Adiabatic Quantum Graph Matching with Permutation Matrix Constraints — Supplementary Material —

Marcel Seelbach Benkner[1] Vladislav Golyanik[2] Christian Theobalt[2] Michael Moeller[1]

[1]University of Siegen [2]Max Planck Institute for Informatics, SIC

This supplementary material provides details on the derivations of the Quantum Graph Matching (QGM) approach as well as more experimental results. The proofs of Lemma (4.1) and Prop. (4.1) of the main matter can be found in Sec. 1. Sec. 2 complements the description of our experiments with the quantum computers, accessible via IBM Quantum Experience. Next, we show additional results of QGM for point cloud alignment in Sec. 3 and nearly isometric non-rigid shape registration in Sec. 4, followed by a discussion of the minor embedding and negative results in Secs. 5 and 6, respectively. Note that D-Wave provides one minute of QPU time on their annealers for research purposes per month. We obtain three minutes in total during the current project and use 80% of this time for our experiments. One minute of time on the quantum annealer can suffice to run hundreds of problems.

1. Proofs for Section 4

In order to ensure that we can incorporate a constraint of the form $A\mathbf{x} = \mathbf{b}$ as a penalty, the corresponding penalty parameter λ has to be chosen such that

$$\min_{\mathbf{x}\in\{0,1\}^n, A\mathbf{x}=\mathbf{b}} \mathbf{x}^T W\mathbf{x} < \min_{\mathbf{x}\in\{0,1\}^n, A\mathbf{x}\neq\mathbf{b}} \mathbf{x}^T W\mathbf{x} + \lambda\|A\mathbf{x}-\mathbf{b}\|^2. \tag{1}$$

A naive estimate for a sufficiently large λ is obtained by using

$$\min_{\mathbf{x}\in\{0,1\}^n, A\mathbf{x}=\mathbf{b}} \mathbf{x}^T W\mathbf{x} \leq \sum_{i,j} \max\{0, W_{i,j}\} \tag{2}$$

and

$$\begin{aligned}
&\min_{\mathbf{x}\in\{0,1\}^n, A\mathbf{x}\neq\mathbf{b}} \mathbf{x}^T W\mathbf{x} + \lambda\|A\mathbf{x}-\mathbf{b}\|^2 \\
\geq &\min_{\mathbf{x}\in\{0,1\}^n, A\mathbf{x}\neq\mathbf{b}} \mathbf{x}^T W\mathbf{x} + \lambda \min_{\mathbf{x}\in\{0,1\}^n, A\mathbf{x}\neq\mathbf{b}} \|A\mathbf{x}-\mathbf{b}\|^2 \\
\geq &\min_{\mathbf{x}\in\{0,1\}^n} \mathbf{x}^T W\mathbf{x} + \lambda \underbrace{\min_{s\in\{-1,1\}^n, A\mathbf{x}\neq\mathbf{b}} \|A\mathbf{x}-\mathbf{b}\|^2}_{=:d} \\
\geq &-\sum_{i,j} \max\{0, -W_{i,j}\} + \lambda d.
\end{aligned} \tag{3}$$

Thus a sufficient upper bound for λ is given by

$$\begin{aligned}
\lambda &> \frac{1}{d}\sum_{i,j}(\max\{0, -W_{i,j}\} + \max\{0, W_{i,j}\}) \\
&= \frac{1}{d}\sum_{i,j}|W_{i,j}|.
\end{aligned}$$

The influence of the linear term $\mathbf{c}^\mathrm{T}\mathbf{x}$ can be calculated in the same way. Alternatively $\mathbf{c}$ can be added to the diagonal of W, since $x_i^2 = x_i$ for $x_i \in \{0,1\}$. The fact that $d = 2$ can be proven by the following reasoning. Consider that there is a violation for the rows so that one row does not sum up to one and $\sum_{k \text{ belongs to a row}} (Ax-b)_k^2 = 1$. Since the sum over all rows of a matrix is the same as the sum over all columns, one constraint for the columns is violated as well. Since the same argument can be made if we switch rows and columns d has to be 2.

1.1. Proof of Proposition (4.1)

Better estimates can be obtained by exploiting more about the specific structure of the constraint. We aim to formulate the optimization problem as an unconstrained optimization problem with row-wise penalty parameters:

$$\min_{\mathbf{x}\in\{0,1\}^{n^2}} \mathbf{x}^\mathrm{T} W\mathbf{x} + \mathbf{c}^T\mathbf{x} + \sum_{i=1}^{2n} \lambda_i((A\mathbf{x})_i - b_i)^2. \tag{4}$$

One can solve the following optimization problems

$$D_{\mathcal{J}_j} = \max_{k\in\mathcal{J}_j}(\sum_i |(W_{k,i} + W_{i,k})| + |W_{k,k}| + |c_k|) \tag{5}$$

for every row or column $\mathcal{J}_j$.

