Supplementary Material for
Q-Match: Iterative Shape Matching via Quantum Annealing

Marcel Seelbach Benkner[1] Zorah Lähner[1] Vladislav Golyanik[2]
Christof Wunderlich[1] Christian Theobalt[2] Michael Moeller[1]

[1]University of Siegen [2]MPI for Informatics, SIC

This supplementary material provides a deeper analysis of the proposed Q-Match approach and more experimental details. This includes:

- Further analysis of the solution quality for individual QUBOs and visualizations of the minor embeddings (Sec. A).
- Derivation of Eq. (6) and proof of Lemma 4.1 from the main matter (Sec. B).
- A toy example, in which all 2-cycles individually do not improve the energy (Sec. C).
- More details on calculating W_s and $\tilde{W}$ (Sec. D).
- A list of our solutions to selected QAPLIB problems *vs* ground truth (Sec. E).

A. Analysis of the Individual QUBOs

We analyze the quality of the solution of the individual QUBOs in dependence of the dimension. The success probability for one QUBO solution is defined as the fraction between the anneals that ended up in the optimum and the number of anneals. The success probability averaged over 20 QUBOs per dimension at the first two iterations, *i.e.,* computations of (5) for a set of 2-cycles, of one instance of the FAUST dataset can be seen in Fig. 1. We see that for increasing dimension the success probability is decreasing and less runs end up in the ground state. One possible way to reverse this trend would be to increase the annealing time, which we left constant at $T_A = 20\mu s$. We also plotted the fraction of QUBOs, where the ground state was among the returned solutions. Leaving the number of anneals constant this probability declined from 1 for $4-24$ worst vertices to 0.4 for 40 worst vertices. To get the optimum for more instances we performed the experiment with 5000 anneals for 40 and 50 worst vertices. In this experiment we found the optimum in 90% or 45% of the cases, respectively.

Note that quantum annealing is a stochastic algorithm. Therefore a success probability P_s is directly linked with the amount of time needed to to get the optimum with, *e.g.,*

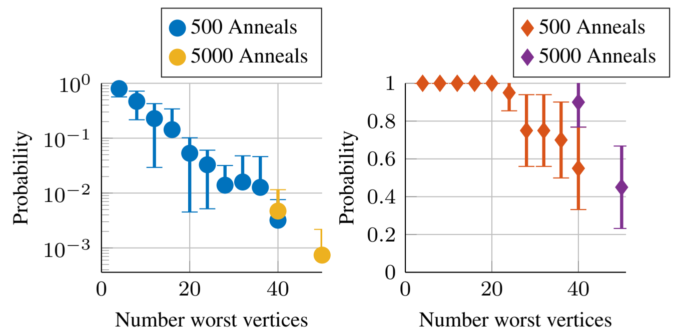

Figure 1. Success probability (left) and the fraction of executions where the best solution is the optimum (right), for different problem sizes and number of anneals. The success probability decreases with the problem size, and, therefore, more anneals are necessary. In the left plot the deviation bar is the standard deviation and in the right plot it is the binomial proportion confidence interval. No lower part of the deviation bar means it goes beyond zero in the left plot.

99% probability:

$$T = \frac{\ln(1-99\%)}{\ln(1-P_s)} T_A, \tag{1}$$

where T_A is the time for one anneal. We also computed the binomial proportion confidence interval for the probability that the optimum of the QUBO is found, with at least one anneal. If the underlying distribution is binomial, then the true probability lies within 20% of our estimates in 95% of cases. If the number of worst vertices is 23 or less the estimate is even closer to the true probability. The binomial confidence interval can be seen in Fig. 1, on the right. In the left plot of Fig. 1 the standard deviation of the success probability for averaging over the 20 individual QUBOs is also depicted. This shows that the success probability strongly varies for the distinct QUBO instances.

A.1. Minor Embedding

In most cases, our problems result in fully connected logical qubit graphs. In Figs. 2 and 3, there are illustrations of the minor embeddings computed by the method of Cai *et al.* [3] (used in Leap 2) and visualized by the problem inspector of Leap 2 [4]. We plot the average maximum chain lengths and average numbers of physical qubits in the obtained minor embeddings in Fig. 4.

