

ILP-Based QUBO Encodings of the Traveling Salesman Problem: Derivation and Benchmarking on Quantum Annealing Hardware

Katarzyna Nałęcz-Charkiewicz

Abstract—The Traveling Salesman Problem (TSP) has been one of the most intensively studied problems in theoretical computer science for decades. Being NP-hard, it is unlikely to be solved in polynomial time; hence, despite numerous classical exact and approximate methods, quantum-inspired approaches are also explored. In this study, we analyze several formulations of the TSP suitable for quantum annealing, focusing not only on qubit requirements but also on practical performance, i.e., how frequently feasible and optimal solutions are obtained. We consider both complete and non-complete graphs, reflecting common real-world scenarios. One of the analyzed formulations corresponds to a previously established Quadratic Unconstrained Binary Optimization (QUBO) model, while the others originate from classical Integer Linear Programming (ILP) formulations. We provide a unified, step-by-step translation of selected compact ILP TSP formulations into QUBO models. We analytically show that for non-complete graphs these encodings are asymptotically smaller than the commonly used baseline formulation. We benchmark the formulations using simulated and quantum annealing to evaluate their feasibility and optimality rates. The results are discussed in the context of current quantum annealing hardware limitations, including QPU size and embedding overhead.

Keywords—Traveling Salesman Problem, Quantum Annealing, QUBO, Integer Programming, Digital Systems, QPU Architecture

I. INTRODUCTION

A. The Quantum Computing Context and Optimization

At the dawn of the “quantum era” [1], the development and improvement of algorithms for quantum computers is of particular importance, even though the currently available hardware is still at an early stage of technological maturity. However, in order to be ready for the moment when the appropriate size and reliability of quantum computers will allow us to exploit their potential to efficiently solve real-world problems, it is important now to learn about their limitations, as well as to improve computational methods using the quantum paradigm.

Solving combinatorial problems is one of the classic challenges of discrete optimization, often involving NP-hard tasks. Besides exact algorithms, heuristic approaches also play a significant role in solving such problems. One example of this

type of algorithm is simulated annealing (SA) — a stochastic optimization method inspired by the annealing process in metallurgy, used to search for approximate minima of the objective function. A similar concept is the basis of the quantum annealing method, which uses the effects of quantum mechanics — in particular quantum tunnelling [2] — in the process of searching the solution space. The device implementing the quantum annealing algorithm was manufactured e.g. by D-Wave [3]. Their quantum annealer (QA) is a special-purpose quantum computer dedicated to optimisation tasks. It has been experimentally shown that, during operation, a quantum annealing processor exhibits quantum coherence and entanglement within subsets of its qubits [4].

In order to solve an optimisation task using a QA, the problem must be presented in the form of an unconstrained quadratic binary optimisation task (QUBO) or, equivalently, as an Ising model [5]. Finding the solution of a problem expressed as a QUBO is equivalent to finding the minimum of a quadratic function of binary variables. It should be emphasised that by expressing a computational problem in the form of a QUBO, we are not restricted to the quantum annealing domain. Problems presented in this form can also be the input to, for example, algorithms in the more common quantum gate model [6], digital annealer (e.g. designed by Fujitsu [7]), or Variational Quantum Eigensolver (VQE).

B. The Traveling Salesman Problem

The Traveling Salesman Problem (TSP) is a classical combinatorial optimization problem, in which the goal is to find the shortest possible route visiting each city (vertex of the graph) exactly once and returning to the starting point. In the symmetric TSP variant (STSP), the distance between two cities is assumed to be the same in both directions. The ubiquity of TSP applications — in fields ranging from logistics and route planning to manufacturing (where it helps e.g. to optimize the order of repetitive machine operations [8]) — makes it a particularly appealing subject of continued research. In this work, we do not restrict the analysis to complete graphs; instead, we also consider sparse graphs of bounded degree, which naturally arise in many practical applications.

Author is with Institute of Computer Science, Faculty of Electronics and Information Technology, Warsaw University of Technology, Poland (e-mail: katarzyna.nalecz-charkiewicz@pw.edu.pl).



C. Challenges in QUBO Formulation

The choice of QUBO encoding has a direct impact on the *hardware feasibility* of a problem: the number of binary variables determines the minimum number of logical qubits required, while the coupling structure determines how many physical qubits are needed after minor embedding onto the QPU topology. For the D-Wave Advantage processor with its Pegasus topology, each logical variable typically requires a chain of several physical qubits, so reducing the variable count and coupling density directly extends the solvable problem size [10].

The TSP formulation as a QUBO model was first proposed by Lucas [9] and has since become a standard benchmark for quantum annealing hardware. However, the Lucas formulation requires $O(n^2)$ binary variables for a graph with n nodes, which rapidly exhausts the qubit budget of current devices. Classical Integer Linear Programming (ILP) has produced several compact subtour-elimination formulations of the TSP that, when translated to QUBO via binary encoding, yield models whose variable count grows more slowly with n for sparse graphs.

D. Related Work

The performance of QUBO and Ising formulations has been studied on quantum annealing hardware and related Ising-machine platforms for several combinatorial problems. Salehi et al. [22] investigate QUBO formulations of TSP variants and report experiments on a quantum annealing device. Stogiannos et al. benchmark D-Wave quantum annealers and hybrid solvers on HCP and TSP instances. Zieliński et al. [21] examine how different 3SAT-to-QUBO transformations affect solution quality on the D-Wave Advantage system. Related work also analyses formulation-level aspects: Papalitsas et al. [23] propose a QUBO model for the TSP with time windows, while Tamura et al. [20] study the impact of binary integer encodings on Ising-machine performance.

