QUANTUM ANNEALING APPROACHES FOR MINIMAL CUT SET IDENTIFICATION IN FAULT TREES

Rola Saidi, Samuel Deleplanque, Stefan Creemers, Luis Fernando Pérez Armas, Mohamed Hibti, Belhassen Zouari

REPRINT | 3361

UCLouvain

CORE | LIDAM
Louvain Institute of Data Analysis and Modeling in economics and statistics

Quantum Annealing Approaches for Minimal Cut Set Identification in Fault Trees

Rola Saidi[a,1], Samuel Deleplanque[b,1], Stefan Creemers[c,1], Luis Fernando Pérez Armas[d,1], Mohamed Hibti[a,1], Belhassen Zouari[e]

[a] *EDF Paris-Saclay, 7 Bd Gaspard Monge, Palaiseau, 91120, France*
[b] *Junia, IEMN, CNRS, 41 Bd Vauban, Lille, 59000, France*
[c] *Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), Voie du Roman Pays 34, Louvain-la-Neuve, 1348, Belgium*
[d] *IESEG School of Management, Univ. Lille, CNRS, UMR 9221 - LEM - Lille Economie Management, 3 Rue de la Digue, Lille, 59000, France*
[e] *ICL-Junia, Université Catholique de Lille, Lille, 59000, France*

Abstract

The identification of Minimal Cut Sets (MCS) in Fault Trees (FT) is central to Probabilistic Safety Assessment (PSA) but remains computationally challenging. While gate-based quantum algorithms have been explored, Quantum Annealing (QA) has not yet been applied to this industrial use case. This work investigates the application of QA to MCS identification by modeling the FT as a boolean function and implementing three existing SAT to Quadratic Unconstrained Binary Optimization (QUBO) encoding strategies. We evaluate these encodings using generated 3-SAT instances as well as realistic FT benchmarks, and conduct experiments on a simulated annealer and two D-Wave architectures. Our results show that the encoding with reusable auxiliary variables is the most efficient in terms of qubit requirements and embedding scalability. In addition, the Zephyr architecture provides improved embeddability compared to Pegasus, reflected in reduced chain break occurrences and better performance on larger instances. Across

Email addresses: rola.saidi@edf.fr (Rola Saidi), samuel.deleplanque@junia.com (Samuel Deleplanque), stefan.creemers@uclouvain.be (Stefan Creemers), l.perezarmas@ieseg.fr (Luis Fernando Pérez Armas), mohamed.hibti@edf.fr (Mohamed Hibti), belhassen.zouari@univ-catholille.fr (Belhassen Zouari)

[1] These authors contributed equally to this work.

these experiments, QA demonstrates the ability to sample diverse valid solutions, making it suitable for exploring the solution space of MCS problems. However, its performance remains limited by hardware constraints and the maximum number of possible shots, which prevent exhaustive identification of all solutions in larger instances.

Keywords: Quantum Annealing, Fault Tree Analysis, Minimal Cut Sets, SAT-to-QUBO

---

1. Introduction

Probabilistic Safety Assessment (PSA) relies extensively on Fault Tree Analysis (FTA) as a fundamental framework for evaluating system reliability in safety-critical domains. FTA is a top-down graphical modeling approach that represents how combinations of component-level failures can propagate to system-level failures in complex engineered systems [1]. Extensions such as dynamic fault trees (FTs) further enrich this framework by incorporating temporal and sequence-dependent behavior, enabling time-dependent risk analysis in more advanced PSA settings [2]. Other modeling enhancements, such as object-based FT constructions, have also been proposed to improve system specification and requirements engineering in safety-critical applications [3]. Due to its expressive power and interpretability, FTA remains a cornerstone of PSA across a wide range of applications, including nuclear power systems and electric power infrastructures [4, 5, 6], as well as next-generation avionic flight control systems [7]. A core computational task in FTA is the identification of all Minimal Cut Sets (MCSs), which are the smallest combinations of basic events whose joint failure causes the system to fail.

The classical approaches that address the MCS problem include two main families of algorithms: boolean methods and Binary Decision Diagrams (BDDs) [8, 9, 10]. Examples of the boolean method include the top-down Method of Obtaining Cut Sets (MOCUS) [11] or the bottom-up Minimal Cut Sets Upward (MICSUP) [12]. Alternatively to boolean manipulation, the transformation of FTs into BDDs represent a relevant method that received extensive attention over the past decades for both static and dynamic PSA. Notable developments include the use of Zero-suppressed Binary Decision Diagrams (ZBDDs) to improve the efficiency of MCS computation in large and sparse FTs [13], as well as zero-suppressed ternary decision diagrams

(ZTDDs) for the analysis of non-coherent FTs [14, 15].More recently, BDD-based techniques have been extended to support dynamic PSA, where temporal ordering, sequence dependency, and state-dependent behavior must be explicitly modeled. Examples include the application of Conditional BDDs (CBDDs) to identify critical failure sequences in dynamic FTs with Priority-AND gates [16], the use of Sequential BDDs for the rapid computation of survival signatures in dynamic FTs [17], and partial BDD approaches enabling near-exact seismic risk quantification in multi-unit PSA models [18]. However, regardless of the method used, the computation of MCSs becomes a resource-intensive task as the system size grows, since the number of possible cut sets increases exponentially with the number of basic events. Even though BDDs allow efficient boolean operations in polynomial time, constructing them requires an optimal ordering of variables, which is itself an NP-complete problem. For large-scale FTs, approximation algorithms are often used, however, they tend to sacrifice accuracy, potentially leading to conservative or overly optimistic risk assessments [19]. This issue is closely related to the well-known truncation problem in PSA, where the contribution of a very large number of MCS must be approximated in practice. In particular, [20] analyzes how truncation thresholds affect the accuracy of PSA results when a large number of cut sets contribute to system failure probabilities.

Recent works have investigated quantum approaches to the MCS problem, mainly under the gate-based paradigm, in an attempt to explore their potential to solve complex problems. One method uses a Quantum Fault Tree (QFT) model [21], where the FT is encoded into a quantum circuit, and applies Quantum Amplitude Amplification (QAA) to enhance the probability of sampling MCSs [22]. Another approach reformulates the problem into a boolean satisfiablitiy problem and uses a Grover-based solver, with divide-and-conquer strategies [23]. A third method frames the problem as a vertex separator problem in a directed graph, where a quantum algorithm generates a superposition of all vertex cuts via movement oracles, followed by classical filtering to identify the minimal ones [24]. While promising, these methods remain limited by the small number of qubits and high noise levels of today's Noisy Intermediate-Scale Quantum (NISQ) computers.

Interestingly, to the best of our knowledge, Quantum Annealing (QA) has not yet been applied to this problem, despite the fact that QA devices currently offer access to a larger number of qubits with less noise compared to gate-based NISQ quantum computers. A key challenge, however, lies in the

fact that QA is inherently designed for optimization problems, whereas in this context we are interested in enumerating all possible solutions. Nevertheless, it remains promising to investigate QA's potential to provide distinct solutions, particularly since FTs can be reformulated as SAT instances, which is a class of problems that has already been addressed using QA. The underlying idea would be to map the MCS problem into an optimization framework, defining an objective function whose minima correspond to the distinct solutions of the original problem, which can then be recovered by repeatedly sampling the QA.

This motivates the objective of the present work, which is to investigate the application of QA for this industrial use case of MCS identification using real-life FT instances. To this end, FTs are first transformed into Conjunctive Normal Form (CNF) and then mapped onto a QUBO formulation that is suitable for QAs. Three different state-of-the-art encoding strategies were studied, in order to evaluate their relative efficiency and scalability. The proposed formulations were benchmarked using both generated 3-SAT instances and real-life FT cases. In addition, we also compared D-Wave's Pegasus and Zephyr architectures. The analysis focused on key performance metrics, including embedding resource requirements, the impact of hardware topology, the ability to recover exhaustive sets of solutions, and computation time.

The remainder of this paper is structured as follows. Section 2 introduces the principles of our approach and presents a number of basic concepts. Section 3 details the three SAT-to-QUBO encoding methods that we studied. Section 4 describes the experimental setup and discusses the results obtained. Section 5 concludes.