B. Derivations and Proofs

B.1. Derivation of Eq. (6) in Sec. 4.1

It is stated in the main paper in Eq. (6), that one can convert the multiplication of cycles from (5) into an additive structure, *i.e.,* that

$$P(\alpha) = \left(\prod_i c_i^{\alpha_i}\right) P_0 = P_0 + \sum_{i=1}^{m} \alpha_i (c_i - I) P_0$$

holds, where I is the identity.

Proof. Consider the case where we only have a single cycle c. Now, the following holds:

$$\begin{aligned} P(\alpha) &= P_0 + \alpha(c - I)P_0 = (1-\alpha)P_0 + \alpha c P_0 \\ &= \begin{cases} P_0, & \text{for } \alpha = 0 \\ cP_0, & \text{for } \alpha = 1 \end{cases} = c^\alpha P_0. \end{aligned}$$

Independent of P_0, we can write:

$$c^\alpha = (1-\alpha)I + \alpha c. \tag{2}$$

Additionally we can write (6) independent of P_0 by applying the inverse permutation from the right side:

$$\prod_i c_i^{\alpha_i} = I + \sum_{i=1}^{m} \alpha_i (c_i - I).$$

Now as an induction step we apply $c_{m+1}^{\alpha_{m+1}}$ from the right:

$$\begin{aligned} & c_{m+1}^{\alpha_{m+1}} \left(\prod_i c_i^{\alpha_i}\right) \\ &= c_{m+1}^{\alpha_{m+1}} (I + \sum_{i=1}^{m} \alpha_i (c_i - I)) \\ &= ((1-\alpha_{m+1})I + \alpha_{m+1} c_{m+1})(I + \sum_{i=1}^{m} \alpha_i (c_i - I)) \\ &= I + \sum_{i=1}^{m+1} \alpha_i (c_i - I) + \alpha_{m+1}(c_{m+1} - I) \sum_{i=1}^{m} \alpha_i (c_i - I) \end{aligned} \tag{3}$$

We want to use that for two disjoint cycles c_k, c_l, the equality $(c_k - I)(c_l - I) = 0$ holds. In all the places where c_k has 0 on the diagonal, c_l has 1, because they are disjoint. This leads to the fact that in the rows, where $(c_k - I)$ is non-zero, $(c_l - I)$ has zero columns or rows, respectively. Therefore the last term in (3) vanishes and the statement is proven. □

B.2. Proof of Lemma 4.1

To prove Lemma 4.1, we first show that the statement is correct for a k-cycle.

Lemma B.1. *Let P be a k-cycle. Then, P can be written as a product of Q and R, where Q and R are permutations that only consist of disjoint 2-cycles.*

Proof. Without loss of generality, let $P = (1\ 2\ 3 \ldots k)$ be the k-cycle. This is possible because rearranging rows and columns does not change the problem. For even k, Q and R take the following form:

$$\begin{aligned} Q &= (1\ 2)(3\ k)(4\ (k-1)) \ldots \left(\left(1 + \frac{k}{2}\right)\ \left(2 + \frac{k}{2}\right)\right), \\ R &= (2\ k)(3\ (k-1)) \ldots \left(\frac{k}{2}\ \left(2 + \frac{k}{2}\right)\right). \end{aligned}$$

It can be easily checked that $P = QR$ holds in this case. For uneven k, one can choose the following Q and R, and the same holds:

$$\begin{aligned} Q &= (1\ 2)(3\ k)(4\ (k-1)) \ldots \left(\left(\frac{k+1}{2}\right)\ \left(1 + \frac{k+1}{2}\right)\right), \\ R &= (2\ k)(3\ (k-1)) \ldots \left(\frac{k-1}{2}\ \left(1 + \frac{k+1}{2}\right)\right). \end{aligned}$$ □

Next, the following argument gives the proof for Lemma 4.1.