E. Objectives and Contributions

This paper makes the following contributions:

- A unified, step-by-step QUBO derivation of five ILP-based TSP formulations, with explicit binary variable counts (Section II-B);
- Analytical bounds showing that compact ILP encodings are asymptotically smaller than the Lucas baseline for non-complete graphs (Section III);
- Experimental benchmarking on D-Wave Advantage QPU and simulated annealing across 17 instances (Section II);
- A discussion of architectural implications for near-term QPU hardware design (Section IV).

Detailed step-by-step mathematical derivations of the QUBO penalty terms for each formulation are provided in the supplementary material available at [29], together with the full experimental data and source code.

II. MATERIALS AND METHODS

A. Problem Definition

This paper considers the Symmetric Traveling Salesman Problem (STSP) defined on an undirected, connected graph. Each vertex represents a city, while the edge weights correspond to the costs of traveling between cities. The optimisation goal is to determine a Hamilton cycle with the minimum total cost.

Unlike classical TSP benchmarks (such as TSPLIB), the analysis is not limited to complete graphs; sparse graphs with a small number of vertices are also an important part of the study. The variant of the problem adopted in many practical applications is closer to real-world scenarios than the complete graph model. This approach, which takes into account general graphs without assuming their completeness, affects the form of the resulting QUBO models, in some cases leading, as we show below, to a smaller asymptotic number of binary variables in the QUBO model.

B. Formulations

1) *Notation*: Let us introduce the following notation: E denotes the set of edges in graph G , and V denotes the set of its vertices, $V = \{1, \dots, n\}$, where each vertex represents a city; c_{ij} is a distance between cities i and j . The Traveling Salesman Problem consists of visiting each of the cities (vertices of the graph) exactly once and then returning to the starting point, so that the total distance traveled is as short as possible.

Please note that the notation used throughout this document differs from that found in the original formulations. These changes were introduced to unify the notation across all formulations for clarity and consistency. Both the notation used in this subsection and the formal (mathematical) representation of the formulations presented here are largely based on the comparative studies of ATSP formulations presented in [11] and [12].

2) *Lucas*: We denote by QUBO_L the QUBO formulation of the TSP proposed by Lucas [9], that uses a positional encoding: each city $i \in \{1, \dots, n\}$ is assigned a tour position $t \in \{1, \dots, n\}$, and the binary variables $p_{i,t}$ indicate whether city i occupies position t . We recall Lucas's construction, in which the energy (Hamiltonian) is defined so that its minimizers correspond to valid Hamiltonian cycles (and, with costs, to optimal TSP tours).

The cost function of the tour is encoded as:

$$H_C = \sum_{\substack{(i,j) \in E \\ i \neq j}} c_{ij} \sum_{t=1}^n p_{i,t} p_{j,t+1}. \quad (1)$$

To ensure that each city appears exactly once in the tour, and that each position is filled by exactly one city, the following penalty terms are added to the energy function:

$$H_{H_1} = \sum_{i=1}^n \left(1 - \sum_{t=1}^n p_{i,t}\right)^2 + \sum_{t=1}^n \left(1 - \sum_{i=1}^n p_{i,t}\right)^2 \quad (2)$$

The subscript H indicates that these terms enforce *Hamiltonicity* (assignment to a Hamiltonian cycle). In situation when

both variables $p_{i,t}$ and $p_{j,t+1}$ are equal 1 and $(i, j) \notin E$, an additional energy penalty should be added (to penalize selecting city i immediately before j):

$$H_{H_2} = \sum_{\substack{(i,j) \notin E \\ i \neq j}} \sum_{t=1}^n p_{i,t} p_{j,t+1}, \quad (3)$$

The complete QUBO Hamiltonian is:

$$H = H_C + A \cdot H_{H_1} + B \cdot H_{H_2} \quad (4)$$

where A and B are penalty weights large enough to discourage constraint violations.

This formulation is widely used and often considered a state-of-the-art QUBO representation of the TSP [13]. It is also the one implemented by D-Wave Systems in their SDK [14] (available in the `dwave-networkx` package). In this positional encoding, there are n^2 binary variables (one $p_{i,t}$ per city–position pair), so the QUBO matrix has dimension $n^2 \times n^2$; equivalently, both the variable count and the matrix dimension grow as $\mathcal{O}(n^2)$, independently of $|E|$.

3) *Miller–Tucker–Zemlin (MTZ)*: We denote by QUBO_{MTZ} the QUBO formulation derived from the classical Miller–Tucker–Zemlin (MTZ) model [15], which is *edge-based*. Here, $x_{ij} \in \{0, 1\}$ is a binary decision variable equal to 1 if directed edge (i, j) is traversed in the tour, and 0 otherwise. The original integer-linear MTZ model is:

$$\min \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j \\ (i,j) \in E}}^n c_{ij} x_{ij} \quad (5)$$

$$x_{ij} \in \{0, 1\}, \quad (i, j) \in E, \quad (6)$$

$$u_1 = 1, \quad u_i \in \{2, \dots, n\}, \quad i = 2, \dots, n, \quad (7)$$

$$\sum_{\substack{i=1 \\ i \neq j}}^n x_{ij} = 1, \quad j = 1, \dots, n, \quad (8)$$

$$\sum_{\substack{j=1 \\ j \neq i}}^n x_{ij} = 1, \quad i = 1, \dots, n, \quad (9)$$

$$u_i - u_j + (n-1)x_{ij} \leq n-2, \quad 2 \leq i \neq j \leq n. \quad (10)$$

The integer ordering variables u_i are encoded in binary as:

$$v = L + \sum_{k=0}^{m_v-1} 2^k b_k^{(v)}, \quad b_k^{(v)} \in \{0, 1\}, \quad (11)$$

$$m_v = \lceil \log_2(U - L + 1) \rceil.$$

For brevity we denote this encoding by $\text{bin}(v; L, U)$ and use it throughout. The ordering variables are encoded as $\text{bin}(u_i; 2, n)$, $i = 2, \dots, n$, using $m_u = \lceil \log_2(n-1) \rceil$ bits per u_i .

