Quantum Information Processing (2025) 24:338
https://doi.org/10.1007/s11128-025-04948-7

SAT, gadgets, Max2XOR, and quantum annealers

Carlos Ansótegui[1] · **Jordi Levy**[2]

Received: 11 October 2024 / Accepted: 20 September 2025 / Published online: 15 October 2025

Abstract
Quantum annealers are presented as quantum computers that can, with a high probability, optimize certain quadratic functions on Boolean variables in constant time. These functions are basically the Hamiltonian of Ising models that reach the ground energy state with a high probability after an annealing process. In some preliminary works, they have been proposed as a way to solve SAT. These Hamiltonians can be seen as Max2XOR problems, i.e., as the problem of finding an assignment that maximizes the number of XOR clauses of at most two variables that are satisfied. In this paper, we focus on introducing several gadgets to reduce SAT to Max2XOR. We show how they can be used to translate SAT instances to initial configurations of a quantum annealer.

Keywords Maximum satisfiability · MaxSAT · Quantum annealers · Satisfiability · SAT · Gadgets

1 Introduction

Quantum annealers

Quantum Computation exploits some particularities of quantum mechanics, such as superposition, interference, and entanglement, to solve hard problems such as those we face in Artificial Intelligence. Current quantum computers are still too small to compete with classical computers for large-scale industrial or practical problems, such as the optimization and AI problems used in the SAT competition, but the rapid increase in the number of qubits could replace them in some of these problems, specifically those with fewer variables but higher time complexity. Most research in quantum computing is focused on the model of quantum circuits, where gates are quantum operators that act on a qubit or a superposition of them.

✉ Jordi Levy
levy@iiia.csic.es

Carlos Ansótegui
carlos@diei.udl.cat

[1] Logic & Optimization Group, Universitat de Lleida, Lleida, Spain

[2] IIIA, CSIC, Barcelona, Spain

An alternative and equivalent model is *Quantum Annealing* [1]. D-Wave Systems Inc.[1] has constructed some computers based on this model of computation with more than 5000 qubits, surpassing (in the number of qubits) the rest of the manufacturers of quantum computers. In these machines, we can only decide the *coupling factor* J_{ij} between certain pairs of qubits i and j, and the *biases* h_i of each qubit i. From an initial situation described by the Hamiltonian $\mathcal{H} = \sum_i \sigma_i^x$, the system evolves into a system described by an Ising Hamiltonian

$$\mathcal{H}_{Ising} = \sum_{i \in V_G} h_i \sigma_i^z + \sum_{(i,j) \in E_G} J_{ij}\, \sigma_i^z\, \sigma_j^z$$

where σ_i^z is the z-Pauli operator acting on qubit i, and $G = (V_G, E_G)$ is a graph describing the architecture of the machine. During the process, the system is tried to be kept in the lowest-energy eigenstate, thus at the end, ideally, we get the lowest-energy eigenstate of the Ising model that encodes a minimization problem. This final state is classical, without superpositions. The probability of success depends on the difference of energy between the lowest-energy state and the next energy level. The Landau–Zener model describes the probability of success (finishing in the minimal energy state) for a two-state system with one 1/2-spin particle with energy gap Δ under the action of a varying external magnetic field at rate c as $p = 1 - e^{\frac{-\pi \Delta^2}{4c}}$. [21] argues that, although the quantum annealers are more complicated systems, the probability can be approximated by this two-level system.

In order to increase the probability of finding this minimum, the experiment is repeated many times. The ranges of biases and couplings depend on the particular model of the quantum annealer.[2] In this paper, for simplicity, we assume that all of them are in the range $h_i \in [-1, 1]$ and $J_{ij} \in [-1, 1]$, applying a renormalization to the weights (see Sect. 3) to ensure this when required. The structure of the machine also depends on the model. For our purposes, we simply assume that G is an undirected sparse graph. In other words, we will idealize quantum annealers as devices that are able to solve—in constant time and with high probability [3] that depends exponentially on the difference between the minimal solution and the next state—a quadratic function where only some quadratic terms are allowed.

In this paper, we will show that (obviating some technical details) a quantum annealer can be seen as a Max2XOR solver with constraint weights in the reals [0, 1]. Therefore, we focus on the problem of finding gadgets to reduce the satisfiability of a CNF formula (SAT) to the satisfiability of the *maximum* number of 2XOR constraints (Max2XOR) and analyze which (theoretical) features of the gadgets are desirable for quantum annealers. In some sense, a subproduct of our contribution could be seen as one more step to assess the maturity and potential of quantum annealers to solve the SAT problem.

[1] https://www.dwavesys.com/.

[2] For old Chimera-base D-Wave 2000 series, they were $h_i \in [-2, 2]$ and $J_{ij} \in [-1, 1]$. In more modern models, they are $h_i \in [-4, 4]$ and $J_{ij} \in [-2, 1]$.

[3] Here, we use high probability in the mathematical sense, as a nonzero probability.

Max2XOR, QUBO, and Ising model

Exclusive OR (XOR), here written $\oplus$, may be an alternative to the use of traditional OR to represent propositional formulas. In practice, many approaches combining SAT and XOR reasoning have been presented [3, 9, 13–20, 25–27]. By writing clauses $x_1 \oplus \cdots \oplus x_k$ as constraints $x_1 \oplus \cdots \oplus x_k = 1$ or $x_1 \oplus \cdots \oplus x_k = 0$, where 1 means true and 0 false, we can avoid the use of negation, because $\neg x \oplus C = k$ is equivalent to $x \oplus C = 1 - k$. The equivalent to the resolution rule for XOR constraints, called XOR resolution rule, is

$$\frac{x \oplus A = k_1 \quad x \oplus B = k_2}{A \oplus B = k_1 \oplus k_2}$$

where A and B are clauses. In the particular case of $A = B = \emptyset$ and $k_1 \neq k_2$, this rule concludes the contradiction $0 = 1$, that we represent as $\square$. The proof system containing only this rule allows us to produce polynomial refutations for any unsatisfiable set of XOR constraints, using Gaussian elimination. Therefore, unless $P = NP$, we cannot polynomially translate any propositional formula into an equivalent conjunction of XOR constraints.[4] However, the SAT problem can be reduced to the Max2XOR problem, i.e., the problem of maximizing the number of 2XOR constraints that can be satisfied simultaneously.

Formally, a Max2XOR problem is a set of pairs

$$\{_{(w_1)}\, C_1 = k_1, \ldots, _{(w_n)}\, C_n = k_n\},$$

where w_i are positive rational numbers (we can also consider the special case of natural weights or the generalization to positive real weights) representing the penalty for violating the constraint, C_i are clauses form by one variable or the XOR disjuntion (or sum modulo two) of two variables, and k_i are either 0 or 1, representing $false$ and $true$, respectively. A solution is an optimal assignment that maximizes the sum of the weights of satisfied constraints. For example, $\{_{(1/2)}\, x = 1,\ _{(1/2)}\, y = 1,\ _{(1/2)}\, x + y = 1\}$ is a Max2XOR problem where the set of solutions (optimal assignments) are exactly the set of solutions of the SAT formula $\{x \vee y\}$, i.e., the assignments that assign x, or y, or both, to one. All these three optimal assignments satisfy two constraints with a total weight equal to one.

A Quadratic Unconstrained Binary Optimization (QUBO) problem is a minimization problem:

$$\min_{x_i \in \{0,1\}} \sum_i a_i x_i + \sum_{i<j} b_{ij} x_i x_j$$

[4] We can translate any OR clause $x_1 \vee \cdots \vee x_k$ into the set of weighted XOR constraints: $\bigcup_{S \subseteq \{1,\ldots,k\}} \{_{(1/2^{k-1})} \bigoplus_{i \in S} x_i = 1\}$ This translation allows us to reduce SAT to MaxXOR. However, the reduction is not polynomial, because it translates every clause of size k into $2^k - 1$ constraints. In Sect. 4, we describe a polynomial reduction that avoids this exponential explosion on expenses of introducing new variables.

Notice that any Max2XOR problem Γ may be translated into an equivalent QUBO problem, taking:

$$a_i = \sum_{(w)\, x_i=0\in\Gamma} w - \sum_{(w)\, x_i=1\in\Gamma} w + \sum_{(w)\, x_i+y=0\in\Gamma} w - \sum_{(w)\, x_i+y=1\in\Gamma} w$$
$$b_{ij} = -2\sum_{(w)\, x_i+x_j=0\in\Gamma} w + 2\sum_{(w)\, x_i+x_j=1\in\Gamma} w$$

Conversely, any QUBO problem may be easily translated into a Max2XOR problem (where all weights w are positive, whereas a_i and b_{ij} may be negative).