Proof. One can first write the permutation P in cycle notation. Then, we decompose each cycle individually as it was shown in Lemma B.1. Note that according to Lemma B.1, the decomposition of the k-cycle does not require additional elements in Q or R that do not occur in the cycle. □

Permutations like Q or R that only consist of 2-cycles are called involutions. The fraction of permutations that are involutions has following asymptotic behavior [9] (Prop. VIII.2.):

$$\frac{I_n}{n!} = \frac{e^{-\frac{1}{4}}}{2\sqrt{\pi n}} n^{-\frac{n}{2}} e^{\frac{n}{2} + \sqrt{n}} \left(1 + O\left(\frac{1}{n^{\frac{1}{5}}}\right)\right). \tag{4}$$

Considering Lemma B.1, a rough estimate is $I_n^2 > n!$. For these permutations, in theory, one step of the cyclic α-expansion would suffice to obtain to the identity (which could be the optimum w.l.o.g.).

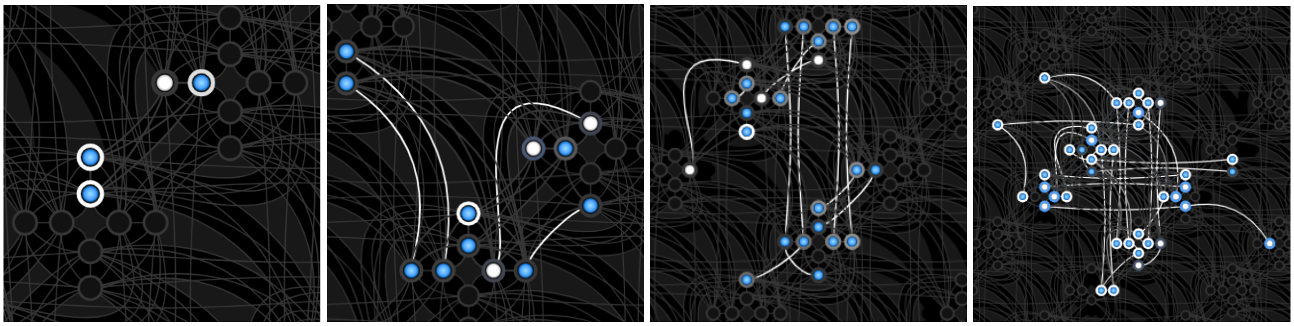

Figure 2. Illustration of the embedding from the D-Wave Leap 2 problem inspector [4] for using $8, 16, 24, 32$ worst vertices. One node depicts a physical qubits. The inner color shows the measured value in the lowest energy state, while the color of the outer ring shows the sign of the bias, *e.g.,* the coefficient of the linear term in the optimization problem (2). The edges depict the chains.