The *range Hamiltonian* enforcing the upper bound on u_i is:

$$H_R = \sum_{i=2}^n \left(\text{bin}(u_i; 2, n) - n - \text{bin}(s_i^{[7]}; 0, n-2) \right)^2. \quad (12)$$

Here, $s_i^{[7]}$ is the slack variable introduced to convert the upper-bound constraint (7) on u_i into an equality.

The *subtour-elimination Hamiltonian* derived from (10) is:

$$H_S = \sum_{\substack{(i,j) \in E \\ 2 \leq i \neq j \leq n}} \left(\text{bin}(u_i; 2, n) - \text{bin}(u_j; 2, n) + (n-1)x_{ij} - \text{bin}(s_{ij}^{[10]}; 0, n-2) - (n-2) \right)^2. \quad (13)$$

The cost Hamiltonian and degree-constraint Hamiltonian are:

$$H_C = \sum_{(i,j) \in E, i \neq j} c_{ij} x_{ij}, \quad (14)$$

$$H_H = \sum_{j=1}^n \left(1 - \sum_{\substack{i=1 \\ (i,j) \in E \\ i \neq j}} x_{ij} \right)^2 + \sum_{i=1}^n \left(1 - \sum_{\substack{j=1 \\ (i,j) \in E \\ j \neq i}} x_{ij} \right)^2. \quad (15)$$

The subscript H in H_H denotes that this term enforces the Hamiltonian-cycle degree conditions (every city has exactly one incoming and one outgoing arc in the tour).

The complete QUBO_{MTZ} Hamiltonian is therefore:

$$H = H_C + A \cdot H_H + B \cdot H_R + C \cdot H_S, \quad (16)$$

where A , B , and C are positive penalty weights.

4) *Desrochers–Laporte (DL)*: We denote by QUBO_{DL} the QUBO formulation derived from the integer programming model proposed by Desrochers and Laporte [16]. It replaces the classical MTZ subtour elimination constraints (7) and (10) with the following ones (see Section 2 and equations (8), (10), (11) in [16] for more details):

$$u_i - u_j + (n-1)x_{ij} + (n-3)x_{ji} \leq n-2, \quad \forall (i, j) \in E, i \neq j, i, j = 2, \dots, n \quad (17)$$

$$1 + (n-3)x_{i1} + \sum_{j=2}^n x_{ji} \leq u_i, \quad \forall i = 2, \dots, n \quad (18)$$

$$u_i \leq n-1 - (n-3)x_{1i} - \sum_{j=2}^n x_{ij}, \quad \forall i = 2, \dots, n \quad (19)$$

Integer variables u_i are encoded using binary encoding (11). The QUBO has the general structure given by equation (16), where H_C , H_H and H_R remain as in QUBO_{MTZ} , and H_S is the sum of squared residuals corresponding to constraints (17)–(19). The explicit expressions for $H_S^{(17)}$, $H_S^{(18)}$, and $H_S^{(19)}$ are provided in the supplementary material [29].

5) *Sherali–Driscoll (SD)*: We denote by QUBO_{SD} the QUBO formulation based on the model introduced by Sherali and Driscoll [17]. Similarly to QUBO_{DL} , this formulation differs from QUBO_{MTZ} in the conditions used to eliminate subtours. An additional variable, y_{ij} , is introduced to represent the order of edge (i, j) in the tour. For more details see Section 1 and equations (15)–(21) in [17].

$$\sum_{j=2}^n y_{ij} + (n-1)x_{i1} = u_i \quad \forall i = 2, \dots, n, \quad (20)$$

$$\sum_{i=2}^n y_{ij} + 1 = u_j \quad \forall j = 2, \dots, n, \quad (21)$$

$$x_{ij} \leq y_{ij} \quad \forall i, j = 2, \dots, n, \quad (22)$$

$$y_{ij} \leq (n-2)x_{ij} \quad \forall i, j = 2, \dots, n, \quad (23)$$

$$(24)$$

$$u_j + (n-2)x_{ij} - (n-1)(1-x_{ji}) \leq y_{ij} + y_{ji} \quad \forall i, j = 2, \dots, n. \quad (25)$$

$$y_{ij} + y_{ji} \leq u_j - (1-x_{ji}) \quad \forall i, j = 2, \dots, n. \quad (26)$$

$$1 + (1-x_{1j}) + (n-3)x_{j1} \leq u_j \quad \forall j = 2, \dots, n. \quad (27)$$

$$u_j \leq (n-1) - (n-3)x_{1j} - (1-x_{j1}), \quad \forall j = 2, \dots, n. \quad (28)$$

The complete QUBO_{SD} Hamiltonian takes the form given by Equation (16), where $H_S = H_S^{(20)} + \dots + H_S^{(28)}$, with each $H_S^{(k)}$ being the squared-residual penalty term corresponding to constraint (k). The explicit expressions are provided in the supplementary material [29].

6) *Gavish-Graves (GG)*: We denote by QUBO_{GG} the QUBO formulation derived from the classical single-commodity flow model introduced by Gavish and Graves [18]. In this formulation, the tour is interpreted as a flow of a single fictitious commodity from all vertices toward a reference vertex, with flow conservation and capacity constraints ensuring tour validity and eliminating subtours.