Any QUBO problem can be translated into an Ising formulation

$$\min_{z_i\in\{-1,+1\}} \sum_i h_i z_i + \sum_{i<j} J_{ij} z_i z_j$$

via the bijection $h_i = \frac{1}{2}a_i + \frac{1}{4}\sum_j b_{ij}$, $J_{ij} = \frac{1}{4}b_{ij}$, and similarly for any Max2XOR problem Γ via $h_i = \frac{1}{2}(\sum_{(w)\, x_i=0\in\Gamma} w - \sum_{(w)\, x_i=1\in\Gamma} w)$ and $J_{ij} = \frac{1}{2}(\sum_{(w)\, x_i+x_j=1\in\Gamma} w - \sum_{(w)\, x_i+x_j=0\in\Gamma} w)$. Therefore, Max2XOR, QUBO, and Ising model annealing are three NP-complete equivalent problems. For example,

- The Max2XOR problem $\{_{(1)}\, x_1 = 0,\ _{(1)}\, x_1 + x_2 = 0\}$,
- The QUBO problem $\min_{x_1,x_2\in\{0,1\}} 2x_1 + x_2 - 2x_1x_2$, and
- The Ising problem $\min_{z_1,z_2\in\{-1,+1\}} \frac{1}{2}z_1 - \frac{1}{2}z_1z_2$

are all of them equivalent. In all these problems, we can multiply all weights/coefficients by the same constant, obtaining an equivalent problem. However, we define the transformations to preserve the *energy gap* between the lowest-energy state and the next classical state.

We can see quantum annealers as hardware to try to some this problem in any of these three presentations. In this paper, to formalize the reduction from SAT to Max2XOR (or equivalently to QUBO or Ising), we will use gadgets.

Gadgets

Traditionally, the word "gadget" is used to denote a finite combinatorial structure that allows translating constraints of one problem to constraints of another. In [28], the notion is formalized, defining a (α, β)-*gadget* as a function from a family of constraints $\mathcal{F}_1$ to another family $\mathcal{F}_2$ that translate every constraint $f(\vec{x})$ in $\mathcal{F}_1$ to a set of β constraints $\{g_i(\vec{x}, \vec{b})\}_{i=1,\ldots,\beta}$ in $\mathcal{F}_2$, where the b's are (fresh) *auxiliary variables* such that, when $f(\vec{x})$ is satisfied for some assignment of the x's, we can find an assignment to the b's that satisfy α constraints $g(\vec{x}, \vec{b})$. Conversely, when $f(\vec{x})$ is falsified, no assignment for the b's satisfies strictly more than $\alpha - 1$ constraints $g(\vec{x}, \vec{b})$. If, additionally, when $f(\vec{x})$ is falsified, some assignment for the b's satisfy exactly $\alpha - 1$ constraints $g(\vec{x}, \vec{b})$, we say that the gadget is *strict*. This definition can be generalized to *weighted* constraints, where the weights are real numbers.[5] Then,

[5] Although a more detailed analysis would show that we can restrict them to be rational numbers.

instead of the number of satisfied constraints, we talk about the sum of the weights of satisfied constraints.

There is a long tradition of the use of gadgets. For instance, the $(k-2, k-2)$-gadget:

$$x_1 \vee \cdots \vee x_k \rightarrow \begin{cases} x_1 \vee x_2 \vee b_1, \\ \neg b_1 \vee x_3 \vee b_2, \\ \cdots, \\ \neg b_{k-4} \vee x_{k-2} \vee b_{k-3}, \\ \neg b_{k-3} \vee x_{k-1} \vee x_k \end{cases} \tag{1}$$

reduces kSAT to 3SAT. The following (weighted) (3.5, 4)-gadget [28] reduces 3SAT to Max2SAT:

$$x_1 \vee x_2 \vee x_3 \rightarrow \begin{cases} {}_{(1/2)}\, x_1 \vee x_3, \ {}_{(1/2)}\, \neg x_1 \vee \neg x_3, \\ {}_{(1/2)}\, x_1 \vee \neg b, \ {}_{(1/2)}\, \neg x_1 \vee b, \\ {}_{(1/2)}\, x_3 \vee \neg b, \ {}_{(1/2)}\, \neg x_3 \vee b, \\ {}_{(1)}\, x_2 \vee b \end{cases} \tag{2}$$

In [2], new gadgets reducing SAT to Max2SAT are introduced. The authors show, formally and empirically, that the new reduction allows solving efficiently the Pigeon Hole Principle, i.e., they prove that there exists a polynomial MaxSAT resolution proof [7] and that a general-purpose MaxSAT solver can solve the resulting Max2SAT formula.

Defining $Opt(P)$ as the maximal number of satisfied constraints of a problem P, $Cost(P)$ as the minimal number of falsified constraints, and $Weight(P)$ the number of constraints in P, we have that, if P is translated into P' using a (α, β)-gadget, then

$$\begin{aligned} Opt(P') &\leq (\alpha - 1)Weight(P) + Opt(P) \\ Cost(P') &\geq (\beta - \alpha)Weight(P) + Cost(P) \end{aligned}$$

with equalities if the gadget is strict. This allows us to obtain an algorithm to maximize P, using an algorithm to maximize P'.

When $\alpha = \beta$, we preserve the cost, hence the satisfiability. Then, given a decision algorithm for P', we get a decision algorithm for P.

Moreover, when we have a p-approximation algorithm for P', we can get an $(1 - \alpha(1-p))$-approximation algorithm for P. Therefore, traditionally, we were interested in gadgets with minimal α in order to minimize the error in the approximation.

If the algorithm is based on deriving empty clauses, i.e., that prove lower bounds for the cost, we will be interested in gadgets that minimize $\beta - \alpha$. Notice that in the case that P is a decision problem, then we can say that P is unsatisfiable if, and only if, $Cost(P') \geq (\beta - \alpha)Weight(P) + 1$.

In this paper, we are interested in maximizing another feature of gadgets, that we will call *energy gap*, and will be introduced later. We will also discuss on the number of auxiliary variables the gadget introduces and how *flexible* is the structure itself of the gadget.

Related work

Santra et al. [24] experimentally study the performance of one of the first quantum annealers, with 108 qubits, on random Max2SAT problems. They showed that the probability of success decreases with clause density due to a shrinking energy gap between the ground and first excited states, and that this probability is not correlated with the runtime of classical MaxSAT solvers such as akmaxsat. Their experiments also highlighted the impact of limited qubit connectivity on the encodings.

Using the Hamiltonian $H = \sum_{(s\cdot x_i \vee s'\cdot x_j)\in\Gamma} \frac{\mathbb{I}-s\cdot\sigma_i^z}{2}\frac{\mathbb{I}-s'\cdot\sigma_j^z}{2}$, where $s, s' \in \{1, -1\}$ represent the sign of the variable in the clause, they get an energy penalty of 1 for every clause violated by an assignment (the energy in an observed state is equal to the number of clauses violated by the assignment it represents). This Hamiltonian can be encoded with the biases $h_i = -1/4 \sum_{(s\cdot x_i \vee s'\cdot x_j)\in\Gamma} s$ and couplings $J_{ij} = 1/4 \sum_{(s\cdot x_i \vee s'\cdot x_j)\in\Gamma} s \cdot s'$. To ensure that all of them are in the range $[-1, 1]$, we have to re-scale the Hamiltonian, multiplying by $1/\max\{\max_i h_i, \max_{i,j} J_{ij}\}$. In the best case, when the density of clauses is low, this factor is 4, which results in an energy penalty of 4 for every violated clause. But, for high clause densities, the factor is smaller, which results in a lower precision in the method.

Chancellor et al. [8] study the translation of kSAT and parity problems to quantum annealing. They observe that $x_1 \vee x_2 \vee x_3$ can be reduced to $x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3 + x_1x_2x_3$, or in general, $x_1 \vee \cdots \vee x_k$ to $1 + \sum_{S\subseteq\{1,\ldots,k\}} (-1)^{|S|+1} \prod_{i\in S} x_i$. Their method introduces dense couplings and requires an exponential number of auxiliary qubits as k increases.

Nüßlein et al. [22] evaluate different reductions of 3SAT to quantum annealing and proposed a new method that improves over those of [8] and [10]. They reported better performance in terms of qubit usage and accuracy, although they did not compare their approach with the methods of [5, 6] or with the new gadgets presented in this paper.

Douglass et al. [12] explore the use of a quantum annealer for constructing SAT filters, translating MaxCUT, Not-All-Equal 3SAT, and 2-in-4-SAT into Hamiltonians. This translation is almost direct. They already noted that solving general SAT would require the introduction of auxiliary (ancillary) variables.