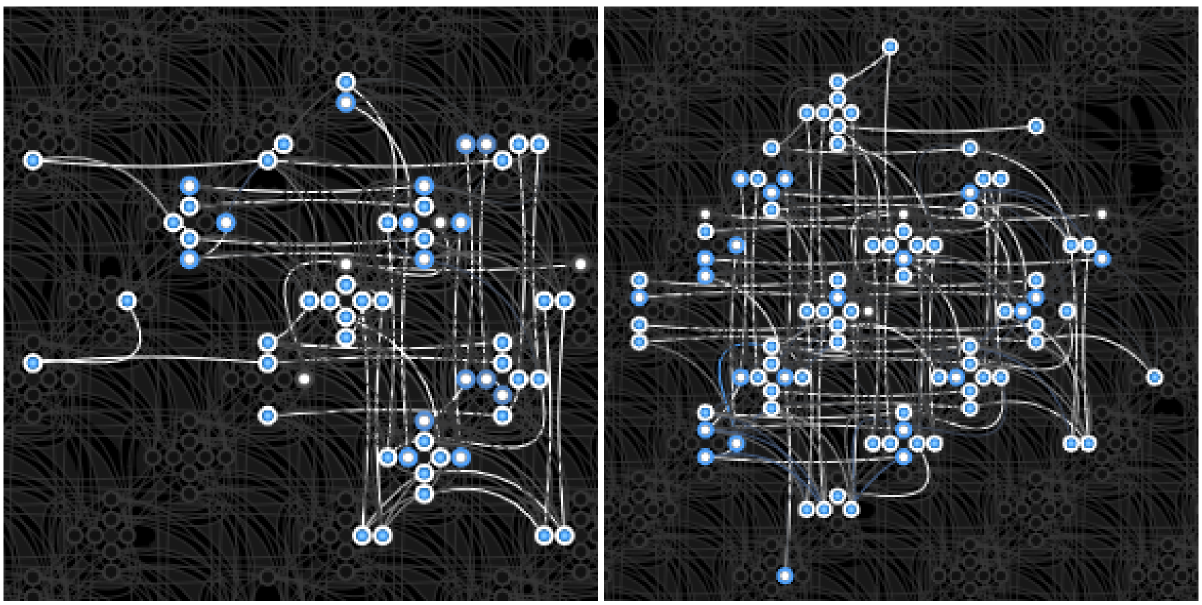

Figure 3. Illustration of the embedding from the D-Wave Leap 2 problem inspector [4] for using 40 and 50 worst vertices. One node depicts a physical qubits. The inner color shows the measured value in the lowest energy state, while the color of the outer ring shows the sign of the bias, *e.g.,* the coefficient of the linear term in the optimization problem (2). The edges depict the chains.

Note on Classical Optimization: If (8) was submodular, it would be possible to efficiently solve it by converting it to a graph cut problem instead of using AQCing. According to Bach [1], a quadratic function $f(\mathbf{x}) := \mathbf{x}^T W \mathbf{x}$ is submodular, if and only if all non-diagonal elements of W are non-positive. Since $\tilde{W}_{ij}$ can have a positive sign, the function is not submodular and, therefore, cannot be efficiently optimized. We tried small alternations of $\tilde{W}$ (*e.g.*, changing the sign of the off-diagonal elements by switching C_i to $-C_i$), but did not succeed in making (8) submodular.

B.3. Simulated Annealing vs. Quantum Annealing for the Subproblems

We also performed Q-Match from Alg. 1 with a simulated annealing solver from [5] for the subproblems. The quality of the results for the FAUST dataset and for QAPLIB can be seen in Figs. 7 and 8. Here we executed the simulated annealing sampler with 5 sweeps and performed 100 runs. Increasing the number of sweeps further did not yield qualitatively different results.

In Fig. 5 the processing time for the subproblems is plotted in dependence of the dimension. If one measures the wall clock time for a query to the D-Wave sampler in the same way one gets results of the order of seconds. This is due to the fact that the time the solver takes for the computation is overshadowed by the time to connect and get access to the machine. To get this time estimate one can simply look at `dwave ping` in ocean. For the QPU access time we already presented the results in dependence of dimension in Fig. 6. A more detailed overview of the different

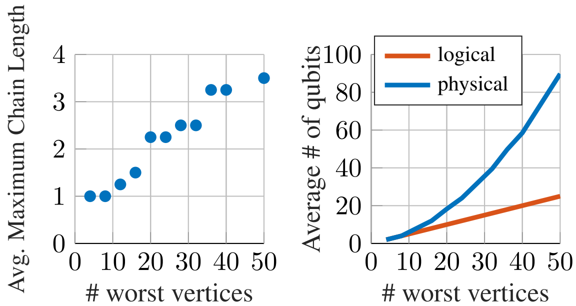

Figure 4. Average maximal chain length and number of physical qubits for increasing problem size averaged over 4 instances. The number of logical qubits increases linearly with the problem size.

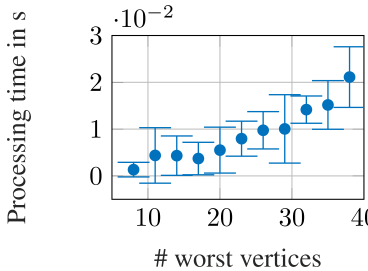

Figure 5. Processing time of the simulated annealing sampler in dependence of the dimension. For one execution we have 100 runs and 5 sweeps as parameters. We averaged the measured time over 10 executions.

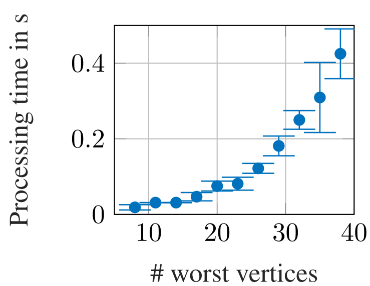

Figure 6. Time to compute the minor embedding in dependence of the dimension of the subproplem. The subproblems stem from the Q-Match algorithm applied to the Faust dataset. We averaged the measured time over 5 executions.

runtimes is given in [6].