The original ILP model by Gavish and Graves includes (5), (6), (8), (9) and:

$$\sum_{j=1}^n g_{ji} - \sum_{j=2}^n g_{ij} = 1 \quad \forall i = 2, \dots, n \quad (29)$$

$$0 \leq g_{ij} \leq (n-1) \cdot x_{ij} \quad \forall i = 1, \dots, n; j = 2, \dots, n, \quad (30)$$

where g_{ij} is an integer variable denoting the amount of flow from vertex i to j . The formulation QUBO_{GG} takes the form:

$$H = H_C + A \cdot H_H + B \cdot H_S, \quad (31)$$

with $H_S = H_S^{(29)} + H_S^{(30)}$:

$$H_S^{(29)} = \sum_{i=2}^n \left(\sum_{j=1}^n \text{bin}(g_{ji}; 0, n-1) - \sum_{j=2}^n \text{bin}(g_{ij}; 0, n-1) - 1 \right)^2, \quad (32)$$

$$H_S^{(30)} = \sum_{i=1}^n \sum_{j=2}^n \left((n-1)x_{ij} - \text{bin}(g_{ij}; 0, n-1) - \text{bin}(s_{ij}^{[30]}; 0, n-1) \right)^2. \quad (33)$$

7) *Finke-Claus-Gunn (FCG)*: The formulation denoted as QUBO_{FCG} was introduced by Finke, Claus, and Gunn [19] and belongs to the class of two-commodity flow models. It is an extension of single-commodity flow formulations, such as the one by Gavish and Graves described in the previous subsection. Two flow variables are introduced for each directed edge (i, j) : w_{ij} (commodity 1) and z_{ij} (commodity 2). Although introduced as continuous, these variables are implicitly integer when x_{ij} is binary.

The complete formulation consists of (5), (6), (8), (9), as well as:

$$\sum_{j=2}^n (w_{1j} - w_{j1}) = n-1, \quad (34)$$

$$\sum_{j=1}^n (w_{ij} - w_{ji}) = -1, \quad \forall i = 2, \dots, n; i \neq j, \quad (35)$$

$$\sum_{j=2}^n (z_{1j} - z_{j1}) = -(n-1), \quad (36)$$

$$\sum_{j=1}^n (z_{ij} - z_{ji}) = 1, \quad \forall i = 2, \dots, n; i \neq j, \quad (37)$$

$$\sum_{j=1}^n (w_{ij} + z_{ij}) = n-1, \quad \forall i = 1, \dots, n, \quad (38)$$

$$w_{ij} + z_{ij} = (n-1) \cdot x_{ij}, \quad i, j = 1, \dots, n. \quad (39)$$

It is worth noting that the formulation contains no inequality constraints; as a result, no slack variables for inequality constraints need to be introduced. However, the integer flow variables w_{ij} and z_{ij} are still encoded in binary form, contributing additional binary variables to the QUBO. The QUBO takes the form of (31). Here $H_S = H_S^{(34)} + \dots + H_S^{(39)}$, with each $H_S^{(k)}$ being the squared-residual penalty for constraint (k). The explicit expressions are provided in the supplementary material [29].

C. Test Instances and Data Generation

The experiments were conducted on a set of Traveling Salesman Problem instances defined on small undirected graphs. It was assumed that the analysed structures are connected and simple graphs, but not necessarily complete. The size of the graphs was chosen so that the corresponding QUBO problems could fit entirely on a QPU. This design assumption was intentionally made to avoid the need for problem decomposition and hybrid solvers, allowing for a direct assessment of the QPU's performance on the formulated models.

The decision not to use commonly used benchmarks, such as the **TSPLIB** library, was dictated by the specific nature of the instances available there. According to the documentation of the collection, almost all cases, with the exception of the

Hamilton cycle problem, are defined for complete graphs. In turn, the available instances for the Hamilton cycle problem are characterised by an excessive number of vertices (over 1000), which makes it impossible to meet the assumption of complete embedding of the problem on the QPU. It should be noted that the final number of qubits used depends both on the processor architecture and the choice of embedding algorithm; in this study, we decided to use the algorithm offered by default by the D-Wave API [14].

The following data sets were used in the research:

- 1) **House of Graphs collection:** Eight Hamiltonian graphs with between 5 and 7 vertices were selected from the repository <https://houseofgraphs.org>. The graph identifiers used are: 1094, 226, 438, 642, 702, 742, 868, and 204.
- 2) **Synthetic and external graphs:** The dataset was supplemented with the FIVE instance [30] and a series of randomly generated synthetic graphs, labelled Experiment_1, Experiment_2 and Experiment_3.

The original graphs from the House of Graphs repository do not have assigned edge weights. They were assigned by random draw from the integers in the range 1 to 100, with uniform distribution. Visualisations of all analysed graph instances are available in the supplementary repository [29]. To obtain reference, exact solutions to the Traveling Salesman Problem in a deterministic manner, the `python_tsp` library [28] and the Bellman–Held–Karp algorithm [26], [27] were used. The complete instance characteristics table is available in the supplementary material [29].

D. QUBO Construction and Scaling

All formulations of the Traveling Salesman Problem in ILP form considered in this paper have been transformed into QUBO form as described in Section II-B, using a *penalty model* in which the problem constraints are encoded as penalty terms added to the objective function. The weights of these components, called Lagrange multipliers, control the relative importance of satisfying the constraints compared to minimising the route cost. In this paper, the values of the Lagrange multipliers were analysed within a fixed range common to all formulations, namely 2, 5, 10 and 20.

Model parameter normalisation was used in the instance preparation process. The QUBO matrix was scaled using a max-absolute normalisation such that the absolute values of all linear and quadratic coefficients did not exceed the prescribed threshold `target_max_bias` (=1.0). Subsequently, the chain strength was determined as the product of a relative scaling factor `chain_strength_ratio` (=1.8) and the maximum absolute QUBO coefficient after scaling.