The following bound for λ_j can be used to obtain an unconstrained optimization problem as in (4):

$$\lambda_j = \max_{k\in\mathcal{J}_j}(\sum_i |(W_{k,i}+W_{i,k})| + |W_{k,k}| + |c_k|) \quad (6)$$
$$+\frac{1}{2}(\max_k \sum_i |(W_{k,i}+W_{i,k})| + |W_{k,k}| + |c_k|) \quad (7)$$
$$= D_{\mathcal{J}_j} + \frac{1}{2} D_{\{1,\ldots,n^2\}} \quad (8)$$

Proof. Let x be an arbitrary element in $\{0,1\}^{n^2}$. We want to show that for the above choice of λ_j

$$\exists \mathbf{p} \in \text{vec}(\mathbb{P}_n) : \mathbf{p}^{\text{T}} W \mathbf{p} + \mathbf{c}^T \mathbf{p} \quad (9)$$
$$\leq \mathbf{x}^{\text{T}} W \mathbf{x} + \mathbf{c}^T \mathbf{x} + \sum_{i=1}^{2n} \lambda_i ((A\mathbf{x})_i - b_i)^2 \quad (10)$$

For the matrix X, which fulfills $x = \text{vec}(X)$, we can construct sets with the property:

$$(i,j) \in I \Rightarrow X_{i,j} = 1 \quad \wedge \quad \forall k \in \{1,...,n\} \setminus \{j\} (i,k) \notin I$$
$$\wedge \quad \forall k \in \{1,...,n\} \setminus \{j\} (k,j) \notin I, \quad (11)$$

We name one of these sets that has the maximal possible number of elements I_{max}. The permutation matrix P with $\mathbf{p} = \text{vec}(P)$ that we want to construct can be any permutation with ones placed at the positions in I_{max}. We can get from X to P by first erasing all ones that are not in the positions I_{max} and then adding $n - |I_{\text{max}}|$ ones.

Consider the set of matrices $(X^{(k)})_{0\leq k \leq H}$ with:

$$X^{(0)} = X$$
$$X^{(H)} = P \quad (12)$$
$$||X^{(k)} - X^{(k-1)}|| = 1$$

These matrices can be constructed if we start from P and erase successively all ones that are not in I_{max}. After that we can insert ones that have an index in common with an element in I_{max}. This set we call B.

Inserting or erasing a one at the j-th column yields maximally to an energy difference of

$$D_{\mathcal{C}_j} = \max_{k\in\mathcal{C}_j}(\sum_i |(W_{k,i}+W_{i,k})| + |W_{k,k}| + |c_k|), \quad (13)$$

where $\mathcal{C}_j$ describe the indizes that belong to the j-th collumn and analogously $\mathcal{R}_j$ describe the indizes that belong to the j-th row. We define f as:

$$f(Y) = \mathbf{y}^{\text{T}} W \mathbf{y} + \mathbf{c}^T \mathbf{y} \quad (14)$$

with $\mathbf{y} = \text{vec}(Y)$. To prove (10) we use the principle of a telescope sum:

$$f(P) - f(X)$$
$$= \sum_{k=0}^{H-1} f(X^{(k+1)}) - f(X^{(k)}) \leq \sum_{k=0}^{H-1} \left| f(X^{(k+1)}) - f(X^{(k)}) \right|$$
$$= \sum_{\{k\in\{0,\ldots,H-1\} | \text{pos}\,(X^{(k+1)} - X^{(k)}) \in B\}} \left| f(X^{(k+1)}) - f(X^{(k)}) \right|$$
$$+ \sum_{\{k\in\{0,\ldots,H-1\} | \text{pos}\,(X^{(k+1)} - X^{(k)}) \notin B\}} \left| f(X^{(k+1)}) - f(X^{(k)}) \right| \quad (15)$$

For the second sum we want to use:

$$\left| f(X^{(k+1)}) - f(X^{(k)}) \right| \leq D_{\{1,\ldots,n^2\}} \quad (16)$$

In the first sum we can make use of the specific column or row and use $D_{\mathcal{J}_j}$. This yields the following ansatz for $\lambda_{\mathcal{R}_j}$, which is the λ_k that belongs to the j-th row:

$$\lambda_{\mathcal{R}_j} = \alpha D_{\mathcal{R}_j} + \beta D_{\{1,\ldots,n^2\}}. \quad (17)$$

