to global spin inversion. By noting that $z_1 z_2 z_3 = z_1 z_{2,3}$ we see that such a three body

Hamiltonian can take the form:

$$H_3 = m\, z_{1,2,3} + \lambda\,(H_{LHZ} - z_1 z_{2,3} z_{1,2,3}). \tag{39}$$

Bibliography

- add later, cite LHZ and Simon Benjamin stabilizer paper

D. Symmetry Based Mappings

subsections I (NC) need to have:

- recursive qubo methods
- chains of 3 local gadgets (thing in second Lechner paper)
- our methods based on symmetry
- Simon Benjamin Stabilizer idea and multidimensional LHZ (one section)
- Possibly others: need to check how distinct some of the work Alejandro Perdomo-Ortiz and Jacob Biamonte have done is, whether these should be their own subsections or just references in exiting ones

VI. STRATEGIES FOR COMBINING METHODS

Should look at ORI graph in [19]

Part II

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

The most general time-independent Hamiltonian for n qubits acted on only by Pauli operators is:

$$H = \alpha I + \sum_{i}^{n} \left(\alpha_i^{(z)} z_i + \alpha_i^{(x)} x_i + \alpha_i^{(y)} y_i \right) + \tag{40}$$

$$\sum_{ij}^{n} \left(a_{ij}^{(zz)} z_i z_j + a_{ij}^{(zx)} z_i x_j + a_{ij}^{(zy)} z_i y_j + a_{ij}^{(xz)} x_i z_j + a_{ij}^{(xx)} x_i x_j + a_{ij}^{(xy)} x_i y_j + a_{ij}^{(yz)} y_i z_j + a_{ij}^{(yx)} y_i x_j + a_{ij}^{(yy)} y_i y_j \right) + \tag{41}$$

$$\sum_{ijk}^{n} \left(a_{ijk}^{(zzz)} z_i z_j z_k + a_{ijk}^{(zzx)} z_i z_j x_k + a_{ijk}^{(zzy)} z_i z_j y_k + a_{ijk}^{(zxz)} z_i x_j z_k + \cdots + a_{ijk}^{(yyy)} y_i y_j y_k \right) + \tag{42}$$

$$\sum_{ijkl\cdots n}^{n} \left(a_{ijkl\cdots n}^{(zzzz\cdots z)} z_i z_j z_k z_l \cdots z_n + a_{ijkl\cdots n}^{(zzz\cdots x)} z_i z_j z_k \cdots x_n + a_{ijkl\cdots n}^{(zzz\cdots y)} z_i z_j z_k \cdots y_n + a_{ijkl\cdots n}^{(zz\cdots xz)} z_i z_j \cdots x_{n-1} z_n + \cdots + a_{ijkl\cdots n}^{(yyy\cdots y)} y_i y_j y_k \cdots y_n \right) \tag{43}$$

We wish to find a 2-local Hamiltonian of the form of Eq. (2) or Eq. (1) that is equivalent to Eq. (40) in terms of reproducing certain desired properties (e.g. same ground state(s), same ground eigenvalue(s), same full eigenspectrum, etc.) to within the desired precision.

VII. BRUTE FORCE NUMERICAL

If our only wish is for the 2-local Hamiltonian to preserve the ground state of the original k-local Hamiltonian, we can optimize the coefficients of the 2-local Hamiltonian so that the difference between the ground state of the original Hamiltonian and the ground state of the 2-local Hamiltonian is minimized under some desired measure. For example, if we wish for the dot product of the lowest energy eigenvector of the original H and that of the new $H_{2-\text{local}}$ to be 0.9, then the following two Hamiltonians are equivalent for our purposes:

$$z_1 z_2 + y_1 x_2 z_3 \mapsto \alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 z_3 + \alpha_4 z_4 + \alpha_5 x_1 + \alpha_6 x_2 + \alpha_7 x_3 + \alpha_8 x_4 + \alpha_9 z_1 z_1 + \alpha_{10} z_1 z_2 + \cdots + \alpha_{72} z_4 x_4 \tag{44}$$

This should be easy to optimize in MATLAB, assuming that 1 auxiliary qubit is enough. Most α are probably 0 because the characteristic polynomial of the left side is of degree 8, so there's far more unknowns than equations. We can then put numerical values above, in place of the α's.

Likewise, if we only wish for the lowest eigenvalue E_0 of the original H to be recovered to within 0.01, for example, we can minimize the function $(E_{0,k-\text{local}} - E_{0,2-\text{local}})^2$ with respect to the α coefficients of the 2-local Hamiltonian until the objective function evaluates to a number smaller than 0.1^2. For example:

$$z_1z_2 + y_1x_2z_3 \mapsto \alpha_1z_1 + \alpha_2z_2 + \alpha_3z_3 + \alpha_4z_4 + \alpha_5x_1 + \alpha_6x_2 + \alpha_7x_3 + \alpha_8x_4 + \alpha_9z_1z_1 + \alpha_{10}z_1z_2 + \cdots + \alpha_{72}z_4x_4 \tag{45}$$

This should be even easier to optimize in MATLAB.

