Quantum Annealing Solutions for the Closest String Problem with D-Wave Systems

Chandeepa Dissanayake

***Abstract*—The Closest String Problem is an NP-complete problem which appears more commonly in bioinformatics and coding theory. Less surprisingly, classical approaches have been pursued with two prominent algorithms being the genetic algorithm and simulated annealing. Latest improvements to quantum computing devices with a specialization in optimization tasks such as D-Wave systems, suggest that an attempt to embed the problem in a model accepted by such systems is worthwhile. In this work, two QUBO formulations have been proposed, with one being a slight modification over the other. Subsequently, an evaluation based on a few simple test cases had been carried out on both formulations. In this regard, the D-Wave annealers have been used, while providing guidelines for optimality on certain platform-specific concerns. For evaluation purposes, a metric termed Occurrence Ratio (OR) has been defined. With minimal hyperparameter tuning, the expected solutions were obtained for every test case and the optimality was guaranteed. To address practical and implementation issues, an inherent decomposition strategy based on the possibility of having substrings has been elucidated to accommodate the restricted qubit count. Conclusively, the need for further investigation on tuning the hyperparameters is emphasized.**

***Index Terms*—Combinatorial optimization, closest string problem, quadratic unconstrained binary optimization models, quantum algorithms, quantum annealing, D-Wave systems**

I. Introduction

FORMAL languages are an important concept in automata theory. Informally, such a formal language is defined as a set of strings constructed using the characters of a finite alphabet[1]. In this context, different scenarios emerge in which the objective is to determine a representative string that exhibits the highest degree of similarity to a given set of strings. The concept of similarity and the need to quantitatively measure it, gives rise to different objective functions and thus different distance measures for determining the difference between two strings. Note that different distance measures will yield different representative strings.

For any two strings of equal length, the Hamming distance is one of the most fundamental distance measures, which has its roots in coding theory[2]. Hamming distance is the number of positions at which the corresponding symbols are different for any two strings of equal length[3]. More formally, it can be defined with the Kronecker delta δ_{ab} as follows. Given two strings s_1, s_2 of length m over the alphabet Σ, the Hamming distance $d(s_1, s_2)$ is given by,

$$d(s_1, s_2) = \sum_{i=1}^{m} f(s_{1i}, s_{2i}) \tag{1}$$

where s_{xi} denotes the i^{th} character of the string s_x ($x = 1$ or $x = 2$ in this definition), and the function f is defined as,

$$f(c_1, c_2) = 1 - \delta_{c_1 c_2} = \begin{cases} 0, & \text{if } c_1 = c_2 \\ 1, & \text{if } c_1 \neq c_2 \end{cases} \tag{2}$$

Note that c_1, c_2 are two variables. In Equation (1), we set $c_1 = s_{1i}$ and $c_2 = s_{2i}$ for each i.

The problem of determining the representative string with the aid of Hamming distance as the distance measure is known as the Closest String Problem (CSP). Formally, elaborating on the definitions in [4], [5]: given a set of strings $S = \{s_1, s_2, s_3, ..., s_n\}$ where each string s_x is defined over the alphabet Σ and is of length m, the goal is to determine a string s_M which minimizes k such that for each string $s_x \in S$,

$$d(s_x, s_M) \leq k \tag{3}$$

CSP is proven to be NP-hard[4]. A special case of the decision problem of CSP called the hitting string problem has been proven to be NP-complete[6]. Therefore, the decision problem of CSP is NP-complete. Thus, exploring algorithms with a lower time complexity for CSP does not purely rely in the interests of applications of the problem. From a theorist's point of view, such investigations could lead to better insights about computationally hard problems and possibly allow further intuitions about unsolved problems such as P vs. NP question.

Coding theory is one of the major fields in which CSP has its applications, mostly in error correction[7], [8]. A reader familiar with the bioinformatics might notice that closest string has to be determined in designing genetic probes and drug target identification [4]. Algorithm presented in [9] for the identification of protein binding sites presents a direct application of finding the closest string representation. Additionally, interesting applications might be found for different problems in the context of Finite and Push-Down Automata in automata theory. [10] discusses a generalization of CSP, called the Closest Substring Problem in the context of regular languages.

Classically, this is one of the problems that had been studied extensively. One of the earliest approximation algorithms, called the Largest Distance Decreasing algorithm has been presented in [11], and determines solutions in polynomial time. The greedy heuristic algorithm in [12] chooses the character

The author is with the Department of Statistics & Computer Science, University of Kelaniya, Dalugama 11300, Sri Lanka (email: chandeepadissanayake@gmail.com)

based on the global evaluation of the Hamming distance and the choice at a given index affects the choice of the symbol for the next. Wave function collapse techniques have been studied in [13] as another heuristic based approach, using the idea of entropy in the original WaveFunctionCollapse constraint [14]. Additionally, a hybrid metaheuristic is discussed in [15]. For guaranteeing the optimality, a recursive exact method has been presented and has been compared against an integer programming formulation of the CSP[12]. Whenever the alphabet is of size 2, [16] presents an exact algorithm called Distance First Algorithm. For the other cases, a polynomial heuristic has been introduced and it is claimed that there is a possibility of obtaining a nearly optimal solution in a reasonable time. Furthermore, [17] presents a set of implementations of the existing algorithms and introduces a parameterized algorithm for the binary case of CSP.

Evidently, none of these approaches were successful in providing an efficient algorithm for the CSP in general. This could very well be due to the intrinsic difficulty in the problem itself, or due to the inherent limitations in the model of computation we rely upon. For example, despite the integer factorization problem not being proven to be NP-complete and exponential speedup hasn't been obtained, Shor's algorithm built on top of a unitary circuit model of quantum computation demonstrated superpolynomial advantage over every existing classical algorithm[18]. Accordingly, it is worthwhile to investigate alternative models of computation for problems that are either NP-complete or appear to be hard.