To obtain the constant β we have to calculate the right side of the inequality:

$$\beta \geq \max_{\{\mathbf{x}\in\{0,1\}^{n^2} | A\mathbf{x}\neq\mathbf{b}\}} \frac{|n - I_{\text{max}}|}{\sum_j |A\mathbf{x}_j - b_j|}. \quad (18)$$

If we have a row, where no element of I_{max} is present, there can be other ones placed there that share a column with a position in I_{max}. If there is no one in that column, then $|A\mathbf{x}_j - b_j| \neq 0$ for the corresponding j. If there are ones then $\sum_j |A\mathbf{x}_j - b_j|$ also increases, since the ones are in places, where they share a column with a position in I_{max}. This yields to the inequality:

$$\beta \geq \max_{\{\mathbf{x}\in\{0,1\}^{n^2} | A\mathbf{x}\neq\mathbf{b}\}} \frac{|n - I_{\text{max}}|}{2 \cdot |n - I_{\text{max}}|} = \frac{1}{2}. \quad (19)$$

To get an estimate for α we now consider a row that contains an element of I_{max}. In the case, that there is only a single one we do not have to delete other ones. Every additional one increases $|A\mathbf{x}_j - b_j|$ also by one. Therefore:

$$\alpha \geq \max_{j,\mathbf{x}\in\{0,1\}^{n^2} | (A\mathbf{x})_j \neq b_j \quad \mathcal{J}_j \cap \text{vec}(I_{\text{max}}) \neq \varnothing} \frac{|\text{vec}(B \setminus I_{\text{max}}) \cap \mathcal{J}_j|}{\left| (A\mathbf{x})_j - b_j \right|} = 1. \quad (20)$$

1.2. Proofs of Lemma (4.1) and Proposition (4.2)

1.2.1 Proof of Lemma (4.1)

Proof. We express the set of permutation matrices in terms of the coefficients $x_{i,j}$:

$$
\begin{aligned}
\mathbb{P}_n = \{X \in \mathbb{R}^{n\times n} | &\forall i,j \in \{1,...,n\} x_{i,j} \in \{0,1\} \\
&\forall j \in \{1,...,n\} \sum_{i=1}^{n} x_{i,j} = \sum_{i=1}^{n} x_{j,i} = 1\} \\
&= \Bigg\{ \begin{pmatrix} 1-\sum_{i=2}^n x_{1,i} & 1-\sum_{i=2}^n x_{2,i} & \dots & 1-\sum_{i=2}^n x_{n,i} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,n} \end{pmatrix} \\
&\in \mathbb{R}^{n\times n} | \forall i \in \{2,...,n\}, j \in \{1,...,n\}\ x_{i,j} \in \{0,1\} \\
&\forall j \in \{1,...,n\} \sum_{i=1}^{n} x_{j,i} = 1 \wedge \sum_{i=2}^{n} x_{i,j} \le 1 \Big\} \\
&= \Bigg\{ \begin{pmatrix} 1-\sum_{i=2}^n x_{1,i} & 1-\sum_{i=2}^n x_{2,i} & \dots & 1-\sum_{i=2}^n x_{n,i} \\ 1-\sum_{i=2}^n x_{2,i} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 1-\sum_{i=2}^n x_{n,i} & x_{n,2} & \cdots & x_{n,n} \end{pmatrix} \\
&\in \mathbb{R}^{n\times n} | \forall i \in \{2,...,n\}, j \in \{1,...,n\}\ x_{i,j} \in \{0,1\} \\
&\forall j \in \{1,...,n\} \sum_{i=1}^{n} x_{j,i} = 1 \wedge \sum_{i=2}^{n} x_{i,j} \le 1 \Big\} \\
&= \Bigg\{ \begin{pmatrix} 1-(n-1)+\sum_{i,j=2}^n x_{i,j} & 1-\sum_{i=2}^n x_{2,i} & \dots & 1-\sum_{i=2}^n x_{n,i} \\ 1-\sum_{i=2}^n x_{2,i} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 1-\sum_{i=2}^n x_{n,i} & x_{n,2} & \cdots & x_{n,n} \end{pmatrix} \\
&\in \mathbb{R}^{n\times n} | \forall i \in \{2,...,n\}, j \in \{2,...,n\}\ x_{i,j} \in \{0,1\} \\
&\sum_{i,j=2}^{n} x_{i,j} \in \{n-2, n-1\}\ \forall j \in \{1,...,n\} \\
&\sum_{i=2}^{n} x_{j,i} \le 1 \wedge \sum_{i=2}^{n} x_{i,j} \le 1 \Big\}
\end{aligned}
$$