Since the minor embedding is computed locally one can directly measure the processing time in dependence of the dimension. In Fig. 6 this measurement is averaged over 5 instances. The simulated annealing sampler takes here already an order of magnitude less time. However since we mostly want to embed a fully connected graph the problem to find an embedding could be solved beforehand and an existing embedding can be reused.

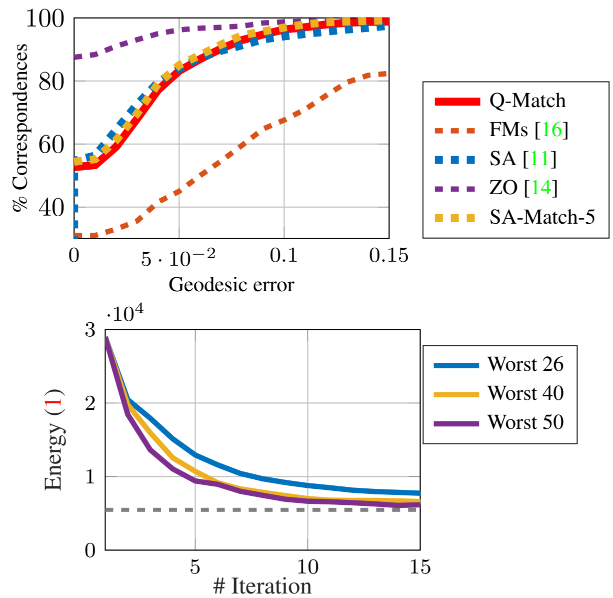

Figure 7. Evaluation of cumulative error [13] (left) and convergence (right) on the FAUST dataset. (Left) We compare against Simulated Annealing [11] without postprocessing and Functional Maps [16]. Dashed lines indicate non-quantum approaches. The results have symmetry-flipped solutions removed, these have an equivalent final energy for all three methods but are not recognized as correct in the evaluation protocol. (Right) We show the convergence of the energy over 30 iterations. The larger the set of worst vertices, the faster our method converges. The dashed grey line shows the optimal energy.

We also state that the emphasis of this work is not on benchmarking the subproblems. If one would do this one could also optimize over the annealing path and could use further features like the extended J-range for more precision and spin reversal transformations to mitigate some systematic errors. Additionally one would also apply formula (1). In this section we only want to provide a rough overview of the computing time. Preliminary work that does an in depth optimization of these algorithms confirms that quantum annealing is highly promising: In [7] QUBOs were found where quantum annealing yields a "time-to-99%- success-probability that is $\sim 10^8$ times faster than simulated annealing running on a single processor core". Reference [12] benchmarks the D-Wave 2000Q on a broader class of problems and confirms the potential of this technology.

C. Failure Case for Individually Optimizing Over the 2-Cycles

We present an example to proof that optimizing over larger sets of 2-cycles is superior to looking at single 2-cycles separately, as is done in [11]. We construct a plane

BO [2]:

	a	b	c	d	e	f	g	h
Optimum	5426670	3817852	5426795	3821225	5386879	3782044	10117172	7098658
Our (Quantum)	5450757	3828405	5485230	3822190	5403238	3797120	10158673	7152966
Our (Simulated Annealing)	5449653	3836716	5445272	3870067	5398333	3783329	10119061	7130826

ESC16 [8], HAD [10]:

	a	b	c	d	e	f	g	h	i	j	a	b	c	d	e
Optimum	68	292	160	16	28	0	26	996	14	8	1652	2724	3720	5358	6922
Our (Quantum)	70	292	160	16	28	0	26	996	14	8	1652	2748	3750	5358	6922
Our (Simulated Annealing)	68	292	160	16	28	0	26	996	14	10	1686	2730	3734	5376	7068

NUG [15]:

	a	b	c	d	e	f	g	h	i	j	k	l	m	n
Optimum	578	1014	1610	1240	1732	1930	2570	2438	3596	3488	3744	5234	5166	6124
Our (Quantum)	618	1026	1650	1296	1882	1936	2606	2574	3712	3632	4004	5550	5348	6352
Our (Simulated Annealing)	594	1076	1694	1322	1802	1976	2682	2516	3714	3630	3940	5508	5514	6480

SCR [18], Rou [17]:

	a	b	c	a	b	c
Optimum	31410	51140	110030	235528	354210	725522
Our (Quantum)	35454	58320	114322	251872	373218	754506
Our (Simulated Annealing)	33672	58606	115822	248982	384354	760738

Table 1. Our solutions for exemplary sets of the QAPLIB dataset with different sizes of quadratic assignment problems.