For each formulation, the number of binary variables and the number of quadratic terms were recorded. These values were treated as an approximate measure of the degree of coupling in the QUBO model and were used to analyse the impact of the problem structure on the effectiveness of annealing algorithms.

E. Solvers and Experimental Setup

All constructed QUBO models were evaluated using two annealing algorithms: classical simulated annealing (SA), serving as a baseline, and quantum annealing (QA). The aim of the experimental methodology was to conduct a comparative analysis of the impact of the problem formulation and the resulting QUBO model structure on the effectiveness of both optimisation processes.

In the case of SA, the algorithm implemented in the D-Wave Ocean SDK package was used [14]. For each instance of the problem and each formulation, a series of three independent solver runs were performed for a given set of parameters. The experiments analysed the impact of the following parameters:

- number of readings (*num_reads*),
- number of steps (*num_sweeps*),
- annealing schedule (*beta_schedule*) – linear and geometric variants,
- penalty coefficient (Lagrange multiplier).

Experiments using QA were conducted on actual quantum processors (QPUs) from D-Wave, made available through the PLGrid infrastructure (<https://plgrid.pl/>). The calculations were performed based on two hardware topologies: Pegasus and Zephyr.

For each instance, the model was embedded on the physical qubit structure using the default minor embedding heuristics offered by the D-Wave API. The number of reads (*num_reads*) was set at a fixed level of 500, which was due to the need for effective management of the computational budget and the limited availability of QPU time. As in the case of SA, a fixed range of Lagrange coefficient values was analysed. Due to the probabilistic nature of quantum computations, each experimental configuration was repeated three times, providing a basic assessment of result variability within the available QPU budget.

F. Evaluation Metrics

The aim of the evaluation was to compare the impact of the QUBO encoding choice on the feasibility and optimality rates obtained by each annealing algorithm. The analysis distinguished between two aspects: the ability to generate solutions that meet the imposed constraints (*feasibility*) and the ability to obtain optimal solutions (*optimality*).

The formal criteria for evaluating the results were defined as follows: a solution was considered feasible if it satisfied all the structural constraints of the TSP problem, i.e. it constituted a correct Hamilton cycle in which each city was visited exactly once. In turn, a solution is considered optimal if it is acceptable and its cost is equal to the known optimal value for a given instance.

In order to compare the formulations, the following measures were introduced:

- *Feasible solution ratio*: the ratio of the number of feasible solutions obtained to the total number of solver runs for a given configuration of parameters and formulation.
- *Optimal solution ratio*: the ratio of the number of optimal solutions obtained to the total number of solver runs for a given parameter configuration and formulation.

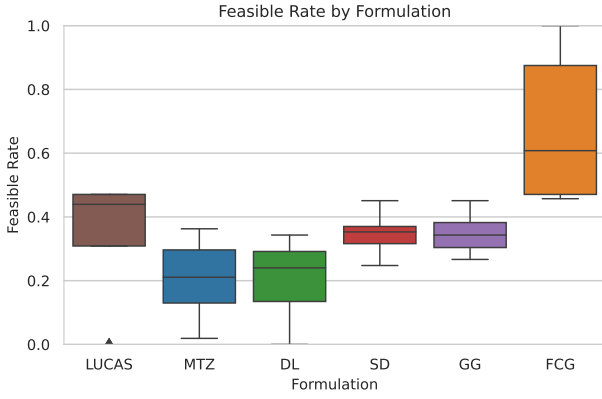


Fig. 1. Average feasible solution ratio obtained with simulated annealing (SA) for different problem formulations. Results are averaged over all tested problem instances and parameter settings.

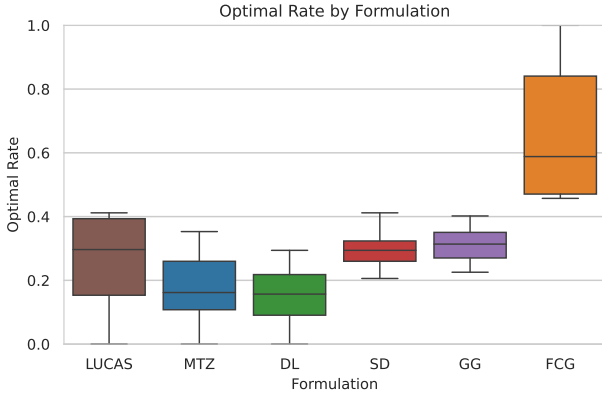


Fig. 2. Average optimal solution ratio obtained with simulated annealing (SA) for different problem formulations. Results are averaged over all tested problem instances and parameter settings.

A single run was understood as one solver call generating *num_reads* samples. The results were aggregated by averaging them over multiple independent runs, different problem instances, and different solver parameter settings.

III. RESULTS

The analysis of the obtained results was primarily aimed at determining the impact of the problem formulation on the effectiveness of optimisation, first in the simulated annealing process (which served as a reference point), and then in quantum annealing.

Figure 1 presents the distribution of the feasible rate measure, and Figure 2 presents the optimal rate measure for the six formulations analysed in the SA algorithm. The highest feasibility and optimality rates were recorded for the FCG formulation. In terms of the feasible rate, the following formulations ranked next: LUCAS, SD and, slightly weaker, GG. In the case of the optimal rate, the GG formulation came second after the winning FCG, while the LUCAS and SD models achieved similar results.

