SAT Strikes Back: Parameter and Path Relations in Quantum Toolchains

Lukas Schmidbauer
Technical University of Applied Sciences Regensburg
Regensburg, Germany
lukas.schmidbauer@othr.de

Wolfgang Mauerer
Technical University of Applied Sciences Regensburg
Siemens AG, Technology
Regensburg/Munich, Germany
wolfgang.mauerer@othr.de

***Abstract*—In the foreseeable future, toolchains for quantum computing should offer automatic means of transforming a high level problem formulation down to a hardware executable form. Thereby, it is crucial to find (multiple) transformation paths that are optimised for (hardware specific) metrics. We zoom into this pictured tree of transformations by focussing on k-SAT instances as input and their transformation to QUBO, while considering structure and characteristic metrics of input, intermediate and output representations. Our results can be used to rate valid paths of transformation in advance—also in automated (quantum) toolchains. We support the automation aspect by considering stability and therefore predictability of free parameters and transformation paths. Moreover, our findings can be used in the manifesting era of error correction (since considering structure in a high abstraction layer can benefit error correcting codes in layers below). We also show that current research is closely linked to *quadratisation* techniques and their mathematical foundation.**

***Index Terms*—Quantum Software, SAT, Pseudo Boolean Function, QUBO, PUBO**

I. Introduction

Although quantum computers promise mathematically backed advantages [1], achieving practical quantum advantage is a herculean task. On the one hand, hardware has to cope with noise, imperfections, limited amount of inhomogeneous qubits and restricted topology, which narrows down potential applicability—also in the field of error correction. On the other hand, preprocessing problem formulations has a significant impact on solution quality, performance and deployability onto current hardware [2]–[11]. However, preprocessing steps (or transformations) occur (in-)between all abstraction layers, starting from an abstract problem formulation down to transpilation onto hardware—leading to mountainous amounts of combinations. Moreover, transformations also encode NP-complete problems (*e.g.*, mapping and routing problem [12], [13]), which complicates finding optimal solutions.

k-Satisfiability (SAT) formulations have an enormous field of applications. Scheduling problems [14], circuit equivalence checking [15], string constraint handling [16] and quantum circuit optimisation [17], [18] are some of many applications. Furthermore, industrial problems induce specific structures and properties into their SAT formulations [19]. It is widely presumed that targeted classical solvers (*e.g.*, Conflict-Driven Clause Learning (CDCL)) are able to exploit these hidden structures. Notably, self-similar [20], community [21], [22] and scale-free structures [23], [24] are among relevant properties for industrial SAT instances. Take into consideration that different solvers are more or less suited for specific properties of SAT instances. For example, (satisfiable) random k-SAT instances can be solved using Stochastic Local Search or Look-Ahead solvers [19]. Transforming a SAT instance to widely used Quadratic Unconstrained Binary Optimisation (QUBO) form (*e.g.*, [25], [26]) makes quantum annealing available as a hardware solver [27]. Similar to classical solvers, different properties result in varying performance. For instance, [28] compares randomly generated 3-SAT instances and their performance on DWave's quantum annealer for two different QUBO formulations.

The performance of SAT solvers depends on the size and structure of the instance specific solution space. This has been widely studied for random 3-SAT instances, where the clause-to-variable ratio $\alpha = \frac{m}{n}$ leads to a phase transition at $\alpha \approx \alpha_C = 4.267$ from satisfiable to non-satisfiable instances [29]. For $\alpha \approx \alpha_d = 3.921$, hard instances, inducing metastable states, can be found [29]. For quantum annealing, Gabor *et al.* [30] show empirically that performance also depends on clause-to-variable ratio α by firstly transforming the input 3-SAT representation to a Maximum Independent Set (MIS) problem and then to QUBO (see Sec. III).

From a software engineering perspective, it is therefore essential to know effects of single and transitive transformations on important metrics to get most out of available quantum hardware. We want to extend above work, by shedding more light on the induced structure and properties, when transforming from k-SAT to QUBO. Fig. 1 shows a multitude of possible transformations and possible paths from k-SAT to QUBO alongside their respective properties. Sec. II introduces fundamental concepts. They are used in Sec. III, which analyses in depth shown transformations from Fig. 1. Sec. IV then analyses discussed transformation paths based on their structure and scaling effects on the metrics shown in Fig. 1. We also discuss implications on quantum software in Sec. V and conclude in Sec. VI.

The paper is augmented by a comprehensive reproduction package [31] and a supplementary website (links in PDF) that allow for extending our work.

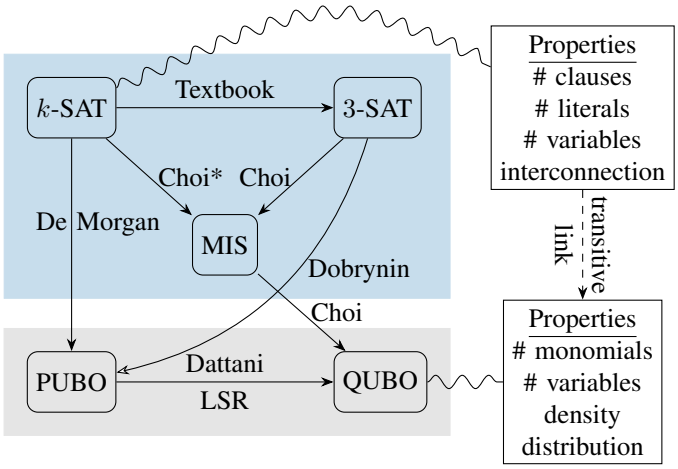

Figure 1: Possible paths from k-SAT to QUBO via known transformations with special focus on properties induced into resulting QUBO. Choi* marks a generalised version of the 3-SAT to MIS transformation. Perceived abstraction layer of *NP-complete* problems coloured in blue and *Optimisation* problems coloured in grey.

II. Fundamentals

A. SAT

SAT is an NP-complete problem that determines if a Boolean formula is satisfiable. Although there is a multitude of possible representations of such formulas, Conjunctive Normal Form (CNF) is a prominently used representation, since every Boolean circuit can be transformed into an equi-satisfiable CNF formula in linear time [32]. CNFs are also a standard representation for annual SAT competition [33].

A SAT formula $\psi(\vec{x})$ in n variables $\vec{x} = (x_1, x_2, \ldots, x_n)$ in CNF is a conjunction of m clauses $C_i, i \in \{1, \ldots, m\}$:

$$\psi(\vec{x}) = \bigwedge_{i=1}^{m} C_i, \ \vec{x} \in \{0,1\}^n \tag{1}$$

A clause C_i consists of a disjunction of positive or negative literals, where a positive literal is a variable x_i and a negative literal is the negation of a variable $\overline{x_i}$. We denote a positive literal (or variable) by l and negative literal (or negated variable) by $\bar{l}$. When the number of literals in clauses $C_i, i \in \{1, \ldots, m\}$ is fixed to $k \in \mathbb{N}$, $\psi(\vec{x})$ is an exact-k-SAT instance. For example,

$$\psi(\vec{x}) = (x_1 \vee x_2 \vee \overline{x_4} \vee x_5) \wedge (\overline{x_1} \vee x_3 \vee \overline{x_4} \vee \overline{x_5}) \tag{2}$$

is an exact 4-SAT formula in variables $\vec{x} = (x_1, x_2, x_3, x_4, x_5) \in \{0,1\}^5$. Hence, there are 2^5 possible assignments for $\vec{x}$. If $\exists \vec{x} \in \{0,1\}^n : \psi(\vec{x}) = 1$, we call $\psi(\vec{x})$ satisfiable. $|C|$ denotes the size of a clause C. The objective of the MAX-k-SAT problem is to find $\vec{x} \in \{0,1\}^n$ such that the number of satisfied clauses is maximised.

B. Maximum Independent Set

The independent set problem asks for a subset of nodes, such that no node pair in the subset is connected via an edge. MIS then asks for the biggest independent set. MIS is an NP-hard problem with an NP-complete decision variant (see SAT).

More formally, let $G(V, E)$ denote an undirected graph. Then, an independent set is a subset of nodes $I \subseteq V$ such that there exists no edge $e = \{i, j\} \in E : i \in I \wedge j \in I$. The maximum independent set is (a) not contained in any other independent set (also called maximal condition) and (b) largest with respect to the cardinality of I [34], [35]. Fig. 2 shows an example graph and a MIS.

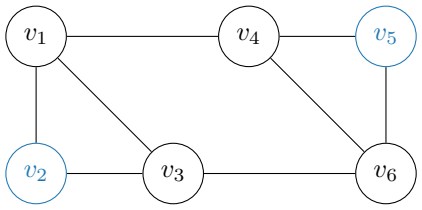

Figure 2: Example of a MIS $I = \{v_2, v_5\}$ coloured in blue. Note that $\{v_1, v_2, v_3\}$ and $\{v_4, v_5, v_6\}$ form a 3-clique (*i.e.*, a fully connected subgraph).