2D-shape where the optimum can be reached with a collection of 2-cycles but each 2-cycle applied individually results in a worse energy starting from a specific permutation. Consider the points depicted in Figure 9. If we chose a, b and ϵ, we can compute the energies of permutations with (9) using Euclidean distances between the points. Possible values would be $b = 10$, $a = 1$ and $\epsilon = 0.1$.

The shape is almost (but not exactly) symmetric with respect to mirroring along a shifted x-axis (Permutation (1 4)(2 5)(3 6)). In our experiment, the second shape is a copy of the first with permuted vertices, and we want to find the correspondence. Let the identity be the optimal solution, and the current permutation is $P_0 = (14)(25)(36)$. Permuting any of the three points on the upper $\{1, 2, 3\}$ to any of the lower points $\{4, 5, 6\}$ on the right causes – despite the correct assignment – a distortion of the (near-) isometry, such that no such 2-cycle improves the cost function when the assignment of the other points remains unchanged. However, applying all three correct 2-cycles at once, allows to pass to the global optimum with a lower energy.

This illustrates that using our cyclic α-expansion iteration step for optimizing over multiple 2-cycles at a time can have significant advantages over a sequence of simple single 2-cycle updates.

D. Calculation of W_s and $\tilde{W}$

Notice that for a given permutation P and a set of cycles C, it is possible to get $\tilde{W}$ without precomputing W_s in roughly the same time as computing W_s. However, if W_s is precomputed for a subset of vertices, $\tilde{W}$ can be computed very efficiently for any set of cycles on this subset. Therefore, we once calculate the expensive W_s, and then evaluate several sets of cycles on it to increase the overall efficiency.

D.1. Calculating W_s

If we want to solve a subproblem of (1) we assume that all correspondences for indices, that are not optimized, stay fixed. Therefore, it is not sufficient to set W_s to a submatrix of W, but we have to add the influence of these fixed correspondences. Given a set $s_M, s_N \subset \{1, \dots, n\}$ of indices

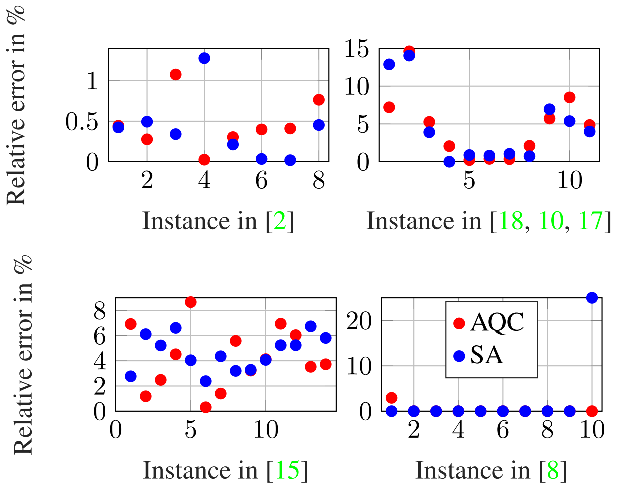

Figure 8. Relative error $\frac{E_{\text{obtained}} - E_{\text{opt}}}{E_{\text{opt}}}$ of our method in percentage for the instances of [2] (upper left), [18] (1-3), [10] (4-8) and [17] (9-11) (upper right), [15] (lower left) and [8] (lower right) in QAPLIB. The problem sizes range between 12 and 30, of which [15] contains the larger ones where we do less well.