$$
\begin{aligned}
&= \Bigg\{ \begin{pmatrix} 2-n+\sum_{i,j=2}^n x_{i,j} & 1-\sum_{i=2}^n x_{2,i} & \dots & 1-\sum_{i=2}^n x_{n,i} \\ 1-\sum_{i=2}^n x_{2,i} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 1-\sum_{i=2}^n x_{n,i} & x_{n,2} & \cdots & x_{n,n} \end{pmatrix} \\
&\in \mathbb{R}^{n\times n} | \forall i \in \{2,...,n\}, j \in \{2,...,n\}\ x_{i,j} \in \{0,1\} \\
&\sum_{i,j=2}^{n} x_{i,j} \in \{n-2, n-1\}\ \forall j \in \{2,...,n\} \sum_{i=2}^{n} x_{j,i} \le 1 \wedge \sum_{i=2}^{n} x_{i,j} \le 1 \Big\} \\
&= \Bigg\{ \begin{pmatrix} 2-n+\sum_{i,j=2}^n x_{i,j} & 1-\sum_{i=2}^n x_{2,i} & \dots & 1-\sum_{i=2}^n x_{n,i} \\ 1-\sum_{i=2}^n x_{2,i} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 1-\sum_{i=2}^n x_{n,i} & x_{n,2} & \cdots & x_{n,n} \end{pmatrix} \\
&\in \mathbb{R}^{n\times n} | \forall i \in \{2,...,n\}, j \in \{2,...,n\}\ x_{i,j} \in \{0,1\} \\
&\sum_{i,j=2}^{n} x_{i,j} \in \{n-2, n-1\}\ \forall j,i,k \in \{2,...,n\} \quad i \neq k \\
&x_{j,i}x_{j,k} = 0 \wedge x_{i,j}x_{k,j} = 0 \Big\}
\end{aligned}
$$

□

1.2.2 Proof of Proposition (4.2)

We want to find sufficient lower bounds for λ_1^j and λ_2 in

$$
\begin{aligned}
\min_{\mathbf{x}\in\{0,1\}^{(n-1)^2}} \quad & \mathbf{x}^{\mathrm{T}}\tilde{W}\mathbf{x} + \tilde{c}^T\mathbf{x} + \\
& \sum_{j=1}^{2(n-1)} \lambda_1^j \left(\sum_{k\in\mathcal{J}_j} x_k\right)\left(\sum_{k\in\mathcal{J}_j} x_k - 1\right) \\
& + \lambda_2 \left(\sum_{i=1}^{(n-1)^2} x_i - (n-1)\right)\left(\sum_{i=1}^{(n-1)^2} x_i - (n-2)\right),
\end{aligned} \tag{21}
$$

so that it coincides with the constrained optimization problem. We make the following ansatz for the λ_2 parameter:

$$
\lambda_2 = \frac{1}{2} D_{\{1,\dots,(n-1)^2\}} \tag{22}
$$

Since the function

$$
\left(\sum_{i=1}^{(n-1)^2} x_i - (n-1)\right)\left(\sum_{i=1}^{(n-1)^2} x_i - (n-2)\right) \tag{23}
$$

increases faster than $2 \cdot H$,when H is the number of entries that need to be switched in order to have $n-1$- or $n-2$-many entries equal to 1.

This choice for λ_2 allows us to only investigate the case, where we have $n-1$ or n ones placed anywhere. Similar to the prior proof we ask, what the worst ratio between ones we have to insert anywhere and ones we have to insert or delete in a particular column is. It can be easily seen that in the worst case scenario all ones are in one column or row. Therefore the ratio is one half. This shows that we can choose:

$$
\lambda_1^j = \frac{1}{2} D_{\mathcal{J}_j} + \frac{1}{2} D_{\{1,\dots,(n-1)^2\}} \tag{24}
$$

□

2. Experiments on IBM Burlington

Several of our experiments are conducted on the IBM five qubits in Burlington [1], which is accessible per cloud. In contrast to D-Wave quantum annealers, the circuit-model is used for the design of this quantum computer. From a purely theoretical perspective there is some equivalence [3] between adiabatic quantum computing and quantum computing with the circuit model, though due to the technical challenges, significant differences remain in practice. One way to execute a time-dependent Hamiltonian $H(t)$ (6) on the IBM machine is to approximate it with L constant Hamiltonians. The Hamiltonians, which are constant in time, can be decomposed and expressed with quantum gates via *trotterization* [7].

More specifically, choosing the value of the parameter L in the piecewise constant approximation of the Hamiltonian is crucial as we loose the (pseudo-)adiabacity for small L, but – as each gate has a certain chance of introducing an error and a long execution time of the circuit makes errors due to decoherence more probable – large L are also prone to fail.