2. Quantum Annealing Approach for Minimal Cut Set problem

Unlike gate-based quantum computing, which relies on discrete quantum circuits and unitary gate operations to manipulate qubit states, QA is an analog approach designed specifically for solving combinatorial optimization problems, where the objective is to find a (binary) configuration that minimizes a given cost function.

The idea of applying QA to the MCS problem relies on seeing the FT as a boolean function that expresses the logical relationships between the Basic Events (BEs) and the Top Event (TE). These boolean expressions can be reformulated as SATisfiability (SAT) problems, which are then encoded as optimization problems suitable for QA.

More precisely, once the FT has been translated into a SAT formulation, it can be further mapped into a QUBO model. This Hamiltonian represents an energy landscape that can be minimized using a quantum annealer such as the hardware developed by D-Wave.

Since the MCS problem involves an exhaustive search, aiming to identify all minimal combinations of basic events leading to the system failure rather than a single optimal solution, the goal of this method is to design the cost function such that its minima correspond to distinct valid solutions of the original SAT problem. By sampling from the low-energy configurations obtained through QA, multiple cut sets can thus be extracted (See Algorithm 1& 2).

Algorithm 1 QA Approach for MCS Identification

Require: FT $G = (V, E)$ with top event TE
Ensure: Set of MCS $\mathcal{M}$

Encode the FT into a CNF formula Φ using Tseitin transformation (Algorithm 3)
Construct a QUBO cost function $Q(x)$ such that:
    $Q(x)$ is minimal if x satisfies Φ, and $Q(x)$ is not minimal otherwise
Submit $Q(x)$ to a quantum annealer
Sample a set of low-energy solutions $\mathcal{S}$
$\mathcal{C} \leftarrow \emptyset$
for all $x \in \mathcal{S}$ **do**
    if x satisfies Φ **then**
        Extract corresponding basic events C_x
        Add C_x to $\mathcal{C}$
    end if
end for
$\mathcal{M} \leftarrow$ MinimalCutSetsFiltering($\mathcal{C}$) (Algorithm 2)
return $\mathcal{M}$

The UML activity diagram in Figure 1 summarizes step by step our approach using three QUBO reductions.

2.1. Quantum Annealing (D-Wave implementation)

QA is a metaheuristic optimization technique that exploits quantum mechanical effects such as *superposition* and *tunneling* to explore the energy landscape of a cost function [25, 26]. This approach is the quantum version

Algorithm 2 MCS Filtering Algorithm

Require: $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ list of cut sets
Ensure: $\mathcal{M}$ list of MCS

```
Sort C in increasing order of cardinality
M ← ∅
for all C_i ∈ C do
    is_minimal ← True
    for all C_j ∈ M do
        if C_j ⊆ C_i then
            is_minimal ← False
            break
        end if
    end for
    if is_minimal then
        Add C_i to M
    end if
end for
return M
```

of the classical simulated annealing [27], which is a probabilistic technique inspired from the phenomenon of systematic cooling of a system in metallurgy to reach the minimum state. It is shown that QA could theoretically [25] and experimentally [28] be advantageous in solving some classical optimization problems over its classical counterpart.

The process of QA is based on the adiabatic theorem of quantum mechanics, which describes the evolution of any quantum system [29]. This theorem states that if the evolution is slow enough, the system remains in the ground state throughout the process. The application of this natural phenomenon consists of starting with a relatively simple initial state represented by a Hamiltonian H_0, whose ground state is easy to prepare. The system then slowly evolves towards a problem Hamiltonian H_P, whose ground state encodes the optimal solution of the problem. The total Hamiltonian over time t is expressed as:

$$H(t) = A(t)H_0 + B(t)H_P, \quad 0 \leq t \leq T, \tag{1}$$

where $A(t)$ and $B(t)$ are time-dependent functions satisfying $A(0) \gg B(0)$ and $A(T) \ll B(T)$. When the annealing ends ($t = T$), the system reaches

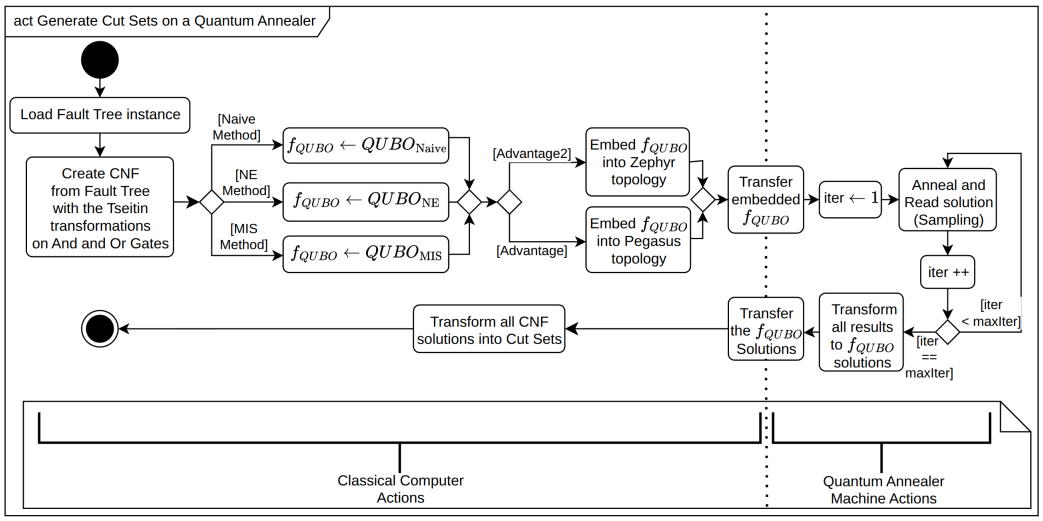


Figure 1: UML activity diagram: QA Approach for MCS identification.

the ground state of H_P, which corresponds to the optimal solution of the encoded optimization problem. In practice, the annealing schedule and the presence of noise determine the quality of the obtained solution.

The D-Wave quantum annealers are purpose-built for solving optimization problems formulated as QUBOs. Their QA process is implemented through a transverse-field Ising model (which is isomorphic to a QUBO). The physical qubits in these systems are implemented as superconducting flux qubits, whose interactions are constrained by the hardware's underlying connectivity topology. In fact, D-wave machines have a static topology that does not necessarily correspond to the QUBO graph. That is why we talk about the "embedding process", which is the process of mapping the model into the physical qubit graph, often using chains of qubits to represent individual logical variables.

Over successive generations, D-Wave has developed different architectures to improve qubit connectivity, scalability, and noise resilience. In our work, we use two D-wave machines: Advantage_system6.4 and Avantage2_system2.1 which correspond to quantum processors with the Pegasus and Zephyr topologies, respectively [30].

As shown in Table 1, the Pegasus topology provides a structured connectivity pattern with up to 15 couplers per qubit. The newer Zephyr

topology increases local connectivity to up to 20 couplers per qubit and improves the regularity and flexibility of the embedding graph. In contrast, Advantage_system6.4 has a larger number of physical qubits than Avantage2_system2.1.

These hardware architectures play a critical role in determining how efficiently logical QUBO variables can be mapped onto physical qubits, a process known as *embedding*. Efficient embeddings minimize chain lengths and reduce noise accumulation, which directly impacts the quality and reliability of the obtained solutions.

Table 1: Comparison of D-Wave quantum annealers used in this work (Pegasus and Zephyr architectures).

Feature	Advantage_system6.4	Advantage2_system2.1
Topology	Pegasus (P16)	Zephyr (Z12)
Number of qubits	5612	4516
Average connectivity	15 couplers per qubit	20 couplers per qubit
Number of couplers	40 088	40 448

2.2. Minimal Cut Sets in Fault Tree Analysis

FTA is a graphical method widely used in risk and reliability analysis of complex engineering systems that allows modeling how component failures lead to system failures [31]. An FT can be formally defined as a Directed Acyclic Graph (DAG), where the nodes are partitioned into three categories: Basic Events (BEs), Intermediate Events (IEs), and the Top Event (TE), and where the edges represent dependencies between the events. The events are usually represented by binary variables that indicate whether a failure of a subsystem or individual component has occurred. These events are linked together using logical gates that translate the dependencies between the system and its basic components.