Adiabatic quantum computation is a quantum computational model that has seen interest in the recent years, particularly due to the development of devices that could leverage the process of quantum annealing for solving different problems, and especially optimization problems. Quantum annealing is involved in the minimization of an energy function through adiabatic evolution, and it requires the re-formulation of the existing problem definition as a Hamiltonian that defines the energy of a quantum system. The reformulation of the problem can be modeled either as an Ising problem or as a Quadratic Unconstrained Binary Optimization (QUBO) problem, with the latter simply being a transformation of the former. Retrospectively, a variety of NP-complete problems have been formulated in this way[19] and had been attempted using different annealing devices. In conclusion, it is worthwhile to investigate such a formulation for the CSP.

The prime outcome of this work is two QUBO formulations for the CSP. The formulated QUBO models have been evaluated on a limited number of simple test cases, where the D-Wave systems were used. D-Wave Systems is a vendor of quantum annealing devices and recently, they have allowed public access to their Quantum Processing Units (QPUs) on the cloud supplemented by an SDK. Subsequently the need for utilizing the number of available qubits and the techniques of achieving it will be discussed. Finally, the need for hyperparameter tuning has also been emphasized.

II. Preliminaries

A. Ising Model

An Ising model is an abstract mathematical model which usually has a large, but finite number of states. It has been used to describe different physical systems and the properties of them such as ferromagnetism in statistical mechanics. In fact, it is convenient to apprehend an Ising model as a lattice structure[1] in which there are lattice sites where the unit cells of the lattice can be located in either spin up or spin down state. In this context, it is important to view a lattice as a graph rearranged in 3-dimensions. Formally, the Ising model is defined as follows.

Consider a lattice structure (as described above) where the set of lattice sites are given by $\wedge$. Every lattice site $k \in \wedge$ has a set of adjacent lattice sites. For each lattice site $k \in \wedge$, there exists a discrete variable σ_k such that $\sigma_k \in \{-1, 1\}$, representing the spin. The spin configuration $\sigma = \{\sigma_k\}_{k \in \wedge}$ is an assignment of spin value to each site. For any 2 adjacent sites, there is an interaction J_{ij} where $i, j \in \wedge$. There is an external parameter denoted by h, which is typically an external magnetic field that interacts with the lattice site. Accordingly, for a given configuration, the energy is given by,

$$H(\sigma) = -\sum_{<ij>} J_{ij}\alpha_i\alpha_j - h\sum_j \sigma_j \tag{4}$$

The $(-)$ sign of the second term of Equation (4) is conventional. Furthermore, note that there are simplifications of this formulation, depending on the specific problems that are being modelled. For a detailed derivation and and explanation refer to [20].

1) Transverse Field Ising Model: The quantum mechanical description of the above Ising model is called the Transverse Field Ising model. In here, the interactions J_{ij} between lattice sites i and j are determined by the spin projections of the involved lattice sites spins along the z-axis. These interactions are also affected by the external magnetic field (which accounted for the external parameter h_j in the classical model), which acts perpendicular to the z-axis in this case, say along x-axis. This necessity of the perpendicularity in this setup, renders the spin projection along z-axis and the x-axis to be non-commuting observables. Consequently, the classical model cannot explain this setup, thus requiring the replacement of the spins with Pauli matrices associated with spin-$1/2$ observables. Accordingly, the energy of any spin configuration is given by the quantum Hamiltonian,

$$H^{'} = -J\left\{\left(\sum_{<i,j>} Z_iZ_j\right) + g\sum_j X_j\right\} \tag{5}$$

where, Z_i and X_i are Pauli matrices (as described already), and J is simply a prefactor, whereas g represents a coefficient which determines the relative strength of the external magnetic field applied when compared to the interactions between neighbouring sites.

[1]Referring to how the ions are arranged in the crystal structure of a metal that exhibits magnetism (Iron in this case)

In one of the foundational papers, quantum annealing has been proposed by introducing quantum fluctuations to the simulated annealing paradigm. It was tested by using the transverse field Ising model[21].

B. Quadratic Unconstrained Binary Optimization (QUBO)

Quadratic Unconstrained Binary Optimization problem is an NP-hard combinatorial optimization problem which attempts to determine a minimum value for a function defined on a binary vector space, along with the corresponding binary vector which results in the minimum value. Formally, given an upper triangular matrix $Q^{n\times n}$, the objective is to determine a binary vector $x^* \in \{0,1\}^n$ such that $argmin\, f(x) = x^*$ where,

$$f(x) = x^T Q x = \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij} x_i x_j \tag{6}$$

for any $x \in \{0,1\}^n$.

The QUBO problem displays a close resemblance with the Ising model formulation. In fact, any such Ising model formulated in terms of spins s_α can be transformed to a QUBO problem, replacing each spin by a binary variable, by applying the following transformation.

$$x_\alpha = \frac{s_\alpha + 1}{2} \tag{7}$$

Generally, for most of the problems it is more convenient to develop the model as a QUBO problem rather than an Ising formulation[19]. Furthermore, by introducing coefficients/penalties for each an every component in the formulation, the penalty Hamiltonian is obtained and it is used to embed the problem in the annealing devices. This is because such quantum annealing devices are capable of solving only unconstrained problems, and if there are constraints embedded into the Hamiltonian, the only possibility is to drastically increase the energy on each constraint violation[19].

C. D-Wave Quantum Annealers

D-Wave is a vendor which allows public access to a set of quantum annealers through a cloud called Leap. They provide a Software Development Kit (SDK) called Ocean SDK, along with a variety of other toolkits which suits different purposes from scientific research to commercial grade applications. In this work, D-Wave Leap cloud was utilized through the use of their Ocean SDK to test and evaluate the QUBO formulations.

In order to solve an optimization problem using the annealers in D-Wave systems, it is required to embed the problem in to the QPUs according to their topology as the first step. Once the problem is formulated as an Ising model or a QUBO problem, it is required to map it into the QPU. In the very early D-Wave QPUs, this mapping was achieved through the Chimera graph[22]. The process of embedding problem variables into the QPU, which is called minor-embedding, was to be manually accomplished. With the recent upgrades up to the Advantage QPUs, the primary topology was also upgraded to Pegasus graphs which has a slightly different internal arrangement of couplers[23]. Furthermore, the D-Wave Ocean SDK was supplemented with routines to implicitly perform minor embedding, thus eliminating the need for the user to be aware of the internal architecture of the QPU being accessed. An interested reader may refer to [24] for further explanation.