Because of the above-mentioned reasons, we are able to perform the adiabatic evolution for only two qubits on the 5 qubit processor until now. Fortunately, using the trick from Sec. 4.3, we can optimise over the 2×2 permutation matrices. In our experiment, we performed 50 time steps. We choose the evolution time 0.1 and use Suzuki expansion of order 2, similar to Kraus *et al.* [5]. The matrix W and the vector c were generated randomly for every of the 20 iterations. Every circuit was executed 8192 times, which is the maximal amount. An exemplary result of an execution is summarized in the histogram in Fig. 1. Although we obtain slightly different distributions each time, the highest peak always coincides with the second column of the optimal permutation, which is $|01\rangle$ here.

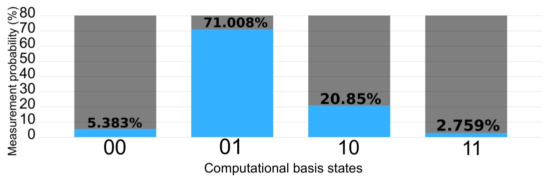

Figure 1: QGM execution histogram over 8192 runs using trotterization on a five qubit processor from IBM Quantum Experience. The states $|00\rangle$ and $|11\rangle$ are suppressed, because of the proposed regularization terms, *i.e.,* since these states violate that the sum of all elements in a column of a permutation matrix equals to 1.

The recent publication [4] about adiabatic quantum computing on machines from IBM Quantum Experience proposes catalyst Hamiltonian, which could improve the results in future and make it possible to succeed in higher dimensions.

3. Point Cloud Matching Example

To illustrate another application of the studied matching problems, we consider the registration of two 3D point clouds by finding correspondences of four pre-selected points in each scene as illustrated in Fig. 2. While this preselection is, of course, a very challenging part of the overall solution, our main goal is to illustrate the solution of a (low dimensional) matching problem via quantum computing – possibly as the solution to a subproblem in an iterative registration algorithm.

To set up the matching problem, we select four arbitrary points in one scene and use the ground truth transformation between the two frames to identify the corresponding points in the other scene. We then set up a matching problem with costs as in (17) by simply using Euclidean distances between the points.

The histogram of energies obtained by all three quantum graph matching formulations over 500 anneals is shown in Fig. 3. The top row illustrates the overall histogram with strong peaks at low energies. As these peaks correspond to permutation matrices, we can conclude that all penalty terms were successful in strongly promoting permutations, with the inserted formulation yielding the best results. Zooming into the leftmost peak (illustrated in the bottom row of Fig. 3), however, reveals that none of the three formulations was successful in consistently predicting the global optimum among the permutation matrices. Considering the probabilities of less than 1% for the row-wise and baseline, and about 5.5% for the inserted formulations to predict the ground truth solution, one must conclude that all algorithms do not provide significantly better solutions than random guessing (which has a success probability of 4.17% for $n = 4$).

In summary, this experiment underlines the great difficulty current quantum hardware still has with problem instances of $n = 4$, *i.e.,* looking for the values of the best 16 qubits that are constrained to representing a permutation matrix. The performance of our three QUBO formulations on this problem can be seen in Fig. 3.

4. Near-Isometric Matching

Fig. 4 shows an example of a matching problem between two shapes that are only approximately related by an isometry, *i.e.,* matching a wolf to a cat in this particular instance. Still modelling the problem as an isometric deformation using costs determined by (17) yields an instance, where the true matching still is the global optimum of the quadratic assignment problem, but where wrong permutations have considerably more similar energies than in the point cloud and isometric shape matching examples.

The success probabilities for the *inserted*, the *baseline* and the *row-wise* QGM variants are 3.8%, 4% and 9.2%, respectively. As we can see in Fig. 5, the energy landscape has a wide range and the energy values corresponding to permutations are closer to each other compared to the case when the isometry assumption is strictly fulfilled. If one looks at the histograms with only permutations, only the *row-wise* QGM shows some trend towards the permutation with the lowest energy. Although its success probability is more than twice as good as random guessing, such a factor to random guessing would still not be sufficient to scale such an algorithm to large n.