Analysis of tabular data (available in the repository [29]) indicates that a relatively small number of experiments resulted in finding acceptable solutions, and even fewer found optimal ones. The success of the FCG formulation suggests that in the

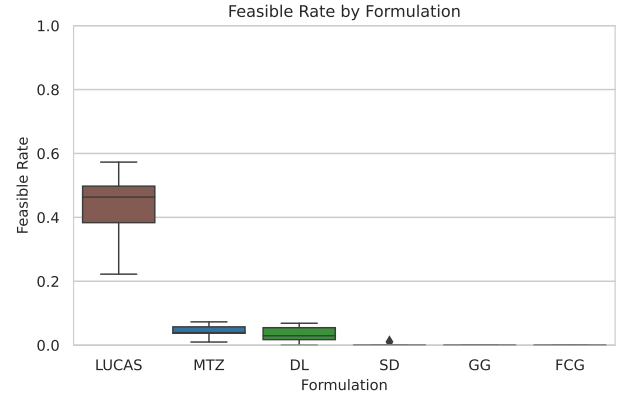


Fig. 3. Average feasible solution ratio obtained with quantum annealing (QA) for different problem formulations. Results are averaged over all tested problem instances and parameter settings.

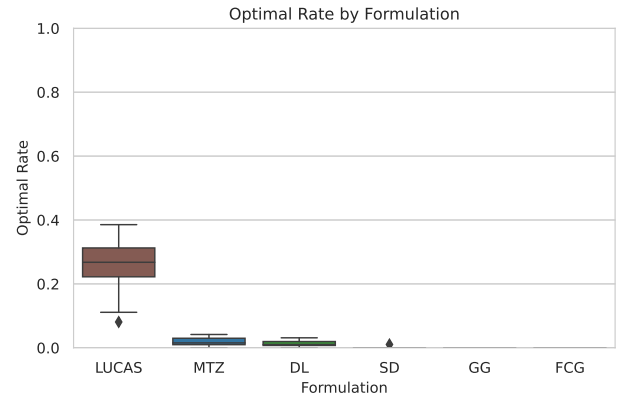


Fig. 4. Average optimal solution ratio obtained with quantum annealing (QA) for different problem formulations. Results are averaged over all tested problem instances and parameter settings.

case of SA, other factors, such as the structure of constraints, have a greater impact on the result than the size of the QUBO matrix itself. For example, the LUCAS formulation, despite having the smallest number of variables used to encode each of the problems considered, did not prove to be the most effective. This confirms the thesis that achieving global minima remains a difficult task and is limited by the characteristics of the heuristic itself, even for small instances such as the test instances considered.

Experiments conducted on a real quantum processor (Figures 3 and 4) show that the effectiveness of QA is even more strongly correlated with the choice of formulation than in the case of SA. As predicted based on previous studies [25], the QA algorithm performed significantly worse than SA – acceptable solutions were found less frequently, and global optimum hits were sporadic. In the case of GG and FCG formulations, the QA algorithm did not generate any feasible solutions. The LUCAS formulation showed the highest effectiveness in the quantum environment, which is consistent with its frequent use as a reference point in the literature on QUBO for the TSP problem [22]–[24].

A comparison of both algorithms in terms of the number of binary variables (Figure 5) and the number of quadratic interactions (Figure 6) leads to the following conclusions: as

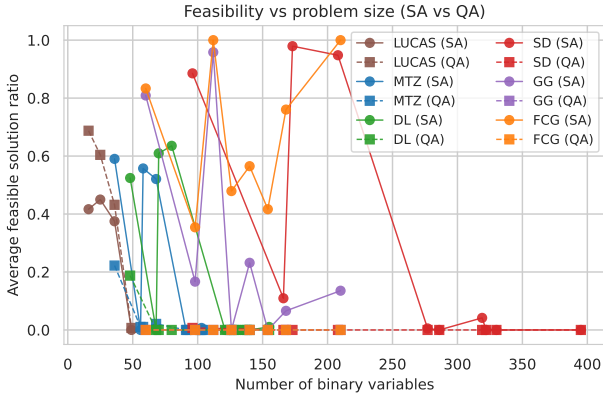


Fig. 5. Comparison of feasible solution ratios obtained with simulated annealing and quantum annealing as a function of the number of binary variables.

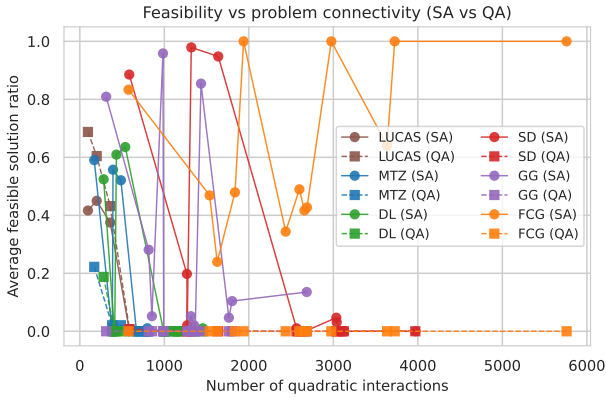


Fig. 6. Comparison of feasible solution ratios obtained with simulated annealing and quantum annealing as a function of the number of quadratic interactions.

the size of the problem increases, measured by the number of variables, a systematic decrease in the feasible rate is observed for both methods, with quantum annealing showing a much steeper decline. A similar trend is observed in the analysis of the number of quadratic terms, treated as a measure of coupling complexity. Within the scope of the investigated problem instances, quantum annealing does not provide an advantage over simulated annealing in terms of its ability to find feasible solutions as the difficulty of the instance increases, as measured by the number of binary variables and quadratic interactions.

An analysis of the asymptotic number of binary variables in the examined QUBO formulations reveals significant differences between Lucas’s positional approach (i.e., node-based formulations) and edge-based formulations. Namely, for the formulations MTZ, DL, SD, GG and FCG, the number of binary variables scales asymptotically as $O(|E| \log |V|)$. In contrast, the formulation LUCAS, which uses positional encoding, requires $O(n^2)$ binary variables, regardless of the graph structure.