C. PBF, PUBO, QUBO

A Pseudo-Boolean Function (PBF) is a function

$$f : \{0,1\}^n \to \mathbb{R} \tag{3}$$

that can be uniquely represented by a multi-linear polynomial [36]:

$$f(x_1, \ldots, x_n) = \sum_{S \subseteq \{1,\ldots,n\}} \alpha_S \prod_{j \in S} x_j, \tag{4}$$

where $\alpha_S \prod_{j \in S} x_j$ is called a monomial of f and $\alpha_S \in \mathbb{R}$. The degree (or order) of a monomial is given by $|S|$.

Polynomial Unconstrained Binary Optimisation (PUBO) refers to the problem of finding $\vec{x} \in \{0,1\}^n$, such that PBF $f(\vec{x})$ is maximised or minimised. QUBO refers to the same problem, while restricting f to be quadratic or in other words only allowing for monomials of degree at most two. For instance, $3x_1x_2x_3$ is a degree-3 monomial, while $\pi x_1 x_2$ is a degree-2 monomial and thus allowed in QUBO problems.

We sometimes refer to the *quadratisation* of a higher-order (*i.e.*, degree greater than two) function $f(\vec{x})$ as the reduction of PUBO to QUBO. Technically, a PBF $f'(\vec{x}, \vec{y})$ is a *quadratisation* of $f(\vec{x})$, if $f'(\vec{x}, \vec{y})$ is a quadratic PBF ($\deg(f') = 2$) in $\vec{x} = x_1, \ldots, x_n$ and $\vec{y} = y_1, \ldots, y_m$, and satisfies [36]:

$$f(\vec{x}) = \min_{\vec{y} \in \{0,1\}^m} f'(\vec{x}, \vec{y}) \ \forall \vec{x} \in \{0,1\}^n. \tag{5}$$

Linked to that is a standard penalty term for iterative *quadratisation* that constrains new variables [36]:

$$\mathrm{p}(x_i, x_j, y_h) = 3y_h + x_i x_j - 2x_i y_h - 2x_j y_h. \tag{6}$$

D. Graph Representations for SAT and PBF

To analyse structural properties, we define the following variable incidence graph $G(V, E)$ for SAT formulas: Let $\psi(\vec{x}) = \bigwedge_i C_i$ be a SAT formula in CNF. Then, $V = \{x_1, \ldots, x_n\}$ is the set of variables. Edges are introduced, whenever two variables (negated or not) occur in the same clause C_i, that is, edge $e = (x_a, x_b) \in E \Leftrightarrow \exists i : x_a \in C_i \wedge x_b \in C_i$. Variable $x \in C$ denotes whether variable x occurs in clause C regardless of negation.

Analogously we define a graph representation for a given PBF $f(\vec{x}) = \sum_i \alpha_i M_i$, where $\alpha_i M_i$ enumerates all possible monomials based on $\vec{x}$. As before nodes $V = \{x_1, \ldots, x_n\}$. Similarly, edges are introduced, whenever two variables occur in the same monomial M_i: $e = (x_a, x_b) \in E \Leftrightarrow \exists i : x_a \in M_i \wedge x_b \in M_i \wedge \alpha_i \neq 0$.

For both graph representations, we disregard self-edges, since in the case of SAT they would either mean that a variable is redundant in a clause or the clause is trivially satisfiable and in the case of PBFs, $x^n = x, n \in \mathbb{N}$. Take into consideration that the same variable pair can occur in multiple clauses or monomials. To keep this information, a multi graph (multiset E) could be used. However, for sake of simplicity, we do not formally introduce a multigraph, but rather denote the number of edges between two nodes by an edge label.

III. Related Work

A. Textbook k-SAT to 3-SAT

The textbook reduction from k-SAT to 3-SAT [35] provides a simple method to reduce the size of clauses. Let $t = 3$ be the target size. In essence, we iteratively replace the last $t-1$ variables by a new variable in each clause C_i larger than t. Hence, the size of C_i reduces by $t-2$ and we introduce a new t-SAT clause that contains the negation of the new variable and the formerly replaced ones. For example, let $\psi(\vec{x}) = (x_1 \vee x_2 \vee \overline{x_3} \vee \overline{x_4} \vee \overline{x_5})$ be a 5-SAT instance. Then,

$$(x_1 \vee x_2 \vee \overline{x_3} \vee \overline{x_4} \vee \overline{x_5}) \xrightarrow{x_6}$$
$$(x_1 \vee x_2 \vee \overline{x_3} \vee x_6) \wedge (\overline{x_6} \vee \overline{x_4} \vee \overline{x_5}) \xrightarrow{x_7}$$
$$(x_1 \vee x_2 \vee x_7) \wedge (\overline{x_7} \vee \overline{x_3} \vee x_6) \wedge (\overline{x_6} \vee \overline{x_4} \vee \overline{x_5}) = \psi'(\vec{x'})$$

$\psi'(\vec{x'})$ is a t-SAT formula that is satisfiable if and only if $\psi(\vec{x})$ is satisfiable. Note that this method can easily be extended to reduce to t-SAT with $t > 3$ formulas.

B. MAX-k-SAT to QUBO

Chancellor *et al.* [37] provide a QUBO mapping of MAX-3-SAT and mentions the easily expandability of his construction to MAX-k-SAT[1]. By counting the number of literals that satisfy a clause for a given assignment via one-hot encoded ancillas, he is able to use penalty terms per clause that match a given MAX-3-SAT problem. Chancellor *et al.* also mention that more efficient (in terms of ancilla bits) constructions are possible. For instance, a 3-SAT clause can be implemented into QUBO by using a single (and not 3, as before) ancilla bit. We demonstrate this mapping, when considering the general mapping of a k-SAT clause to PUBO, which is then transformed to QUBO via *quadratisation*.

Nüßlein *et al.* [38] expand the idea of counting the number of literals that satisfy a clause by using binary encoding—lowering the number of ancillas per clause to grow logarithmically in k for $k \geq 4$ in a MAX-k-SAT problem. To be more precise, the number of variables in the resulting QUBO N is given by:

$$N = n + m \cdot r(k), \tag{7}$$

where n is the number of variables, m is the number of clauses and

$$r(k) = \begin{cases} 0 & \text{if } k = 2, \\ 1 & \text{if } k = 3, \\ \lceil \log_2(k+1) \rceil + r(\lceil \log_2(k+1) \rceil) & \text{if } k \geq 4. \end{cases}$$

Note that both methods scale linearly in the number of clauses.

C. 3-SAT to QUBO

Recall the definition of SAT and MIS from Sec. II. We now shortly review a known reduction from 3-SAT to MIS (Choi) [39]. Then, we can use a known QUBO formulation for the MIS instance.

Let $\psi(\vec{x}) = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ be a 3-SAT instance in CNF, that is, each clause C_i, $i \in \{1, \ldots, m\}$ consists of at maximum 3 literals (*i.e.*, (negated) variables $x_1, \ldots, x_n$): $C_i = (l_{i_1} \vee l_{i_2} \vee l_{i_3})$. For each literal in a clause C_i, we create a group of fully connected nodes in graph $G_{\text{SAT}}(V, E)$ (see Fig. 2) [39]. Additionally, every conflicting literal pair (*i.e.*, $l_{i_s} = \overline{l_{j_t}}$; $i \neq j$) in G_{SAT} introduces an edge (i_s, j_t). Then, the following statements are logically equivalent:

- $\psi(\vec{x})$ is satisfiable.
- G_{SAT} has a MIS of size m.

The known QUBO formulation of a MIS [39] incorporates its graph G and was recently used by Zielinski *et al.* [40] to compare it to other transformations to QUBO. Its optimisation function is given by [41]:

$$\gamma(x_1, \ldots, x_n) = \sum_{i \in V} c_i x_i - \sum_{(i,j) \in E} J_{ij} x_i x_j, \tag{8}$$

where $c_i = 1$ is the weight for nodes in the graph[2] and $J_{ij} > 1$ $\forall (i,j) \in E$. Maximising function $\gamma(x_1, \ldots, x_n)$ solves the MIS problem. In particular, the set of nodes $\text{mis}(G) = \{i \in V : x_i^* = 1\}$, where $(x_1^*, \ldots, x_n^*) = \arg\max_{(x_1, \ldots, x_n) \in \{0,1\}^n} \gamma(x_1, \ldots, x_n)$, corresponds to MIS.