Bian et al. [5, 6] using a 2048 qubits quantum annealer, explore the feasibility of solving SAT, without restrictions on clause size. They decomposed formulas using Tseitin encodings into constraints of at most three variables, which are then mapped into Hamiltonians.[6] They use multiple qubits per logical variable, ensuring (using heuristics) that there is a path of coupled qubits that connect all these occurrences. Although the authors did not explicitly frame their work in terms of gadgets, their decomposition can be naturally understood in this way.

Choi [11] analyzes the problem of allocating variables into qubits and models it as a variant of graph embedding, where in the process, some transformations are allowed,

[6] For instance, we can decompose $x_1 \vee x_2 \vee x_3 \vee x_4$ as $\{b_1 \leftrightarrow x_1 \vee x_2, b_2 \leftrightarrow x_3 \vee x_4, b_1 \vee b_2\}$. Then, each one of these constraints or subformulas contributes to the Hamiltonian. For instance, $b_1 \leftrightarrow x_1 \vee x_2$ contributes by adding $5/2 + 1/2x_1 + 1/2x_2 - b_1 + 1/2x_1x_2 - x_1b_1 - x_2b_1$ to the Hamiltonian. (Abusing from the notation, we identify the variable, that takes Boolean values 0 or 1, with the qubit representing it, which takes values +1 or -1 when measured). To ensure that the coupling factors are in the range $[-1, 1]$, in some cases, they have to re-normalize the entire Hamiltonian, dividing all factors by a constant.

basically the contractions of paths into edges. Bian et al. [4] also analyze some methods with the same purpose.

2 Preliminaries

A k-ary *constraint function* is a Boolean function $f : \{0,1\}^k \to \{0,1\}$. A *constraint family* is a set $\mathcal{F}$ of constraint functions (with possibly distinct arities). A *constraint*, over variables $V = \{x_1, \ldots, x_n\}$ and constraint family $\mathcal{F}$, is a pair formed by a k-ary constraint function $f \in \mathcal{F}$ and a subset of k variables, noted $f(x_{i_1}, \ldots, x_{i_k})$, or $f(\vec{x})$ for simplicity. A *(weighted) constraint problem* or *(weighted) formula* P, over variables V and constraint family $\mathcal{F}$, is a set of pairs (weight, constraint) over V and $\mathcal{F}$, where the weight is a positive rational number, denoted $P = \{_{(w_1)}\ f_1(x^1_{i_1}, \ldots, x^1_{i_{k_1}}), \ldots, _{(w_m)}\ f_m(x^m_{i_1}, \ldots, x^m_{i_{k_m}})\}$. The *weight* of a problem is $Weight(P) = w_1 + \cdots + w_m$.

An *assignment* is a function $I : \{x_1, \ldots, x_n\} \to \{0,1\}$. We say that an assignment I *satisfies* a constraint $f(x_{i_1}, \ldots, x_{i_k})$, if $I(f(\vec{x})) =_{\text{def}} f(I(x_{i_1}), \ldots, I(x_{i_k})) = 1$. The value of an assignment I for a constraint problem $P = \{_{(w_i)}\ f_i(\vec{x})\}_{i=1,\ldots,m}$, is the sum of the weights of the constraints that this assignment satisfies, i.e., $I(P) = \sum_{i=1}^{m} w_i\, I(f_i(\vec{x}))$. We also define the sum of weights of unsatisfied clauses as $\overline{I}(P) = \sum_{i=1}^{m} w_i\, (1 - I(f_i(\vec{x})))$. An assignment I is said to be *optimal* for a constraint problem P, if it maximizes $I(P)$. We define this optimum as $Opt(P) = \max_I I(P)$. We also define $Cost(P) = \min_I \overline{I}(P)$, i.e., the minimum sum of weights of falsified constraints, that fulfills $Opt(P) + Cost(P) = Weight(P)$.

Many optimization problems may be formalized as the problem of finding an optimal assignment for a constraint problem. In the following, we define some of them:

1. MaxkSAT is the constraint family defined by the constraint functions of the form $f(x_1, \ldots, x_r) = l_1 \vee \cdots \vee l_r$, where every l_i may be either x_i or $\neg x_i$ and $r \leq k$.
2. MaxkXOR is the constraint family defined by the constraint functions $f(x_1, \ldots, x_r) = x_1 \oplus \cdots \oplus x_r$ and $f(x_1, \ldots, x_r) = x_1 \oplus \cdots \oplus x_r \oplus 1$, where $r \leq k$.

In this paper, we are interested in the problem Max2XOR, which has got very little attention in the literature, probably because it is quite similar to MaxCUT.

Next, we define gadgets as a transformation from constraints into sets of weighted constraints (or problems). These transformations can be extended to define transformations of problems into problems.

Definition 1 Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two constraint families. A (α, β)-*gadget* from $\mathcal{F}_1$ to $\mathcal{F}_2$ is a function that, for any constraint $f(\vec{x})$ over $\mathcal{F}_1$ returns a weighted constraint problem $P = \{_{(w_i)}\ g_i(\vec{x}, \vec{b})\}_{i=1,\ldots,m}$ over $\mathcal{F}_2$ and variables $\{\vec{x}\} \cup \{\vec{b}\}$, where $\vec{b}$ are auxiliary variables distinct from $\vec{x}$, such that $\beta = \sum_{i=1}^{m} w_i$ and, for any assignment $I : \{\vec{x}\} \to \{0,1\}$:

1. If $I(f(\vec{x})) = 1$, for any extension of I to $I' : \{\vec{x}\} \cup \{\vec{b}\} \to \{0,1\}$, $I'(P) \leq \alpha$ and there exist one of such extension with $I'(P) = \alpha$.
2. If $I(f(\vec{x})) = 0$, for any extension of I to $I' : \{\vec{x}\} \cup \{\vec{b}\} \to \{0,1\}$, $I'(P) \leq \alpha - 1$ and if, additionally, there exists one of such extension with $I'(P) = \alpha - 1$ we say the gadget is strict.

Lemma 1 *The composition of a* (α_1, β_1)*-gadget from* $\mathcal{F}_1$ *to* $\mathcal{F}_2$ *and a* (α_2, β_2)*-gadget from* $\mathcal{F}_2$ *to* $\mathcal{F}_3$ *results into a* $(\beta_1 (\alpha_2 - 1) + \alpha_1,\ \beta_1\beta_2)$*-gadget from* $\mathcal{F}_1$ *to* $\mathcal{F}_3$.

Proof The first gadget multiplies the total weight of constraints by β_1 and the second by β_2. Therefore, the composition multiplies it by $\beta = \beta_1\beta_2$.

For any assignment, if the original constraint is falsified, the optimal extension after the first gadget satisfies constraints with a weight of $\alpha_1 - 1$, and falsifies the rest $\beta_1 - (\alpha_1 - 1)$. The second gadget satisfies constraints for a weight of $\alpha_2 - 1$ of the falsified plus α_2 of the satisfied. Therefore, the composition satisfies constraints with a total weight $(\alpha_2 - 1)(\beta_1 - (\alpha_1 - 1)) + \alpha_2(\alpha_1 - 1) = \beta_1(\alpha_2 - 1) + \alpha_1 - 1$.

If the original constraint is satisfied, the optimal extension after the first gadget satisfies α_1 and falsifies the rest $\beta_1 - \alpha_1$. After the second gadget, the weight of satisfied constraints is $(\alpha_2 - 1)(\beta_1 - \alpha_1) + \alpha_2\alpha_1 = \beta_1(\alpha_2 - 1) + \alpha_1$.

The difference between both situations is one; hence, the composition is a gadget, and $\alpha = \beta_1(\alpha_2 - 1) + \alpha_1$. □

3 Quantum annealers as Max2XOR solvers

As we mention in the introduction, a quantum annealer can be seen as a Max2XOR solver. Roughly speaking, it is able to reach the state that minimizes the energy of the Ising Hamiltonian $\mathcal{H} = \sum_i h_i\sigma_i^z + \sum_{(i,j)\in E} J_{ij}\sigma_i^z\sigma_j^z$, where σ_i^z is the z-Pauli operator acting on qubit i. For every linear term $h_i\sigma_i^z$ of this Hamiltonian, we can get a 2XOR constraint $_{(2h_i)}\, x_i = 0$, when $h_i > 0$, or $_{(-2h_i)}\, x_i = 1$, when $h_i < 0$. Similarly, for each quadratic term $J_{ij}\sigma_i^z\sigma_j^z$, we get a constraint $_{(2J_{ij})}\, x_i \oplus x_j = 1$, when $J_{ij} > 0$, or $_{(-2J_{ij})}\, x_i \oplus x_j = 0$, when $J_{ij} < 0$. Conversely, any Max2XOR problem may be mapped into an Ising Hamiltonian.

