GPS: Improvement in the formulation of the TSP for its generalizations type QUBO

Saúl González Bermejo,¹ Guillermo Alonso-Linaje,¹ and Parfait Atchade-Adelomou^{2,3}

¹Universidad de Valladolid C/Plaza de Santa Cruz, 8, 47002 Valladolid (Spain) *

²Engineering Department Research Group on Data Science for the Digital Society La Salle -
Universitat Ramon Llull Carrer de Sant Joan de La Salle, 42 08022 Barcelona (Spain) †

³Lighthouse Disruptive Innovation Group, LLC 7 Broadway Terrace,
Apt 1 Cambridge MA 02139 Middlesex County, Massachusetts (USA) ‡

(Dated: Oct 2021)

We propose a new binary formulation of the Travelling Salesman Problem (TSP), with which we overcame the best formulation of the Vehicle Routing Problem (VRP) in terms of the minimum number of necessary variables. Furthermore, we present a detailed study of the constraints used and compare our model (GPS) with other frequent formulations (MTZ and native formulation). Finally, we have carried out a coherence and efficiency check of the proposed formulation by running it on a quantum annealing computer, D-Wave_2000Q6.

KeyWords: Quantum Computing, Quantum Annealing, Optimización combinatoria, QUBO, TSP, VRP

I. INTRODUCCIÓN

It is known that the best current QUBO formulation of the Traveling Salesman Problem (TSP) requires N^2 binary variables (IX A)). However, when this formulation is extended to more general problems such as Vehicle Routing Problem (VRP), its modelling is no longer efficient in terms of the number of variables because it requires the use of new variables to transform it into QUBO formulation. The number of the qubits, are crucial in today's quantum computing, so it is necessary to develop models focused on reducing the number of required variables.

The VRP encompasses two different problems: one in which the distance travelled by vehicles subject to capacity restrictions is minimized [1–4] and another, in which the time spent it takes vehicles to complete their routes is minimized. In this article, everything related to the VRP will optimize the time to complete these routes.

When modelling this problem, it is necessary to establish relationships between the distances travelled by the vehicles. For this reason, it is required that the distance calculation function between journeys has to be a linear function if we want to reduce the number of auxiliary variables used.

The best QUBO models of the TSP that can represent distance linearly use more than N^3 variables (native TSP formulation). There is indeed a formulation that uses $N^2 \log_2(N)$ variables (MTZ [5]), but its results are not as satisfactory when applying the *annealing* algorithms on it.

Our purpose in this work is to present a new QUBO model of the TSP whose the travelled distance's calculation is linear, and that uses only $3N^2$ variables, considerably improving the existing TSP models (both of N^3

and $N^2 \log_2(N)$ variables). Furthermore, this new formulation of the TSP will be generalized to define the most efficient formulation in terms of the number of variables of the VRP.

The document is organized as follows. Section (II) shows previous work on the TSP algorithm and its derivatives. Section (III) will explore adiabatic computation. In the section (IV), the general formulations of TSP and MTZ will be explored. Section (V) presents our TSP proposal with the improvements in numbers of qubits for quantum projects of this NISQ era and beyond. A generalization of our contribution is seen in Section (VI) where we propose an optimal VRP at the qubit level. In Section (VII), we present the results of the entire study carried out. In the section (VIII), we discuss the results and contrast the contributions. Finally, Section (IX) concludes the work carried out, and we open ourselves to some lines of the future of the proposed model.

II. RELATED WORK

In the mid-1920s, these two references, [6, 7] were the first articles to provide a solution to the minimal spanning tree (MST) problem. Based on these works, the mathematical researcher, Joseph B. Kruskal Jr, applied these solutions to the TSP [8], giving life to some of the first solutions to this problem that will arise during the next decades.

Almost at the end of the sixties, this work [9] offered a compilation and synthesis of the research on the travelling salesman problem. Its authors began by defining the problem and presenting several relevant theorems. They also classified and detailed the solution techniques and computational results. Before that, in the mid-1960s, the TSP started to emerge in many different contexts. This article [10] highlights some applications that began to gain space in everyday life, such as vehicle routing or job shop scheduling problems. Other applications such as planning, logistics and the manufacture of electronic

* saul.gonzalez.bermejo@alumnos.uva.es

† parfait.atchade@salle.url.edu

‡ parfait.atchade@lighthouse-dig.com

circuits became of particular interest.

By making a few small modifications to the original TSP, we could apply it in many fields such as SWP [11] and DNA sequencing [12, 13] among others. In this last application, the concept of ‘city’ would come to be fragments of DNA and the idea of ‘distance’, a measure of similarity between the pieces of DNA. In many applications, additional restrictions such as resource limits or time windows make the problem considerably difficult.

Computationally, the TSP [14] is an NP-Hard problem within combinatorial optimization. As an NP-Hard problem, it is computationally complex, and heuristics are continually being developed to get as close as possible to the optimal solution. However, considering the computational complexity nature of these problems, the new approach that quantum computing takes is very promising.

Many works are related to the standard/native TSP or some related variant in a quantum environment within this new approach. For example, in this work [15], the authors introduced a different quantum annealing scheme based on a path-integral Monte Carlo processes to address the symmetric version of the Travelling Salesman Problem (sTSP). In these other articles [16, 17], the authors did a comparative study using the D-Wave platform to evaluate and compare the efficiency of quantum annealing with classical methods for solving standard TSP.

In this reference, [18] several comparisons of heuristic techniques were made for some TSPLIB problems, both symmetric and asymmetric, and their results have been compared to other methods such as Self Organizing Maps and Simulated Annealing. In both cases, the local search technique was applied to the results found with Wang’s Recurrent Neural Network with “Winner Takes All”. The computational complexity of their heuristics method got $O(n^2 + n)$ [19], considered competitive when compared to Self Organizing Maps, which has a complexity of $O(n^2)$ [20]. The authors went on to show that the CAN technique has a computational complexity of $O(n^2 \log(n))$ [21], while the Simulated Annealing technique has a complexity of $O(n^4 \log(n))$ [22]. Results that are quite far from what we propose in this article.