Zielinski *et al.* [40] structure 3-SAT clauses into four types, depending on (negated) literals. They present an algorithm to search for valid QUBO representations[3] $Q_i, i \in \{1, \ldots, m\}$ for single 3-SAT clauses and then combine each Q_i to a QUBO Q, using methods presented in [37]. This method results in a QUBO size of $n + m$, where n is the number of variables

[1] Technically, Ising Spin Glasses are used, which can easily be mapped to QUBO by variable substitution.

[2] SAT reduces to an unweighted graph.

[3] Note that there are infinitely many valid representations.

in the original SAT instance and m is the number of clauses. Note that defining types of k-SAT clauses, as before for 3-SAT, scales exponentially in k. However, textbook reductions from k-SAT to 3-SAT can be applied prior to type definition (see Sec. II).

This method is closely linked to *quadratisation* techniques found in the works of Boros *et al.* [36], [42], [43]: We can reformulate the above stated as firstly defining a family of valid *quadratisations* for clause types that use a single ancilla variable. Alternatively, one can use the methods presented in Sec. III-E to formulate higher-order PBFs $f_i, i \in \{1, \dots, m\}$ and then apply one of many reductions from [44] to arrive at a similar quadratic function for clauses. Secondly, a PBF for the original 3-SAT instance is determined by summation:

$$f = \sum_{i=1}^{m} f_i. \tag{9}$$

Dobrynin *et al.* [45] point to a similar argument by incorporating different penalty terms in their PUBO to QUBO transformation. In [46], the $n+m$ approach provides different penalty terms with fewer interactions than the standard penalty (see Sec. II). Despite these penalties not obeying the *quadratisation* criteria, they preserve at least one minimum[4]. Although, every isolated k-SAT clause C_i has $2^k - 1$ satisfying variable assignments, penalty terms that do not obey the *quadratisation* criteria potentially reduce the set of valid possible solutions for isolated clauses. However, since clauses $C_i, i \in \{1, \dots, m\}$ typically share variables or literals in a given k-SAT formula, it is important to preserve all local satisfying solutions for clauses. If not, satisfying variable assignments for the given k-SAT formula do not map to minimum in QUBO. Take into consideration that the effect of this local inaccuracy is especially pronounced on k-SAT formulas with high inter-clause connectivity, high number of clauses or high k. We therefore emphasize the importance to adhere to the *quadratisation* criteria (see Eq. 5), when building upon small solution sets.

D. k-SAT to MIS to QUBO

Through a similar argument for the reduction from 3-SAT to MIS (see Sec. III-C), we extend this study by comparing QUBO mappings for k-SAT instances. Analogously, each exact k-SAT clause C_i creates a fully connected sub-graph in G with k nodes. As before, conflicting literals in different sub-graphs are connected. Note that if there are conflicting literals in the same sub-graph, they are already connected by construction. Hence, for exact k-SAT with m clauses, there are $k \cdot m$ nodes in the graph.

As a side note, the size of resulting QUBO is at least twice the number of variables squared, when all literals are used in the SAT formula. As with the 3-SAT to MIS construction, choosing $J_{ij} > min(c_i, c_j)$, $c_i > 0$ is required.

[4]We refer to the penalties given for $(a \vee b \vee \overline{c})$ and $(a \vee \overline{b} \vee \overline{c})$.

E. k-SAT to PUBO

Let $\psi(\vec{x})$ be an exact k-SAT formula in CNF with clauses C_i, $i \in \{1, \dots, m\}$. For example, let $C_1 = (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$ and $C_2 = (x_2 \vee x_3 \vee \overline{x_4})$. Then, contrary to Choi's reduction, we can encode a negated variable x_i as $1 - x_i$. The same applies for clause negation, by using DeMorgan's rules for multiple variables (see also [45]). Hence, equivalent PBFs for clauses C_1 and C_2 are given by

$$\begin{aligned} f_{C_1}(x_1, x_2, x_3) &= 1 - x_1 x_2 x_3 \\ f_{C_2}(x_2, x_3, x_4) &= 1 - (1 - x_2)(1 - x_3)x_4. \end{aligned} \tag{10}$$

Their sum then encodes the decision problem:

$$f_{\text{SAT}}(x_1, x_2, x_3, x_4) = f_{C_1}(x_1, x_2, x_3) + f_{C_2}(x_2, x_3, x_4).$$

A similar construction for 3-SAT formulas can be found in [47]. Note that if a clause C_i evaluates to true the corresponding PBF f_{C_i} evaluates to 1. If C_i evaluates to false, then f_{C_i} evaluates to 0—effectively encoding C_i into a maximisation problem. The minimisation problem can easily be obtained by negating f_{C_i}. For both maximisation and minimisation problem, the SAT formula is satisfiable, iff

$$\exists \vec{x} \in \{0, 1\}^n : |f_{\text{SAT}}(\vec{x})| = m. \tag{11}$$

If $|f_{\text{SAT}}(\vec{x})| = s$, then there are s many satisfied clauses for the assignment $\vec{x}$, which can be used for the generalised optimisation of MAX-SAT.

In general, for exact k-SAT formulas, this leads to monomials of degree k in the resulting PBF. Monomials of degree smaller than k occur whenever non-negated variables appear in a clause C_i. Note that if a clause C_i consists of k negated variables (*e.g.*, $(\overline{x_1} \vee \overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$), then the resulting PBF has a single degree-k monomial (*e.g.*, $x_1 x_2 x_3 x_4$). Monomials can can be *quadratised* via either $\lceil log_2(k) \rceil - 1$ or a single extra variable—depending on whether α_S is positive or negative [43], [44]. Although redundant clauses are excluded from the input SAT formula, the resulting degree-k monomial can occur in other transformed PBFs f_{C_j}. For example, let $C_j = (x_1 \vee \overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$. Then $x_1 x_2 x_3 x_4$ occurs in $f_{C_j}(\vec{x}) = 1 - (1 - x_1)x_2 x_3 x_4$ [5], which offers potential to efficiently *quadratise* multiple occurrences at once.

Conversely, assume a clause C_i consists of k distinct and non-negated variables (*i.e.*, no negative literal). Then, the resulting PBF f_{C_i} has all possible monomials of degree i, $i \in \{1, \dots, k\}$. Since there are $\binom{k}{i}$ many possible degree-i monomials, clause C_i introduces

$$\sum_{i=0}^{k} \binom{k}{i} = 2^k - 1$$

[5]We do not consider prefactor α here.

many monomials. If one would brute force reduce all degree-i ($i > 2$) monomials via a single variable in every clause[6], there would be

$$m \cdot \sum_{i=3}^{k} \binom{k}{i} = m \cdot \left(2^k - \binom{k}{2} - k - 1\right)$$

many new variables in f_{SAT}. However, arbitrary SAT instances usually do not fit into one of the above categories (all positive or all negative literals) and hence a reduction method exploiting the inner structure of f_{SAT} is needed.

F. PUBO to QUBO

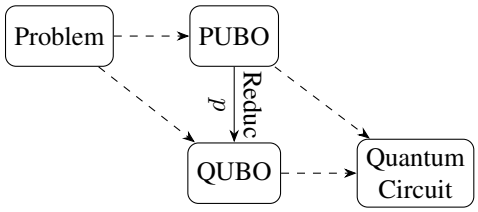

Figure 3: Abstracted PUBO / QUBO relation. (Dashed) lines: (Multiple intermediate) transformation(s).

Let PUBO and QUBO denote two formulations for the same problem (see Fig. 3). It is possible to encode both PUBO and QUBO into quantum circuits and transpile them onto hardware that features at most two qubit operations. On the one hand the PUBO formulation leads to necessary decompositions of higher-order gates. On the other hand, the QUBO formulation has to cope with encoding the same information and therefore has potentially more gates than the original PUBO. Reducing the degree of monomials in a PUBO f can benefit circuit metrics (*i.e.*, circuit depth, distribution of gates and number of gates), when, for example, creating a Quantum Approximate Optimisation Algorithm (QAOA) circuit [2]. Note that these results stem from an empirical study, featuring an industry relevant Job-Shop Scheduling problem with at most degree-4 monomials. Contrary to that, Campbell and Dahl [48] execute small instances of the four corner graph colouring problem with QAOA and COBYLA and found better results when directly using their PUBO formulation.

There are many possible transformations from PUBO f_P to QUBO f_Q—including monomial-wise reductions or considering specific monomial properties [44]. When structural properties cannot be leveraged to simplify f_P, new variables are introduced to reduce the degree of f_P. A fast *quadratisation* algorithm (Local Structure Reduction (LSR)) was published in [49], which uses iterative *quadratisation* and has a free parameter $p \in [0, 1]$. Parameter p influences properties of the resulting QUBO f_Q. For instance, a higher value of p can lead to less variables, but higher degree-2 density in f_Q and vice versa.