In the following, we review with further detail the related work on constructing Hamiltonians for SAT and MaxSAT from the perspective of gadgets.

When in [24], they say that every 2SAT clause $x_i \vee x_j$ contributes to the Hamiltonian as $\frac{\mathbb{I}-\sigma_i^z}{2}\frac{\mathbb{I}-\sigma_j^z}{2} = \frac{1}{4}\mathbb{I} - \frac{1}{4}\sigma_i^z - \frac{1}{4}\sigma_j^z + \frac{1}{4}\sigma_i^z\sigma_j^z$, they are implicitly defining a $(1, 3/2)$-gadget from 2SAT to 2XOR:

$$x_i \vee x_j \rightarrow \begin{cases} {}_{(1/2)}\, x_i = 1, \\ {}_{(1/2)}\, x_j = 1, \\ {}_{(1/2)}\, x_i \oplus x_j = 1 \end{cases} \tag{3}$$

In the following, we only describe the translation of clauses $x_1 \vee \cdots \vee x_k$ where all variables are positive. We can easily generalize the transformation when any of the variables x is negated, simply by recalling that $\neg x \oplus y = k$ is equivalent to $x \oplus y = 1 - k$, and $\neg x \oplus \neg y = k$ is equivalent to $x \oplus y = k$.

Notice also that we normalize the weights to ensure that the gap between the sum of weights when the original constraint is falsified $(\alpha - 1)$ and the sum when it is satisfied (α) is just one. Later, when from the Max2XOR problem, we construct the Hamiltonian, we can multiply the weights for the maximal value that still allows biases and couplings to be inside their corresponding ranges $[-1, 1]$, while we try

to maximize this gap. For example, in the case of the previous gadget $\{_{(1/2)}\, x_1 = 1,\ _{(1/2)}\, x_2 = 1,\ _{(1/2)}\, x_1 \oplus x_2 = 1\}$, when we transform it into a Hamiltonian $-\frac{1}{4}\sigma_1^z - \frac{1}{4}\sigma_2^z + \frac{1}{4}\sigma_1^z\sigma_2^z$, we could multiply all weights by 4 and still getting all coefficients in the range $[-1, 1]$. This allows us to enlarge the energy gap between the lowest-energy state and the next classical state to 4. We call this quantity the *energy gap* of the gadget, noted ΔE. Remark that, as defined below, this energy gap only corresponds to the minimal difference of energy levels when we consider the translation of *just one* clause. When more clauses are involved, if we want to keep biases and coupling inside their corresponding ranges, the situation is more complicated, obtaining smaller energy differences when normalizing. However, what we define as *energy gap* is a good indicator of how good the gadget is, as it has been experimentally observed in [23].

Formally,

Definition 2 Given a gadget P to Max2XOR or, in general, a Max2XOR problem P, we define *energy gap* as

$$\Delta E = \min\left\{\min_i \frac{1}{|h_i|}, \min_{i,j} \frac{1}{|J_{ij}|}\right\}$$

where, remember that

$$h_i = \tfrac{1}{2}\left(\sum_{(w)\, x_i=0\in P} w - \sum_{(w)\, x_i=1\in P} w\right)$$
$$J_{ij} = \tfrac{1}{2}\left(\sum_{(w)\, x_i\oplus x_j=1\in P} w - \sum_{(w)\, x_i\oplus x_j=0\in P} w\right)$$

Assuming that the problem P is simplified, i.e., contains a unique constraint of the form $x_i = 0$ or $x_i = 1$, and a unique constraint of the form $x_i \oplus x_j = 0$ or $x_i \oplus x_j = 1$, this simplifies to

$$\Delta E = \min_{(w)\, C\in P} \frac{2}{w}$$

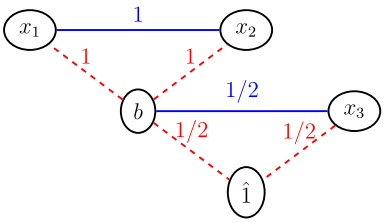
As we mentioned, Nüßlein et al. [22] propose two ways to reduce Max3SAT to QUBO and compares them with previous methods. Here, we will only discuss the reduction that they call NÜSSLEIN^{n+m}, for which they report the best results (the other ones make use of more additional variables). In this reduction, each clause $x_1 \vee x_2 \vee x_3$ contributes to the QUBO problem as:

$$2x_1x_2 - 2x_1b - 2x_2b + x_3b - x_3 + b$$

Formalized as a contribution to an Ising Hamiltonian, this is:

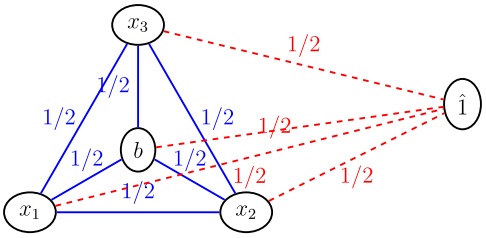
$$H = \frac{z_1z_2}{2} - \frac{z_1b}{2} - \frac{z_2b}{2} + \frac{z_3b}{4} - \frac{z_3}{4} - \frac{b}{4}$$

We can formalize it as the $(5/2, 9/2)$-gadget with $\Delta E = 2$ from 3SAT to Max2XOR represented in Fig. 1. Notice that the weights in the 2XOR constraints correspond to



$$
\begin{aligned}
&_{(1)}\ x_1 \oplus x_2 = 1\\
&_{(1)}\ x_1 \oplus b = 0\\
&_{(1)}\ x_2 \oplus b = 0\\
&_{(1/2)}\ x_3 \oplus b = 1\\
&_{(1/2)}\ x_3 = 1\\
&_{(1/2)}\ b = 1
\end{aligned}
$$

Fig. 1 Graphic representation of the $(5/2, 9/2)$-gadget with $\Delta E = 2$ from 3SAT to Max2XOR proposed in [22]. Blue edges represent equal-one constraints, and red dashed edges are equal-zero constraints



$$
\begin{aligned}
&_{(1/2)}\ x_1 \oplus x_2 = 1\\
&_{(1/2)}\ x_1 \oplus x_3 = 1\\
&_{(1/2)}\ x_2 \oplus x_3 = 1\\
&_{(1/2)}\ x_1 \oplus b = 1\\
&_{(1/2)}\ x_2 \oplus b = 1\\
&_{(1/2)}\ x_3 \oplus b = 1\\
&_{(1/2)}\ x_1 = 1\\
&_{(1/2)}\ x_2 = 1\\
&_{(1/2)}\ x_3 = 1\\
&_{(1/2)}\ b = 1
\end{aligned}
$$

Fig. 2 Graphic representation of the $(3, 5)$-gadget with $\Delta E = 4$ from 3SAT to Max2XOR obtained with the method proposed in [8]

the coefficients of the Hamiltonian multiplied by 2. In the rest of the paper, we will use this graphical representation where solid blue lines between x and y represent $x \oplus y = 1$ and red dashed lines represent $x \oplus y = 0$. Constraints $x = 0$ and $x = 1$ are graphically represented using lines to a special node $\hat{1}$.

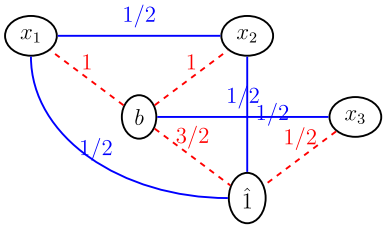
Chancellor et al. [8] propose a method to translate kSAT to QUBO. It is based on the equivalence of the kSAT constraints $x_1 \vee \cdots \vee x_k$ and the polynomial $1 + \sum_{S \subseteq \{1,\ldots,k\}} (-1)^{|S|+1} \prod_{i \in S} x_i$ and the use of a new qubit to represent terms of size bigger than two. There are still different ways to select the coupling factors. For a convenient election, we can get the gadget represented in Fig. 2.