It is also possible to preserve more eigenvalues and/or eigenvectors than just the lowest, for example by minimizing the objective function:

$$\sum_{i}^{i_{\max}} A(E_{i,k-\text{local}} - E_{i,2-\text{local}})^2 + B\frac{1}{\langle\psi_i|\psi_i\rangle} \tag{46}$$

where one can choose A and B according to whether preserving the eigenvalues (A) or the eigenvectrs (B) is more important.

The more auxiliary qubits included in the 2-local Hamiltonian, the more parameters that can be optimized in attempt to make the ground state arbitrarily close to the orginal ground state.

Pros

- Very general.
- Can control which aspects of the 2-local Hamiltonion are most important
- Can likely even gadgetize a "universal" Hamiltonian into an Ising one of larger sized Hilbert space

Cons

- Doing the transformation costs more than solving the problem itself (only useful for testing what's theoretically possible).

Bibliography

- In June 2017 Ref. https://arxiv.org/pdf/1706.03637.pdf was posted on arXiv but it is not clear to me why they have the constraint (7d), the definiion of their matrix norm in (7c) doesn't seem to be given as far as I can see, and their (7a) uses a square while (7d) uses an absolute value which means that one of them is being treated more preferentially than the other, even when they say they are (think they are) using "equal weights".

VIII. RECURSIVE GADGET REDUCTION (ORIGINAL)

Summary

Group all k-local terms together and express their sum as a sum of products of commuting matrices s_{ij}:

$$H_{k-\text{local}} = \sum_i \prod_j s_{ij} + H_{(k-1)-\text{local}}. \tag{47}$$

Define auxiliary qubits labeled by a_{ij} and make the transformation:

$$\sum_i \prod_j s_{ij} \mapsto \frac{\Delta}{4}\left(9 - \sum_i \left(\sum_j z_{a_{ij}}\right)^2 - \sum_j z_{a_{ij}}^2 + \frac{1}{\sqrt[3]{6\Delta}} x_{a_{ij}}\right) + \Lambda. \tag{48}$$

The result will be a $(k-1)$-local Hamiltonian with the same low-lying spectrum as $H_{k-\text{local}}$ for large enough Δ.

Cost

The number of auxiliary qubits needed is the number of s_{ij} matrices in Eq. (47).

Pros

-
-

Cons

-
-

Example

Bibliography

- Kempe et al.

IX. RECURSIVE GADGET REDUCTION (BETTER)

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

A. Oliviera & Terhal (2005)

Summary

Cost

Pros

-

-

Cons

-

-

Example

Bibliography

B. Bravyi et al (2008)

Summary

Cost

Pros

-

-

Cons

-

-

Example

Bibliography

C. Duan et al (2011)

Summary

Cost

Pros

-

-

Cons

-

-

Example

Bibliography

D. Babbush et al. (2013)

Summary

Cost

Pros

-

-

Cons

-

-

Example

Bibliography

E. Cao et al. 2013

Summary

Cost

Pros

-
-

Cons

-
-

Example

Bibliography

X. ONE-STEP GADGET REDUCTION

Jordan & Fahri

Summary

Cost

Pros

-
-

Cons

-
-

Example

Bibliography

XI. NON-PERTURBATIVE EMBEDDINGS

Subasi & Jarzynski (2016)

Summary

Cost

Pros

-
-

Cons

-
-

Example

Bibliography

XII. 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?

Part III

Appendix

XIII. NOTATION

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

$$q_i = \mathbf{1}^{\otimes i-1} q \mathbf{1}^{\otimes (N-i+1)} \tag{49}$$

$$q_i \bar{q}_j = \mathbf{1}^{\otimes i-1} q \mathbf{1}^{\otimes (j-1+i)} \bar{q} \mathbf{1}^{\otimes (N-j+1)}. \tag{50}$$

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:

$$H\psi_n = E_n \psi_n. \tag{51}$$

Eq. 51 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.

XIV. 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:

$$H = (1 - z_{12})(1 - z_{13})(1 - z_{23}) + \ldots + (1 - z_{23})(1 - z_{24})(1 - z_{34}) + z_{12} z_{13} z_{14} z_{23} z_{24} z_{34}. \tag{52}$$

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:

$$H = 2(1 - z_{12})(1 - z_{13})(1 - z_{23}) + \ldots + (1 - z_{23})(1 - z_{24})(1 - z_{34}) + \tag{53}$$

$$z_{14}z_{24}z_{34}(z_{12}z_{13} + z_{12}z_{23} + z_{13}z_{23} - z_{12} - z_{13} - z_{23} + 1). \tag{54}$$

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.

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.