Both of the above topologies are incapable of mapping any given problem formulation as a one-to-one mapping from problem variables to physical qubits on the QPU. In such scenarios, several physical qubits are chained together to represent a single problem variable during the minor-embedding stage. Whenever the problem is solved in the QPU, these chained qubits are constrained to have the same value in every solution. In case this constraint is violated, the chain is said to be broken and the embedding does not represent the problem of interest anymore. During the annealing stage, the chains can be broken when attempting to minimize the energy function. “chain_strength" parameter is used to counter-act the tendency to break the chains[25]. A more detailed discussion and guidelines to set the value of this parameter can be found in [25].

The next step is to use a sampler to extract low energy states from the minor-embedded problem in the QPU. D-Wave provides different mechanisms for this based on the type of the device in which the problem is actually solved in, which they call the “solver". Since the interest of this work is to attempt the problem on real QPUs, “QPU Solvers" have been utilized. For other types of solvers, refer to [26]. A sampler implements the process of sampling based on the solver that we have opted to use.

QPUs are probabilistic by design. D-Wave annealers are of no exception for this. Hence, the most sensible approach of obtaining a valid solution is to run the given problem as arbitrarily many times as possible and utilize the statistical trends observed, to determine the solution. A single run is called a “read" or an “annealing cycle". Multiple such reads increases the diversity of the solution space and in turn allows to determine the probability of obtaining the respective solution, which would not have been possible otherwise. In this context, when submitting any problem to the “solver", D-Wave allows to specify a parameter called “num_reads" whose value denotes the number of annealing cycles [25].

III. Formulation of the CSP as a QUBO Problem

The problem is initially reformulated to a form which requires minimization of the sum of the Hamming distances between a candidate solution s_c (a string over the alphabet Σ) and each string (over the alphabet Σ) in the given set S. This is transformed again into another form in which it is required to determine each symbol s_{ci} of the candidate solution (the string s_c), such that the contribution to the sum of the Hamming distances by selecting the symbol s_{ci} is the minimum among the sum of Hamming distances by selecting other symbols from a finite alphabet. Based on the use of Hamming distance as the distance measure, it is inferred that for each symbol in the candidate solution, the symbol which results in the minimum sum of Hamming distance lies within the subset $\Sigma_i \subset \Sigma$ comprised only with the symbols of the strings in S, at the same position as s_{ci}. The QUBO formulation is

then presented incorporating two conflicting constraints, which are used to counter-act on the effect of the other. Ultimately, in order to eliminate the need for computing a piecewise function which involves an additional step while preparing the embedding of the problem when using the Kronecker-delta, another QUBO formulation is given, building on the fact that digital computers represent symbols in a numerical form. However, it is worthwhile to emphasize that this is not to gain a speedup in the optimization process, but to eliminate an additional step during the embedding.

A. Reformulation: Horizontal Minimzation of the Sum of Hamming Distances

Referring to the definition given for the CSP by the Equations (1), (2) and (3), it is established that k in Equation (3) should be minimized. Consider that, as given above, we have chosen a string s_c over alphabet Σ, as a candidate solution to the CSP. Let $D(s_c)$ be the sum of Hamming distances of the string s_c with the strings $s_x \in S$. Therefore, we can define $D(s_c)$ as follows.

$$D(s_c) = \sum_{x=1}^{n} d(s_c, s_x) = d(s_c, s_1) + d(s_c, s_2) + ... + d(s_c, s_n) \tag{8}$$

According to Equation (3), following extended constraint can be deduced. For s_M,

$$d(s_M, s_1) + d(s_M, s_2) + ... + d(s_M, s_n) \leq nk \tag{9}$$

That is,

$$D(s_M) \leq nk \tag{10}$$

Hence, the CSP can be restated as: Determine a string s_M that minimizes k in Equation (10). Here, in order to minimize k, it is required to minimize $D(s_c)$ and,

$$argmin_{s_c} \{D(s_c)\}_{s_c \in \Sigma^m} = s_M \tag{11}$$

where Σ^* is the language containing all the strings of length m over the alphabet Σ.

If the strings $s_x \in S$ are arranged in an $n \times m$ matrix, where each row correspond to a string, then the objective in the Equation (10) above can be portrayed as minimizing the Hamming distance between the candidate solution and the strings in the rows of the matrix - horizontally.

B. Reformulation: Vertical Minimization of the Sum of Hamming Distances

Let the string s_c and s_x be expressed as $s_c = s_{c1}s_{c2}...s_{cm}$ and $s_x = s_{x1}s_{x2}...s_{xm}$, where s_{ci} and s_{xi} are the symbols at the i^{th} position of the strings s_c, s_x respectively. By using the definition in the Equation (1), we can expand the RHS of the Equation (8) as follows.

$$\begin{aligned} D(s_c) = &\{f(s_{c1}, s_{11}) + f(s_{c2}, s_{12}) + ... + f(s_{cm}, s_{1m})\} \\ &+ \{f(s_{c1}, s_{21}) + f(s_{c2}, s_{22}) + ... + f(s_{cm}, s_{2m})\} \\ &+ ... \\ &+ \{f(s_{c1}, s_{n1}) + f(s_{c2}, s_{n2}) + ... + f(s_{cm}, s_{nm})\} \end{aligned}$$

Rearranging,

$$\begin{aligned} D(s_c) = &\{f(s_{c1}, s_{11}) + f(s_{c1}, s_{21}) + ... + f(s_{c1}, s_{n1})\} \\ &+ \{f(s_{c2}, s_{12}) + f(s_{c2}, s_{22}) + ... + f(s_{c2}, s_{n2})\} \\ &+ ... \\ &+ \{f(s_{cm}, s_{1m}) + f(s_{cm}, s_{2m}) + ... + f(s_{cm}, s_{nm})\} \end{aligned}$$

$$\therefore D(s_c) = \sum_{i=1}^{m} \sum_{x=1}^{n} f(s_{ci}, s_{xi}) \tag{12}$$

Since it is established that $D(s_c)$ is required to be minimized in Equation (10), we can conclude, by considering the Equation (12), that $\Delta_i(s_c)$ for each $i = 1, 2, ..., m$ must be minimized, where $\Delta_i(s_c)$ is defined as,

$$\Delta_i(s_c) = \sum_{x=1}^{n} f(s_{ci}, s_{xi}) \tag{13}$$

This is achievable because there are no terms that correlates the distance between each symbol position i, as suggested by the Equation (12). An important implication of this result is that it is possible to decompose CSP to m sub-problems that can be solved independently.