The leaves of the graph correspond to the set of BEs, which represent elementary component failures that can occur according to a given probability distribution. The root node corresponds to the top event, representing the system-level failure under analysis. The nodes connecting the basic events to the top event are the intermediate events IEs.

Each IE and the TE is defined as the output of a boolean function applied to their input events. More precisely, for any intermediate or top event v, its

associated variable can be written as:

$$x_v = f_v(x_{u_1}, \ldots, x_{u_k})$$

where $u_1, \ldots, u_k$ are the direct predecessors of node v. In practice, these boolean functions correspond to logical gates, which can be expressed using standard boolean operators as used in SAT formulations:

- AND gates output 1 only if all input events occur and are represented by the conjunction operator $\wedge$:

$$x_v = x_{u_1} \wedge x_{u_2} \wedge \cdots \wedge x_{u_k}.$$

- OR gates output 1 if at least one input event occurs and is represented by the disjunction operator $\vee$:

$$x_v = x_{u_1} \vee x_{u_2} \vee \cdots \vee x_{u_k}.$$

The identification of MCSs is one of the most relevant tasks for qualitative analysis of system failures since it allows to identify what combinations of BEs cause the TE to occur [31]. MCSs can be defined as an irreducible set of components whose failure causes the entire system to fail. This means that the failure of the system occurs only when all the elements within an MCS fail.

More formally, let f_{FT} denote the Boolean function associated with the FT, mapping the values of the BEs to the value of the TE.

Definition 1 (Cut Set). A subset $C \subseteq \{BE_i\}_{i=1}^{N_{BE}}$ is a cut set if:

$$f_{FT}(C) = 1,$$

where $f_{FT}(C) = 1$ means that assigning $BE_i = 1$ for all $BE_i \in C$ implies $TE = 1$, and $TE = 0$ otherwise.

Definition 2 (Minimal Cut Set). A subset $C \subseteq \{BE_i\}_{i=1}^{N_{BE}}$ is a Minimal Cut Set if:

$$f_{FT}(C) = 1 \quad \text{and} \quad f_{FT}(C') = 0 \quad \forall C' \subset C.$$

Identifying MCS is crucial in FTA as they provide an efficient way to understand and mitigate the vulnerabilities of a system. This allows designing more reliable systems by reducing the failure probabilities of the most critical combinations of BEs that lead to system failures.

2.3. From Fault Trees to Conjunctive Normal Form (CNF)

In this section, we want to transform the FT into a SAT problem expressed in CNF, which is a boolean formula represented as a conjunction of clauses, each of which is a disjunction of literals. Formally, it can be written as:

$$\Phi = \bigwedge_{i=1}^{m} \left(\bigvee_{j=1}^{k_i} l_{ij} \right), \tag{2}$$

where l_{ij} denotes a literal (i.e., a boolean variable or its negation), m the number of clauses, and k_i the number of elements in clause i. The objective is to determine all the truth assignments for propositional variables that make this logical boolean formula true. These assignments would correspond to all the cut sets of the FT that lead to system failure.

As explained previously, in FTs, event nodes are represented as binary variables indicating whether the corresponding event occurs, and are connected by logical gates that model the dependencies of the system on its basic components. By encoding these relationships between component failures, FT instances can be transformed into SAT problems expressed in CNF using Tseitin transformations [32]. Unlike the naive approach of using Morgan laws that may lead to an exponential growth of the problem, Tseitin transformations ensure a polynomial-size reduction while preserving satisfiability.

The Tseitin transformation is applied iteratively to each gate in the FT and introduces auxiliary variables for intermediate logic gate output (See Algorithm 3):

1. Assign a unique variable z to each gate output.
2. Add a set of CNF clauses that encode the equivalence between z and the gate's input variables according to its logical operation (see Table 2).
3. Recursively apply the transformation to all gates until the full FT is encoded.

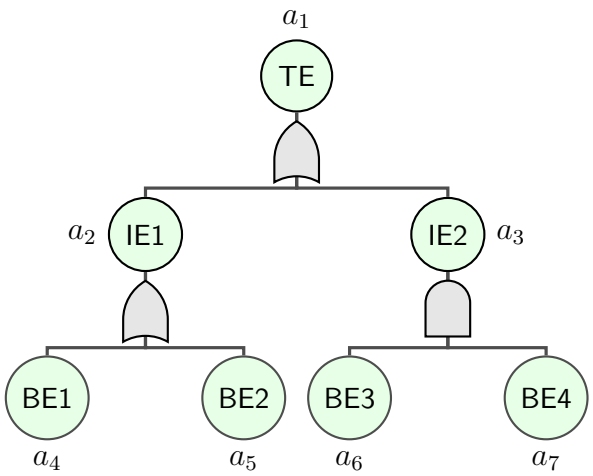
Finally, a unit clause (z_{TE}) is added to the formula, where z_{TE} denotes the variable associated with the TE. This constraint enforces that the encoded formula corresponds to system failure, ensuring that any satisfying assignment represents a combination of BEs leading to the occurrence of the TE.

The resulting CNF has this unit clause and clauses with 2 or 3 literals, which motivated the exploration of 2-SAT and 3-SAT quantum solvers.

Table 2: Tseitin transformations for basic logic gates.

Gate type	Boolean expression	Tseitin clauses (CNF)
AND ($z = x_1 \wedge x_2$)	$z \leftrightarrow (x_1 \wedge x_2)$	$(\neg z \vee x_1)$, $(\neg z \vee x_2)$, $(z \vee \neg x_1 \vee \neg x_2)$
OR ($z = x_1 \vee x_2$)	$z \leftrightarrow (x_1 \vee x_2)$	$(z \vee \neg x_1)$, $(z \vee \neg x_2)$, $(\neg z \vee x_1 \vee x_2)$
NOT ($z = \neg x_1$)	$z \leftrightarrow \neg x_1$	$(z \vee x_1)$, $(\neg z \vee \neg x_1)$

Figure 2 illustrates a simple example of transforming a small FT into CNF using Tseitin transformations.



$$F = a_1 \wedge (a_1 \vee \bar{a}_2) \wedge (a_1 \vee \bar{a}_3) \wedge (\bar{a}_1 \vee a_2 \vee a_3) \\ \wedge (a_2 \vee \bar{a}_4) \wedge (a_2 \vee \bar{a}_5) \wedge (\bar{a}_2 \vee a_4 \vee a_5) \\ \wedge (\bar{a}_3 \vee a_6) \wedge (\bar{a}_3 \vee a_7) \wedge (a_3 \vee \bar{a}_6 \vee \bar{a}_7).$$

Figure 2: Example of a FT reduction to CNF using Tseitin transformations.

3. SAT-TO-QUBO encoding methods

To run on D-Wave's QAs, the CNF formulas must be reformulated into a QUBO problem in such a way that assignments satisfying all CNF clauses correspond to the minima of the QUBO. In this work, we implemented and

Algorithm 3 Recursive Tseitin Encoding of a FT into CNF

Require: FT $G = (V, E)$ with gate types, top event TE
Ensure: CNF formula Φ, variable map var_map

$\Phi \leftarrow \emptyset$, $c \leftarrow 1$, var_map $\leftarrow \emptyset$
for all $v \in V$ **do**
    var_map$[v] \leftarrow c$; $c \leftarrow c + 1$
end for
function EncodeGate($inputs, output, type$)
    if $|inputs| = 1$ **then**
        return
    else if $|inputs| = 2$ **then**
        if $type = AND$ **then**
            Add $(\neg output \vee inputs_1)$
            Add $(\neg output \vee inputs_2)$
            Add $(output \vee \neg inputs_1 \vee \neg inputs_2)$
        else
            Add $(output \vee \neg inputs_1)$
            Add $(output \vee \neg inputs_2)$
            Add $(\neg output \vee inputs_1 \vee inputs_2)$
        end if
    else
        Split $inputs$ into L, R
        $z_L \leftarrow c$, $z_R \leftarrow c + 1$, $c \leftarrow c + 2$
        EncodeGate($L, z_L, type$)
        EncodeGate($R, z_R, type$)
        EncodeGate($\{z_L, z_R\}, output, type$)
    end if
end function
for all $v \in V$ such that v is a gate **do**
    $output \leftarrow$ var_map$[v]$
    $inputs \leftarrow \{$var_map$[u] : u \in children(v)\}$
    EncodeGate($inputs, output, type(v)$)
end for
Add (var_map$[TE]$) to Φ
return Φ, var_map, $c - 1$

compared three SAT-to-QUBO encoding strategies. It is important to note that the resulting solutions correspond to cut sets, but not necessarily to MCS, since explicitly encoding the minimality constraint would require less tractable QUBO formulations. However, extracting the MCS can be done through a polynomial-time filtering step applied after the QA stage.