Mapping a given logical quantum circuit onto hardware (that depends on given QUBO) often requires the use of SWAP-gates to accommodate for missing connections. Finding balance between connectivity in QUBO and number of qubits is performance-relevant and thus should be considered by an automated (quantum) toolchain. Percentile p is a simple tunable parameter for this task.

[6]This term is a lower bound, since $\lceil log_2(k) \rceil - 1$ extra variables are required for positive degree-k monomials [43].

IV. Experiments

Recall that introduced MAX-k-SAT to QUBO formulations, as well as Choi's reduction via MIS scale with the number of clauses, but do not consider inter-clause relations (*i.e.*, the inner structure). Contrary, the iterative reduction is able to exploit the inner structure of the resulting PUBO formulation, when *quadratising* to QUBO—not only requiring less extra variables, but also influencing structure of resulting QUBO. Hence, comparing these methods is of interest for the following experiments. Our experiments also extend depicted research by considering structural properties of input, intermediate and output problem representations.

A. Setup

Let V be the set of variables. Then $L = V \cup \{\overline{x} : x \in V\}$ is the set of literals, where $\overline{x}$ is the negation of x. For all experiments, we randomly sample k literals from L and repeat that step m times (number of clauses). Since the number of actually used variables in $\psi(\vec{x})$ can be less than $|V|$, we use the number of actually used variables as the x-axis of our graphs. At first we shed light on the scaling effects and then analyse the structural properties of (intermediate) representations for a single instance. Take into consideration that, for $k = 3$, textbook reduction has no effect on its input. Hence, Choi and Choi*, as well as Dobry. and DeMorg. perform similarly among all figures for $k = 3$.

B. Results

Fig. 4, 5 and 6 have equal structure: The x-axis represents the number of variables in the input k-SAT instance. Each horizontal facet shows one of the tested variants, that is, Choi (Textbook reduction to 3-SAT, followed by MIS to QUBO), Choi* (generalised version, *i.e.*, direct MIS to QUBO), Dobry. (Textbook reduction to 3-SAT, followed by direct PUBO mapping and *quadratisation*) and DeMorg. (direct PUBO mapping, followed by *quadratisation*). Fig. 9 also gives a bird eye view on their relation. Each vertical facet represents k in the input instance, while the colour indicates the number of clauses m in the input instance.

Fig. 4 shows the number of variables in resulting QUBO on its y-axis (log scale). For each method, the number of clauses increases the number of variables. For both Choi and Choi*, the increase is linear in the total size of clauses, with respect to the input 3- or k-SAT instance. However, Choi has to cope with increased total clause size due to introducing extra variables (see Sec. III-A) and hence, scales worse than $k \cdot m$. In particular, the resulting 3-SAT formula has $(k-2) \cdot m$ clauses and therefore a total size of $3(k-2) \cdot m$ [7]. The effects of the textbook reduction are also visible in Dobry., which

[7]Note that $3(k-2) \cdot m > k \cdot m \Leftrightarrow k > 3 \; \forall m \in \mathbb{N}$.

leads to small dependence on the number of variables in k-SAT, since variables in 3-SAT clauses after textbook reduction contain at least $1/3$ new (negated) variables (depending on the particular pair choice in iterative textbook reduction). Conversely, DeMorg. scales with the number of variables in k-SAT, but introduces less variables up to $k = 7$ compared to Choi* and Dobry. and up to $k = 10$ for Choi and Dobry..

Fig. 5 shows the number of monomials (mostly interactions) in QUBO on its y-axis (log scale). DeMorg. and Dobry. scale similarly with according arguments as before for Fig. 4. For Choi and Choi*, the number of monomials decreases as the number of variables in k-SAT increases. Recall that their construction (see Sec. III) introduces interconnections in MIS graph, whenever there are conflicting literals in different clauses. Hence, increasing the number of variables in randomly sampled k-SAT instances, decreases the expected number of conflicts. Interestingly, this effect also transitively applies through textbook reduction to Choi, since the size of cliques reduces, although the total size of 3-SAT clauses increases compared to k-SAT.

Fig. 6 shows the total time for the k-SAT to QUBO reduction in seconds on its y-axis (log scale), which is relevant for time constraint applications. It is evident that the runtime mostly depends on the number of clauses and k for Choi, Choi* and Dobry.—with Dobry. featuring consistently lower values than Choi, albeit both using textbook reduction. Up to $k = 20$, generalised Choi* is faster than Choi, although it has to generate 20-cliques in MIS graph. In other words, the number of edges in the MIS graph per k-SAT clause is quadratic in k, whereas the textbook reduction introduces $k-2$ 3-SAT clauses per k-SAT clause. Take into consideration that directly mapping k-SAT to PUBO introduces an exponential amount of monomials in the number of positive literals, due to term expansion. We therefore do not recommend this method beyond a certain $k_\alpha > 20$. However, DeMorg. outperforms Choi and Choi* up to $k = 10$ and more than 132 clauses. Note however that the underlying implementation can be further optimised in time.

Fig. 7 shows the ratio of actual to possible clauses and degree-2 monomials on the x- and y-axis respectively. Each method tends to generate sparse QUBO representations for $k \geq 10$. For $k = 3$, DeMorg. and Dobry. result in significantly denser QUBOs than Choi and Choi*, since they do not introduce additional variables. For $k = 10$, Choi* and DeMorg. provide higher density due to less introduced variables, as discussed for Fig. 4. Also note that the randomly sampled input k-SAT instances become less dense as k increases.

Fig. 8 shows the number of variables in QUBO on its y-axis (log scale). Contrary to previous figures, where $p = 1$, its horizontal facets show different percentiles p (see Sec. III-F) for *quadratisation* and method DeMorg.. In general, higher percentile $p \in [0, 1]$ leads to less variables in QUBO, since higher p leads to the iterative reduction of variable pairs that occur in more monomials per iteration. Note however that finding the minimum number of new variables for a valid *quadratisation* is NP-hard [36]—indicating potential instabilities in the heuristic polynomial time approach. Be that as it may, Fig. 8 suggests a relatively stable process for the tested input instances—supporting the general statement for p.

Apart from scaling of metrics, the internal structure and the effect of transformations in each step are relevant to extrapolate our findings to other input problems. Fig. 9 shows this exemplary for a single 6-SAT instance with 5 clauses and 14 variables. Each path of transformations has intermediate representations, which are shown as graphs, introduced in Sec. II. Note that comparing graphs visually can be deceiving, when using arbitrary node positions and ordering. Therefore, only for the visualisation in Fig. 9, we sort nodes according to the highest clique they occur in (see Alg. 1), where MAX_CLIQUE(G) returns a list of nodes, contained in a maximum clique of G and operation "+" (Alg. 1: l. 3) appends the set of nodes C to list V'. In addition to node positioning,

Algorithm 1 Sorted nodes by clique size.

Input Graph $G(V, E)$
Output Sorted list of nodes V'

1: **while** $V \neq \emptyset$ **do**
2: $\quad C \leftarrow$ MAX_CLIQUE(G)
3: $\quad V' \leftarrow V' + C$; $V \leftarrow V \setminus C$
4: **end while**
5: return V'

we show the difference in nodes and edges to the previous representation by colouring new nodes and edges in red. Also recall that edge labels correspond to multiplicity (see Sec. II). Hence, an edge e with label 3 between nodes i and j corresponds to 3 edges. Therefore, pair $\{i, j\}$ occurs in 3 clauses (in case of SAT) or 3 monomials (in case of PBF). Fig. 9 shows a subtle characteristic for Choi's reductions, namely k-cliques, where k depends on the input SAT instance. Hence, shown MIS graph of Choi* has 5 6-cliques, since there are 5 clauses and the input is a 6-SAT instance. Likewise, shown MIS graph of Choi has 20 3-cliques, since there are 20 clauses in the input 3-SAT instance. Note that the size of cliques is upper bounded by k, since a variable in clause C_i cannot be in conflict with all variables of another clause C_j—due to the construction of k-SAT instances. Since we exclude self-edges or linear terms in PBF (see Sec. II), the resulting QUBO graph for Choi(*) is isomorph to the MIS graph.