One of the generalizations of the TSP, known as the VRP, was studied on the D-Wave platform [23, 24]. In tasks where routing and planning capacity (time) is required, the TSP with time windows (TSPTW) was generalized [25, 26], and has high inherent complexity and presents enormous resolution difficulties. In these references [11, 27, 28], the authors modelled combinatorial optimization problems in which social workers visit their patients at their respective homes and attend to them at a specific time, called Social Workers’ Problem (SWP). SWP is a significant problem because additional time constraints allow more realistic scenarios to be modelled than native TSP. The optimal or near-optimal solution for such issues is important in minimizing distance and time and environmental problems such as reducing fuel consumption.

The generalization of the TSP that we will use in work will be the VRP. However, there are other TSP derivatives, such as the Job Shop Scheduling Problem (JSSP) [29] that are not included in the study of this work.

During state of the art carried out, we have found several articles [23, 24, 30] that solve the TSP and VRP (focusing on minimizing distance and not time) for annealing computers. However, the number of variables is still intractable for the current size of quantum computers. For this reason, we propose a new TSP formulation with a representation of the linear distance that uses only $3N^2$ variables, which we will use to outperform the current best VRP modelling in terms of the number of required variables.

III. ADIABATIC QUANTUM COMPUTERS

Adiabatic computation was born from the use of the adiabatic theorem [31, 32] to perform the calculations. Using the tunnel effect to go from the global minimum of a simple Hamiltonian to the global minimum of the problem of interest. One of the market leaders for this type of computing is D-Wave, which roughly solves the quadratic unconstrained binary optimization problem (QUBO). The QUBO formulation (1) is suitable for running a D-Wave architecture; however, QUBO can be mapped to the Ising [33] model and thus be used in computers based on quantum gates, for example, IBMQ, Rigetti, Xanadu (strawberryfields), etc.

The problems that D-Wave quantum computers are prepared to solve are those that consist in finding the minimum of a function of the following form:

$$\sum_{i=1}^n b_i x_i + \sum_{i=1}^n \sum_{j=1}^n q_{i,j} x_i x_j, \quad (1)$$

where the variables $x_i \in \{0, 1\}$ and the coefficients $b_i, q_{i,j} \in \mathbb{R}$.

We have then that, given a problem, we need to model it with the above structure where the variables that form the solution will only take the values 0 or 1. Let us observe that, by taking the variables x_i the values 0 or 1, it is true that $x_i^2 = x_i$. Therefore, we can group the linear terms with the quadratic terms and express the above equation in matrix format:

$$x^t Q x, \quad (2)$$

with $x \in \{0, 1\}^n$ and $Q \in \mathfrak{M}_{n \times n}$ which is compactly representing the QUBO formulation.

IV. TSP FORMULATION

As discussed in the introduction, we will develop the QUBO model of the TSP in which we have worked and

which improves the number of variables required from the previous models whose distance calculation function is linear. To do this, first, we will present the models that we will improve by analyzing their strengths and weaknesses.

A. Native Formulation

In this section, we will recall the formulation of the native TSP [30] to analyze it in detail. This modelling, which has been defined in [34], despite appearing in a very natural way which facilitates its understanding, requires N^3 variables to be implemented. For today's quantum computers, due to qubit number limitations, we could hardly solve cases with several cities less than 15.

The variables that appear in this model are the variables $x_{i,j,t}$ such that $i, j \in \{0, \dots, N+1\}$ and $t \in \{0, \dots, N\}$. Let us consider that the variables $x_{i,i,t}$ do not exist in this model.

The interpretation of the variables $x_{i,j,t}$ is simple, since $x_{i,j,t} = 1$ if at instant t we traverse the edge that connects the cities i and j , and $x_{i,j,t} = 0$ for all other cases.

We can define the objective function of the native (Native in the sense of general, the most used) TSP[30] as:

$$\sum_{u=0}^{N+1} \sum_{v=0}^{N+1} \sum_{t=0}^N x_{u,v,t} d_{u,v}. \quad (3)$$

where $d_{u,v}$ represents the distance between nodes u and v . This objective function is subject to a series of restrictions:

- *Constraint 1.* The salesman must leave each city once.

$$\text{For each } u \in \{0, \dots, N\}: \sum_{v=0}^{N+1} \sum_{t=0}^N x_{u,v,t} = 1. \quad (4)$$

- *Constraint 2.* Each city must be reached once.

$$\text{For each } v \in \{1, \dots, N+1\}: \sum_{u=0}^N \sum_{t=0}^N x_{u,v,t} = 1. \quad (5)$$

- *Constraint 3.* If the salesman leave a city, he cannot return to it later. This constraint ensures that no unconnected cycles are formed as a solution. There are two ways of posing this constraint.

- Imposing that once he leaves a city he cannot return to it.

For each $u \in \{1, \dots, N+1\}$:

$$\sum_{v=0}^{N+1} \sum_{t=0}^N \sum_{w=0}^{N+1} \sum_{j=t+1}^N x_{u,v,t} x_{w,u,j} = 0. \quad (6)$$

- Imposing that once he arrives in a city, at the next moment, he must leave it.

For each $t \in \{0, \dots, N-1\}$, $u, v \in \{0, \dots, N\}$:

$$x_{u,v,t} \left(1 - \sum_{w=1}^{N+1} x_{v,w,t+1}\right) = 0. \quad (7)$$

This formulation requires N^3 variables. Next, we will analyze another model used to define the TSP, which, although it uses fewer variables, offers worse results when working with *annealing*.

B. MTZ formulation.

The idea of this formulation is to consider the variables $x_{i,j} = 1$ if the edge that connects the cities i and j appears in the solution path, where $x_{i,j} = 0$ for all other cases. Once we have these variables, we can establish order on the route employing a set of variables that will represent the moment the salesman arrives at that city (the variable u_i expressed in binary format, will take the integer value t if the city i is reached in the t -th position.). This model requires $N^2 \log_2(N)$, greatly improving the number of variables in the general formulation. However, when implemented using *annealing* it presents surprisingly bad results. This is because the *annealing* algorithm gets stuck trying to minimize the part of the objective function generated by the constraint of the equation (11), since the representation of integers in their binary format has the disadvantage that close numbers such as $2^n - 1$ and 2^n differ by a large number of qubits, so from the *annealing* they are perceived as very different solutions.