3.1. Direct SAT-to-QUBO encoding

We propose a first, naive method that aims to directly transform a 3-SAT or the mixed FT SAT instances into a single QUBO formulation. This approach processes each clause independently and converts it into a logical constraint. For the clauses of two literals, a QUBO reduction is done following Table 3 [33]. For the clauses of three literals, the constraints take the form of inequalities (e.g., "$\geq$" conditions), which are more difficult to map into QUBO penalty terms. All variables are binary ($x_i \in \{0, 1\}$). The objective is to construct an energy function such that all satisfying assignments of the Boolean formula correspond to its global minima, while any violation of the constraints incurs a strictly positive penalty.

Table 3: Clauses and their corresponding quadratic penalties.

Clause	Quadratic penalty
$x_i \vee x_j$	$(1 - x_i - x_j + x_i x_j)$
$x_i \vee \bar{x}_j$	$(x_j - x_i x_j)$
$\bar{x}_i \vee x_j$	$(x_i - x_i x_j)$
$\bar{x}_i \vee \bar{x}_j$	$(x_i x_j)$

In [34] the logical OR constraint $x \vee y \equiv (x + y \geq 1)$ is translated into a quadratic penalty P term suitable for QUBO formulations:

$$P = 1 - x - y + xy,$$

where $P > 0$ is a penalty weight function. It evaluates to 0 when the constraint is satisfied (i.e., at least one variable is 1), and to 1 when it is violated (i.e., both variables are 0).

We now generalize this to a three-variable OR:

$$x_1 \vee x_2 \vee x_3 \equiv (x_1 + x_2 + x_3 \geq 1).$$

The corresponding penalty function becomes: $P_1 = 1 - x_1 - x_2 - x_3 + x_1 x_2 + x_1 x_3 + x_2 x_3 - x_1 x_2 x_3$. The truth table 4 below confirms that this function is

Table 4: Truth table and penalty values for the clause $x_1 \vee x_2 \vee x_3$.

x_1	x_2	x_3	$x_1 \vee x_2 \vee x_3$	Penalty value
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

zero when the OR is true and equals 1 only when all inputs are 0 (i.e., the OR is false).

The cubic term $x_1x_2x_3$ must be quadratized to meet QUBO requirements. Following the method described in [35], we introduce an auxiliary variable y_{12} such that $y_{12} = x_1x_2$, and rewrite the cubic term as $y_{12}x_3$. To enforce the identity $y_{12} = x_1x_2$, we add a penalty term:

$P_2 = x_1x_2 - 2x_1y_{12} - 2x_2y_{12} + 3y_{12}$.

Let λ be a weighting coefficient applied to P_2, chosen so that all cases where $y_{12} \neq x_1x_2$ are appropriately penalized in the following QUBO formulation:

$$\begin{aligned} E &= P_1 + \lambda P_2 \\ &= (1 - x_1 - x_2 - x_3 + y_{12} + x_1x_3 + x_2x_3 - y_{12}x_3) \\ &\quad + \lambda\,(x_1x_2 - 2x_1y_{12} - 2x_2y_{12} + 3y_{12})\,. \end{aligned} \tag{3}$$

Table 5 reports an empirical search for a suitable value of λ to penalize cases where $y_{12} \neq x_1x_2$. For $\lambda = 1$, the function E does not satisfy all the constraints due to the case $(x_1, x_2, x_3, y_{12}) = (1, 1, 0, 0)$, which results in a zero value for $P_1 + P_2$. This issue is resolved as soon as $\lambda \geq 2$.

3.2. Effective 3-SAT QUBO Encoding with Reduced Auxiliary Variables

This strategy, that we refer to as the "New Encoding" (NE) method, is introduced in [36]. The idea is to reduce the 3-CNF formula to a 2-CNF form by reusing auxiliary variables, thus minimizing QUBO size and using less logical qubits. For the 2-CNF clauses we use the previously introduced reduction according to Table 3.

Table 5: Truth table for the quadratization of the cubic term $x_1x_2x_3$. The auxiliary variable y_{12} is introduced to rewrite the cubic interaction as the quadratic term $y_{12}x_3$, with $y_{12} = x_1x_2$.

x_1	x_2	x_3	y_{12}	$x_1x_2 = y_{12}$	P_1	P_2	$P_1 + P_2$	$P_1 + 2P_2$
0	0	0	0	1	1	0	1	1
0	0	0	1	0	2	3	5	8
0	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**
0	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
1	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
0	0	1	1	0	0	3	3	6
0	1	0	1	0	1	1	2	3
1	0	0	1	0	1	1	2	3
0	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**0**
1	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**
1	1	0	0	0	-1	1	0	1
0	1	1	1	0	0	1	1	2
1	1	1	0	0	0	1	1	2
1	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**0**
1	0	1	1	0	0	1	1	2
1	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**

For instance, a 3-literal clause $a \vee b \vee c$ is encoded by introducing an auxiliary variable u such that:

$$a \vee u, \quad \text{subject to } u = b \vee c.$$

The resulting QUBO is:

$$E(a, u) = au - a - u,$$

$$E(u, b, c) = bc - 2bu - 2cu + b + c + u,$$

$$E(a, b, c, u) = bc + au - 2bu - 2cu - a + b + c.$$

In order to use fewer auxiliary variables, the idea is to reuse the same auxiliary variable for redundant literal pairs across different clauses. The intuition behind this approach is that auxiliary variables represent intermediate logical combinations of literals, which only need to be encoded once in the QUBO formulation. By reusing these variables, we avoid introducing redundant representations of the same logical relationship, thereby reducing the total number of variables and consequently, the number of required qubits.

For instance, consider a clause $a \vee b \vee c$, which is encoded using an auxiliary variable $u = b \vee c$. Now, if another clause $d \vee b \vee c$ appears, a naive approach would introduce a new auxiliary variable $v = b \vee c$. However, since the pair (b, c) already appears in the first clause, the same auxiliary variable u can be reused. As a result, instead of constructing two independent encodings:

$$E(a, b, c, u) + E(d, b, c, v),$$

we reuse the auxiliary variable and obtain:

$$E(a, b, c, u) + E(d, b, c, u).$$

In practice, the method proceeds as follows:

1. Identify all pairs of literals that appear in multiple clauses.
2. Count the frequency of co-occurrence of each pair across clauses.
3. Sort these pairs in descending order of frequency.
4. Assign the same auxiliary variable to the most frequent pairs.

This replacement scheme is expected to minimize the total number of auxiliary variables,when redundant literal pairs exist across clauses. As a result, it leads to a more compact QUBO formulation, reducing the number of required qubits and simplifying the embedding onto QA hardware.

3.3. SAT to Maximum Independent Set

This approach reduces the SAT problem to a Maximum Independent Set (MIS) problem, a known polynomial-time transformation suitable for D-Wave devices [37]. According to [37], this reduction of the 3-SAT problem may allow us to obtain graphs whose properties are more in line with the computational characteristics of analog machines. This is the motivation of why we also explore this method for analyzing FT instances.