Bian et al. [5, 6] propose a method that can be used to reduce kSAT (or even more general formulas) to Max2XOR. It is based on the translation of each constraint of the form $x_1 \vee x_2 \leftrightarrow x_3$ as a contribution of

$$\frac{z_1 z_2}{2} - z_1 z_3 - z_2 z_3 + \frac{z_1}{2} + \frac{z_2}{2} - z_3 + \frac{5}{2}$$

to the Hamiltonian of a Ising model. Implicitly, it defines the non-strict $(3, 9/2)$-gadget with $\Delta E = 2$:

$$(x_1 \vee x_2 \leftrightarrow b) \rightarrow \begin{cases} _{(1/2)}\ x_1 = 0\\ _{(1/2)}\ x_2 = 0\\ _{(1)}\ b = 1\\ _{(1/2)}\ x_1 \oplus x_2 = 1\\ _{(1)}\ x_1 \oplus b = 0\\ _{(1)}\ x_2 \oplus b = 0 \end{cases} \tag{4}$$



$_{(1/2)}\, x_1 = 0$
$_{(1/2)}\, x_2 = 0$
$_{(1/2)}\, x_3 = 1$
$_{(3/2)}\, b = 1$
$_{(1/2)}\, x_1 \oplus x_2 = 1$
$_{(1)}\, x_1 \oplus b = 0$
$_{(1)}\, x_2 \oplus b = 0$
$_{(1/2)}\, x_3 \oplus b = 1$

Fig. 3 Graphic representation of the (4, 6)-gadget with $\Delta E = 4/3$ obtained by Tseitin encoding of a 3SAT clause, as proposed by [5, 6]

Notice that the values of the polynomial coefficients correspond to the weights of the constraints in the gadget. The independent term $5/2$ is only introduced in order to ensure that the ground-state energy is zero. Recall that this is just a convention, and the values of this term are not restricted to any range.

The application of the Tseitin encoding also can be seen as a gadget. For instance, the (2, 2)-gadget to reduce 3SAT

$$x_1 \vee x_2 \vee x_3 \rightarrow \begin{cases} x_1 \vee x_2 \leftrightarrow b \\ b \vee x_3 \end{cases} \tag{5}$$

The composition of this gadget (5) with both (3) and (4), results in the (4, 6)-gadget with $\Delta E = 1$ from 3SAT to Max2XOR used in [6] to translate 3SAT clauses into Hamiltonian components, and represented here in Fig. 3.

4 New gadgets from SAT to Max2XOR

As we have seen in the previous section, the construction of Hamiltonians to solve a problem P with a quantum annealer is basically the search for a gadget from the constraints in P to Max2XOR. In general, there are three aspects that we would like to optimize in this gadget. First, we want to reduce the number of auxiliary variables (or ancillary variables in quantum terminology) because they imply using more qubits. Second, we want to relax the structure of the gadget, and in general avoid using gadgets with a dense structure (such as cliques), because there are limitations in the architecture of the quantum annealer, i.e., the graph of allowed couplings. Third, we want to maximize the energy gap, because this reduces the probability of errors.

4.1 A gadget from 3SAT to Max2XOR

An easy way to reduce SAT to Max2XOR is to reduce SAT to 3SAT, using the gadget (1), this to Max2SAT, using the gadget (2) [2, 28], and this to Max2XOR, using the gadget (3).

The concatenation of the gadget (2) and the gadget (3) results into the (2, 3)-gadget with $\Delta E = 4$ described in Figure 4. This gadget is similar to the gadgets described in

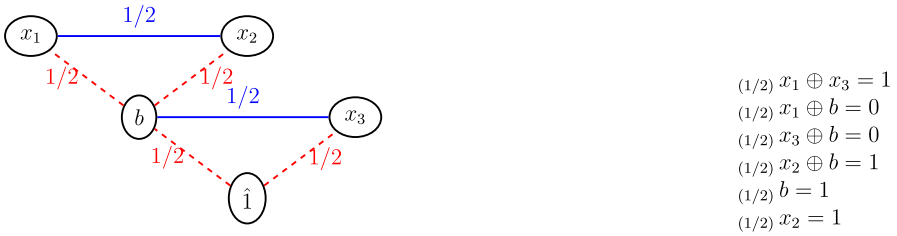


Fig. 4 Graphical representation of the $(2, 3)$-gadget with $\Delta E = 4$ reducing Max3SAT to Max2XOR and based on [28] gadget from Max3SAT to Max2SAT

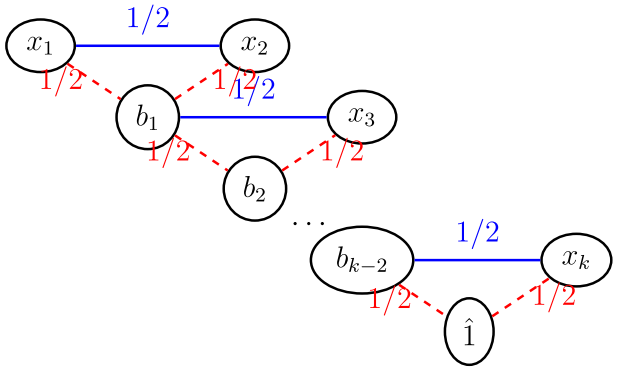


Fig. 5 Graphical representation of $T^0(x_1 \vee \cdots \vee x_k, \hat{1})$, defining a $(k-1, 3/2(k-1)$-gadget with $\Delta E = 4$ form kSAT to Max2XOR that introduces $k-2$ variables. For $k = 3$, this gadget corresponds to the one in Fig. 4

Figures 1, 2, and 3. However, this gadget is optimal in terms of minimizing α, β, ΔE and the number of auxiliary variables.

However, when we concatenate gadget (1) with the gadget in Figure 4 to obtain a reduction from kSAT to Max2XOR, for $k \geq 4$, we get a $(2(k-2),\ 3(k-2))$-gadget with $2k-5$ auxiliary variables and $\Delta E = 4$. This one is not optimal, in the sense that, as we will see below, there exist other gadgets that introduce fewer variables with the same ΔE, and smaller α and β parameters.

In our experience, in general, if we concatenate two gadgets, even if both are optimal (in some parameter), the resulting gadget can be suboptimal (in that parameter). Therefore, it is usually better to compute *direct* gadgets. In the following, we describe direct and better reductions (for some of the parameters) from SAT to Max2XOR Fig. 5.

4.2 A regular-like gadget

Here, we present a new gadget inspired by the Refined-Regular gadget from SAT to Max2SAT introduced by [2]. The reduction is based on a function T^0 that takes as parameters a SAT clause and a variable. The use of this variable is for technical

reasons, to make simpler the recursive definition and proof of Lemma 2. Later (see Theorem 1), it will be replaced by the constant one.

Definition 3 Given a SAT clause $x_1 \vee \cdots \vee x_k$ and an auxiliary variable b, define the Max2XOR problem $T^0(x_1 \vee \cdots \vee x_k, b)$ recursively as follows:

$$T^0(x_1 \vee x_2, b) = \begin{cases} {}_{(1/2)}\, x_1 \oplus x_2 = 1 \\ {}_{(1/2)}\, x_1 \oplus b = 0 \\ {}_{(1/2)}\, b \oplus x_2 = 0 \end{cases}$$

for binary clauses, and

$$T^0(x_1 \vee \cdots \vee x_k, b) = T^0(x_1 \vee \cdots \vee x_{k-1}, b') \cup \begin{cases} {}_{(1/2)}\, b' \oplus x_k = 1 \\ {}_{(1/2)}\, b' \oplus b = 0 \\ {}_{(1/2)}\, b \oplus x_k = 0 \end{cases}$$

for $k \geq 3$.

Lemma 2 *Consider the Max2XOR problem $T^0(x_1 \vee \cdots \vee x_k, b_{k-1})$. We have:*

1. *For any assignment $I : \{x_1, \ldots, x_k\} \to \{0, 1\}$, the extension of the assignment as $I'(b_i) = I(x_{i+1})$, for $i = 1, \ldots, k-1$, maximizes the number of satisfied 2XOR constraints, that is $2\,(k-1)$.*
2. *For any assignment $I : \{x_1, \ldots, x_k, b_{k-1}\} \to \{0, 1\}$ satisfying $I(b_{k-1}) = 1$, the extension of the assignment as*

$$I'(b_i) = \begin{cases} 1 & \text{if } I(x_{i+2}) = \cdots = I(x_k) = 0 \\ I(x_{i+1}) & \text{otherwise} \end{cases}$$

for $i = 1, \ldots, k-2$, maximizes the number of satisfied 2XOR constraints, that is $2\,(k-2)$, if $I(x_1) = \cdots = I(x_k) = 0$, or $2\,(k-1)$ otherwise.