From the Equations (12) and (13), it is evident that the CSP can be restated as follows: For each symbol position $i = 1, 2, ..., m$, determine the symbol s_{ci} which minimizes $\Delta_i(s_c)$. i.e., for every i, $\Delta_i(s_M)$ is minimum.

Following the same matrix arrangement as in the previous section, this formulation can be portrayed as the iterative minimization of the Hamming distance between string generated by repeating the i^{th} symbol of the candidate solution n times and the string in the i^{th} column of the matrix, for each i - vertically.

C. Reduced Search Space for each Symbol Position

The candidate solution s_c is defined as a string over the same alphabet Σ, which is the alphabet for the strings in set S. However, according to the definition given in the Equation (2), if $s_{ci} \notin \Sigma_i$ then, for all $x = 1, 2, ..., n$, $f(s_{ci}, s_{xi}) = 1$, where $\Sigma_i = \{s_{xi} : s_{xi}$ is the symbol at the i^{th} position of the string $s_x \in S\}$ for each i. Therefore, the following can be stated. If $\omega \in \Sigma_i$, $\omega' \notin \Sigma_i$ and $s_{c'}$ is a string defined over Σ where $s_{c'i} \notin \Sigma_i$ then,

$$\sum_{x=1}^{n} f(\omega, s_{xi}) < \sum_{x=1}^{n} f(\omega', s_{xi}) \implies \Delta_i(s_c) < \Delta_i(s_{c'}) \tag{14}$$

It is guaranteed by the Equation (14) that the symbol at i^{th} position of the string s_M is one of the symbols at the i^{th} position of any of the strings $s_x \in S$. Accordingly, the most important result here is that we can reduce the search space of the s_{ci} to be the set Σ_i, instead of Σ. Note that c' in $s_{c'}$ is used to emphasize that the entire string may not be different from s_c, but just a symbol (the use of ' in the subscript).

Therefore, for a given s_{ci} if $\Delta_i(s_c)$ for a given symbol position i is minimum then $s_{Mi} = s_{ci}$ is probable. Assuming the minimality of $\Delta_i(s_c)$, if $\nexists \omega \in \Sigma_i$ such that $s_{ci} \neq \omega$ and

$\Delta_i(s_c) = \Delta_i(\omega)$, then $s_{Mi} = s_{ci}$ is guaranteed. Conclusively, we can state the following. For each $i = 1, 2, ..., m$,

$$argmin_x \left\{ \sum_{j=1}^{n} f(s_{xi}, s_{ji}) \right\}_{x=1}^{n} = c_{Mi} \tag{15}$$

D. QUBO Formulation of the CSP

According to the result given by the Equation (14), it is guaranteed that the symbol corresponding to the minimum distance for each symbol position i exists in Σ_i. Therefore, per each symbol position, allocating n binary variables (to account for each string), the following Hamiltonian results.

$$H_B = B \sum_{i=1}^{m} \sum_{x=1}^{n} \alpha_{xi} \sum_{y=1}^{n} f(s_{xi}, s_{yi}) \tag{16}$$

Here, the binary variable α_{xi} denotes whether the symbol s_{xi} is chosen as the symbol s_{ci} in the candidate solution s_c. B is the Lagrange multiplier[2]. The Equation (16) represents the objective term.

The ground state of the Hamiltonian H_B alone corresponds to the state where for all $x = 1, 2, ..., n$ and $i = 1, 2, ..., m$, $\alpha_{xi} = 0$. Not all variables should take the value 0. Optimally and ideally, for every x, only one variable in the set $\{\alpha_{x1}, \alpha_{x2}, ..., \alpha_{xm}\}$ is allowed to take the value 1, as only one symbol is to be chosen for s_{Mi}. Evidently, these two constraints conflict with each other. Nevertheless, by introducing penalty terms to the Hamiltonian in the Equation (16), the lowest energy can still be attributed to the the state which satisfies both of the constraints. Accordingly, following Hamiltonian represents both of the penalty terms.

$$H_A = A \sum_{i=1}^{m} \sum_{x=1}^{n} \{1 - \alpha_{xi}\} + A \sum_{i=1}^{m} \sum_{x=1}^{n} \left\{ \alpha_{xi} \sum_{y=x+1}^{n} \alpha_{yi} \right\} \tag{17}$$

Here, the first penalty term accounts for the constraint that disallows all variables from taking the value 0. In fact, it adds up a penalty for every variable that is set to 0. Including quadratic terms progressively for each pair of variables, per each symbol position, the second penalty term increases the penalty assigned for configurations in which there are more than one variable with value 1, for each symbol position. A is the Lagrange multiplier associated with the Hamiltonian H_A. Notably, the same Lagrange multiplier was used for both of the penalty terms, signifying that both of the constraints are of equal importance in determining s_M.