The MIS problem is an NP-hard combinatorial optimization problem defined on an undirected graph $G = (X, E)$, where the goal is to find the largest subset of vertices $S \subseteq X$ such that no two vertices in S are adjacent [38]. Given a SAT instance with variables v_j and clauses C_i of size $k_i \in \{1, 2, 3\}$, the reduction proceeds as follows:

1. **One vertex per literal occurrence:** for each clause C_i containing k_i literals, we create k_i vertices, each associated with a binary variable $y_{i,\ell} \in \{0, 1\}$:

$$y_{i,\ell} = \begin{cases} 1 & \text{if the literal } \ell \text{ in clause } i \text{ is selected in the MIS,} \\ 0 & \text{otherwise.} \end{cases}$$

2. **Clause constraint:** all vertices belonging to the same clause C_i are fully connected, forming a clique of size k_i. This ensures that at most one literal per clause can be selected.
3. **Variable consistency:** for each variable v_j, all occurrences of v_j and $\neg v_j$ across all clauses are connected pairwise. This forbids selecting both a variable and its negation.

This construction yields a graph with $\sum_i k_i$ vertices. Its density depends on the recurrence of variables across clauses, which impacts the ease of embedding the resulting QUBO onto quantum hardware.

Then, for the obtaining of the SAT assignments, if the MIS solver returns an independent set S of size $|C|$, then the SAT instance is satisfiable. Each selected vertex corresponds to one satisfying literal per clause. A satisfying assignment is extracted as:

$$v_j = \begin{cases} 0 & \text{if the chosen literal corresponds to } \neg v_j, \\ 1 & \text{otherwise.} \end{cases}$$

If no MIS of size $|C|$ exists, the SAT instance is unsatisfiable.

4. Experiments & Results

In this section, we report the results of several experiments that were run on a simulated annealer and two quantum annealers of D-Wave using real FT instances as well as generated 3-SAT instances. These tests also include comparisons of the three encoding strategies regarding the embedding requirements, the influence of the quantum annealer's topology, the number of distinct solutions, as well as the runtime.

4.1. Tested Instances

For this study, we considered 51 instances of industrial FTs, provided by EDF (Électricité de France), ranging from small examples with only 4 nodes to large-scale systems with up to 1773 nodes. The corresponding SAT formulations varied from 5 variables and 7 clauses in the smallest system to 2247 variables and 3625 clauses in the largest one.

In addition to these industrial cases, we used generated 19 sets of 30 feasible 3-SAT instances to test the algorithms on controlled, synthetic problems. The number of clauses was set to vary from 5 to 30, increasing by 5 at each step, while the clause-to-variable ratio α ranged from 2 to 6, with a step of 1. The corresponding number of variables was then derived from these parameters. We discarded infeasible sets, such as those with fewer than three variables, as well as specific redundant configurations.

It is important to highlight the fundamental differences between the industrial FT instances and the generated 3-SAT instances. The SAT formulations derived from real FTs are highly structured, as they originate from logical representations of physical systems where dependencies between components are explicitly defined through AND and OR gates. This structured and hierarchical nature often introduces regular patterns that can be exploited by solvers, potentially making these instances easier to handle despite their larger size. Due to Tseitin encoding, the formulas are a mix of clauses with two and three literals plus one single literal clause that represents the TE.

In contrast, the generated 3-SAT instances are synthetic and randomly constructed, with each clause composed of three literals chosen independently. They do not reflect any physical system but serve as theoretical benchmarks to evaluate solver performance under controlled and challenging conditions. Consequently, while FT-derived SAT instances are representative of practical, real-world problems, generated 3-SAT instances are valuable

for testing algorithmic robustness and scalability. The combination of both types of instances therefore provides a balanced and comprehensive evaluation framework.

4.2. Embedding and Resources

The three SAT-to-QUBO encoding strategies were compared in terms of embedding requirements, specifically the required number of logical qubits and the estimated number of required physical qubits on D-Wave hardware. Logical qubits represent the variables in the QUBO model, while physical qubits are the actual hardware qubits on D-Wave. Because the hardware has limited connectivity, one logical qubit often needs to be mapped onto a chain of several physical qubits. This is why the number of physical qubits is typically larger than the number of logical qubits.

Note that in our implementation, the unit clause corresponding to the TE is fixed in the QUBO model of the NE method, which reduces the number of variables by one.

Results on both the 3-SAT instances and the set of FT-derived instances (Figure 3) show that the NE method slightly outperforms the Naive method, particularly in larger instances where redundancy exists. It required less logical qubits and thus less physical qubits, confirming that its simplifications are effective at scale. The MIS method, on the other hand, proved to be the most resource-demanding approach in terms of qubit count, as expected. Nonetheless, its graph-structured formulation still makes it a candidate for further investigation as it may enable the sampling of a larger number of distinct solutions.

To better understand the conditions under which the NE encoding provides improvements, we further analyzed the relationship between literal pairs redundancy accross different clauses and qubit reduction. Figure 4 shows for each instance, the total number of repeated literal pairs in the CNF formula together with the logical and physical qubit reductions obtained by the NE encoding compared to the Naive method.

As expected, the advantage of the NE method becomes more significant for instances that contain redundant literal pairs across clauses. When such redundancy is present, fewer auxiliary variables are required, which directly reduces the number of logical qubits in the resulting QUBO model. In contrast, when no redundancy is present, the number of logical qubits of the NE and the naive methods differs only by one unit, due to the fact that the unit clause corresponding to the TE is fixed to 1 in the QUBO formulation.

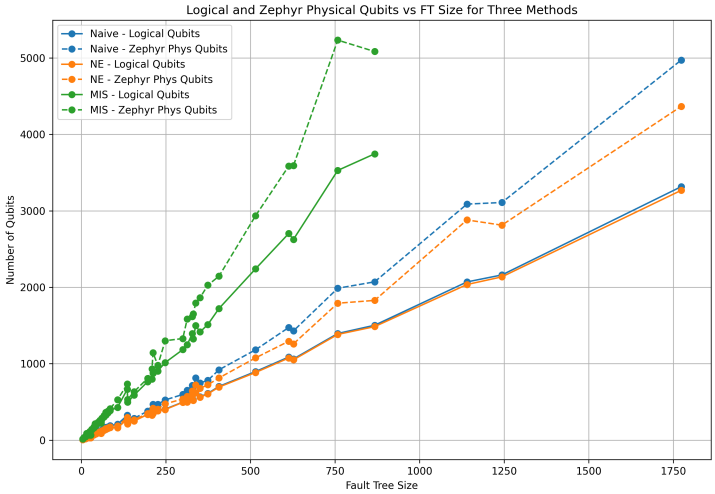


Figure 3: Logical versus physical qubits for the three encoding methods.

The figure also shows that the amount of redundancy tends to increase with the size of the FT. Larger instances naturally contain more clauses and therefore a higher probability that pairs of literals appear repeatedly across clauses. This explains why the advantage of the NE method becomes more noticeable for larger FTs.

Interestingly, a reduction in the number of required physical qubits is observed even for instances where no logical qubit reduction occurs. This effect arises from the embedding process onto the D-Wave hardware topology. The structure of the QUBO generated by the NE encoding results in a slightly sparser graph, even when the number of logical variables is nearly identical. Since the minor-embedding procedure must map this graph onto hardware (that has limited connectivity), small differences in graph structure can lead to shorter chains of physical qubits representing each logical variable. As a result, the NE formulation can still reduce the total number of physical qubits required by the embedding despite having nearly the same number of logical variables.

Overall, these results confirm that the NE encoding is particularly beneficial when clause redundancy is present, while still providing modest improvements in physical resource requirements even in the absence of redundancy.

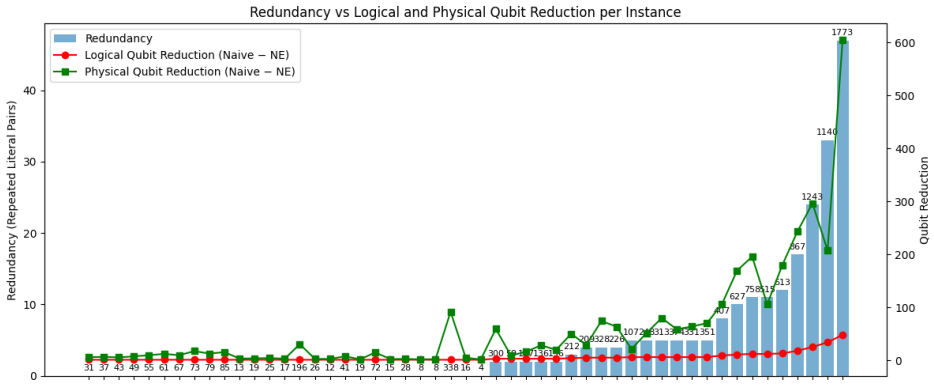


Figure 4: Redundancy in FT CNF clauses and corresponding logical and physical qubit reduction achieved by the NE encoding compared to the Naive encoding. The numbers above the bars indicate the size of the corresponding FT.