This is in contrast to the methods that map a k-SAT instance directly to PUBO—leading to monomials of degree less than or equal to k. Since, in general, the *quadratisation* of PUBO requires additional variables and constraint terms (see Sec. III-F), resulting QUBOs change in structure and variables. Note that each PUBO graph is isomorphic to its 3- or k-SAT graph up to edge labels, which increase. Increasing edge labels are a result of degree-$t, t < k$ monomials. After *quadratisation*, each QUBO graph has edge label 1, which is not shown explicitly. Fig. 9 shows many more red edges in QUBO in the direct mapping (DeMorgan) compared to firstly reducing to 3-SAT (Dobrynin). Red edges in QUBO are a result of the quadratic penalty term (see Eq. 6) and quadratic

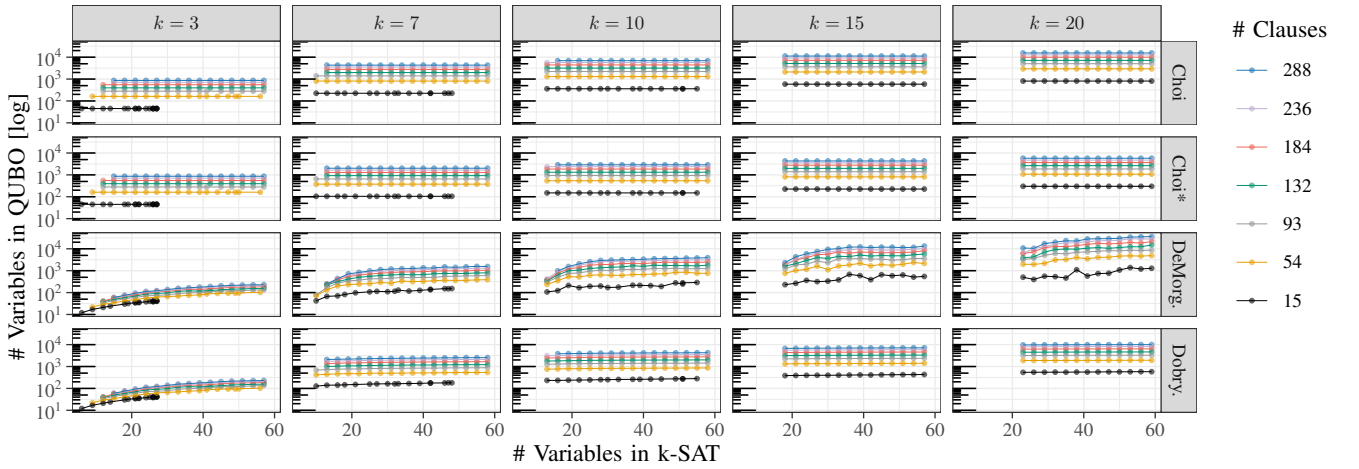

Figure 4: Number of variables in k-SAT (x-axis) vs number of variables in resulting QUBO (y-axis), coloured by the number of clauses. Horizontal facet: Transformation path. Vertical facet: k in k-SAT. For DeMorg. and Dobry., $p = 1$.

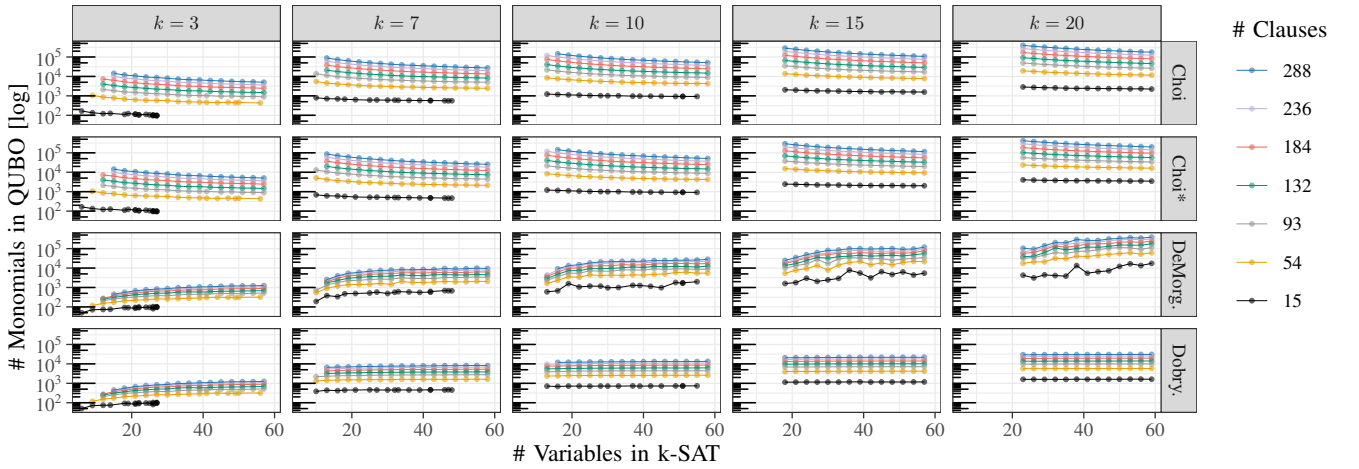

Figure 5: Number of variables in k-SAT (x-axis) vs number of monomials in resulting QUBO (y-axis), coloured by the number of clauses. Horizontal facet: Transformation path. Vertical facet: k in k-SAT. For DeMorg. and Dobry., $p = 1$.

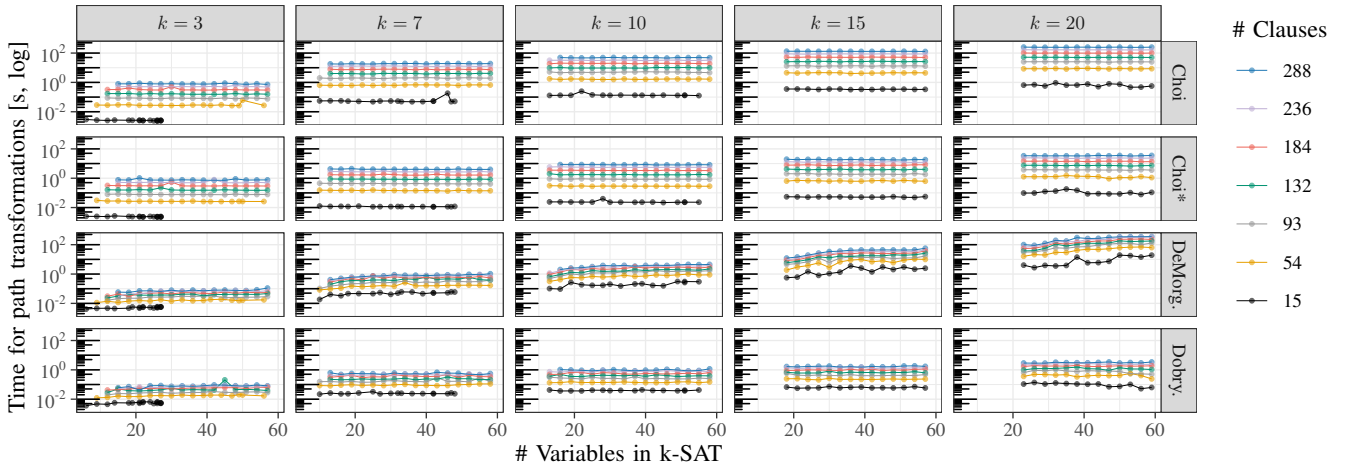

Figure 6: Number of variables in k-SAT (x-axis) vs transformation time (y-axis), coloured by the number of clauses. Horizontal facet: Transformation path. Vertical facet: k in k-SAT. For DeMorg. and Dobry., $p = 1$.

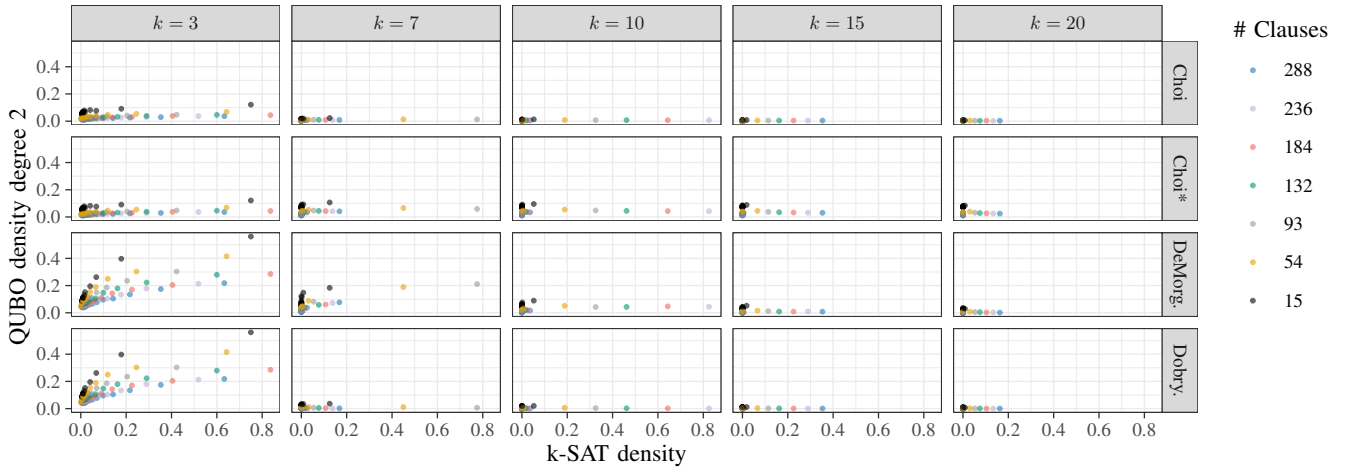

Figure 7: k-SAT density (*i.e.*, ratio of actual to possible clauses; x-axis) vs QUBO density for degree-2 monomials (*i.e.*, ratio of actual to possible degree-2 monomials) (y-axis), coloured by the number of clauses. Horizontal facet: Transformation path. Vertical facet: Value of k in k-SAT. For DeMorg. and Dobry., $p = 1$.