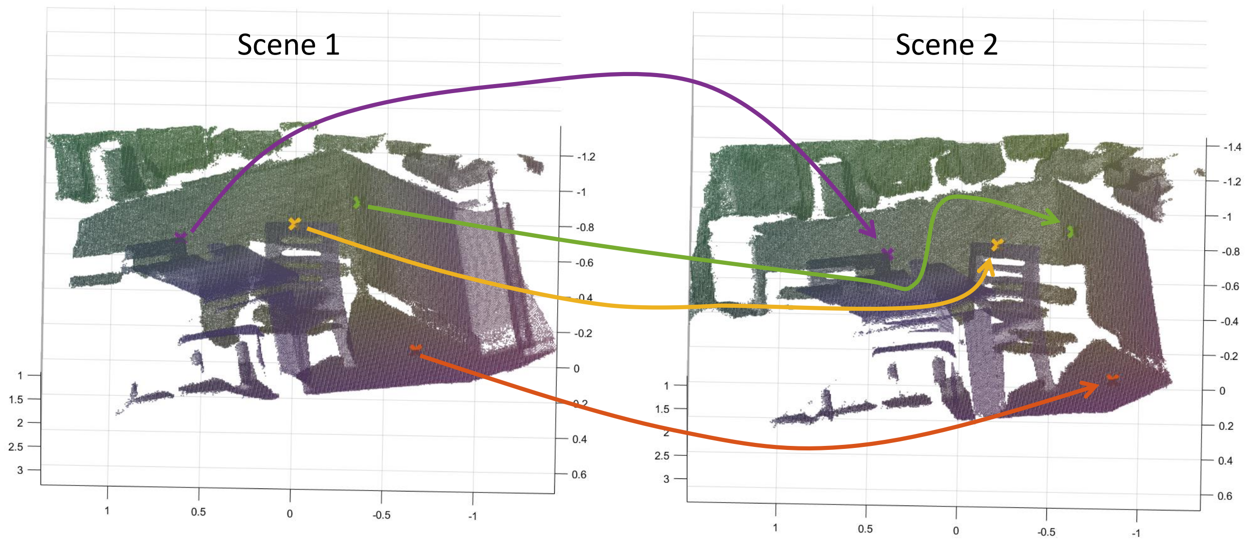

Figure 2: Illustrating the correct matching between four selected keypoints in two frames of the 7-Scenes 'Redkitchen' dataset, as provided at https://3dmatch.cs.princeton.edu/

5. Embedding to the Chimera Graph

The embeddings to the Chimera graph for different dimensions with the row-wise formulation can be seen in Fig. 6. While the number of logical qubits grows quadratically with n, the number of physical qubits required to embed those to the Chimera graph grows as n^4. For $n = 2$, the length of the longest chain is two physical qubits. For $n = 3$ and $n = 4$, the chain length does not exceed four and six, respectively.

6. Discussion of Negative Results

Since physical quantum computing is an emerging technology, reporting and discussing negative results on the early stage is of high relevance for the community. Insights of this section can help in choosing a promising direction for improvements and future research.

As discussed in section 5.2 we claim that the failure to provide the ground state with more probability than random guessing is due to the experimental errors in the coupling parameters. To back this up we provide the smallest and largest values of couplings and biases of the regularization and data term in table 1.

The couplings and the biases are scaled, so that they fit the feasible region of the annealer. $Q_{\text{reg}}, q_{\text{reg}}$ do yield constant energies for all permutations and therefore $Q_{\text{prob}}, q_{\text{prob}}$ contains the information, which permutation is optimal. The sum are the real, physical couplings

$$Q = Q_{\text{reg}} + Q_{\text{prob}}$$

Table 1: Illustrating the range of values arising from the penalty to constrain each formulation to permutation matrices ($Q_{\text{reg}\,*}$ and $q_{\text{reg}\,*}$) and from the actual problem costs ($Q_{\text{prob}\,*}$ and $q_{\text{prob}\,*}$). As we can see the constraints contribute more to the quadratic coupling matrix by a factor of around 6 for the inserted, 13 for the row-wise, and almost 55 for the baseline.

	Row-wise	Inserted	Baseline
Q_{max}	-0.887	-0.987	-0.973
Q_{min}	-1.037	-1.295	-1.009
q_{max}	-1.937	-8.884	-116.772
q_{min}	-2.277	-9.753	-120.772
$Q_{\text{reg max}}$	0.962	1.121	0.983
$Q_{\text{reg min}}$	-0.962	-0.705	-0.991
$q_{\text{reg max}}$	4.017	10.885	118.772
$q_{\text{reg min}}$	0.140	7.753	118.772

and biases

$$q = q_{\text{reg}} + q_{\text{prob}}.$$

One can see that most of the accessible range of the coupling parameters has to make sure that the output is a permutation.