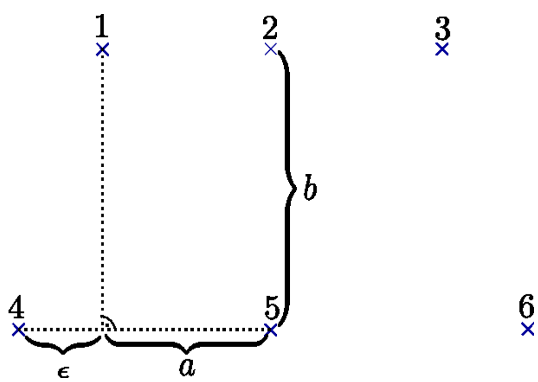

Figure 9. Exemplary shape to show that individually applying 2-cycles does not suffice. Because of the length ϵ the points are only nearly symmetric along an x-axis. The shape is invariant under the permutation $(13)(46)$.

which indicate the subproblem of W (in Q-Match, these are the sets I_M, I_N), and a previous permutation P, we calculate W_s as follows:

$$(W_s)_{ikjl} = W_{ikjl} + \sum_{(v_M, v_N) \in V} W_{ikv_Mv_N} + W_{v_Mv_Njl}. \tag{5}$$

Here $V \subset M \times N$ is the set of correspondences indicated by a permutation P, with removed all tuples in V which contain entries from s_M, s_N, *i.e.*, $(v_M, v_N) \in V$ if $P(v_M) = v_N$ and $v_M \notin v_M, v_N \notin v_N$. This results in a $k^2 \times k^2$ matrix where each entry contains the sum of $\mathcal{O}(|C|)$

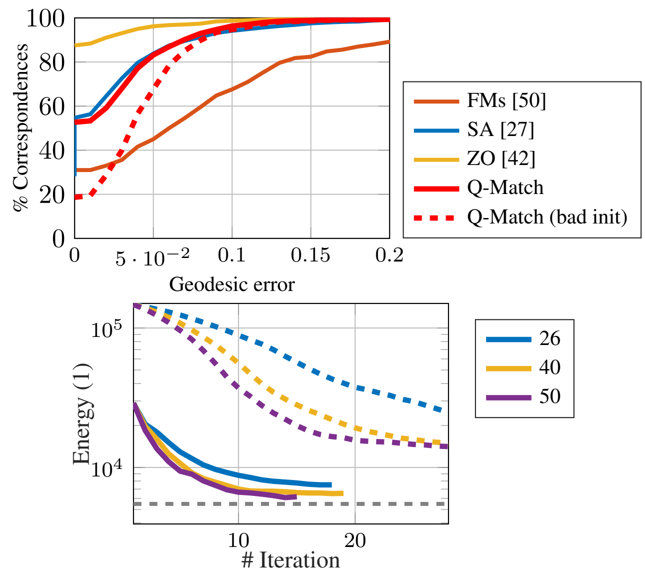

Figure 10. Quantitative experiments comparing our method using the optimal descriptor initialization (solid lines) with the worst descriptor initialization (dashed lines). (Left) Cumulative geodesic error (left) and convergence (right) is shown on the FAUST dataset, otherwise equivalent to Fig. 4. The dashed gray line is the ground truth value.

basic operations ($W_{ijkl} = |d(i,j) - d(k,l)|$, where all $d(\cdot,\cdot)$ are precomputed), resulting in $\mathcal{O}(k^4|C|)$. The computation of each entry can be parallelized.

D.2. Calculating $\tilde{W}$

Since we converted (1) into a QUBO (2), W_s also needs to be converted into $\tilde{W}$, *i.e.,* the matrix describing the energy for the chosen combination of cycles. Since the cycles are sampled from s_M, s_N, $\tilde{W}$ can be computed from the entries of W_s, as we defined in (7) (and repeated here):

$$\tilde{W}_{ij} = \begin{cases} E(C_i, C_j) & \text{if } i \neq j, \\ E(C_i, C_i) + E(C_i, P_0) + E(P_0, C_j) & \text{otherwise.} \end{cases} \tag{7}$$

$E(C_i, C_j)$ can be calculated as two matrix-vector multiplications (1), however, since the vectors are vectorized permutation matrices with exactly k non-zero entries, they can be written as two sums over k entries. This is a $m \times m$ matrix. Computing every entry separately leads to a complexity of $\mathcal{O}(m^2 k)$. In our setting with 2-cycles, $m = \frac{1}{2}k$ holds, therefore, we reach a complexity of $\mathcal{O}(k^3)$. Note that usually $|C| \gg k$, and calculating $\tilde{W}$ is a lot more efficient than calculating W_s (see Fig. 6).