Proof See Lemma 3, where a more general result is proved. □

Theorem 1 *The translation of every clause $x_1 \vee \cdots \vee x_k$ into $T^0(x_1 \vee \cdots \vee x_k, \hat{1})$ defines a $(k-1, 3/2(k-1))$-gadget from kSAT to Max2XOR that introduces $k-2$ auxiliary variables and has $\Delta E = 4$.*

Proof The soundness of the reduction is based on the second statement of Lemma 2. Assume that we force the variable $\hat{1}$ to be interpreted as one, i.e., $I(\hat{1}) = 1$, or that, alternatively, we simply replace it by the constant 1. Any assignment satisfying the clause can extend to satisfy at least $2(k-1)$ 2XOR constraints from $T^0(x_1 \vee \cdots \vee x_k, \hat{1})$, with a total weight equal to $k-1$. Any assignment falsifying it can only be extended to satisfy at most $2(k-2)$ 2XOR constraints, with a total weight $k-2$. Hence, $\alpha = k-1$. The number of 2XOR constraints is $3(k-1)$, all of them with weight $1/2$. Therefore, $\beta = 3/2(k-1)$. □

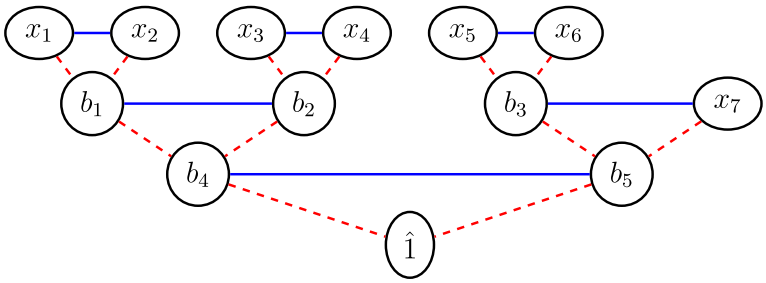


Fig. 6 A graphical representation of a Tree-like gadget for 7SAT. All constraints have weight 1/2

4.3 A tree-like gadget

In this section, we describe the Tree-like gadget, which is a generalization of the Regular-like gadget. Before describing the new gadget, we introduce an example for the clause $x_1 \vee \cdots \vee x_7$, in Fig. 6.

Definition 4 Given a clause $x_1 \vee \cdots \vee x_k$ and an auxiliary variable b, define the Max2XOR problem $T^t(x_1 \vee \cdots \vee x_k, b)$ recursively and non-deterministically as follows:

$$T^t(x_1 \vee \cdots \vee x_k,\ b) = \begin{cases} \{x_1 \oplus x_2 = 1, x_1 \oplus b = 0, x_2 \oplus b = 0\} & if\ k = 2 \\ T^t(x_1 \vee \cdots \vee x_{k-1},\ b') \cup & if\ k \geq 3 \\ \{b' \oplus x_k = 1, b' \oplus b = 0, x_k \oplus b = 0\} & \\ T^t(x_2 \vee \cdots \vee x_k,\ b') \cup & if\ k \geq 3 \\ \{b' \oplus x_1 = 1, b' \oplus b = 0, x_1 \oplus b = 0\} & \\ T^t(x_1 \vee \cdots \vee x_r,\ b') \cup & if\ k \geq 4, where \\ T^t(x_{r+1} \vee \cdots \vee x_k,\ b'') \cup & 2 \leq r \leq k-2 \\ \{b' \oplus b'' = 1, b' \oplus b = 0, b'' \oplus b = 0\} & \end{cases}$$

where all XOR constraints have weight 1/2.

Notice that the definition of T^t is not deterministic. When $k = 3$ we have two possible translations (applying the second or third case), and when $k \geq 4$ we can apply the second, third, and fourth cases, with distinct values of r, getting as many possible translations as possible binary trees with k leaves.

Notice also that the definition of T^0 corresponds to the definition of T^t where we only apply the first and second cases. Therefore, T^t is a generalization of T^0.

Lemma 3 *Consider the set of 2XOR constraints of $T^t(x_1 \vee \cdots \vee x_k, b)$ without weights. We have:*

1. *There are $3(k-1)$ constraints.*
2. *Any assignment satisfies at most $2(k-1)$ of them.*
3. *Any assignment $I : \{x_1, \ldots, x_k\} \to \{0, 1\}$ can be extended to an optimal assignment satisfying $I(b) = I(x_1 \vee \cdots \vee x_k)$ and $2(k-1)$ constraints.*
4. *Any assignment I satisfying $I(x_1 \vee \cdots \vee x_k) = 0$ and $I(b) = 1$ can be extended to an optimal assignment satisfying $2(k-2)$ constraints.*

Proof The first statement is trivial.

For the second, notice that for, each one of the $k-1$ *triangles* of the form $\{a \oplus b = 1, a \oplus c = 0, b \oplus c = 0\}$, any assignment can satisfy at most two of the constraints of each triangle.

The third statement is proved by induction. The base case, for binary clauses, is trivial. For the induction case, if we apply the second or third options of the definition of T, the proof is similar to Lemma 2. In the fourth case, assume by induction that there is an assignment extending $I : \{x_1, \ldots, x_r\} \to \{0, 1\}$ that verifies $I(b') = I(x_1 \vee \cdots \vee x_r)$ and satisfies $2(r-2)$ constraints of $T^t(x_1 \vee \cdots \vee x_r, b')$. Similarly, there is an assignment extending $I : \{x_{r+1}, \ldots, x_k\} \to \{0, 1\}$ that verifies $I(b'') = I(x_{r+1} \vee \cdots \vee x_k)$ and satisfies $2((k-r)-2)$ constraints of $T^t(x_{r+1} \vee \cdots \vee x_k, b'')$. Both assignments do not share variables, therefore, they can be combined into a single assignment. This assignment can be extended with $I(b) = b' \vee b''$. This ensures that it will satisfy two of the constraints from $\{b' \oplus b'' = 1, b' \oplus b = 0, b'' \oplus b = 0\}$. Therefore, it verifies $I(b) = I(x_1 \vee \cdots \vee x_k)$ and satisfies $2(r-2)+2((k-r)-2)+2 = 2(k-1)$ constraints of $T^t(x_1 \vee \cdots \vee x_k, b)$. Since this is the maximal number of constraints we can satisfy, this assignment is optimal.

The fourth statement is also proved by induction. The base case, for binary clauses, is trivial. For the induction case, consider only the application of the fourth case of the definition of T (the other cases are quite similar). We have to consider three possibilities:

If we extend $I(b') = I(b'') = 1$, by induction, we can extend I to satisfy $2(r-2)$ constraints of $T^t(x_1 \vee \cdots \vee x_r, b')$ and $2((k-r)-2)$ constraints of $T^t(x_{r+1} \vee \cdots \vee x_k, b'')$. It satisfies 2 constraints from $\{b' \oplus b'' = 1, b' \oplus b = 0, b'' \oplus b = 0\}$, hence a total of $2(k-3)$ constraints.

If we extend $I(b') = I(b'') = 0$, by the third statement of this lemma, we can extend I to satisfy $2(r-1)$ constraints of $T^t(x_1 \vee \cdots \vee x_r, b')$ and $2((k-r)-1)$ constraints of $T^t(x_{r+1} \vee \cdots \vee x_k, b'')$. It does not satisfy any constraints from $\{b' \oplus b'' = 1, b' \oplus b = 0, b'' \oplus b = 0\}$, hence a total of $2(k-2)$ constraints.

Finally, if we extend $I(b') = 1$ and $I(b'') = 0$ (or vice versa), by induction, we can extend I to satisfy $2(r-2)$ constraints of $T^t(x_1 \vee \cdots \vee x_r, b')$ and by the third statement $2((k-r)-2)$ constraints of $T^t(x_{r+1} \vee \cdots \vee x_k, b'')$. It also satisfies 2 constraints from $\{b' \oplus b'' = 1, b' \oplus b = 0, b'' \oplus b = 0\}$, hence a total of $2(k-2)$ constraints. □

Theorem 2 *The translation of every clause $x_1 \vee \cdots \vee x_k$ into $T^t(x_1 \vee \cdots \vee x_k, \hat{1})$ defines a $(k-1, 3/2(k-1))$-gadget from kSAT to Max2XOR that introduces $k-2$ auxiliary variables, with $\Delta E = 4$.*

Proof The theorem is a direct consequence of the two last statements of Lemma 3. □

If we compare Theorems 1 and 2, we observe that both establish the same values for $\alpha = k-1$ and $\beta = 3/2(k-1)$ in the (α, β)-gadget, and both introduce the same number $(k-2)$ of auxiliary variables. Therefore, a priori the parallel translation is a generalization of the sequential translation, but it does not have a clear (theoretical) advantage.

However, in the context of quantum annealers, we recall that not all possible couplings are allowed, only the ones in the graph defined by the architecture. To circumvent

this issue, the typical approach is to make *copies* (using extra qubits) of the variables involved in the Max2XOR constraint whose coupling is not allowed. In other words, if we have a constraint $_{(w)}\, x_i \oplus x_j = 1$ in the Max2XOR problem, but $(i, j) \notin E$, we have to find a path and replace the constraint by $\{_{(1)}\, x_i \oplus x_{k_1} = 0, \ldots, _{(1)}\, x_{k_{r-1}} \oplus x_r = 0, _{(w)}\, x_r \oplus x_j = 1\}$

In this sense, the more *flexible* structure of the parallel translation can help to fit more easily (i.e., with fewer extra qubits) the Max2XOR constraints generated by the gadget into the architecture of the quantum annealer.