By combining the Hamiltonians given in the Equations (17) and (16), it can be concluded that the following Hamiltonian H describes the QUBO formulation for CSP.

$$H = H_A + H_B \tag{18}$$

E. An Alternative QUBO Formulation

Even though the optimization strategy in quantum annealing is not constructed upon a gradient-based optimization algorithm, it is generally preferred to avoid conditional statements in the problem formulation of an optimization problem as they result in discontinuous functions[27]. Additionally, despite the fact that problem embedding procedure is not performance critical and being done by using a modern programming language with a compiler that is optimized to handle logical operations as efficiently as the arithmetic operations (or vice-versa depending on the exact architecture being used), one might prefer to formulate the problem in the form of a continuous function. Based on the internal numerical representation of symbols in digital computers[3], this is achievable. The formulated function will not be continuous by its definition, but will be entirely based on arithmetic operations. Let $C : \Sigma \to \mathbb{R}$ be a bijection and $C(s_{xi})$ denote the value of the mapping for the symbol at i^{th} position of the string s_x. Using this mapping C, the following Hamiltonian $H_B^{'}$ can be used as an alternative objective function instead of the Hamiltonian given by the Equation (16).

$$H_B^{'} = B \sum_{i=1}^{m} \sum_{x=1}^{n} \alpha_{xi} \sum_{y=1}^{n} \frac{(C(s_{xi}) - C(s_{yi}))^2}{(C(s_{xi}) - C(s_{yi}))^2 + 1} \tag{19}$$

Notice that in this case, each value is scaled to the range $[0, 1)$. However, this is to facilitate tuning of the Lagrange multipliers whereas in most of the classical optimization algorithms it is to trigger faster convergence and to avoid biases[27].

Similar to the Equation (18), combining Hamiltonians given in the Equations (17) and (19), we arrive at the following alternative formulation for CSP, where $H^{'}$ gives the Hamiltonian.

$$H^{'} = H_A + H_B^{'} \tag{20}$$

F. Choosing Values for the Lagrange Parameters

In both of the Hamiltonians given by the Equations (18) and (20), it is required to determine values for the Lagrange multipliers A and B. Evidently, they could be plugged in with arbitrary values based on the requirements of the problem of interest. It depends on whether or not it is feasible to violate constraints at the cost of further minimizing the objective. The following guidelines are followed in this regard.

It is advisable to set $B = 1$. The constraints specified above must never be violated. Therefore, it must be assured that $max\{H_B\} < min\{H_A\}$. Since $max\{H_B\} = Bmn(n-1)$, we need to ensure that $min\{H_A\} > mn(n-1)$. In order to determine $min\{H_A\}$, its two terms can be considered independently by assigning either $\alpha_{xi} = 0$ or $\alpha_{xi} = 1$ $\forall x, i$, to see that

$$min\{H_A\} = \begin{cases} Am, & \text{if } n = 2 \\ \frac{Am(n-1)(n-2)}{2}, & \text{if } n = 3 \text{ or } n = 4 \\ Amn, & \text{if } n > 4 \end{cases} \tag{21}$$

The proof of the Equation (21) is given in the Appendix. Considering the distribution of the symbols in each position from 1 to m, $\exists A \in \mathbb{R}$ such that

$$B < A \leq \left\lceil \frac{Bmn(n-1)}{\lambda} \right\rceil$$

[2]This is identified as a Lagrange parameter in D-Wave Systems[24]

[3]By using encoding standards such as ASCII

where,

$$\lambda = \frac{min\{H_A\}}{A} = \begin{cases} m, & \text{if } n = 2 \\ \frac{m(n-1)(n-2)}{2}, & \text{if } n = 3 \text{ or } n = 4 \\ mn, & \text{if } n > 4 \end{cases}$$

which guarantees the optimal solution. Hence, for the purpose of analysis, a range of values from $\left(B, \left\lceil \frac{Bmn(n-1)}{\lambda} \right\rceil \right]$ can be attempted for A. If, by inspection, it appears that all the strings are mostly similar, then it is recommended to start with a value closer to the lower bound of possible values for A. i.e., closer to B. Otherwise, start with a value closer to the upper bound of A.

When using the Hamiltonian $H^{'}$, it is pertinent to observe that the values for these Lagrange parameters could be slightly different, typically with a minimal impact on the results. However, this minor difference assumes significance when the obtained solutions of H are slightly different from the solutions of $H^{'}$.

G. Important Concerns in the D-Wave Systems

Each quadratic term in either of the QUBO formulations given in Equation (18) and Equation (20) translates to a connection between a pair of qubits in the working graph. In the minor embedding, these two qubits corresponds to the two binary variables in the quadratic term concerned. For every $i = 1, 2, ..., m$, the interactions between all pairs of terms in the second constraint(term) of H_A can be represented by an undirected complete graph K_n. According to [23], allowing longer chains, it is possible to derive minor-embeddings for up to $n = 12M - 10$, where M is the working graph size in the Pegasus topology. Advantage4.1, which was the default, publicly available and the preferred QPU choice at the time of experimentation, has a graph size of 16 [28]. i.e., $M = 16$. Therefore, the the upper bound is $n = 182$. If each one of the m sub-problems is to be solved independently, Advantage4.1 QPU with P16 (Pegasus working graph with 16x16 unit cells) graph will be able to solve CSP instances with a maximum number of strings of 182. For any other instance, if m_{max} is the maximum number of sub-problems that is minor-embedded at once, then the maximum number of strings is $\lfloor 182/m_{max} \rfloor$. Subsequently it can be concluded that if the number of strings in a given CSP exceeds 182, it cannot be solved with a QPU leveraging the P16 graph.

As briefly explained in the preliminaries, given a problem graph Q that is obtained by representing the variables in the QUBO formulation as vertices and interactions between the variables given by the quadratic terms as edges, typically it is required to embed Q in the working graph F as a minor. However, there are instances where there exists a one-to-one mapping of vertices f from graph Q to graph F where each edge of Q is an edge of F. If such a mapping f exists, then Q is a subgraph of F and a subgraph embedding is possible[29]. In such cases, the optimum embedding does not contain chains. K_4 is a subgraph of the working graph of the P16[23]. Accordingly, all problem graphs $Q = K_q$ such that $1 \leq q \leq 4$ has a subgraph embedding in P, where P denotes the working graph of P16. Consequently, all instances of CSP such that $n \leq 4$ has a subgraph embedding. Therefore, whenever the number of strings (n) is less than or equal to 4 for the given problem instance, "chain_strength" parameter can be set to 0. However, for other cases a non-zero value is required, which can be assigned by considering the maximum possible value for the QUBO.