There exists already ways to deal with these problems, which we want to try out in further experiments. One possibility would be to use extended J-range parameter [2]. The easiest way for this would be to use the virtual graph embedding instead of the default one (EmbeddingComposite). First attempts in this direction show that using the virtual graph embedding requires a lot of computation time. For

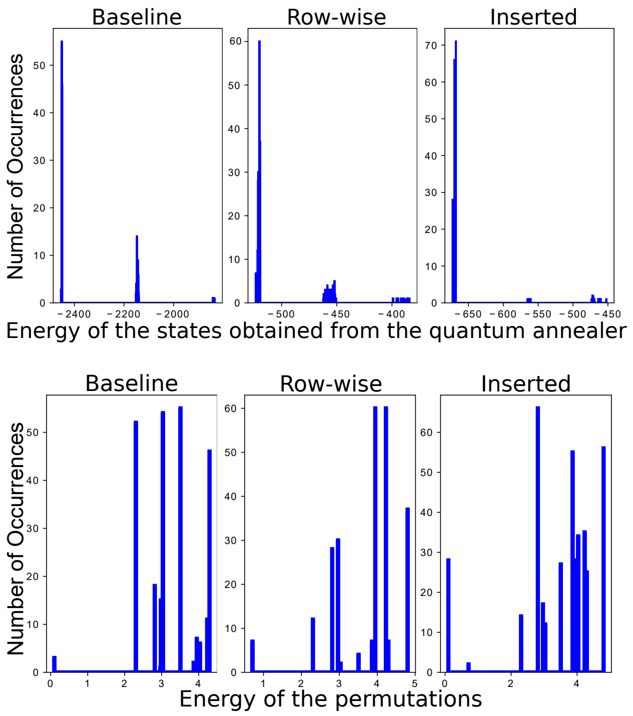

Figure 3: The histograms show the states obtained for the point matching problem in Fig. 2. The top row shows the overall histogram obtained over 500 anneals while the bottom row is a (rescaled) zoom into the leftmost peaks of the upper row, which corresponds to actual permutation matrices.

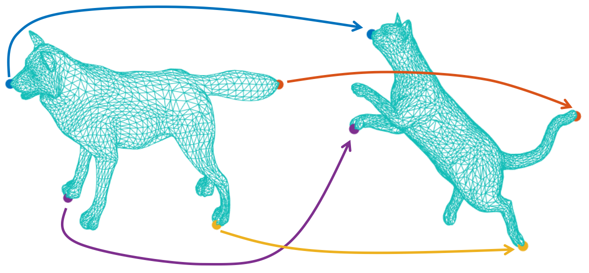

Figure 4: Illustrating a matching problem in which the shapes to be matched are only approximately related by an isometry.

one instance, where we got an error warning we had to invest 12% of our access time.

As reported in the main paper we first used a long annealing path with a break and at some point switched to using $20 \mu s$. Although a longer annealing time can often be used to enhance the success probability, if we additionally look a the time it takes to perform the experiment until we

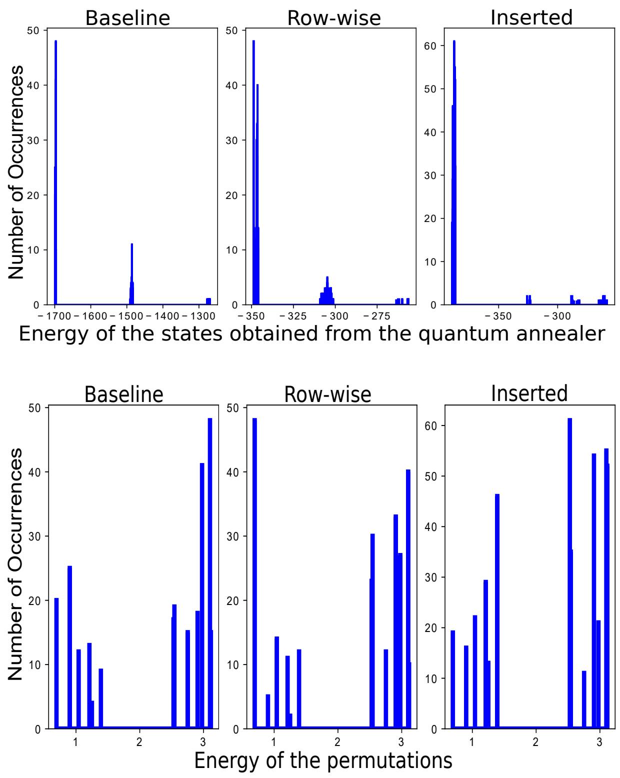

Figure 5: The histograms show the states obtained for the point matching problem in Fig. 4. The top row shows the overall histogram obtained over 500 anneals while the bottom row is a (rescaled) zoom into the leftmost peaks of the upper row, which corresponds to actual permutation matrices.

one gets the optimum with for example 99% certainty then according to [6] $20 \mu s$ seems to be the better choice, if less than 512 qubits are used.