Next, let us define the variables needed for this model.

- $x_{i,j}$ such that $i, j \in \{0, \dots, N+1\}$. They will represent the edges that appear in the route.
- Let $H := \lfloor \log_2(N+1) \rfloor + 1$ and then define $u_i := \sum_{h=0}^H u_{i,h}$ such that $i \in \{0, \dots, N+1\}$. As discussed in the previous paragraph, these variables represent the order in which the cities are travelled.
- To be able to handle the previous variables, it is necessary to include the following variables: $slack_{i,j,h}$ such that $i, j \in \{0, \dots, N+1\}$ and $h \in \{0, \dots, H_1\}$ with $H_1 := \lfloor \log_2(N+1) \rfloor + 2$. These auxiliary variables will work as slack variables to transform inequalities into equalities.

Once the model variables have been defined, let us analyze the constraints that must be met.

Similar to the native TSP, we can define the MTZ objective function as follows:

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} x_{i,j} d_{i,j}, \quad (8)$$

where $d_{i,j}$ represents the distance between cities i and j . In this case, the restrictions to be satisfied are as follows:

- *Constraint 1*: The salesman must leave each city once.

$$\text{For each } i \in \{0, \dots, N\}: \sum_{j=1}^{N+1} x_{i,j} = 1. \quad (9)$$

- *Constraint 2*: Each city must be reached once.

$$\text{For each } j \in \{1, \dots, N+1\}: \sum_{i=1}^N x_{i,j} = 1. \quad (10)$$

- *Constraint 3*: There must be an order between the cities so that there is a single cycle that runs through all of them. Then, for each $i, j \in \{0, \dots, N+1\}$:

$$u_i - u_j + (N+2)x_{i,j} + \sum_{h=0}^{H_1} 2^h \text{slack}_{i,j,h} = N+1. \quad (11)$$

This restriction guarantees that if $x_{i,j} = 1$, what means, from city i we travel to city j , then the value of u_j must be greater than u_i . This ensures that the u_i variables provide a meaningful order.

Once the two most common QUBO models of the TSP have been presented, let us analyze the formulation with which we improve the number of variables of the previous two.

V. GPS FORMULATION

To develop this model, we take the variables $x_{i,j,r}$ with $i, j \in \{0, \dots, N+1\}$ and $r \in \{0, 1, 2\}$. In all the modelling, the variables $x_{i,j,r}$ such that $i = j$ are not considered. We work with directional edges, that is, if in the model the edge (i, j) appears, we will understand that first we go through node i and immediately after that we go to j . Let us analyze what each variable represents:

- $x_{i,j,0} = 1$ means that the edge (i, j) does not appear in the path and the node i is reached earlier than the j .
- $x_{i,j,1} = 1$ means that the edge (i, j) appears in the path, so the node i is reached earlier than the j .
- $x_{i,j,2} = 1$ means that the edge (i, j) does not appear in the path, and the node j is reached earlier than the i .

From the definition of our variables, we can define the distance travelled through the following objective function as:

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1}. \quad (12)$$

The constraints that must be met are:

- *Constraint 1*: For each i, j one and only one of the 3 cases of r must be given, so

$$\text{For all } i, j: \sum_{r=0}^2 x_{i,j,r} = 1. \quad (13)$$

- *Constraint 2*: Each node must be exited once.

$$\text{For each } i \in \{0, \dots, N\}: \sum_{j=0}^{N+1} x_{i,j,1} = 1. \quad (14)$$

- *Constraint 3*: Each node must be reached once.

$$\text{For each } j \in \{1, \dots, N+1\}: \sum_{i=0}^N x_{i,j,1} = 1. \quad (15)$$

- *Constraint 4*: If node i is reached before j , then node j is reached after i , so, for all $i, j \in \{0, \dots, N+1\}$ such that $i \neq j$:

$$x_{i,j,2} = 1 - x_{j,i,2}. \quad (16)$$

It would also have to be specified for $r = 0$ and $r = 1$, however this restriction is sufficient since by (13) it is implicit.

- *Constraint 5*: If node i is reached before node j and node j is reached before node k , then node i must be reached before k . This condition will prevent the route from returning to a node from which it had already exited, thus preventing cycles from forming. We then arrive at the penalty function equation (17).

$$\sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N (x_{j,i,2} x_{k,j,2} - x_{j,i,2} x_{k,i,2} - x_{k,j,2} x_{k,i,2} + x_{k,i,2}). \quad (17)$$

With only the cases in which $i \neq j$, $i \neq k$ and $j \neq k$ will be taken in the summation and in the annex (X) we will provide the approach followed to arrive at it.

The following is deduced from the equation (17). We have $x_{i,j,2} = 0$ if i is reached before j and $x_{i,j,2} = 1$ in the case where j is reached before i . Thus, with the previous equation we penalize these

following cases in which $x_{i,j,2} = 0, x_{j,k,2} = 0$ and $x_{i,k,2} = 1$ and $x_{i,j,2} = 1, x_{j,k,2} = 1$ and $x_{i,k,2} = 0$ which lead to cases in which it would be forming cycles (for these two situations we have that the value of the parentheses is 1 and for the rest 0). In (31) we have a similar situation for our VRP formulation, where we offer more details. For this condition, we must have directly constructed a penalty function that avoids erroneous cases without first posing linear conditions through which to generate its corresponding penalty function.

Formulated in this way we have managed to reduce the number of variables required from $N^2 \log_2 N$ to $3N^2$, achieving very noticeable reductions when working with large problems. Once we have this formulation, let us see how we can generalize it to the VRP.

VI. VRP FORMULATION

As discussed in the introduction, this model is oriented to be optimal concerning the number of qubits used. However, this generalization does not appear as naturally as expected because it requires a delicate step to get the constraints of the equation (31). To do this, we will detail each step and explain each constraint step by step.