This means that for sparse graphs where $|E| = O(n)$, all formulations analysed other than Lucas lead to asymptotically smaller QUBO instances in terms of the number of binary variables. In practice, this could be an advantage in the context

of solving the problem using quantum annealing algorithms, as the size of the QUBO is one of the key factors affecting the difficulty of embedding the problem on the QPU architecture and the quality of the solutions obtained. However, it should be emphasised that in the computational experiments carried out, due to the small size of the graphs tested (which, let us recall, is a direct consequence of performing calculations for instances that are entirely embeddable on a QPU), the potential benefits of this property could not be observed.

IV. DISCUSSION

This paper analyses alternatives to the LUCAS formulation, which is currently the state-of-the-art representation of the Traveling Salesman Problem in the form of a QUBO model. Five other formulations are presented, describing in detail their transformation from Traveling Salesman Problem models to QUBO using the penalty model technique. Practical verification was carried out on selected small-scale instances, chosen so as to enable their full embedding on a quantum processing unit (QPU). The behaviour of both the simulated annealing (SA) algorithm, serving as a reference point, and quantum annealing (QA) were examined.

While in the case of simulated annealing the choice of problem formulation was of key importance, for quantum annealing the size of the problem proved to be the determining factor for efficiency. The number of binary variables—including slack variables, which are necessary to encode integer variables in binary form and transform inequality constraints into equality constraints—directly translates into the dimension of the QUBO matrix. The results achieved by both algorithms deviated from the optimal ones. This situation is due to several factors: the heuristic nature of both methods (no guarantee of finding the global optimum), the complexity of the energy landscape of optimisation problems and, in the case of QA, the limited computational budget.

As predicted, the effectiveness of optimisation (especially in the QA process) deteriorated as the size of the problem increased. This phenomenon is a consequence of the increasing difficulty of embedding on the QPU and the greater complexity of the energy landscape, which makes it difficult for the system to reach its minimum energy state. In the tested range, the SA algorithm proved to be significantly more effective than QA. The trends of a decrease in the number of solutions found with increasing problem complexity (number of variables and interactions) were similar for both algorithms, but the decrease was incomparably more rapid in the case of QA. The only exception was the formulation of FCG in the SA process, where a flattening of results was observed for the largest instances.

Improving the efficiency of quantum annealing may require methods that go beyond hardware parameter tuning, including work at the model level (e.g., using alternative integer encoding methods instead of the default ones available in the D-Wave API) and embedding optimisation.

We are aware of the limitations of this work, such as the focus on small instances (resulting directly from the methodological assumptions) and the lack of manual optimisation of the embedding process. Further research should

include the analysis of more diverse TSP instances, larger-scale calculations with a larger QPU budget, and tests of alternative methods of encoding variables and constraints.

Although the transformations themselves rely on standard penalty-based techniques, providing explicit QUBO representations of these formulations enables their direct comparison and benchmarking. To the best of our knowledge, this work provides the first explicit derivation of QUBO formulations for several classical ILP-based TSP models, including the SD, DL, GG, and FCG formulations.

From an architectural perspective, the results highlight that the choice of QUBO encoding is as important as the hardware architecture itself. Compact ILP-based formulations reduce the logical qubit count but introduce denser coupling structures, which may increase chain lengths in the minor embedding and thus the effective noise level on current QPUs. This trade-off is a key consideration for the design of digital quantum co-processors and hybrid quantum-classical systems for combinatorial optimization.

V. CONCLUSIONS

In conclusion, among the formulations of the Traveling Salesman Problem on a quantum processor that were tested, Lucas's model remains the most effective. Although its asymptotic spatial complexity is higher than that of the other five formulations, the limitations of current hardware have prevented empirical confirmation of the hypothetical advantage of models with lower theoretical spatial complexity. With the current state of technology, relying solely on native QPU calculations (excluding hybrid solvers), full verification of these advantages remains impossible.

ACKNOWLEDGMENT

We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within computational grants nos. PLG/2025/018189, PLG/2025/018523 and PLG/2025/018808.

During the preparation of this manuscript, the author used Gemini 3 Flash as well as multiple versions of ChatGPT to (1) assist with the translation of the manuscript, (2) correct errors in spelling and grammar, (3) improve the text flow, and (4) support the development of computational code. The author has reviewed and edited the output and takes full responsibility for the content of this publication.

The data supporting the findings of this study are available in the Zenodo repository at <https://doi.org/10.5281/zenodo.20528483> [29].