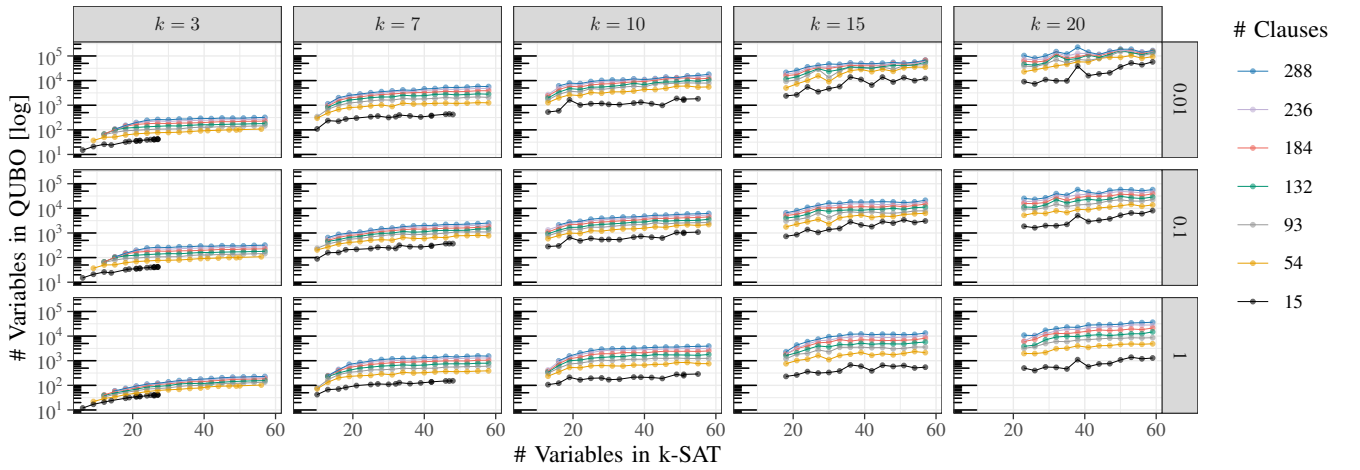

Figure 8: Number of variables in k-SAT (x-axis) vs number of variables in resulting QUBO (y-axis), coloured by the number of clauses. Horizontal facet: Percentile p for direct mapping to PUBO (DeMorgan). Vertical facet: k in k-SAT.

monomials. For example, term $x_1x_2x_3x_4 \overset{x_1x_2=y_1}{\longrightarrow} y_1x_3x_4 + p(x_1, x_2, y_1) \overset{x_3x_4=y_2}{\longrightarrow} y_1y_2 + p(x_1, x_2, y_1) + p(x_3, x_4, y_2)$ leads to a new edge $\{y_1, y_2\}$ and at least 2 more edges per penalty term.

In summary, *quadratisation* spreads the information encoded by PUBO over new variables. However, directly mapping k-SAT to PUBO leads to more concentrated regions of high and low connectivity, while first mapping to 3-SAT leads to a more uniform distribution of graph connectivity. The varying spread of interactions is also visible for Choi's reductions. One can take advantage of these variations in later steps, when, for example, transpiling a circuit onto specific hardware topologies.

C. Qualitative Extrapolation

Motivated by structure inducing industry relevant problems and based on discussed results for random exact k-SAT instances, we make qualitative arguments for different input k-SAT formulas with varying structure. This enables estimating structural properties of resulting QUBO for structured or degenerate input instances. We therefore consider:

(1) High degeneracy in k
(2) High inter-clause connections (communities/cliques)

In the following, we assume that the input is no longer necessarily given as an exact k-SAT instance. Therefore, clauses $C_i, i \in \{1, \dots, m\}$ can have different size—up to k. Let $\psi(\vec{x}) = \bigwedge_i C_i$ be a SAT formula in CNF.

a) Textbook k-SAT to 3-SAT: The transformation from k-SAT to 3-SAT solely depends on the number of clauses of size $k > 3$. Therefore, inter-clause relations do not affect this transformation (2). Note that due to newly introduced variables, there cannot be redundant clauses. A high degeneracy in k or less clauses of size k lead to less new variables (1). To be more precise, for a clause of size k, $k - 3$ new

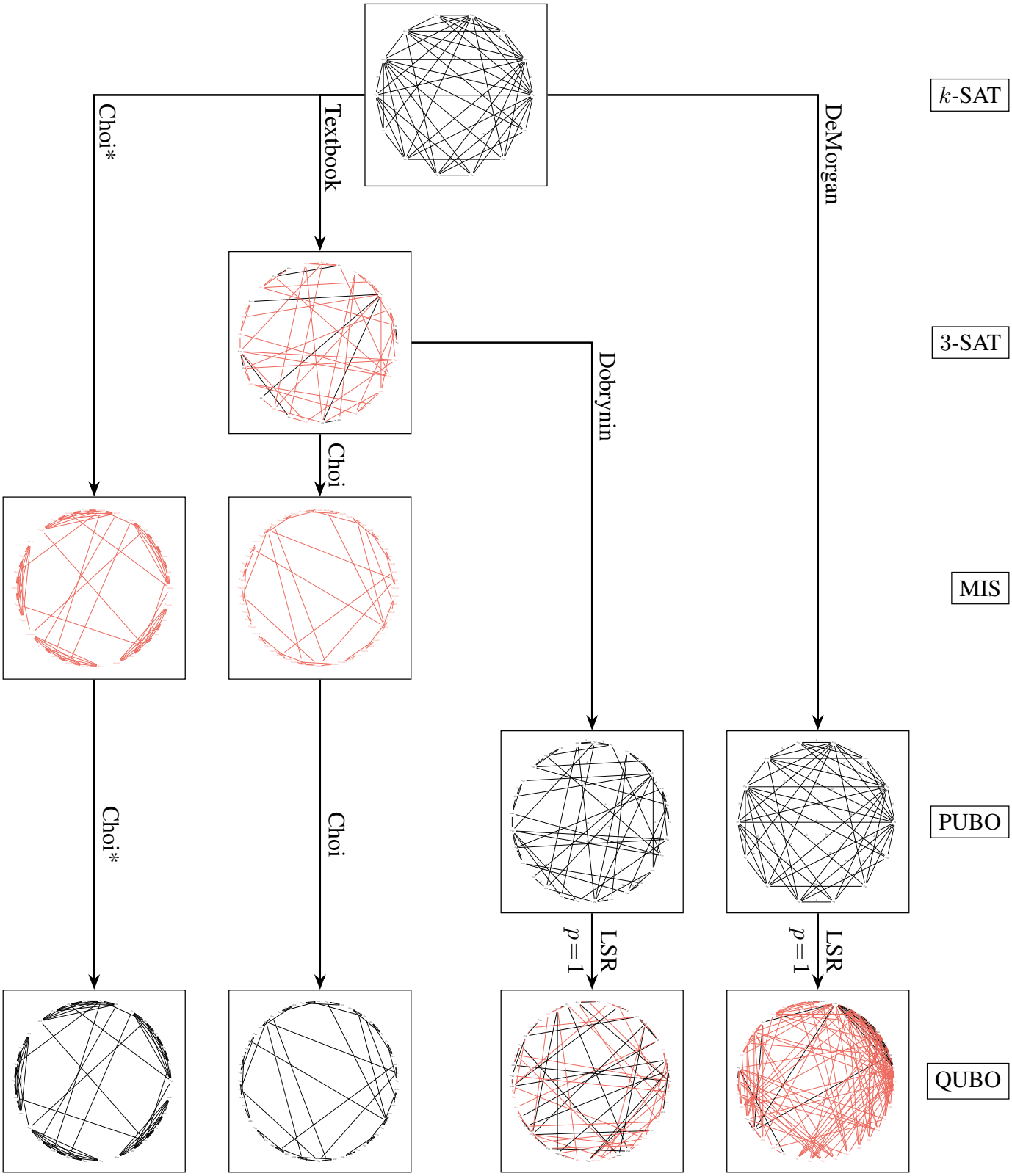

Figure 9: Structural graph evolution of transformations and transformation paths, starting at a 6-SAT instance with 5 clauses and 14 variables. Choi* marks a generalised version of Choi's 3-SAT to MIS to QUBO transformation. Nodes are placed, according to the largest clique they occur in—leading to clustered nodes. Newly introduced nodes and edges are shown in red compared to the last representation. Due to their unique construction, MIS graphs do not adopt edges or nodes.