A. Original formulation $5N^2Q$

For this VRP, we will consider that N is the number of cities and Q is the number of available vehicles. We first present the variables that will form the problem. We then take the following set of variables.

$$x_{i,j,r,q} \text{ with } i, j \in \{0, \dots, N+1\}, \quad (18)$$

$$r \in \{0, 1, 2, 3, 4\} \text{ and } q \in \{1, \dots, Q\}$$

In all the modelling, the variables $x_{i,j,r}$ such that $i = j$ are not considered. The variables i, j refer to the cities must travel to, and the variable q refers to the vehicle. The nodes 0 and $N+1$ correspond to the starting and ending points. Note that they may be the same node but we will separate them for convenience in the formulation. The values $d_{i,j}$ with $i, j \in \{0, \dots, N+1\}$ correspond to the distance between node i and j . Let us dive into the interpretation of each variable:

- $x_{i,j,0,q} = 1$ means that the vehicle q travels to the cities i and j , does not travel across the edge (i, j) and arrives at the city i before the j .
- $x_{i,j,1,q} = 1$ means that the vehicle q travels to the cities i and j travels across the edge (i, j) (that is, once it passes through the city i the next city it reaches is the j) and therefore the city i is reached earlier than the city j .

- $x_{i,j,2,q} = 1$ means that the vehicle q travels through the cities i and j and arrives at the city j earlier than at the i .
- $x_{i,j,3,q} = 1$ means that the vehicle q does not go through the cities i and j , and the city i is reached earlier than the city j . Note that $x_{i,j,3,q}$ can take the value 1 whether the vehicle q passes through one of both cities or neither of them.
- $x_{i,j,4,q} = 1$ means that the vehicle q does not travel to the cities i and j , and the city j is reached earlier than the city i .

Even if no vehicle passes through the objects i and j , the formulation must establish an order between them. However, this restriction does not make the modelling meaningless, since we can assume that if the vehicles are ordered in the order of $\{1, \dots, Q\}$, then i will be reached before j if the vehicle that passes through node i has a lower number than the one that passes through node j . Once the interpretation of each variable is explained, let us analyse the constraints that must be met.

- *Constraint 1:* For each i, j, q , one and only one of the possibilities must be met for r , so:

$$\text{For all } i, j, q: \sum_{r=0}^4 x_{i,j,r,q} = 1, \quad (19)$$

- *Constraint 2:* Each vehicle has to fulfill that it leaves the starting position. For this situation, we are going to impose that:

$$\text{For all } q: \sum_{j=1}^{N+1} x_{0,j,1,q} = 1, \quad (20)$$

No vehicle can return to the starting position from a city, so:

$$\text{For all } q: \sum_{i=0}^{N+1} x_{i,0,1,q} = 0, \quad (21)$$

- *Constraint 3:* Every vehicle must finish in the final position. For this, it must be fulfilled that:

$$\text{For all } q: \sum_{i=0}^N x_{i,N+1,1,q} = 1, \quad (22)$$

No vehicle can leave the final position. We then have that:

$$\text{For all } q: \sum_{j=0}^{N+1} x_{N+1,j,1,q} = 0. \quad (23)$$

Vehicles that do not travel on any road will meet all constraints when taking the following condition:

$$x_{0,N+1,1,q} = 1.$$

- *Constraint 4:* The vehicle must leave once and only once from each city, then:

$$\text{For each } i \in \{1, \dots, N\}: \sum_{q=1}^Q \sum_{j=1}^{N+1} x_{i,j,1,q} = 1. \quad (24)$$

- *Constraint 5:* The vehicle must arrive once and only once to each city, then:

$$\text{For each } j \in \{1, \dots, N\}: \sum_{q=1}^Q \sum_{i=0}^N x_{i,j,1,q} = 1. \quad (25)$$

- *Constraint 6:* The city i is reached before the city j does not depend on each vehicle. Therefore, for all the vehicles that either arrive at city i earlier than j , or arrive at city j earlier than i . Introducing the auxiliary variables $a_{i,j}$, we have the following constraint. For all $i, j \in \{1, \dots, N\}$:

$$\sum_{q=1}^Q x_{i,j,0,q} + x_{i,j,1,q} + x_{i,j,3,q} = a_{i,j}Q. \quad (26)$$

It will then be true that for each i, j or $a_{i,j} = 1$, which means that the city i is reached earlier than the city j and therefore for each q we will have $x_{i,j,r,q} = 1$ for any value of the r in which i is reached before j , or $a_{i,j} = 0$, and, we will have $x_{i,j,r,q} = 0$ for all the vehicles and for values r where i is reached before j .

- *Constraint 7:* If the vehicle q arrives in the city j , then the vehicle q must leave the city j . For this we impose the constraint that for $i \in \{0, \dots, N\}$, $j \in \{1, \dots, N\}$ and $q \in \{1, \dots, Q\}$:

$$x_{i,j,1,q} \left(1 - \sum_{k=1}^{N+1} x_{j,k,1,q}\right) = 0. \quad (27)$$

Let us now impose the conditions that make vehicles run on a tour.

- *Constraint 8:* It must be fulfilled that either the vehicle pass through the city i before the j or arrive before to the city j than the i . Therefore, it must be verified that, for $i \in \{0, \dots, N\}$, $j \in \{1, \dots, N\}$ and $q \in \{1, \dots, Q\}$:

$$x_{i,j,0,q} + x_{i,j,1,q} + x_{i,j,3,q} = 1 - (x_{j,i,0,q} + x_{j,i,1,q} + x_{j,i,3,q}). \quad (28)$$

- *Constraint 9:* If city i is reached before j and city j is reached before city k , then city i must be reached before city k . This condition will prevent the vehicle from returning to a city it has already passed through and therefore prevents a cycle from forming.