4.3. Influence of the Quantum Annealer's Topology

We compared the Pegasus topology with the recently introduced Zephyr topology, integrated into D-Wave's sixth-generation Advantage2 QA, for the three SAT-to-QUBO methods on both generated 3-SAT instances and FT-derived instances. For each FT instance, the number of physical qubits required for embedding was estimated 10 times per topology to account for variability in the embedding process. The results, plotted in Figure 5, show the average number of required physical qubits for the three methods on both topologies.

Across all three methods, embedding on Pegasus consistently required more physical qubits and longer chain lengths than embedding on Zephyr. Consequently, the MIS-based method, being the most resource-intensive, reached the embeddability limit earlier than the Naive and NE methods. Similarly, for Naive and MIS methods, the maximum embeddable instance size was reached sooner on Pegasus than on Zephyr, meaning that some instances could be embedded on Zephyr but not on Pegasus.

4.4. Exhaustivity

To compare the QUBO formulations in terms of the number of distinct solutions, tests on D-Wave's Simulated Annealer (SA) were run with 4000 shots, due to limited access to real QA hardware. The number of solutions obtained was compared with those from an exact SAT solver, for both generated 3-SAT instances and certain small FT instances. For the 3-SAT

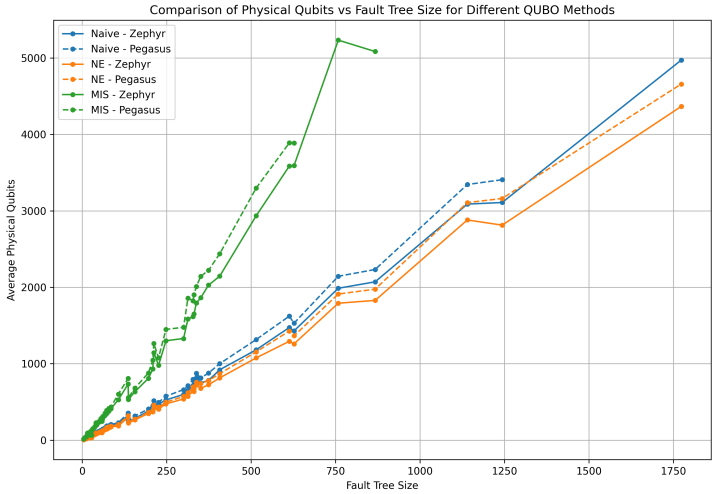


Figure 5: Pegasus versus Zephyr physical qubits for the three encoding methods.

instances, which are relatively small compared to the FT SAT instances, all three QUBO formulations on SA were able to sample all solutions for the smallest systems. For larger 3-SAT instances (e.g., 13 variables and 25 clauses), SA found slightly fewer solutions than the exact solver.

For FT instances, which include a mix of 1-, 2-, and 3-literal clauses, both NE and Naive formulations were able to recover the full set of solutions for small FT instances, as shown in Table 6. For example, instances of size 4, 8, and 13 nodes produced exactly the same number of solutions as the exact SAT solver. This confirms that both QUBO encodings correctly capture the structure of the SAT problem.

For larger instances, the number of distinct solutions returned by SA becomes smaller than the number of solutions reported by the exact solver, which could be attributed to sampling limitations of SA rather than deficiencies in the QUBO formulation. In addition, as instance size grows, the number of distinct solutions obtained approaches the number of shots, indicating that the solution space may be too large to be adequately sampled. The largest FT instance solvable with all three methods on SA had 85 nodes (SAT with 111 variables and 166 clauses); for larger instances, the simulator could no longer find valid solutions.

NE and Naive showed similar results for all instances, confirming that the NE method of reusing auxiliary variables does not affect solution qual-

Table 6: Number of solutions found with SA across NE, Naive, and MIS encodings, compared with the exact SAT solver results.

FT size	NE #Sol.	Naive #Sol.	MIS #Sol.	Exact SAT #Sol.
4	7	7	5	7
8	31	31	17	31
8	62	63	35	63
13	201	201	76	201
15	480	481	263	481
16	2841	3129	1596	8191
17	2878	3152	1579	8191
19	2497	2523	847	4041
19	977	979	406	1001
25	3796	3835	2623	–
49	3854	3894	3590	–
55	3854	3894	3590	–
85	3698	3816	3392	–

ity while reducing resource requirements. In contrast, the MIS formulation consistently returns fewer solutions than the NE and Naive encodings. This suggests that the MIS mapping does not preserve the full SAT solution space.

Some tests were also run on D-Wave QA hardware, using both Pegasus and Zephyr topologies. As expected, some instances were only testable on Zephyr due to its higher qubit connectivity. Moreover, for certain instances, the number of shots on Pegasus could not exceed 4000, whereas up to 5000 shots were possible on Zephyr, which can lead to a higher number of retrieved solutions for larger instances. However, in Table 7, we report results obtained with the same number of shots (4000) for some selected FT instances on both topologies to ensure a fair comparison.

For small instances where no chain breaks are observed on either topology, Pegasus and Zephyr produce identical or very similar numbers of solutions, with Pegasus occasionally returning slightly more solutions. These results are also very close to those obtained with SA, indicating that both architectures correctly sample the solution space in this regime.

As the instance size increases, a clear difference emerges between the two architectures: Zephyr consistently returns a higher number of valid solutions than Pegasus for larger instances. A more detailed examination shows that this transition coincides with the appearance of chain breaks on Pega-

Table 7: Updated comparison of QA results on Zephyr (ZE) and Pegasus (PE) topologies versus SA for the NE method.

FT size	PE #Sol.	ZE #Sol.	SA #Sol.	PE avg chain break	ZE avg chain break
4	7	7	7	0	0
8	31	31	31	0	0
8	63	63	62	0	0
13	201	201	201	0	0
15	480	447	480	0	0
16	2978	2867	2878	0	0
17	2972	2906	2841	0	0
19	950	902	977	0	0
19	2224	2221	2497	0	0
25	3721	3827	3796	1.18×10^{-5}	0
49	3883	3992	3854	2.69×10^{-6}	0
55	3857	3993	3832	2.38×10^{-6}	0
61	3830	3981	3751	8.55×10^{-6}	0
67	3816	3990	3755	1.94×10^{-6}	0
73	3851	3981	3743	3.55×10^{-6}	0
79	3778	3981	3723	9.80×10^{-6}	0
85	3703	3990	3698	6.06×10^{-6}	0

sus. While no chain breaks are observed for small instances, Pegasus exhibits small but non-zero chain break fractions for larger problems, whereas Zephyr shows none across all tested instances. Although this observation does not establish a direct causal relationship, it supports the hypothesis that the improved connectivity of Zephyr leads to more stable embeddings, thereby reducing chain breaks and improving sampling performance for larger instances.

To assess whether the observed differences of number of solutions could be attributed to statistical fluctuations, a paired t-test was performed across all tested instances. The test yielded a t-statistic of 2.73 and a p-value of 0.0093, indicating a statistically significant difference between Pegasus and Zephyr at the 1% significance level.

To further investigate this behavior, repeated experiments (30 runs) were conducted on selected instances (Table 8). The results confirm that for a small instance (FT.inst.a1434), Pegasus achieves a slightly higher average number of solutions, whereas for a larger instance affected by chain breaks (FT.inst.9), Zephyr outperforms Pegasus.

Table 8: Average number of valid solutions and average chain break fractions over 30 runs for selected FT instances on Pegasus (PE) and Zephyr (ZE).