4.4 A clique-like gadget

The previous gadgets have an energy gap $\Delta E = 4$, but they are not optimal with respect to the number of auxiliary variables. We can obtain gadgets that only require $\mathcal{O}(\log k)$ auxiliary variables, on expenses of decreasing the energy gap and increasing the density of constraints. Therefore, there is a trade-off between optimizing the energy gap and the number of auxiliary variables.

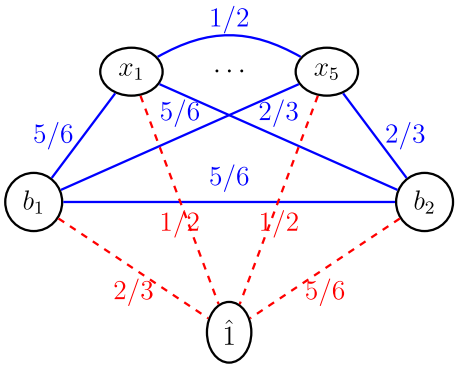
Definition 5 Given a clause $x_1 \vee \cdots \vee x_k$, where k is a power of 2, define the Max2XOR problem $T^c(x_1 \vee \cdots \vee x_k)$ as follows:

$$T^c(x_1 \vee \cdots \vee x_k) = \begin{cases} _{(1/2)}\, x_i = 1 & i = 1, \ldots, k \\ _{(2^{j-1})}\, b_j = 1 & j = 1, \ldots, \log k - 1 \\ _{(1/2)}\, x_i \oplus x_j = 1 & 1 \le i < j \le k \\ _{(2^{i+j-1})}\, b_i \oplus b_j = 1 & 1 \le i < j \le \log k - 1 \\ _{(2^{j-1})}\, x_i \oplus b_j = 1 & i = 1, \ldots, k \\ & j = 1, \ldots, \log k - 1 \end{cases}$$

Theorem 3 *The transformation of every clause $x_1 \vee \cdots \vee x_k$ into $T^c(x_1 \vee \cdots \vee x_k)$ defines a (α, β)-gadget from kSAT to Max2XOR that introduces $\log k - 1$ auxiliary variables, with*

$$\alpha = \frac{1}{2} k\,(k-1)$$
$$\beta = \frac{11\,k^2 - 15\,k + 4}{12}$$
$$\Delta E = 2^5\, k^{-2}$$

Proof The intuition for the construction of the gadget is: Apart from the variables x_i (with weight 1), we add auxiliary variables b_j with weight 2^j and a constant $\hat{0}$ with weight 1. The sum of all weights is $2k - 1$. The weight of every constraint $x \oplus y = 1$ is equal to one-half of the weight of x by the weight of y. No matter how x's are assigned, if at least one is assigned to one, we can assign the b's such that the weight of variables assigned to one is k and the weight of the assigned to zero is $k - 1$, or vice versa. This maximizes the weight of satisfied assignments that is $k(k-1)/2 = \alpha$. Conversely, if all x's are assigned to zero, the best we can do is assign all b's to one, which results in weight $(k+1)(k-2)/2 = \alpha - 1$ for the satisfied constraints.



$$
\begin{aligned}
&(1/2)\ x_i = 1\\
&(2/3)\ b_1 = 1\\
&(5/6)\ b_2 = 1\\
&(1/2)\ x_i \oplus x_j = 1\\
&(5/6)\ x_i \oplus b_1 = 1\\
&(2/3)\ x_i \oplus b_2 = 1\\
&(5/6)\ b_1 \oplus b_2 = 1
\end{aligned}
$$

where $i, j = 1, \ldots, 5$ and $i \neq j$

Fig. 7 Graphic representation of the clique-like $(10, 52/3)$-gadget from 5SAT to Max2XOR, with $\Delta E = 12/5$, and optimal number of 2 auxiliary variables

More formally, let I be an assignment satisfying $x_1 \vee \cdots \vee x_k$. Let $r = \sum_{i=1}^{k} I(x_i) \in [1, k]$. We extend I such that $I(b_i)$ is the ith bit of the binary representation of the number $k - r$. Therefore, $\sum_{i=1}^{\log k-1} I(b_i)\, 2^i$ is either $k - r$ or $k - r - 1$. And the sum of weights of variables assigned to one, $w_1 \stackrel{def}{=} \sum_{i=1}^{\log k-1} I(b_i)\, 2^i + \sum_{i=1}^{k} x_i$ is either k or $k - 1$. For the ones assigned to zero (including $\hat{0}$), we have $w_0 \stackrel{def}{=} \sum_{i=1}^{\log k-1} (1 - I(b_i))\, 2^i + \sum_{i=1}^{k} (1 - I(x_i)) + 1 = 2k - 1 - w_1$. Therefore, the sum of satisfied constraints is $w_1\, w_0/2 = k(k-1)/2$.

If I falsifies the clause, we can set $I(b_i) = 1$ and get $w_1 = \sum_{i=1}^{\log k-1} 2^i = k-2$ and $w_0 = k+1$. In this case, we get satisfied constraints for a total weight $(k+1)(k-2)/2 = k(k-1)/2 - 1$. □

When k is not a power of 2, the gadget with a minimal number of auxiliary variables is not straightforward to generalize. In Figure 7, we represent the gadget for $k = 5$ with 2 auxiliary variables.

5 Computing gadgets automatically

As in [2], we have created a Mixed Integer Programming model to compute automatically gadgets from SAT to Max2XOR with the MIP solver Gurobi.

Given a kSAT clause and a set of new auxiliary variables b_j, the MIP program explores the subsets of weighted XOR clauses of length at most 2 that can be constructed with the k input variables and the b_j variables. The weights of the XOR clauses are represented with continuous variables in the interval [0, 1].

The MIP model incorporates constraints to ensure that the selected subset of XOR clauses fulfills the definition of a (α, β)-gadget, and the objective function maximizes the energy gap ΔE. The MIP model is solved to optimality.

In the following table, we report the α, β and ΔE values found with the MIP program for some combinations of k and number of auxiliary variables:

	Number of aux. vars. 1	2	3
k=3	$\alpha = 2$ $\beta = 3$ $\Delta E = 4$		
k=4	$\alpha = 6$ $\beta = 10$ $\Delta E = 2$	$\alpha = 3$ $\beta = 9/2$ $\Delta E = 4$	
k=5	-	$\alpha = 10$ $\beta = 52/3$ $\Delta E = 12/5$	$\alpha = 4$ $\beta = 6$ $\Delta E = 4$

As we can see, the gadgets found by the MIP program presented in the diagonal of the table have the same values as the Tree-like gadget, while the gadgets under the diagonal have the same values as the Clique-like gadget, certifying that, for small values of k, they are optimal respect to the energy gap and the number of auxiliary variables, respectively.

6 Conclusions

Quantum annealers *may* offer in the future an alternative to solve the SAT problem more efficiently. To take maximum advantage of this new computation architecture, we need to take into account several aspects, such as the number of qubits that need our reformulation of the SAT problem, the energy gap, and how adaptable is our encoding to the graph describing the allowed couplings of the qubits in the quantum annealer (what may incur into the use of additional auxiliary variables).

Tree-like gadgets help us to maximize the energy gap using a linear number of auxiliary variables and are rather *flexible* to adapt them efficiently to different graph architectures. On the other hand, Clique-like gadgets require a logarithmic number of auxiliary variables, (i.e., fewer qubits), but with a lower energy gap, and may use additional qubits to accommodate the dense structure of the gadget to the architecture of the quantum annealer.

Acknowledgements This work was supported by the project PROOFS BEYOND (grant number PID2022-138506NB-C21) funded by the AEI.

Author Contributions Both authors contributed to the research in the same way.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data Availability No datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no Conflict of interest.