E. Exact Solutions on QAPLIB

Since the relative error of the QAP is not invariant under shifts of W, we also report our results on QAPLIB in

Table 1. Here, it becomes clear again that we reach the optimum for virtually all instances in ESC16 and HAD.

F. Non-Optimal Initialization

Due to a sign error in our original experiments on the FAUST dataset, we ran them with the worst possible descriptor based initialization instead of the best. As expected the accuracy is not as high and the algorithm converges slower, but Q-Match does not break completely with a very bad initialization. We see this as an indicator that finding high quality solutions for larger subproblems leads to a very robust pipeline. See Figure 10 for the exact results.

References

[1] Francis Bach. Submodular functions: from discrete to continuous domains. *Mathematical Programming*, 175(1):419–459, 2019. 3

[2] Rainer E. Burkard and Josef Offermann. Entwurf von schreibmaschinentastaturen mittels quadratischer zuordnungsprobleme. *Zeitschrift für Operations Research*, 1977. 5, 6

[3] Jun Cai, William G. Macready, and Aidan Roy. A practical heuristic for finding graph minors. *arXiv e-prints*, 2014. 2

[4] D-Wave Systems, Inc. Leap datasheet v10. https://www.dwavesys.com/sites/default/files/Leap_Datasheet_v10_0.pdf, 2021. online; accessed on the 25.02.2021. 2, 3

[5] D-Wave Systems, Inc. Simulated annealing sampler. https://github.com/dwavesystems/dwave-neal/blob/master/docs/reference/sampler.rst, 2021. online; accessed on the 25.02.2021. 3

[6] D-Wave Systems, Inc. Solver computation time. https://docs.dwavesys.com/docs/latest/timing_qa_cycle_time.html, 2021. online; accessed on the 15.03.2021. 4

[7] Vasil S. Denchev, Sergio Boixo, Sergei V. Isakov, Nan Ding, Ryan Babbush, Vadim Smelyanskiy, John Martinis, and Hartmut Neven. What is the computational value of finite-range tunneling? *Phys. Rev. X*, 6:031015, Aug 2016. 4

[8] Bernhard Eschermann and Hans-Joachim Wunderlich. Optimized synthesis of self-testable finite state machines. In *20th International Symposium on Fault-Tolerant Computing (FFTCS 20)*, 1990. 5, 6

[9] Philippe Flajolet and Robert Sedgewick. *Analytic combinatorics*. Cambridge University Press, 2009. 2

[10] Scott W Hadley, Franz Rendl, and Henry Wolkowicz. A new lower bound via projection for the quadratic assignment problem. *Mathematics of Operations Research*, 1992. 5, 6

[11] Benjamin Holzschuh, Zorah Lähner, and Daniel Cremers. Simulated annealing for 3d shape correspondence. In *Conference of 3D Vision (3DV)*, 2020. 4

[12] Michael Jünger, Elisabeth Lobe, Petra Mutzel, Gerhard Reinelt, Franz Rendl, Giovanni Rinaldi, and Tobias Stollenwerk. Performance of a quantum annealer for ising ground state computations on chimera graphs. *arXiv preprint arXiv:1904.11965*, 2019. 4

[13] Vladimir G. Kim, Yaron Lipman, and Thomas A. Funkhouser. Blended intrinsic maps. *ACM Trans. Graph.*, 30(4), 2011. 4

[14] Simone Melzi, Jing Ren, Emanuele Rodolà, Abhishek Sharma, Peter Wonka, and Maks Ovsjanikov. Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 2019. 4

[15] Christopher E. Nugent, Thomas E. Vollmann, and John Ruml. An experimental comparison of techniques for the assignment of facilities to locations. *Operations Research*, 1968. 5, 6

[16] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *Trans. Graphics*, 31(4):30, 2012. 4

[17] Catherine Roucairol. *Du sequentiel au parallele: la recherche arborescente et son application a la programmation quadratique en variables 0 et 1*. PhD thesis, Université Pierre et Marie Curie, 1987. 5, 6

[18] Michael Scriabin and Roger C. Vergin. Comparison of computer algorithms and visual based methods for plant layout. *Management Science*, 1975. 5, 6