To introduce this constraint, we will directly calculate a penalty function worth 0 in the correct cases and 1 in those that are not. To facilitate the understanding of the penalty function, we are going to take, for i, j, k, q , the following variables:

$$\begin{aligned} a_{i,j} &= x_{i,j,0,1} + x_{i,j,1,1} + x_{i,j,3,1} \\ a_{j,k} &= x_{j,k,0,1} + x_{j,k,1,1} + x_{j,k,3,1} \\ a_{i,k} &= x_{i,k,0,1} + x_{i,k,1,1} + x_{i,k,3,1}. \end{aligned} \quad (29)$$

Remember that it is not necessary to introduce these conditions because the constraint (26) establishes the correct values of the variables $a_{i,j}$. Therefore, $a_{i,j} = 1$ means that the city i is reached before the city j and the same with j and k . Also, it is very important to remember that due to the same constraint (26), we can take any of the vehicles as a reference. In this case, we have taken the first vehicle as a reference.

In this way, fixed i, j, k , we have the 3 variables $a_{i,j}, a_{j,k}, a_{i,k}$. Remember that $a_{i,j}, a_{j,k}, a_{i,k}$ only take the values 0 or 1. Also, let us note that the cases that lead to values of the variables for which cycles can be formed and that we must discard are $(a_{i,j}, a_{j,k}, a_{i,k}) = (0, 0, 1)$ and $(a_{i,j}, a_{j,k}, a_{i,k}) = (1, 1, 0)$.

In the case $(0, 0, 1)$ we would have that the city j is reached after the i , the k after the j , and yet the city k is reached rather than i , which is absurd. The case $(1, 1, 0)$ cannot be given either, since it reaches i before j and j before k , so it cannot be that we also reach k before i . We therefore must construct a penalty function so that for $f(a_{i,j}, a_{j,k}, a_{i,k})$ it holds that $f(0, 0, 1) > 0$, $f(1, 1, 0) > 0$ y $f(a_{i,j}, a_{j,k}, a_{i,k}) = 0$ for all other cases. A function that satisfies these conditions is from the equation (30).

$$f(a_{i,j}, a_{j,k}, a_{i,k}) := a_{i,j}a_{j,k} - a_{i,j}a_{i,k} - a_{j,k}a_{i,k} + a_{i,k}^2. \quad (30)$$

then, adding to the cost function the equation (31)

$$\lambda \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N (a_{i,j}a_{j,k} - a_{i,j}a_{i,k} - a_{j,k}a_{i,k} + a_{i,k}^2), \quad (31)$$

we will have that the best solutions will be those that comply with this constraint.

- *Constraint 10:* The objective we seek is to minimize vehicle travel time. What we could do is to see how long each vehicle takes to complete the route and try to minimize as much of the time as possible. However, this function soon becomes complex

so we have decided to develop a different idea that simplifies the process and smoothes the objective function. If we impose the condition that all vehicles travel less distance than the distance travelled by vehicle number 1, we will have that minimizing the maximum of the distances will be equivalent to minimizing the distance travelled by the first vehicle. We then have the following condition. For each $q \in \{2, \dots, Q\}$:

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1,q} \leq \sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1,1}. \quad (32)$$

We transform this inequality into equality by taking once again $D_{max} := \sum_{i=0}^n \max_j \{d_{i,j}\}$ and the variables $b_{h,q}$ (the variables $b_{h,q}$ are like in (11) slack variables and they are in their binary expression) in:

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1,q} + \sum_{h=0}^{h_{max}} 2^h b_{h,q} - \sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1,1} = 0. \quad (33)$$

Under these conditions the function to be minimized corresponds to:

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1,1}. \quad (34)$$

This condition has the disadvantage that we are eliminating solutions where it is another vehicle that travels the longest distance. Let us explore how to avoid this problem and get more flexibility in the model to make it easier for the Quantum Annealing to find the optimum one. We can establish an auxiliary variable D and we set that the distance travelled by each vehicle must be less than this variable, that is to say:

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} d_{i,j} x_{i,j,1,q} \leq D, \text{ for all } q \in \{1, \dots, Q\}. \quad (35)$$

The variable D is an integer, so we must treat it in some way in order to include it in the model. As we explained in the introduction of the section dedicated to the formulation of the MTZ model (IV B), it is convenient to try to avoid the binary representation of integer variables. To do so, we can express D as a combination of the distances between edges by taking $D = \sum_{i=0}^{N+1} \sum_{j=0}^{N+1} x_{i,j} b_{i,j}$. Thus after imposing the constraint (35) we have that the function to minimize is D .

Thanks to this modelling of the VRP we have been able to reduce the number of variables required to the order of $5N^2Q$. However, we have managed to reduce it even further to $3N^2Q$, which is detailed in (IX B). However, we have preferred to present this other model due to its easy understanding.

VII. RESULTS

Before analyzing the results, we are pleased with our GPS formulation's positive results achieved.

Let us observe in the different tables the comparison of the number of qubits, time during which the D-Wave Quantum Annealing simulator has been executed, and the length of the path found. The sign "-" represents that the algorithm did not find a possible way during the elapsed time (in minutes). In this examples, the cities which form the TSP to solve are the vertex of the regular polygon with these number of vertex.

	GPS	General	MTZ
Number of qubits	75	100	140
Elapsed Time (min)	0.332	0.08	0.569
Path Length (m)	5.65	5.65	5.65

Table I. In this scenario of 4 cities, we set comparison with the 3 models, MTZ, native TSP and GPS. The comparison is based on the number of times to find the solution, the distance travelled, and the number of qubits. We can appreciate the good performance of our GPS model, and above all the savings it offers us in the number of qubits.

	GPS	General	MTZ
Number of qubits	147	294	266
Elapsed Time (min)	0.337	0.39	1.338
Path Length (m)	6.00	6.00	8.46

Table II. In this scenario of 6 cities, we set comparison with the 3 models, MTZ, native TSP and GPS. The comparison is based on the number of times to find the solution, the distance travelled, and the number of qubits. We can appreciate the good performance of our GPS model, and above all the savings it offers us in the number of qubits.

These results have been obtained using a simulator because we would require access to a quantum computer for a time similar to that needed to perform the simulations (in some cases more than an hour). However, it is the benefits of modeling with few qubits (such as GPS modelling) will be much more notable when these problems are implemented in real quantum computers. Other studies that did not require many hours of the quantum computer were carried out on the DW_2000Q_6. In the discussion section, we detail some interesting cases.