Instance	Size	PE mean	PE chain break	ZE mean	ZE chain break
FT.inst.a1434	19	927.57	2.86×10^{-5}	881.57	0
FT.inst.a1001	17	2876.03	5.63×10^{-6}	2884.50	0
FT.inst.9	25	3777.07	7.42×10^{-6}	3797.17	0

A statistical analysis using Welch's t-test indicates that these differences are statistically significant for some instances. In particular, the advantage of Zephyr for FT.inst.9 is highly significant (p-value $\approx 2.8 \times 10^{-5}$), while no significant difference is observed for FT.inst.a1001 (p-value ≈ 0.58). Conversely, Pegasus significantly outperforms Zephyr for FT.inst.a1434 (p-value $\approx 3.0 \times 10^{-6}$). These results indicate that the relative performance of the two topologies remains instance-dependent.

Overall, these results show that quantum annealers can retrieve a diverse set of valid solutions, demonstrating their ability to explore the solution space for small and medium-sized instances. However, as instance size increases, exhaustivity becomes increasingly limited by the fixed sampling budget, which leads to incomplete coverage of the solution space. In this regime, a trade-off emerges between embedding stability and sampling performance: while both

architectures behave similarly on small problems, Zephyr provides more robust performance for larger instances, likely due to its higher connectivity and reduced susceptibility to chain breaks.

4.5. Runtime

To assess computational efficiency, we compared the runtime of the QA on D-Wave's Zephyr topology with that of SA for several FT instances, as shown in Table 9.

Table 9: Quantum Annealing (ZE) timing results compared to SA runtime for selected FT instances.

Instance	Size	ZE QPU time (s)	ZE embedding (s)	ZE total QA time (s)	SA runtime (s)
FT.inst.a65	8	0,46984	0,000757	0,470597	83,8359
FT.inst.5	13	0,52448	0,001861	0,526341	115,7224
FT.inst.a29	15	0,49856	0,001414	0,499974	130,4178
FT.inst.a1434	19	0,50224	0,002908	0,505148	145,3695
FT.inst.a954	16	0,51664	0,002865	0,519505	196,2131
FT.inst.9	25	0,54776	0,005427	0,553187	243,1154
FT.inst.11	31	0,5504	0,004057	0,554457	308,4998
FT.inst.a132	41	0,5344	0,005954	0,540354	486,8613
FT.inst.16	49	0,58312	0,011037	0,594157	523,1273
FT.inst.18	55	0,53496	0,007993	0,542953	571,9218
FT.inst.20	61	0,52584	0,010434	0,536274	650,9974
FT.inst.22	67	0,59592	0,009546	0,605466	719,5723
FT.inst.24	73	0,61568	0,010285	0,625965	789,9033
FT.inst.a1433	72	0,58088	0,011177	0,592057	828,5588
FT.inst.28	85	0,55368	0,01167	0,56535	954,3569

The QA total runtime accounts for both the QPU execution time and the embedding time required to map the logical problem onto the hardware. For all tested instances, the total QA runtime remains consistently below one second, ranging approximately between 0.47 s and 0.63 s. The embedding time increases with the size of the instance, reflecting the growing complexity of the embedding procedure on the hardware graph. However, it remains small compared to the QPU execution time and does not significantly affect the overall QA runtime. In contrast, SA requires significantly longer computation times to complete the same number of annealing runs (4000 shots), with runtimes ranging from approximately 80 s to nearly 950 s depending on the instance size. This corresponds to a difference of up to three orders of magnitude compared to QA.

In Figure 6, we plot the distribution of the execution time for the NE method. These results show that the runtime of QA is significantly lower

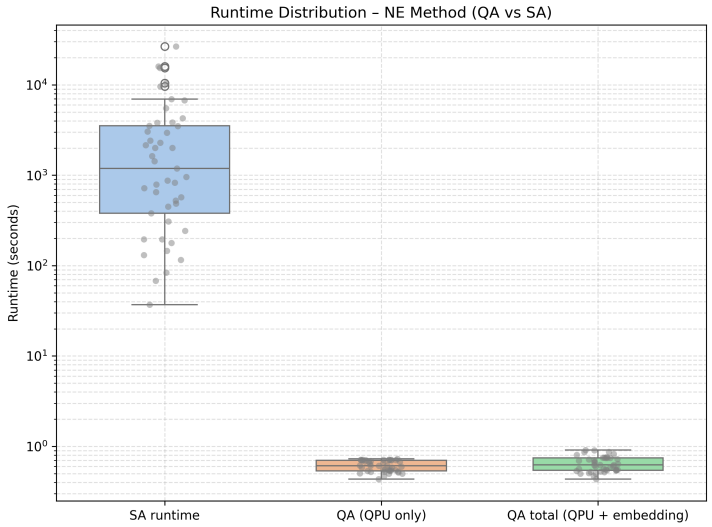


Figure 6: Comparison of SA and QA runtimes for the NE method.

than that of SA for a comparable number of solutions. Although QA does not guarantee the retrieval of all valid solutions due to its limited sampling and probabilistic nature, this speedup represents a strong advantage. In particular, such a runtime gain could be highly beneficial in real-life scenarios where QA could be used iteratively. First to explore a subset of solutions rapidly, then to block already found ones and to re-anneal in order to discover new solutions. This hybrid exploration strategy could help to exploit QA's efficiency while improving solution exhaustivity over multiple runs.

5. Conclusion and Perspectives

This work investigated the applicability of QA to the MCS problem in the context of PSA. By evaluating three different SAT-to-QUBO encoding strategies on both generated 3-SAT instances and real industrial FT systems, we moved beyond idealized benchmarks and examined QA performance on realistic PSA scenarios. Among the three encodings, the SAT-to-QUBO encoding with reusable auxiliary variables demonstrated the best scalability as it required significantly fewer qubits and thus showed more efficient embeddings.

In addition, we assessed the capabilities of D-Wave's Pegasus and the newly released Zephyr architecture. The Zephyr topology improved embed-

dability thanks to its increased connectivity, enabling larger and more complex FT-derived problems to be mapped onto the hardware. To evaluate the quality of the obtained solutions, we analyzed the exhaustivity of both SA and QA. For small instances, SA results confirmed that the NE and Naive encodings correctly recover all satisfying assignments, validating that the QUBO formulations preserve the structure of the SAT solution space. On QA hardware, both architectures were able to retrieve diverse sets of valid solutions for small and medium-sized instances, demonstrating their ability to explore the solution space under favorable embedding conditions, with Zephyr showing improved robustness reflected in the absence of chain breaks for larger instances. In addition, we also investigated the runtimes of both QA and SA approaches. Our results show that QA is significantly faster than SA, and that the speedup increases with the size of the instance. On average, QA is 794 times faster than SA (for the largest instance, QA is even 1688 times faster than SA).

Despite these encouraging results, several limitations of this study should be acknowledged. First, the size of the FT instances that can be embedded remains constrained by the current scale of QA hardware and by the minor-embedding process required to map QUBO interaction graphs onto the hardware topology. Even with the improved connectivity of the Zephyr architecture, very large industrial fault trees cannot yet be fully embedded. Another limitation arises from the fact that the QUBO formulation encodes the satisfiability of the CNF representation of the FT. As a consequence, the QA samples all satisfying assignments corresponding to valid cut sets, whereas the PSA objective is to identify only the MCS. In the present approach, minimality is enforced through a classical post-processing filtering step. Although this filtering can be performed in polynomial time with respect to the number of sampled cuts, it may become costly in practice when the number of retrieved cuts grows exponentially with the size of the system. Designing QUBO formulations that directly target MCS would therefore significantly improve efficiency by reducing the number of required samples.

More generally, QA currently acts as a heuristic sampling approach for the MCS problem rather than a fully exhaustive solver. Its current limitations in scalability and solution completeness restrict its direct applicability to large-scale or real-time industrial PSA scenarios. Its main advantage lies in its ability to rapidly sample diverse solutions, potentially exploring different regions of the solution space. The objective of this work was therefore to explore both the potential and the current limitations of QA in the context

of PSA applications.