TABLE I
THE EXPECTED CLOSEST STRING FOR EACH SET OF STRINGS

Set	Set of Strings	Expected Closest String
#1	{"aaa","aaa","ddd"}	"aaa"
#2	{"aaa","aaa","ddd","ddd","ddd"}	"ddd"
#3	{"aaa","aaa","ded","ded","ded","ddd"}	"ded"
#4	{"abcdef","ghijkl","abcghi","xyzjkl", "abcmno"}	"abcjkl"

IV. Experimental Evaluation using D-Wave Systems

D-Wave systems' Leap cloud has been utilized as the platform and the Python SDK was used to build a script that would embed and solve the problem[4]. Refer to the preliminaries for further explanation on the D-Wave annealers. It is worthwhile to emphasize that the purpose of this evaluation is simply to establish empirical evidence for the validity of QUBO formulations. This would not serve as a rigorous or exhaustive testing attempt across many possible test cases.

Both QUBOs were tested for 4 sets of strings. The first 3 sets were handcrafted and the last one was created by ChatGPT[30]. Refer to the Table I for the sets of strings. Each set of strings was evaluated separately. For each set, "num_reads" was set to 100. Furthermore, when implementing Hamiltonian $H^{'}$ in the Equation (7), ASCII representation was used as the bijection C.

A. Determining the Chain Strength

Since Set #1 has only 3 strings, it allowed for the "chain_strength" parameter value to be set to 0. However for the other sets, it has to be assigned to a non-zero value as the number of strings in the set exceeded 4. The exact value assigned was dependent on the number of strings in the respective set and the distribution of the symbols in each set. By considering the influence of these two factors, we can discern four distinct cases which would determine the value for the "chain_strength", which is enumerated in ascending order as follows.

1) Less number of strings with a low variance in the distribution of symbols in the strings
2) Less number of strings with a high variance in the distribution of symbols in the strings
3) Higher number of strings with a low variance in the distribution of symbols in the strings
4) Higher number of strings with a high variance in the distribution of symbols in the strings

It can be observed that out of the two factors, the number of strings takes precedence over the variance in the distribution

[4]Codebase: https://github.com/chandeepadissanayake/qubo_csp

of symbols within the strings. Increasing the number of strings directly leads to an increased number of chains in the minor-embedding. While it may not be immediately evident that a high variance in symbol distribution directly causes chain breaks, it indirectly leads to a greater contribution to the overall energy within the Hamiltonian, thus diminishing the significance of the terms representing the chains. The comparison w.r.t. the number of strings and the distribution of symbols implied within the cases outlined above are relative to a given baseline. One such evident baseline, which is used in this work, is "chain_strength" $= 0$ for $n \leq 4$ as elaborated above. Each and every case above refers to $n > 4$. For the case 1, the general heuristic is to start off with a slightly higher value than the baseline. Accordingly, a preferred starting point for case 1 is "chain_strength" $= 1$. For the subsequent cases 2, 3, 4; values with increasing magnitude should be used. Accordingly, for the string set #1, baseline value is used and in the sets #2, #3 and #4, which can be considered as problem instances pertaining to the cases 2, 3 and 4 respectively, a range of values for "chain_strength" is attempted and the value that produced the optimum results is reported. It is worth noting that, despite the D-Wave documentation specifying that the "chain_strength" value is considered a hyperparameter[31], it is beneficial to establish a set of guidelines to arbitrarily limit the range of possible values.

B. Occurrence Ratio of a Solution

Due to the intrinsic probabilistic nature of the QPUs in D-Wave systems, the optimal solution may not be returned in every annealing cycle. Based on the QUBO formulations in the Equations (18) and (20), it is evident that multiple configurations of output can present the same solution in many cases. Therefore, in order to determine the significance of a solution among the other solutions in the solution space, a metric is desired. Note that the different configurations of output which present the same output string as the solution must be treated as the respective output string occurring multiple times. For a given solution/output string P, the Occurrence Ratio OR_P is defined as follows.

$$OR_P = \frac{N_P}{\sum_Q N_Q} \tag{22}$$

where, N_P represents the number of occurrences of P in the solution space. It should be evident that $\sum_Q N_Q =$ num_reads.

A higher occurrence ratio implies a higher likelihood of the solution being obtained from a D-Wave annealer. Maximum Occurrence Ratio (MOR) is the maximum of all possible OR_P values, i.e., the maximum OR_P that happens for any solution. MOR will aid in comparing the likelihood of the most occurring solution in the solution space against the likelihood of a given solution P.

C. Results

Refer to the Table II and Table III for the results when the QUBO formulations given by the Equations (18) and (20) are used. Column "P" refers to the minimum energy solution obtained for each set of strings. The columns "A", and "B" refer to the corresponding values for the respective Lagrange parameters whereas column "γ" refers to the value for "chain_strength" used. "OR_P" and "MOR" columns hold the same definitions as defined in the preceding section. The minimum energy solution P has always been reported disallowing any chain breaks by setting the appropriate value for "chain_strength" γ while repeating "num_reads" $(= 100)$, over several attempts to determine the maximum OR_P value possible for P.