REFERENCES

- [1] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018. <https://doi.org/10.22331/q-2018-08-06-79>
- [2] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model," *Physical Review E*, vol. 58, no. 5, pp. 5355–5363, 1998. <https://doi.org/10.1103/physreve.58.5355>
- [3] D-Wave Quantum Inc., "D-Wave Quantum Inc.," 2026. [Online]. Available: <https://www.dwavequantum.com/>
- [4] T. Lanting, A.J. Przybysz, A. Yu. Smirnov *et al.*, "Entanglement in a quantum annealing processor," *Physical Review X*, vol. 4, no. 2, p. 021041, 2014. <https://doi.org/10.1103/PhysRevX.4.021041>
- [5] D-Wave Systems, "Ising-QUBO Transformations," 2026. [Online]. Available: https://docs.dwavequantum.com/en/latest/quantum_research/qubo_ising.html#qpu-qubo-ising-transformations
- [6] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," *arXiv preprint arXiv:1411.4028*, 2014. <https://doi.org/10.48550/arXiv.1411.4028>
- [7] M. Aramon, G. Rosenberg, E. Valiante *et al.*, "Physics-inspired optimization for quadratic unconstrained problems using a digital annealer," *Frontiers in Physics*, vol. 7, p. 48, 2019. <https://doi.org/10.3389/fphy.2019.00048>
- [8] M. M. Flood, "The traveling-salesman problem," *Operations Research*, vol. 4, no. 1, pp. 61–75, 1956. <https://doi.org/10.1287/opre.4.1.61>
- [9] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, p. 5, 2014. <https://doi.org/10.3389/fphy.2014.00005>
- [10] D-Wave Systems, "Reformulating a Problem," 2026. [Online]. Available: https://docs.dwavequantum.com/en/latest/quantum_research/reformulating.html
- [11] T. Öncan, I. K. Altınel, and G. Laporte, "A comparative analysis of several asymmetric traveling salesman problem formulations," *Computers & Operations Research*, vol. 36, no. 3, pp. 637–654, 2009. <https://doi.org/10.1016/j.cor.2007.11.008>
- [12] A. V. Orman and H. P. Williams, "A Survey of Different Integer Programming Formulations of the Travelling Salesman Problem," in *Optimization, Econometric and Financial Analysis*, E. J. Kontoghiorghe and C. Gatu, Eds. Berlin: Springer, 2007, pp. 91–104. https://doi.org/10.1007/3-540-36626-1_5
- [13] J. Nüßlein, T. Gabor, C. Linnhoff-Popien, S. Feld, "Algorithmic QUBO formulations for k -SAT and Hamiltonian cycles," in *Proc. GECCO '22 Companion*, 2022, pp. 2240–2246. <https://doi.org/10.1145/3520304.3533952>
- [14] D-Wave Documentation, "D-Wave Ocean SDK," 2026. [Online]. Available: <https://docs.ocean.dwavesys.com/en/stable/>
- [15] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM*, vol. 7, no. 4, pp. 326–329, 1960. <https://doi.org/10.1145/321043.321046>
- [16] M. Desrochers and G. Laporte, "Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints," *Operations Research Letters*, vol. 10, no. 1, pp. 27–36, 1991. [https://doi.org/10.1016/0167-6377\(91\)90083-2](https://doi.org/10.1016/0167-6377(91)90083-2)
- [17] H. D. Sherali and P. J. Driscoll, "On tightening the relaxations of Miller-Tucker-Zemlin formulations for asymmetric traveling salesman problems," *Operations Research*, vol. 50, no. 4, pp. 656–669, 2002. <https://doi.org/10.1287/opre.50.4.656.2865>
- [18] B. Gavish and S. C. Graves, "The traveling salesman problem and related problems," Working Paper GR-078-78, Operations Research Center, MIT, 1978.
- [19] G. Finke, A. Claus, and E. Gunn, "A two-commodity network flow approach to the traveling salesman problem," *Congresses Numeration*, vol. 41, pp. 167–178, 1984.
- [20] K. Tamura, T. Shirai, H. Katsura *et al.*, "Performance comparison of typical binary-integer encodings in an Ising machine," *IEEE Access*, vol. 9, pp. 81032–81039, 2021. <https://doi.org/10.1109/ACCESS.2021.3081685>
- [21] S. Zieliński, J. Nüßlein, J. Stein *et al.*, "Influence of different 3SAT-to-QUBO transformations on the solution quality of quantum annealing: A benchmark study," in *Proc. GECCO '23 Companion*, 2023, pp. 2263–2271. <https://doi.org/10.1145/3583133.3596330>
- [22] Ö. Salehi, A. Glos, and J. A. Mischczak, "Unconstrained binary models of the travelling salesman problem variants for quantum optimization," *Quantum Information Processing*, vol. 21, no. 2, p. 67, 2022. <https://doi.org/10.1007/s11128-021-03405-5>
- [23] C. Papalitsas, T. Andronikos, K. Giannakis, *et al.*, "A QUBO model for the traveling salesman problem with time windows," *Algorithms*, vol. 12, no. 11, p. 224, 2019. <https://doi.org/10.3390/a12110224>
- [24] H. Irie, G. Wongpaisarnsin, M. Terabe *et al.*, "Quantum annealing of vehicle routing problem with time, state and capacity," in *Quantum Technology and Optimization Problems*, S. Feld and C. Linnhoff-Popien, Eds. Cham: Springer International Publishing, 2019, pp. 145–156. https://doi.org/10.1007/978-3-030-14082-3_13

- [25] K. Nałęcz-Charkiewicz, A. Das, T. Chatterjee *et al.*, “Graph coarsening approach to the vehicle routing problem: An approximation strategy,” *IEEE Access*, vol. 13, pp. 22459–22472, 2025.
<https://doi.org/10.1109/ACCESS.2025.3534677>
- [26] R. Bellman, “Dynamic Programming Treatment of the Travelling Salesman Problem,” *Journal of the ACM*, vol. 9, no. 1, pp. 61–63, 1962.
<https://doi.org/10.1145/321105.321111>
- [27] M. Held and R. M. Karp, “A dynamic programming approach to sequencing problems,” *Journal of the Society for Industrial and Applied Mathematics*, vol. 10, no. 1, pp. 196–210, 1962.
<https://doi.org/10.1137/0110015>
- [28] python-tsp contributors, “python-tsp: A library for solving the Traveling Salesman Problem,” 2024. [Online]. Available: <https://github.com/fillipe-gsm/python-tsp>
- [29] K. Nałęcz-Charkiewicz, “Experimental artefacts for QUBO formulations of the Traveling Salesman Problem,” Zenodo, 2026.
<https://doi.org/10.5281/zenodo.20528482>
- [30] J. Burkardt, “TSP Data Sets,” 2024. [Online]. Available: <https://people.sc.fsu.edu/~jburkardt/datasets/tsp/tsp.html>