variables are introduced. Take into consideration that there are variations of the introduced textbook reduction (see Sec. II) that, for instance, choose different variable pairs in the iterative process. This affects the inner structure and interconnectivity of resulting 3-SAT clauses.

b) Choi(*): Recall that in a first step Choi's reduction creates fully connected sub-graphs for literals in each clause. Hence, a high degeneracy in k (1) directly influences size of cliques in MIS graph. Since resulting QUBO graph is isomorphic to MIS graph, it is also directly influenced. Since Choi's reduction introduces edges, whenever conflicting literals occur in different clauses, a high conflicting inter-clause connectivity (2) leads to more densely connected cliques in MIS graph. Take into consideration that influence from textbook reduction are transitively applied, when firstly reducing to 3-SAT.

c) SAT to PUBO: Directly mapping SAT to PUBO, as shown in Sec. III, is locally dependent on clauses and their structure. For the number of introduced monomials, differentiating between positive and negative variables is decisive, as the number of monomials scales exponentially with the number of positive variables in clauses. Hence, a high degeneracy in k (1) or reducing the size of clauses results in less monomials in the resulting PUBO. The number of variables in PUBO is independent of structural properties in $\psi(\vec{x})$. Compared to random instances, a high inter-clause connection (2) potentially increases chances that a variable pair occurs in multiple monomials—enabling to reduce the same variable pair in multiple monomials at once and thereby decreasing the number of introduced variables when transforming to QUBO. As before, transitive effects from textbook reduction apply—potentially decreasing inter-clause connectivity.

V. Implications on Quantum Software

At a certain k_α it becomes infeasible to encode k_α-SAT clauses directly into PUBO form. Although we demonstrate scaling behaviour in time and size for k, the concrete value for k_α depends on a concrete application. An upstream reduction of clause size by, for example, textbook reduction (see Sec. II), makes PUBO a well suited abstraction for quantum toolchains—also considering their expressivity in other problem formulations [2] and potential speedup over QUBO in annealing [50]. Choi(*)'s reduced QUBO has high connectivity among former clauses in k-SAT. It can also be combined with textbook reductions, to increase the number of variables, but decrease clique-size and keep its regular structure. Ultimately a hardware executable representation is needed and hence further transformation steps. A quantum toolchain can leverage these results by choosing suitable paths and parameters based on later steps. For example, a quantum hardware that features high local, but low global connectivity, benefits from the structure of Choi(*)'s reduction, whereas a low qubit count hardware benefits from PUBO to QUBO reduction in the case of 3-SAT and parameter $p = 1$.

Higher order formulations (*i.e.*, increasing k) of k-SAT, tend towards sparsely connected QUBOs, which emphasises the need for efficient classical data structures, representing monomials in QUBO. This aspect is of interest in high performance data centres [51], where communication time is a vital aspect of performance. Moreover, estimating time for preparatory steps can be beneficial for schedulers that manage work loads on restricted (quantum) nodes.

The mapping problem (*i.e.*, finding a hardware executable circuit) is also relevant for Quantum Error Correcting Codes (QECC), since they typically add additional gates or qubits to introduce redundancy (see [52], [53] for more information about QECC). Since circuit depth or the number of additional qubits scales with the number of correctable errors, it is sensible to also consider hardware topology and hardware noise in advance (*e.g.*, by reducing circuit size or depth by a pre-optimised hardware specific mapping). This aspect can be combined with characteristics that are introduced by transformation paths in an automated quantum toolchain (*e.g.*, connectivity and error rates around additional qubits used for QECC).

Overall, our results indicate relatively stable scaling behaviour, paving the way for extrapolation. Hence, optimising metrics for error correcting codes, hardware and ultimately performance and solution quality is possible in a predictable manner.

VI. Conclusion and Outlook

As there are many transformation paths and free parameters from an abstract problem formulation down to deploying it onto hardware, selecting an optimal route requires the characterisation of global influence of single transformations. We set the starting point on k-SAT abstraction and investigate four transformation paths down to QUBO—accompanied by intermediate representations and their respective properties (3-SAT, MIS and PUBO). Our empirical study finds relatively stable property development among all four paths, which enables predicting their behaviour for larger inputs. We also give qualitative arguments to extrapolate our findings beyond random k-SAT inputs. Although properties develop predictably, their magnitude and structure differs (*e.g.*, # clauses, # variables, # monomials, distribution). An automated quantum toolchain can leverage these differences by optimising for metrics most relevant for a given hardware and therefore improve performance, runtime or solution quality.

The mountainous amount of possible paths and free parameters gives rise to further studies of property development among a wider range of abstraction layers. For instance, the energy spectra of a resulting optimisation problem (PUBO or QUBO) can be interesting in view of (quantum) solvers. Other studies can also extend our work by incorporating further metrics, ranked by their importance to the target representation.

Acknowledgements We thank Simon Thelen for many active discussions and Maja Franz for valuable comments on the ideas presented in this paper. We acknowledge support from German Federal Ministry of Education and Research (BMBF), funding program "Quantum Technologies—from Basic Research to Market", grant #13N15647 and #13N16092. WM acknowledges support by the High-Tech Agenda Bavaria.

References

[1] L. K. Grover, "A fast quantum mechanical algorithm for database search," 1996. [Online]. Available: https://arxiv.org/abs/quant-ph/9605043

[2] L. Schmidbauer, K. Wintersperger, E. Lobe, and W. Mauerer, "Polynomial reduction methods and their impact on qaoa circuits," in *IEEE International Conference on Quantum Software (QSW)*, 07 2024.

[3] M. Schönberger, I. Trummer, and W. Mauerer, "Quantum-inspired digital annealing for join ordering," *Proc. VLDB Endow.*, vol. 17, no. 3, p. 511–524, nov 2023. [Online]. Available: https://doi.org/10.14778/3632093.3632112

[4] K. J. Mesman, F. Battistel, E. Reehuis, D. d. Jong, M. J. Tiggelman *et al.*, "Q-profile: Profiling tool for quantum control stacks applied to the quantum approximate optimization algorithm," in *2024 IEEE International Conference on Quantum Software (QSW)*, 2024, pp. 116–124.

[5] N. Quetschlich, F. J. Kiwit, M. A. Wolf, C. A. Riofrio, L. Burgholzer *et al.*, "Towards application-aware quantum circuit compilation," in *2024 IEEE International Conference on Quantum Software (QSW)*, 2024, pp. 135–142.

[6] M. Schnaus, L. Palackal, B. Poggel, X. Runge, H. Ehm *et al.*, "Efficient encodings of the travelling salesperson problem for variational quantum algorithms," in *2024 IEEE International Conference on Quantum Software (QSW)*, 2024, pp. 81–87.

[7] A. Wright, M. Lewis, P. Zuliani, and S. Soudjani, "T-count optimizing genetic algorithm for quantum state preparation," in *2024 IEEE International Conference on Quantum Software (QSW)*, 2024, pp. 58–68.

[8] S. Reale and E. D. Nitto, "Quantum graph pursuit: Analysis of the advantages and challenges of a quantum dynamic combinatorial optimization model from a software developer perspective," in *2024 IEEE International Conference on Quantum Software (QSW)*, 2024, pp. 24–34.

[9] R. Wille, L. Berent, T. Forster, J. Kunasaikaran, K. Mato *et al.*, "The mqt handbook : A summary of design automation tools and software for quantum computing," in *2024 IEEE International Conference on Quantum Software (QSW)*, 2024, pp. 1–8.

[10] T. Yue, W. Mauerer, S. Ali, and D. Taibi, *Challenges and Opportunities in Quantum Software Architecture*. Cham: Springer Nature Switzerland, 2023, pp. 1–23. [Online]. Available: https://doi.org/10.1007/978-3-031-36847-9_1

[11] M. Periyasamy, A. Plinge, C. Mutschler, D. D. Scherer, and W. Mauerer, "Guided-spsa: Simultaneous perturbation stochastic approximation assisted by the parameter shift rule," in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 01, 2024, pp. 1504–1515.