Despite these limitations, our study indicates that QA remains a promising direction for PSA, especially as hardware and embedding techniques continue to evolve. Several research directions emerge from this work. These include developing more compact SAT encodings through formula compression, designing FT-to-SAT transformations that minimize auxiliary variables, and formulating QUBOs that better target minimality properties of cut sets.Additionally, a comparison with state-of-the-art classical MCS solvers would provide valuable insights into the practical competitiveness of QA-based approaches. Besides, advanced QA strategies such as reverse annealing, iterative solution-blocking, and hybrid quantum–classical loops have the potential to improve significantly the exploration of the solution space and move QA closer to exhaustive MCS identification.

References

[1] Enno Ruijters and Mariëlle Stoelinga, *Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools*, Computer Science Review, Vol. 15–16, pp. 29–62, 2015. DOI: 10.1016/j.cosrev.2015.03.001.

[2] M. Čepin, B. Mavko, A dynamic fault tree, *Reliability Engineering & System Safety*, Vol. 75, No. 1, 2002, pp. 83–91.

[3] M. Čepin, Optimization of safety equipment outages improves safety, *Reliability Engineering & System Safety*, Vol. 77, No. 1, 2002, pp. 71–80.

[4] A. Volkanovski, M. Čepin, B. Mavko, Application of the fault tree analysis for assessment of power system reliability, *Reliability Engineering & System Safety*, Vol. 94, No. 6, 2009, pp. 1116–1127.

[5] S. Baek, G. Heo, Application of dynamic fault tree analysis to prioritize electric power systems in nuclear power plants, *Energies*, Vol. 14, No. 14, 2021, Art. no. 4119.

[6] D. Zhang, J. Zhou, Research on reliability of nuclear power plant steam generator liquid level control system based on dynamic fault tree, *Nuclear Physics Review*, Vol. 38, No. 4, 2021, pp. 479–486.

[7] I. Dagal, Probabilistic fault tree analysis and dynamic redundancy optimization for next-generation avionic flight control systems, *Reliability Engineering & System Safety*, Vol. 266, Part B, 2026, Art. no. 111841.

[8] O. Coudert and J.-C. Madre, *Implicit and incremental computation of primes and essential primes of boolean functions*, Proceedings of the Design Automation Conference, pp. 36–39, 1992.

[9] S. B. Akers, *Binary decision diagrams*, IEEE Transactions on Computers, vol. C-27, no. 6, pp. 509–516, 1978. DOI: 10.1109/TC.1978.1675141.

[10] Olivier P. M. Nusbaumer, *Analytical solutions of linked fault tree probabilistic risk assessments using binary decision diagrams with emphasis on nuclear safety applications*, PhD Thesis, ETH Zurich, 2007.

[11] J. B. Fussell and W. E. Vesely, *A New Methodology For Obtaining Cut Sets For Fault Trees*, Transactions of the American Nuclear Society, vol. 15, no. 1, pp. 262–263, 1972.

[12] P. K. Pande, M. E. Spector, P. Chatterjee, *Computerized fault tree analysis: TREEL and MICSUP*, Technical Report, Operations Research Centre, University of California, Berkeley, 1975.

[13] V. Matuzas, S. Contini, Dynamic labelling of BDD and ZBDD for efficient non-coherent fault tree analysis, *Reliability Engineering & System Safety*, Vol. 144, 2015, pp. 183–192.

[14] R. Remenyte-Prescott, J. Andrews, Analysis of non-coherent fault trees using ternary decision diagrams, *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, Vol. 222, No. 2, 2008, pp. 127–138.

[15] W.S. Jung, Zero-suppressed ternary decision diagram algorithm for solving non-coherent fault trees in probabilistic safety assessment of nuclear power plants, *Nuclear Engineering and Technology*, Vol. 56, No. 6, 2024, pp. 2092–2098.

[16] S. Zhou, Z. Li, J. Xiang, Reliability analysis of dynamic fault trees with Priority-AND gates using conditional binary decision diagrams, *Reliability Engineering & System Safety*, Vol. 253, 2025, Art. no. 110495.

[17] S. Wang, et al., Rapid computation of survival signature for dynamic fault tree based on sequential binary decision diagram and multidimensional array, *Reliability Engineering & System Safety*, Vol. 253, 2025, Art. no. 110552.

[18] J.Y. Yoon, S.H. Han, Seismic risk quantification for multi-unit probabilistic safety assessment based on the partial binary decision diagram approach, *Reliability Engineering & System Safety*, Vol. 265, Part A, 2026, Art. no. 111581.

[19] M. Hibti, *D6.12: NEASQC Approximations-like QPSA Algorithms State of the Art*, EDF R&D, 2024.

[20] M. Čepin, Analysis of truncation limit in probabilistic safety assessment, *Reliability Engineering & System Safety*, Vol. 87, No. 3, 2005, pp. 395–403.

[21] G. San Martín Silva, T. Parhizkar, E. López Droguett, *Quantum Fault Trees*, Department of Civil & Environmental Engineering, University of California, Los Angeles, and The B. John Garrick Institute for the Risk Sciences, UCLA.

[22] G. San Martín Silva and E. López Droguett, "Quantum fault trees and minimal cut sets identification," *Reliability Engineering & System Safety*, vol. 262, p. 111147, 2025.

[23] M. Rennela, S. Brand, A. Laarman, and V. Dunjko, *Hybrid Divide-and-Conquer Approach for Tree Search Algorithms*, in *Proceedings of the 27th International Conference on Principles and Practice of Constraint Programming (CP)*, 2021.

[24] A. Zaiou, *Quantum machine learning approaches for graphs and sequences: application to nuclear safety assessment*, PhD Thesis, Université Paris-Nord – Paris XIII, 2022.

[25] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model," *Physical Review E*, vol. 58, no. 5, pp. 5355–5363, 1998.

[26] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, p. 015002, 2018.

[27] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[28] J. Brooke, D. Bitko, T. F. Rosenbaum, and G. Aeppli, *Quantum annealing of a disordered magnet*, Science, 284(5415):779–781, 1999. doi: 10.1126/science.284.5415.779. PMID: 10221904.

[29] T. Kato, "On the adiabatic theorem of quantum mechanics," *Journal of the Physical Society of Japan*, vol. 5, no. 6, pp. 435–439, 1950.

[30] D-Wave Systems Inc., "D-Wave QPU Solver Properties (Documentation)," available at: `https://docs.dwavequantum.com/en/latest/quantum_research/solver_properties_specific.html#qpu-solver-properties-specific`

[31] E. Ruijters and M. Stoelinga, "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools,"

[32] G. S. Tseytin, *On the complexity of derivation in propositional calculus*, in *Studies in Constructive Mathematics and Mathematical Logic*, Part II, Seminars in Mathematics, pp. 115–125, Steklov Mathematical Institute, 1970.

[33] F. Glover, G. Kochenberger, R. Hennig, et al., *Quantum bridge analytics I: a tutorial on formulating and using QUBO models*, Annals of Operations Research, 314:141–183, 2022. doi: 10.1007/s10479-022-04634-2.

[34] F. Glover, G. Kochenberger, R. Hennig, and Y. Du, "Quantum bridge analytics I: A tutorial on formulating and using QUBO models," *Annals of Operations Research*, vol. 314, no. 1, pp. 141–183, 2022.

[35] Ö. Salehi, A. Glos, and J. A. Miszczak, "Unconstrained binary models of the travelling salesman problem variants for quantum optimization," *Quantum Information Processing*, vol. 21, no. 2, p. 67, 2022.

[36] Xiaotian Li, Shunsuke Tsukiyama, Koji Nakano, Victor Parque, Yasuaki Ito, Takumi Kato, Yuya Kawamata, Kaiki Ii, *Effectively encoding satisfiability problems into quadratic unconstrained binary optimization models for quantum computing*, International Journal of Parallel, Emergent and Distributed Systems, 2025, `http://dx.doi.org/10.1080/17445760.2025.2472205`.

[37] S. Deleplanque, L. F. Pérez Armas, and S. Creemers, "solqhealer: Quantum procedures for rendering infeasible solutions feasible: A proof of concept with the maximum independent set problem and 3-SAT," *Journal of Heuristics*, vol. 31, no. 3, pp. 1–28, 2025.

[38] M. R. Garey and D. S. Johnson, "Strong" NP-completeness results: Motivation, examples, and implications, *Journal of the ACM*, 25(3):499–508, 1978.