References

1. Aharonov, D., van Dam, W., Kempe, J., Landau, Z., Lloyd, S., Regev, O.: Adiabatic quantum computation is equivalent to standard quantum computation. SIAM J. Comput. **37**(1), 166–194 (2007). https://doi.org/10.1137/S0097539705447323
2. Ansótegui, C., Levy, J.: Reducing SAT to Max2SAT. In: Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 1367–1373. ijcai.org, (2021). https://doi.org/10.24963/ijcai.2021/189
3. Baumgartner, P., Massacci, F.: The taming of the (X)OR. In: John W. Lloyd, Verónica Dahl, Ulrich Furbach, Manfred Kerber, Kung-Kiu Lau, Catuscia Palamidessi, Luís Moniz Pereira, Yehoshua Sagiv, and Peter J. Stuckey, editors, *Computational Logic - CL 2000, First International Conference, London, UK, 24-28 July, 2000, Proceedings*, volume 1861 of *Lecture Notes in Computer Science*, pages 508–522. Springer, (2000). https://doi.org/10.1007/3-540-44957-4_34
4. Bian, Z., Chudak, F., Israel, R., Lackey, B., Macready, W.G., Roy, A.: Discrete optimization using quantum annealing on sparse Ising models. Front. Phys. **2**, 56 (2014). https://doi.org/10.3389/fphy.2014.00056
5. Bian, Zhengbing, Chudak, Fabián A., Macready, William G., Roy, Aidan, Sebastiani, Roberto, Varotti, Stefano.: Solving SAT and MaxSAT with a quantum annealer: Foundations and a preliminary report. In: Clare Dixon and Marcelo Finger, editors, *Frontiers of Combining Systems - 11th International Symposium, FroCoS 2017, Brasília, Brazil, September 27-29, 2017, Proceedings*, volume 10483 of *Lecture Notes in Computer Science*, pages 153–171. Springer, 2017. https://doi.org/10.1007/978-3-319-66167-4_9
6. Bian, Z., Chudak, F.A., Macready, W.G., Roy, A., Sebastiani, R., Varotti, S.: Solving sat (and maxsat) with a quantum annealer: foundations, encodings, and preliminary results. Inf. Comput. **275**, 104609 (2020). https://doi.org/10.1016/j.ic.2020.104609
7. Bonet, Maria Luisa, levy, Jordi, manyà, Felip: Resolution for Max-SAT. Artif. Intell. **171**(8–9), 606–618 (2007). https://doi.org/10.1016/j.artint.2007.03.001
8. Chancellor, N., Zohren, S., Warburton, P.A., Benjamin, S.C., Roberts, S.: A direct mapping of max k-sat and high order parity checks to a chimera graph. Sci. Rep. (2016). https://doi.org/10.1038/srep37107
9. Chen, J.: Building a hybrid SAT solver via conflict-driven, look-ahead and XOR reasoning techniques. In: Oliver Kullmann, editor, *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, volume 5584 of *Lecture Notes in Computer Science*, pages 298–311. Springer, (2009). https://doi.org/10.1007/978-3-642-02777-2_29
10. Choi, V.: Adiabatic quantum algorithms for the NP-complete maximum-weight independent set, exact cover and 3SAT problems, (2010a)
11. Choi, V.: Minor-embedding in adiabatic quantum computation: II. minor-universal graph design. Quantum Inform. Process. **10**(3), 343–353 (2010). https://doi.org/10.1007/s11128-010-0200-3
12. Douglass, Adam, King, Andrew D., Raymond, Jack: Constructing SAT filters with a quantum annealer. In: Marijn Heule and Sean Weaver, editors, *Theory and Applications of Satisfiability Testing – SAT 2015*, pp. 104–120. Springer, (2015)
13. Heule, Marijn, van Maaren, Hans: Aligning CNF- and equivalence-reasoning. In: *SAT 2004 - The Seventh International Conference on Theory and Applications of Satisfiability Testing, 10-13 May 2004, Vancouver, BC, Canada, Online Proceedings*, (2004)
14. Heule, Marijn, Dufour, Mark, van Zwieten, Joris, van Maaren, Hans: March_eq: Implementing additional reasoning into an efficient look-ahead SAT solver. In: Holger H. Hoos and David G. Mitchell, editors, *Theory and Applications of Satisfiability Testing, 7th International Conference, SAT 2004,*

Vancouver, BC, Canada, May 10-13, 2004, Revised Selected Papers, volume 3542 of *Lecture Notes in Computer Science*, pages 345–359. Springer, (2004). https://doi.org/10.1007/11527695_26
15. Laitinen, Tero, Junttila, Tommi A., Niemelä, Ilkka: Equivalence class based parity reasoning with DPLL(XOR). In: *IEEE 23rd International Conference on Tools with Artificial Intelligence, ICTAI 2011, Boca Raton, FL, USA, November 7-9, 2011*, pp. 649–658. IEEE Computer Society, (2011). https://doi.org/10.1109/ICTAI.2011.103
16. Laitinen, Tero, Junttila, Tommi A., Niemelä, Ilkka: Extending clause learning SAT solvers with complete parity reasoning. In: *IEEE 24th International Conference on Tools with Artificial Intelligence, ICTAI 2012, Athens, Greece, November 7-9, 2012*, pp. 65–72. IEEE Computer Society, (2012a). https://doi.org/10.1109/ICTAI.2012.18
17. Laitinen, Tero, Junttila, Tommi A., Niemelä, Ilkka: Conflict-driven XOR-clause learning. In: Alessandro Cimatti and Roberto Sebastiani, editors, *Theory and Applications of Satisfiability Testing - SAT 2012 - 15th International Conference, Trento, Italy, June 17-20, 2012. Proceedings*, volume 7317 of *Lecture Notes in Computer Science*, pp. 383–396. Springer, (2012b). https://doi.org/10.1007/978-3-642-31612-8_29
18. Li, Chu Min: Integrating equivalency reasoning into Davis-Putnam procedure. In: Henry A. Kautz and Bruce W. Porter, editors, *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence, July 30 - August 3, 2000, Austin, Texas, USA*, pp. 291–296. AAAI Press / The MIT Press, (2000a)
19. Li, Chu Min: Equivalency reasoning to solve a class of hard sat problems. Inf. Process. Lett. **76**(1–2), 75–81 (2000). https://doi.org/10.1016/S0020-0190(00)00126-5
20. Li, Chu Min: Equivalent literal propagation in the dll procedure. Discret. Appl. Math. **130**(2), 251–276 (2003). https://doi.org/10.1016/S0166-218X(02)00407-9
21. Mehta, V., Jin, F., Raedt, H., Michielsen, K.: Quantum annealing for hard 2-satisfiability problems: distribution and scaling of minimum energy gap and success probability. Phys. Rev. A **105**, 062406 (2022). https://doi.org/10.1103/PhysRevA.105.062406
22. Nüßlein, Jonas, Zielinski, Sebastian, Gabor, Thomas, Linnhoff-Popien, Claudia, Feld, Sebastian: Solving (max) 3-sat via quadratic unconstrained binary optimization. In: Jirí Mikyska, Clélia de Mulatier, Maciej Paszynski, Valeria V. Krzhizhanovskaya, Jack J. Dongarra, and Peter M. A. Sloot, editors, *Computational Science - ICCS 2023 - 23rd International Conference, Prague, Czech Republic, July 3-5, 2023, Proceedings, Part V*, volume 14077 of *Lecture Notes in Computer Science*, pages 34–47. Springer (2023). https://doi.org/10.1007/978-3-031-36030-5_3
23. Rodriguez, Pol, Ballester, Rocco, Levy, Jordi, Ansotegui, Carlos, Cerquides, Jesus: Implementing 3-SAT gadgets for quantum annealers with random instances. In: *24th International Conference on Computational Science, ICCS 2024* (2024). https://doi.org/10.1007/978-3-031-63778-0_20
24. Santra, Siddhartha, Quiroz, Gregory, Steeg, Greg Ver, Lidar, Daniel A.: Max 2-SAT with up to 108 qubits. New J. Phys. **16**(4), 045006 (2014). https://doi.org/10.1088/1367-2630/16/4/045006
25. Soos, Mate: Enhanced gaussian elimination in DPLL-based SAT solvers. In: Daniel Le Berre, editor, *POS-10. Pragmatics of SAT, Edinburgh, UK, July 10, 2010*, volume 8 of *EPiC Series in Computing*, pp. 2–14. EasyChair, (2010)
26. Soos, Mate, Meel, Kuldeep S.: BIRD: engineering an efficient CNF-XOR SAT solver and its applications to approximate model counting. In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 1592–1599. AAAI Press, (2019). https://doi.org/10.1609/aaai.v33i01.33011592
27. Soos, M., Nohl, K., Castelluccia, C.: Extending SAT solvers to cryptographic problems. In: Oliver Kullmann, editor, *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, volume 5584 of *Lecture Notes in Computer Science*, pp. 244–257. Springer, (2009). https://doi.org/10.1007/978-3-642-02777-2_24
28. Trevisan, L., Sorkin, G.B., Sudan, M., Williamson, D.P.: Gadgets, approximation, and linear programming. SIAM J. Comput. **29**(6), 2074–2097 (2000)