	GPS	General	MTZ
Number of qubits	243	648	522
Elapsed Time (min)	1.209	1.177	2.676
Path Length (m)	6.122	9.58	11.46

Table III. In this scenario of 8 cities, we set comparison with the 3 models, MTZ, native TSP and GPS. The comparison is based on the number of times to find the solution, the distance travelled, and the number of qubits. We can appreciate the good performance of our GPS model, and above all the savings it offers us in the number of qubits.

	GPS	General	MTZ
Number of qubits	363	1210	770
Elapsed Time (min)	3.316	3.087	4.175
Path Length (m)	12.51	10.978	-

Table IV. In this scenario of 10 cities, we set comparison with the 3 models, MTZ, native TSP and GPS. The comparison is based on the number of times to find the solution, the distance travelled, and the number of qubits. We can appreciate the good performance of our GPS model, and above all the savings it offers us in the number of qubits.

	GPS	General	MTZ
Number of qubits	507	2028	1066
Elapsed Time (min)	7.992	9.677	10.578
Path Length (m)	14.286	12.28	-

Table V. In this scenario of 12 cities, we set comparison with the 3 models, MTZ, native TSP and GPS. The comparison is based on the number of times to find the solution, the distance travelled, and the number of qubits.

VIII. DISCUSSIONS

Once the different models had been implemented, we achieved the following results. Through the results of the figure (1) up to the figure (4), the good performance of our formulation compared to the general TSP [30] can be observed. An almost identical operation is seen with the generic TSP, except that we are improving at least the number of qubits for the same cases in our proposal. Although the time difference is not significant again, the difference between path lengths is. Let us remember that the advantage of the formulation in which we have worked is based on improving the number of qubits used. We then have that the larger the problems we are working on, the better this difference will be appreciated in the

number of variables.

The MTZ model does not offer positive results. This is since *Annealing* presents many difficulties to find minimum expressions in which the representation of integers appears in their binary format. This is because although the numbers $2^k - 1$ and 2^k are close, they are not close in their binary form since they differ in k variables, so the *annealing* tends to present bad results. Apart from that adjusting, the Lagrange coefficient of this type of constraint is also a complicated task.

General and GPS modelling show better results. While it is true that general modelling gives slightly better results, it requires the use of a higher number of qubits. This may be since the function to be optimized for this model has a smaller number of local minima where the *Annealing* can get stuck or to a bad fit of the Lagrange coefficients.

The problem on which the simulations are carried out consists in finding the optimal path when the points are placed on the vertices of the regular polygons that have the same number of vertices as nodes in our problem.

Figure 1. Path length comparison for $N = 9$. In this graph, we see how the length of the solution paths for the case of 9 Cities is very similar so that both models give good results.

One of the behaviours and results that we believe is important to mention is the following. We realized that it is even more important to consider the number of edges that our model generates. The vertex/connections in a quantum computer are limited and define our quantum computer's typology and quality for error mitigation. Thus, a model that produces many edges (direct links) may request more from a computer than another generates few. The figure (5) offers us a comparative study between our GPS model and the native TSP. This figure shows the exponential behaviour and the number of interconnections that each model offers. Our model improves the number of qubits and gives us a great result reducing the number of connections a lot. The native TSP behaves as $0.8(N+2)^5$ while the GPS as $2(N+2)^3$.

Figure 2. Time comparison for $N = 9$. This graph shows the time taken to carry out the executions in the case of 9 cities. Although it seems that there is a lot of difference, it only represents 10% of the total time, which, as we have seen in other experiences, is not significant.

Figure 3. Path length comparison for $N = 11$. For the example of 11 cities, we can observe that the outcomes are quite similar. Although the time difference is not significant again, the difference between path lengths is. Let us remember that the advantage of the modelling we have worked is based on improving the number of qubits used. We then have that the larger the problems we are working on, the better that difference will be appreciated in the number of variables.

One aspect of GPS worth commenting on here is to generalize it also to be used for the Cutting-plane method. We must change the current constraint (17) since this methodology only works with linear constraints. The way to do this is as follows. For each i, j, k :

- $x_{j,i,2} + x_{k,j,2} \leq 2x_{k,i,2} + w_{i,j,k}^1$
- $x_{j,i,2} + x_{k,j,2} \geq 2x_{k,i,2} - w_{i,j,k}^2$

In these equations, the variables $w_{i,j,k}^p$ are auxiliaries. The purpose of these variables is to satisfy the said constraints. These two restrictions are satisfied by all cases of

Figure 4. Time comparison for $N = 11$. For the example of 11 cities, we can observe that the outcomes are quite similar because although there is a mean difference of about 20 seconds between the results of both simulations, the experience with this problem and other similar ones is that this very small difference does not affect the results on the length of the solution path.

Figure 5. In this figure we can appreciate the exponential behaviour and the number of interconnections that each model offers. Our model (GPS) improves the number of qubits and gives us a great result reducing the number of connections a lot. The Native_TSP behaves as $0.8(N+2)^5$ while the GPS as $2(N+2)^3$.

$(x_{j,i,2}, x_{k,j,2}, x_{k,i,2})$ except for $(0, 0, 1)$ (because it doesn't satisfy the second constraint) and $(1, 1, 0)$ (because it doesn't satisfy the first constraint).

IX. CONCLUSIONS AND FURTHER WORK

The modelling of both the TSP and the VRP developed in this work has given outstanding results to be used to solve both problems and the tasks that derive from them. We have improved the number of variables over other similar ones that provide good results in the condition of having a linear distance calculation function. The improvement in our models represents a fairly significant order of magnitude because we went from N^3 variables to $3N^2$. The figure (6) and (7) summarizes the major contribution of this article.

Figure 6. Comparison of the different models based on the number of qubits. This graph shows the behaviour and evolution of the numbers of qubits for each model. We see the best performance of our GPS model compared to the other models.

Figure 7. Benchmark between MTZ and GPS model based on the number of qubits. We can appreciate that for 30 cities, GPS model needs 2700 qubits while the MTZ 4458.

The results obtained from our VRP formulation and all the experiments carried out maintain the number of variables QN^2 and allow us to offer the community new formulations that minimize the time it takes for vehicles to travel.