TABLE II
RESULTS USING HAMILTONIAN H ON D-WAVE LEAP CLOUD QPUS

Set #	P	A	B	γ	OR_P	MOR
#1	"aaa"	2	1	0	1.00	1.00
#2	"ddd"	3	1	1	0.99	0.99
#3	"ded"	5	1	6	0.53	0.53
#4	"abcjkl"	4	1	5	0.22	0.22

TABLE III
RESULTS USING HAMILTONIAN $H^{'}$ ON D-WAVE LEAP CLOUD QPUS

Set #	P	A	B	γ	OR_P	MOR
#1	"aaa"	2	1	0	1.00	1.00
#2	"ddd"	3	1	1	0.97	0.97
#3	"ded"	5	1	6	0.51	0.51
#4	"abcjkl"	4	1	5	0.22	0.22

Since sets #1 and #2 contained similar strings within them, values closer to the lower bound of A were used. Even though the set #3 contained similar strings, a larger number of strings suggested the possibility of a constraint violation if a lesser value is used for A, which was confirmed by running it with $A = 2$. The outcome of this case is not reported as it was not optimal. Instead, with slightly a higher value, i.e., by setting $A = 5$ the expected solution was obtained. Set #4, contained similar, yet slightly different strings with a comparatively concentrated distribution of symbols, prompting the test case to be done with slightly a lesser value for A. Conclusively, in all cases, the expected closest string has been obtained and the MOR is minimal for every solution. It is worthwhile to note that given the random nature of the QPUs, these values for OR/MOR may not be exactly reproducible, but proximal outcomes should be expected.

It can be observed that the OR_P values for sets #3 and #4 are comparatively lower than what was obtained for sets #1 and #2. This can mostly be attributed to the largely narrow energy landscape created by the narrow distribution of the symbols in the sets #1 and #2. The requirement of a higher "chain_strength" in the cases #3 and #4 has augmented this effect by dispersing out a narrow distribution, possibly into a one with many local minima. It is possible that tuning of the Lagrange multipliers and the "chain_strength" could have alleviated this effect to some extent, but it was not a concern in this work. It should also be emphasized that for a noisy optimization process, it is cumbersome to point out the exact cause.

V. Discussion

One of the most interesting aspects of both of the QUBO formulations is that they have resulted in formulations where there is no interaction between qubits representing symbols at different positions across the strings. Therefore, the problem can be decomposed to sub-problems at the level of individual symbols. Thus, such sub-problems can be independently solved on the QPU. For a larger number of strings, this approach could be followed, decomposing the strings to substrings of equal size, which can be directly embedded in the QPU at once.

Consequently, the number of symbols in the strings does not affect the possibility of being solvable on a given QPU architecture. However, the number of strings remains to be a limiting factor. In fact, as calculated previously, an instance of CSP with a maximum of 182 strings can be solved with P16 QPU architecture in D-Wave systems. Theoretically, the number of qubits required under both QUBO formulations is mn. However, in the P16 architecture minor-embedding can lead up to chains with the length 16/17 for a single node in the problem graph and hence the exact number of qubits required is strictly dependent on the architecture of the QPU.

VI. Conclusions and Recommendations

Both of the derived QUBO formulations can be used for solving the CSP on a quantum annealer. For CSP, the given constraints should not be violated and hence during hyperparameter tuning, the constraints should be strictly enforced. Guidelines for tuning the Lagrange parameters and the "chain_strength" have been provided and one may significantly improve the results by following these guidelines. The constraints specified above could be relaxed for different variants of the problem. Based on algorithms such as the greedy heuristic algorithm in [12], it might be possible to investigate different QUBO formulations for the CSP, possibly relying on the global evaluation of the Hamming distance. An interesting experimental direction for future work based on these formulations is to evaluate the behavior of the minimum energy and the corresponding solutions across the domain of the values permitted for the Lagrange parameters, which may provide insights on hyperparameter tuning for sets of strings in different scales.

VII. Acknowledgments

The author would like to express his heartfelt gratitude to Dr. Chinthanie Weerakoon, a Senior Lecturer at the Department of Statistics and Computer Science, University of Kelaniya, for her unwavering support and guidance in the publication of this paper. Furthermore, he wants to extend his thanks to Dr. Anuradha Mahasinghe, a Senior Lecturer at the Department of Mathematics, University of Colombo, for mentoring him and sparking his interest in the adiabatic model of quantum computation. He also wishes to thank Dr. Sachintha Pitigala, a Senior Lecturer at the Department of Statistics and Computer Science, University of Kelaniya, for his support to the author in the academic arena.

References

[1] M. Sipser, *Introduction to the theory of computation*, 2nd ed. Boston: Thomson Course Technology, 2006, 431 pp., ISBN: 978-0-534-95097-2.

[2] R. W. Hamming, "Error detecting and error correcting codes," *The Bell system technical journal*, vol. 29, no. 2, pp. 147–160, 1950, Publisher: Nokia Bell Labs.

[3] W. N. Waggener, *Pulse code modulation techniques: with applications in communications and data recording*. New York: Van Nostrand Reinhold, 1995, 368 pp., ISBN: 978-0-442-01436-0.

[4] J. K. Lanctot, M. Li, B. Ma, S. Wang, and L. Zhang, "Distinguishing string selection problems," *Information and Computation*, vol. 185, no. 1, pp. 41–55, 2003, Publisher: Elsevier.

[5] M. Li, B. Ma, and L. Wang, *On the closest string and substring problems*, Feb. 17, 2000. arXiv: cs/0002012. [Online]. Available: http://arxiv.org/abs/cs/0002012 (visited on 08/20/2023).

[6] R. Fagin, "Generalized first-order spectra and polynomial-time recognizable sets," *Complexity of computation*, vol. 7, pp. 43–73, 1974, Publisher: Providence, RI.

[7] L. Gąsieniec, J. Jansson, and A. Lingas, "Efficient approximation algorithms for the hamming center problem," in *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, 1999, pp. 905–906.

[8] M. Frances and A. Litman, "On covering problems of codes," *Theory of Computing Systems*, vol. 30, no. 2, pp. 113–119, Apr. 1, 1997, ISSN: 1433-0490. DOI: 10.1007/BF02679443. [Online]. Available: https://doi.org/10.1007/BF02679443 (visited on 08/20/2023).

[9] G. D. Stormo and G. W. Hartzell 3rd, "Identifying protein-binding sites from unaligned DNA fragments.," *Proceedings of the National Academy of Sciences*, vol. 86, no. 4, pp. 1183–1187, 1989, Publisher: National Acad Sciences.