7. Beyond Quantum Computing

In addition to the numerical experiments using quantum computing, we also briefly tested the effect of our three reformulations on methods that are inspired by physical systems. In Fig. 7, we compare the success probability of our three formulations for simulated annealing using random instances of graph matching problems. As we can see, the row-wise and - even more so - the inserted formulations yield results clearly superior to the baseline method, indicating that our analysis might be of use beyond quantum computing. Due to the $\mathcal{NP}$-hard nature of the underlying problem, it is to be expected that the overall success probability still decreases exponentially with increasing n (for a fixed number of simulated annealing runs).

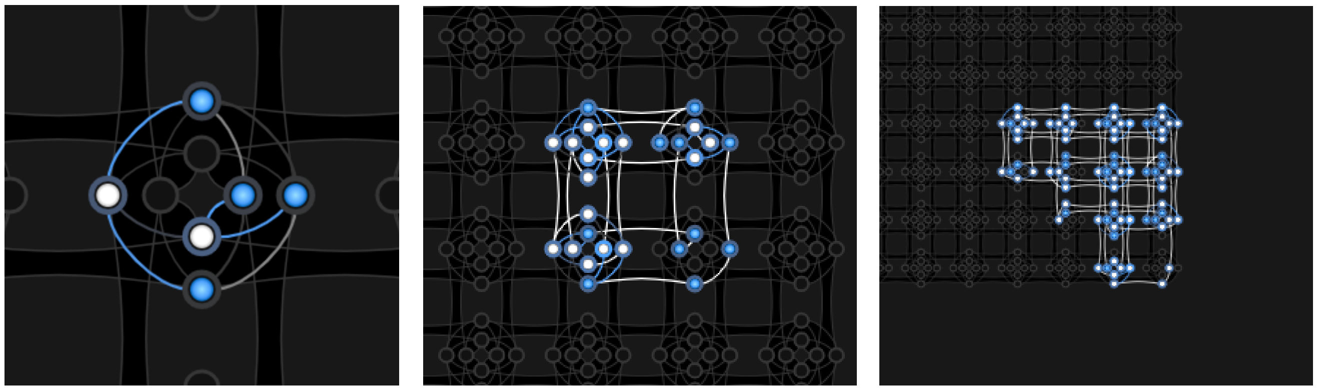

Figure 6: QGM minor embeddings to the chimera graph, for $n = 2$ (left), $n = 3$ (middle) and $n = 4$ (right). The number of logical qubits grows quadratically with n, and the number of physical qubits required to embed the logical qubits to the Chimera graph grows as n^4 on the current generation of D-Wave annealers. The white and blue circles denote measured values zero and one, respectively, and the lines between the qubits denote couplers. Grey lines connect the physical qubits with the same resulting values. A subset of those build chains and represent a single logical qubit. Blue lines connect different measured values.

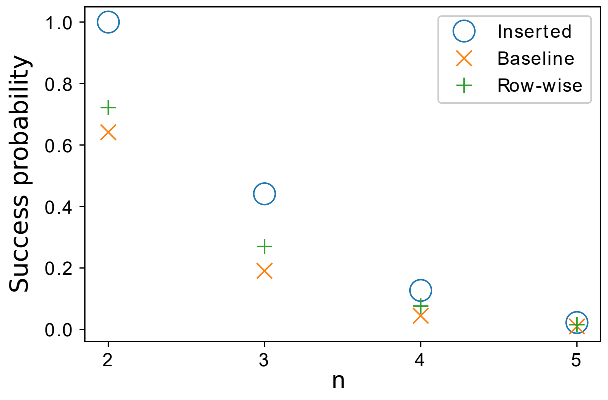

Figure 7: Percentage of the 5000 runs that found the optimum with simulated annealing averaged over 10 random problem instances.

References

[1] IBM Q experience. https://quantum-computing.ibm.com/ http://www.research.ibm.com/quantum. Accessed: 2020 January and February. 3

[2] Performance tuning for d-wave quantum processors. https://www.dwavesys.com/sites/default/files/2_Wed_Am_PerfTips.pdf. Accessed: 2020 July. 5

[3] D. Aharonov et al. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008. 3

[4] C. Cao, J. Xue, N. Shannon, and R. Joynt. Speedup of the quantum adiabatic algorithm using delocalization catalysis. *arXiv preprint arXiv:2007.11212*, 2020. 4

[5] M. Hebenstreit, D. Alsina, J. I. Latorre, and B. Kraus. Compressed quantum computation using a remote five-qubit quantum computer. *Phys. Rev. A*, 95:052339, May 2017. 4

[6] A. D. King and C. C. McGeoch. Algorithm engineering for a quantum annealing platform. *arXiv preprint arXiv:1410.2628*, 2014. 6

[7] M. A. Nielsen and I. Chuang. Quantum computation and quantum information, 2002. 3