[12] C. Zhang, A. B. Hayes, L. Qiu, Y. Jin, Y. Chen *et al.*, "Time-optimal qubit mapping," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 360–374. [Online]. Available: https://doi.org/10.1145/3445814.3446706

[13] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons *et al.*, "On the qubit routing problem," 2019. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2019/10397/

[14] M. Akram, N. Maas, P. Sanders, and D. Schreiber, "Engineering optimal parallel task scheduling," 2024. [Online]. Available: https://arxiv.org/abs/2405.15371

[15] A. Kuehlmann, V. Paruthi, F. Krohm, and M. Ganai, "Robust boolean reasoning for equivalence checking and functional property verification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 12, p. 1377–1394, Dec. 2002. [Online]. Available: http://dx.doi.org/10.1109/TCAD.2002.804386

[16] V. Havlena, L. Holík, O. Lengál, and J. Síč, "Cooking string-integer conversions with noodles." Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SAT.2024.14

[17] I. Shaik and J. van de Pol, "Optimal layout synthesis for deep quantum circuits on nisq processors with 100+ qubits," 2024. [Online]. Available: https://arxiv.org/abs/2403.11598

[18] J. Yang, Y. A. Kharkov, Y. Shi, M. J. H. Heule, and B. Dutertre, "Quantum circuit mapping based on incremental and parallel sat solving." Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SAT.2024.29

[19] C. Ansótegui, M. L. Bonet, J. Giráldez-Cru, and J. Levy, "Structure features for sat instances classification," *Journal of Applied Logic*, vol. 23, p. 27–39, Sep. 2017. [Online]. Available: http://dx.doi.org/10.1016/j.jal.2016.11.004

[20] ——, *The Fractal Dimension of SAT Formulas*. Springer International Publishing, 2014, p. 107–121. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-08587-6_8

[21] C. Ansótegui, J. Giráldez-Cru, and J. Levy, *The Community Structure of SAT Formulas*. Springer Berlin Heidelberg, 2012, p. 410–423. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31612-8_31

[22] Z. Newsham, V. Ganesh, S. Fischmeister, G. Audemard, and L. Simon, *Impact of Community Structure on SAT Solver Performance*. Springer International Publishing, 2014, p. 252–268. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-09284-3_20

[23] C. Ansótegui, M. L. Bonet, and J. Levy, *On the Structure of Industrial SAT Instances*. Springer Berlin Heidelberg, 2009, p. 127–141. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04244-7_13

[24] C. Ansótegui, M. L. Bonet, and J. Levy, "Towards industrial-like random sat instances," in *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, ser. IJCAI'09. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009, p. 387–392.

[25] M. Schönberger, I. Trummer, and W. Mauerer, "Quantum optimisation of general join trees," in *Proceedings of the International Workshop on Quantum Data Science and Management*, ser. QDSM '23, 08 2023.

[26] M. Franz, T. Winker, S. Groppe, and W. Mauerer, "Hype or heuristic? quantum reinforcement learning for join order optimisation," in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 01, 2024, pp. 409–420.

[27] W. v. d. Schoot, D. Leermakers, R. Wezeman, N. Neumann, and F. Phillipson, "Evaluating the q-score of quantum annealers," in *2022 IEEE International Conference on Quantum Software (QSW)*. IEEE, Jul. 2022. [Online]. Available: http://dx.doi.org/10.1109/QSW55613.2022.00017

[28] T. Krüger and W. Mauerer, "Quantum annealing-based software components: An experimental case study with SAT solving," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, p. 445–450. [Online]. Available: https://doi.org/10.1145/3387940.3391472

[29] M. Mézard and R. Zecchina, "Random k-satisfiability problem: From an analytic solution to an efficient algorithm," *Physical Review E*, vol. 66, no. 5, Nov. 2002. [Online]. Available: http://dx.doi.org/10.1103/PhysRevE.66.056126

[30] T. Gabor, S. Zielinski, S. Feld, C. Roch, C. Seidel *et al.*, *Assessing Solution Quality of 3SAT on a Quantum Annealing Platform*. Springer International Publishing, 2019, p. 23–35. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-14082-3_3

[31] W. Mauerer and S. Scherzinger, "1-2-3 reproducibility for quantum software experiments," in *IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 1247–1248.

[32] G. S. Tseitin, *On the Complexity of Derivation in Propositional Calculus*. Springer Berlin Heidelberg, 1983, p. 466–483. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-81955-1_28

[33] N. Froleyks, M. Heule, M. Iser, M. Järvisalo, and M. Suda, "Sat competition 2020," *Artificial Intelligence*, vol. 301, p. 103572, Dec. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.artint.2021.103572

[34] S. Butenko and P. M. Pardalos, "Maximum independent set and related problems, with applications," Ph.D. dissertation, USA, 2003, aAI3120100.

[35] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms, fourth edition*. MIT Press, 2022. [Online]. Available: https://books.google.de/books?id=HOJyzgEACAAJ

[36] E. Boros and P. L. Hammer, "Pseudo-boolean optimization," *Discrete Applied Mathematics*, vol. 123, no. 1, pp. 155–225, 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166218X01003419

[37] N. Chancellor, S. Zohren, P. A. Warburton, S. C. Benjamin, and S. Roberts, "A direct mapping of max k-sat and high order parity checks to a chimera graph," *Scientific Reports*, vol. 6, no. 1, Nov. 2016. [Online]. Available: http://dx.doi.org/10.1038/srep37107

[38] J. Nüßlein, T. Gabor, C. Linnhoff-Popien, and S. Feld, "Algorithmic qubo formulations for k-sat and hamiltonian cycles," in *Proceedings*

of the Genetic and Evolutionary Computation Conference Companion, ser. GECCO '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 2240–2246. [Online]. Available: https://doi.org/10.1145/3520304.3533952

[39] V. Choi, "Adiabatic quantum algorithms for the np-complete maximum-weight independent set, exact cover and 3sat problems," 2010. [Online]. Available: https://arxiv.org/abs/1004.2226

[40] S. Zielinski, J. Nüßlein, J. Stein, T. Gabor, C. Linnhoff-Popien *et al.*, "Pattern qubos: Algorithmic construction of 3sat-to-qubo transformations," *Electronics*, vol. 12, no. 16, p. 3492, Aug. 2023. [Online]. Available: http://dx.doi.org/10.3390/electronics12163492

[41] V. Choi, "Minor-embedding in adiabatic quantum computation: I. the parameter setting problem," 2008. [Online]. Available: https://arxiv.org/abs/0804.4884

[42] E. Boros and A. Gruber, "On quadratization of pseudo-boolean functions," 2014.

[43] E. Boros, Y. Crama, and E. Rodríguez-Heck, "Compact quadratizations for pseudo-boolean functions," *Journal of Combinatorial Optimization*, vol. 39, no. 3, pp. 687–707, 2019. [Online]. Available: https://doi.org/10.1007%2Fs10878-019-00511-0

[44] N. Dattani, "Quadratization in discrete optimization and quantum mechanics," 2019. [Online]. Available: https://arxiv.org/abs/1901.04405

[45] D. Dobrynin, A. Renaudineau, M. Hizzani, D. Strukov, M. Mohseni *et al.*, "Energy landscapes of combinatorial optimization in ising machines," *Physical Review E*, vol. 110, no. 4, Oct. 2024. [Online]. Available: http://dx.doi.org/10.1103/PhysRevE.110.045308

[46] J. Nüßlein, S. Zielinski, T. Gabor, C. Linnhoff-Popien, and S. Feld, *Solving (Max) 3-SAT via Quadratic Unconstrained Binary Optimization*. Springer Nature Switzerland, 2023, p. 34–47. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-36030-5_3

[47] M. Hizzani, A. Heittmann, G. Hutchinson, D. Dobrynin, T. Van Vaerenbergh *et al.*, "Memristor-based hardware and algorithms for higher-order hopfield optimization solver outperforming quadratic ising machines," in *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2024, pp. 1–5.

[48] C. Campbell and E. Dahl, "QAOA of the highest order," in *2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*, 2022, pp. 141–146.

[49] L. Schmidbauer, E. Lobe, I. Schaefer, and W. Mauerer, "It's quick to be square: Fast quadratisation for quantum toolchains," 12 2024. [Online]. Available: https://arxiv.org/abs/2411.19934

[50] S. Nagies, K. T. Geier, J. Akram, D. Bantounas, M. Johanning *et al.*, "Boosting quantum annealing performance through direct polynomial unconstrained binary optimization," 2025. [Online]. Available: https://arxiv.org/abs/2412.04398

[51] K. Wintersperger, H. Safi, and W. Mauerer, *QPU-System Co-design for Quantum HPC Accelerators*. Springer International Publishing, 2022, p. 100–114. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-21867-5_7

[52] J. Roffe, "Quantum error correction: an introductory guide," *Contemporary Physics*, vol. 60, no. 3, p. 226–245, Jul. 2019. [Online]. Available: http://dx.doi.org/10.1080/00107514.2019.1667078

[53] N. P. Breuckmann and J. N. Eberhardt, "Quantum low-density parity-check codes," *PRX Quantum*, vol. 2, no. 4, Oct. 2021. [Online]. Available: http://dx.doi.org/10.1103/PRXQuantum.2.040101