[10] Y.-S. Han, S.-K. Ko, T. Ng, and K. Salomaa, "Closest substring problems for regular languages," *Theoretical Computer Science*, A Fascinating Rainbow of Computation – Honoring Gheorghe Păun on the Occasion of His 70th Birthday, vol. 862, pp. 144–154, Mar. 16, 2021, ISSN: 0304-3975. DOI: 10.1016/j.tcs.2020.09.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030439752030503X (visited on 08/20/2023).

[11] X. Liu, K. Fu, and R. Shao, "Largest distance decreasing algorithm for the closest string problem," *Journal of Information and Computational Science*, vol. 1, pp. 287–292, Dec. 1, 2004.

[12] O. L. Vilca and M. J. Salvatierra, *A recursive exact algorithm for the closest string problem*, Rochester, NY, Apr. 9, 2022. DOI: 10.2139/ssrn.4079638. [Online]. Available: https://papers.ssrn.com/abstract=4079638 (visited on 08/12/2023).

[13] "A heuristic solution to the closest string problem using wave function collapse techniques." (), [Online]. Avail-

able: https://ieeexplore.ieee.org/document/9932602 (visited on 08/12/2023).

[14] M. Gumin, *Wave function collapse algorithm*, version 1.0, original-date: 2016-09-30T11:53:17Z, Sep. 2016. [Online]. Available: https://github.com/mxgmn/WaveFunctionCollapse (visited on 08/12/2023).

[15] S. R. Mousavi, "A hybrid metaheuristic for closest string problem," *International Journal of Computational Biology and Drug Design*, vol. 4, no. 3, pp. 245–261, 2011, ISSN: 1756-0756. DOI: 10.1504/IJCBDD.2011.041413.

[16] X. Liu, S. Liu, Z. Hao, and H. Mauch, "Exact algorithm and heuristic for the closest string problem," *Computers & Operations Research*, vol. 38, no. 11, pp. 1513–1520, Nov. 1, 2011, ISSN: 0305-0548. DOI: 10.1016/j.cor.2011.01.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054811000219 (visited on 08/12/2023).

[17] S. Yuasa, Z.-Z. Chen, B. Ma, and L. Wang, "Designing and implementing algorithms for the closest string problem," *Theoretical Computer Science*, Frontiers of Algorithmics, vol. 786, pp. 32–43, Sep. 27, 2019, ISSN: 0304-3975. DOI: 10.1016/j.tcs.2018.05.017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304397518303372 (visited on 08/12/2023).

[18] P. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, Nov. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[19] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, 2014, ISSN: 2296-424X. DOI: 10.3389/fphy.2014.00005. arXiv: 1302.5843[cond-mat,physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1302.5843 (visited on 05/21/2023).

[20] E. Ising, "Beitrag zur Theorie des Ferromagnetismus," *Zeitschrift für Physik*, vol. 31, no. 1, pp. 253–258, Feb. 1925, ISSN: 0044-3328. DOI: 10.1007/BF02980577. [Online]. Available: http://link.springer.com/10.1007/BF02980577 (visited on 08/26/2023).

[21] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse ising model," *Physical Review E*, vol. 58, no. 5, pp. 5355–5363, Nov. 1, 1998, ISSN: 1063-651X, 1095-3787. DOI: 10.1103/PhysRevE.58.5355. arXiv: cond-mat/9804280. [Online]. Available: http://arxiv.org/abs/cond-mat/9804280 (visited on 08/26/2023).

[22] M. W. Johnson, P. Bunyk, F. Maibaum, *et al.*, "A scalable control system for a superconducting adiabatic quantum optimization processor," *Superconductor Science and Technology*, vol. 23, no. 6, p. 065 004, 2010, Publisher: IOP Publishing.

[23] K. Boothby, P. Bunyk, J. Raymond, and A. Roy, *Next-generation topology of d-wave quantum processors*, Feb. 28, 2020. DOI: 10.48550/arXiv.2003.00133. arXiv: 2003.00133[quant-ph]. [Online]. Available: http://arxiv.org/abs/2003.00133 (visited on 09/17/2023).

[24] "Getting started with d-wave solvers — d-wave system documentation documentation." (), [Online]. Available: https://docs.dwavesys.com/docs/latest/doc_getting_started.html (visited on 08/26/2023).

[25] D.-W. Systems, "Programming the d-wave QPU : Parameters for beginners," D-Wave Systems, Whitepaper 14-1045A-A, Sep. 17, 2020. [Online]. Available: https://www.dwavesys.com/media/qvbjrzgg/guide-2.pdf.

[26] "Workflow: Formulation and sampling — d-wave system documentation documentation." (), [Online]. Available: https://docs.dwavesys.com/docs/latest/c_gs_workflow.html#samplers (visited on 09/02/2023).

[27] A. R. Parkinson, R. Balling, and J. D. Hedengren, "Optimization methods for engineering design," *Brigham Young University*, vol. 5, no. 11, 2013.

[28] "QPU-specific characteristics — d-wave system documentation documentation." (), [Online]. Available: https://docs.dwavesys.com/docs/latest/doc_physical_properties.html (visited on 09/20/2023).

[29] C. Klymko, B. D. Sullivan, and T. S. Humble, "Adiabatic quantum programming: Minor embedding with hard faults," *Quantum information processing*, vol. 13, pp. 709–729, 2014, Publisher: Springer.

[30] "Introducing ChatGPT." (), [Online]. Available: https://openai.com/blog/chatgpt (visited on 08/13/2023).

[31] "Programming the d-wave QPU: Setting the chain strength." (), [Online]. Available: https://www.dwavesys.com/resources/white-paper/programming-the-d-wave-qpu-setting-the-chain-strength/ (visited on 09/22/2023).

Chandeepa Dissanayake is a senior undergraduate student of computer science at the University of Kelaniya, Sri Lanka. He has research interests in theoretical computer science, quantum computing, and quantum information science. He was awarded the second prize at the IEEE SA P2834 Student Challenge 2022 and several national awards for his innovative projects in computer science.