Future work will apply the ideas developed in the QUBO model of these problems to similar ones. In particular, we will look for other variants of the TSP to use the modelling of this that we have carried out.

Compliance with Ethics Guidelines

Funding: This research received no external funding.

Institutional review: This article does not contain any studies with human or animal subjects.

Informed consent: Informed consent was obtained from all individual participants included in the study.

Data availability: Data sharing not applicable. No new data were created or analyzed in this study. Data sharing is not applicable to this article.

APPENDIX

A. TSP formulation N^2 .

There is a TSP modeling that requires N^2 variables, where these are the following:

$$x_{i,t} \text{ such as } i \in \{0, \dots, N+1\} \text{ and } t \in \{0, \dots, N+1\}. \quad (36)$$

Under this formulation $x_{i,t} = 1$ denotes that the city i is reached at position t . The distance calculation function with this formulation is as follows

$$\sum_{i=0}^{N+1} \sum_{j=0}^{N+1} \sum_{t=0}^N d_{i,j} x_{i,t} x_{j,t+1}, \quad (37)$$

where $d_{i,j}$ represents the distance between the node i and the node j . This expression has the problem that the distance formulation has terms of degree two and when trying to generalize this idea to other types of problems such as the VRP it will become a 4 degree constraint making use of a large number of auxiliary variables to convert it to QUBO type format.

B. Improved model $3N^2Q$

In the previous modelling, we can improve the number of variables used from $5N^2Q$ to $3N^2Q$ since certain variables are redundant. Let us see how we can do this. Let us take the set of variables

$$x_{i,j,r,q} \text{ with } i < j \in \{0, \dots, N+1\}, r \in \{0, 1, 2\} \text{ and } q \in \{1, \dots, Q\}$$

In all the modelling, the variables $x_{i,j,r}$ such that $i = j$ are not considered. Let us analyze the interpretation of each variable. For each edge (i, j) , different cases depending on whether a vehicle passes through both cities, which city is visited before the other and whether the edge is travelled or not.

- $x_{i,j,0,q} = 1$ means that the city i is reached earlier than the j and the edge (i, j) is not travelled.
- $x_{i,j,1,q} = 1$ means that the vehicle q travels the cities i and j , it reaches the city i before the j and it travels the edge (i, j) .
- $x_{i,j,2,q} = 1$ means that the city j is reached earlier than the i and the edge (j, i) is not travelled.

This new simplification keeps constraints (20), (22), (24), (25), (27) and (32) defined in the same way as the first proposal of the VRP formulation, so we will only focus on the changes of the remaining constraints:

- **Constraint 1:** For each i, j, q , one and only one of the possibilities must be met for r , so:

$$\text{For all } i, j, q: \sum_{r=0}^2 x_{i,j,r,q} = 1, \quad (38)$$

- *Constraint 6:* That the city i is reached before the city j does not depend on each vehicle. Therefore, for all the vehicles that either arrive at city i earlier than j , or arrive at city j earlier than i . Introducing the auxiliary variables $a_{i,j}$, we have the following constraint. For all $i, j \in \{1, \dots, N\}$:

$$\sum_{q=1}^Q x_{i,j,0,q} + x_{i,j,1,q} = a_{i,j}Q. \quad (39)$$

- *Constraint 8:* It must be fulfilled that either the vehicle pass through the city i before the j or arrive before to the city j than the i . Therefore, it must be verified that, for $i \in \{0, \dots, N\}$, $j \in \{1, \dots, N\}$ and $q \in \{1, \dots, Q\}$:

$$x_{i,j,0,q} + x_{i,j,1,q} = 1 - (x_{j,i,0,q} + x_{j,i,1,q}). \quad (40)$$

- *Constraint 9:* If the city i is reached before j and the city j is reached before the city k , then the city i must be reached before the city k . This condition will prevent the vehicle from returning to a city it has already passed through and therefore prevents a cycle from forming.

$$\lambda \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N (a_{i,j}a_{j,k} - a_{i,j}a_{i,k} - a_{j,k}a_{i,k} + a_{i,k}^2), \quad (41)$$

X. RESTRICTION PENALTY

Let us analyze the system that must be solved to build the penalty function from the equation (17). Our penalty function $P(a_{i,j}, a_{j,k}, a_{i,k})$ must satisfy that $P(0,0,1) = 1$, $P(1,1,0) = 1$ and $P(a_{i,j}, a_{j,k}, a_{i,k}) = 0$ for the rest of the cases. Let us call the variables $a_{i,j} = x$, $a_{j,k} = y$, $a_{i,k} = z$ to simplify the notation. Then, we arrive at the quadratic function P , as is shown as follows:

$$P(x, y, z) = c_1x^2 + c_2xy + c_3xz + c_4y^2 + c_5yz + c_6z^2$$

Imposing the previous restrictions, we have the following system of equations.

- $P(0,0,1) = 1$ So that $c_6 = 1$.
- $P(0,1,0) = 0$ So that $c_4 = 0$.
- $P(0,1,1) = 0$ So that $c_5 + c_6 = 1 \Rightarrow c_5 = -1$.
- $P(1,0,0) = 0$ So that $c_1 = 0$.
- $P(1,0,1) = 0$ So that $c_1 + c_3 + c_6 = 0 \Rightarrow c_3 = -1$
- $P(1,1,0) = 1$ So that $c_2 = 1$.

So far, we have a system of 6 equations with six certain compatible unknowns. First, however, an additional restriction must be verified. Let us verify if it is met.

$$- P(1,1,1) = 0. \quad \sum_{i=1}^6 c_i = 1 - 1 - 1 + 1 = 0.$$

So that indeed all the requirements are met.

We then have that the following function which is a penalty function for the constraint (17).

$$P(a_{i,j}, a_{j,k}, a_{i,k}) = a_{i,j}a_{j,k} - a_{i,j}a_{i,k} - a_{j,k}a_{i,k} + a_{i,k}^2$$

- [1] P. Toth and D. Vigo, *The vehicle routing problem* (SIAM, 2002).
- [2] P. Toth and D. Vigo, Models, relaxations and exact approaches for the capacitated vehicle routing problem, *Discrete Applied Mathematics* **123**, 487 (2002).
- [3] T. K. Ralphs, L. Kopman, W. R. Pulleyblank, and L. E. Trotter, On the capacitated vehicle routing problem, *Mathematical programming* **94**, 343 (2003).
- [4] P. Atchade-Adelomou, G. Alonso-Linaje, J. Albo-Canals, and D. Casado-Fauli, qrobot: A quantum computing approach in mobile robot order picking and batching problem solver optimization, *Algorithms* **14**, 10.3390/a14070194 (2021).
- [5] C. E. Miller, A. W. Tucker, and R. A. Zemlin, Integer programming formulation of traveling salesman problems, *Journal of the ACM (JACM)* **7**, 326 (1960).
- [6] O. Boruvka, On a minimal problem, *Práce Moravské Přírodovědecké Společnosti* **3**, 37 (1926).
- [7] O. Boruvka, Príspevek k řešení otázky ekonomické stavby elektrovodných sítí (contribution to the solution of a problem of economical construction of electrical networks), *Elektronický obzor* **15**, 153 (1926).
- [8] J. B. Kruskal, On the shortest spanning subtree of a graph and the traveling salesman problem, *Proceedings of the American Mathematical society* **7**, 48 (1956).
- [9] M. Bellmore and G. L. Nemhauser, The traveling salesman problem: A survey, *Operations Research* **16**, 538 (1968).
- [10] J. K. Lenstra and A. H. G. R. Kan, Some simple applications of the travelling salesman problem, *Journal of the Operational Research Society* **26**, 717 (1975).
- [11] A. P. Adelomou, E. G. Ribé, and X. V. Cardona, Formulation of the social workers' problem in quadratic unconstrained binary optimization form and solve it on a quantum computer, *Journal of Computer and Communications* **8**, 44 (2020).
- [12] J. Y. Lee, S.-Y. Shin, T. H. Park, and B.-T. Zhang, Solving traveling salesman problems with DNA molecules encoding numerical values, *Biosystems* **78**, 39 (2004).
- [13] P. Ball, DNA computer helps travelling salesman, *Nature* 10.1038/news000113-10 (2000).
- [14] G. Pataki, Teaching integer programming formulations using the traveling salesman problem, *SIAM review* **45**, 116 (2003).
- [15] R. Martoňák, G. E. Santoro, and E. Tosatti, Quantum annealing of the traveling-salesman problem, *Physical Review E* **70**, 057701 (2004).
- [16] R. H. Warren, Adapting the traveling salesman problem to an adiabatic quantum computer, *Quantum information processing* **12**, 1781 (2013).
- [17] R. H. Warren, Small traveling salesman problems, *Journal of Advances in Applied Mathematics* **2** (2017).
- [18] F. Greco, *Traveling salesman problem* (BoD-Books on Demand, 2008).
- [19] J. Wang, Primal and dual assignment networks, *IEEE Transactions on Neural Networks* **8**, 784 (1997).
- [20] K.-S. Leung, H.-D. Jin, and Z.-B. Xu, An expanding self-organizing neural network for the traveling salesman problem, *Neurocomputing* **62**, 267 (2004).
- [21] E. Cochran and J. E. Beasley, The co-adaptive neural network approach to the euclidean travelling salesman problem, *Neural Networks* **16**, 1499 (2003).
- [22] G.-y. Liu, Y. He, Y. Fang, and Y. Qiu, A novel adaptive search strategy of intensification and diversification in tabu search, in *International Conference on Neural Networks and Signal Processing, 2003. Proceedings of the 2003*, Vol. 1 (IEEE, 2003) pp. 428–431.
- [23] S. Feld, C. Roch, T. Gabor, C. Seidel, F. Neukart, I. Galter, W. Mauere, and C. Linnhoff-Popien, A hybrid solution method for the capacitated vehicle routing problem using a quantum annealer, *Frontiers in ICT* **6**, 13 (2019).
- [24] H. Irie, G. Wongpaisarnsin, M. Terabe, A. Miki, and S. Taguchi, Quantum annealing of vehicle routing problem with time, state and capacity, in *International Workshop on Quantum Technology and Optimization Problems* (Springer, 2019) pp. 145–156.
- [25] F. Focacci, A. Lodi, and M. Milano, A hybrid exact algorithm for the tsptw, *INFORMS journal on Computing* **14**, 403 (2002).
- [26] S. Edelkamp, M. Gath, T. Cazenave, and F. Teytaud, Algorithm and knowledge engineering for the tsptw problem, in *2013 IEEE Symposium on Computational Intelligence in Scheduling (CISched)* (IEEE, 2013) pp. 44–51.
- [27] P. Atchade-Adelomou, E. Golobardes-Ribé, and X. Vilasis-Cardona, Using the variational-quantum-eigensolver (VQE) to create an intelligent social workers schedule problem solver, in *Lecture Notes in Computer Science* (Springer International Publishing, 2020) pp. 245–260.
- [28] P. Atchade-Adelomou, E. Golobardes-Ribé, and X. Vilasis-Cardona, Using the parameterized quantum circuit combined with variational-quantum-eigensolver (vqe) to create an intelligent social workers' schedule problem solver (2020), arXiv:2010.05863 [quant-ph].
- [29] D. Applegate and W. Cook, A computational study of the job-shop scheduling problem, *ORSA Journal on computing* **3**, 149 (1991).
- [30] C. Papalitsas, T. Andronikos, K. Giannakis, G. Theocharopoulou, and S. Fanarioti, A qubo model for the traveling salesman problem with time windows, *Algorithms* **12**, 224 (2019).
- [31] M. Born and V. Fock, Beweis des adiabatsatzes, *Zeitschrift für Physik* **51**, 165 (1928).
- [32] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Quantum computation by adiabatic evolution, arXiv preprint quant-ph/0001106 (2000).
- [33] C. C. McGeoch, Adiabatic quantum computation and quantum annealing: Theory and practice, *Synthesis Lectures on Quantum Computing* **5**, 1 (2014).
- [34] A. Lucas, Ising formulations of many np problems, *Frontiers in physics* **2**, 5 (2014).