Modeling the Cyclic Bandwidth Problem in QUBO for Quantum Annealing

Philippe Codognet[1][0000−0002−6254−6389] and Eric Monfroy[2][0000−0001−7970−1368]

[1] JFLI, Tokyo, Japan
[2] University of Angers, LERIA, France
codognet@is.s.u-tokyo.ac.jp Eric.Monfroy@univ-angers.fr

Abstract. We consider a graph labeling problem, i.e., the CYCLIC BANDWIDTH problem, and its formulation in QUBO, the input language for quantum computers based on quantum annealing. To this end, we first consider a constraint programming model based on table constraints and then we derive from this model the QUBO formulation and its penalty matrix. We also detail an analysis of this QUBO model in terms of number of qubits and their required inter-qubit connections in order to estimate the suitability of implementing such a solution on quantum annealers, i.e., the D-Wave Advantage system with a specific graph topology for qubit couplers.

1 Introduction

The new domain of quantum optimization refers to the use of quantum computers to solve combinatorial optimization problems. It has been gaining increasing attention over the past decade. This interest spans both the gate-model paradigm, with the development of the Quantum Approximate Optimization Algorithm (QAOA) [13], and the quantum adiabatic computing paradigm, with the development of Quantum Annealing (QA) [20, 14]. The Quantum Annealing (QA) paradigm was proposed more than two decades ago as an alternative to the gate-based model in quantum computing. It involves modeling a problem using the Hamiltonian (energy function) of a system, where the ground state corresponds to the solution of the original problem. The computation starts from a simple initial Hamiltonian with a known ground state, which is easy to prepare. Through adiabatic evolution, the system gradually transitions to the ground state of the problem Hamiltonian [28]. From a computational point of view, it is related to simulated annealing. However, it takes advantage of the *quantum tunneling* effect to overcome energy barriers and escape local minima, aiming to reach the ground state, rather than relying on (simulated) thermal effects. In the last decade, the development of quantum devices such as D-Wave quantum annealers [7], as well as more recently the "quantum-inspired" hardware such as Fujitsu Digital Annealer [4], makes it possible to experiment QA on a variety of abstract or real-life problems [38], especially combinatorial optimization problems [29, 36].

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

Many examples of classical NP-complete problems have been defined as Ising models in the seminal work of Lucas [25], and the formulation of problems in terms of Ising models is very close to, and indeed equivalent to, a formulation in Quadratic Unconstrained Binary Optimization (QUBO), a simple but powerful paradigm for modeling various types of combinatorial problems, see for instance [16] for a tutorial introduction and [30] for details. Therefore QUBO has become in the last years the standard input language for all quantum and quantum-inspired annealing hardware.

We are therefore interested in this paper in modeling in QUBO the Cyclic Bandwidth Problem, a graph-based combinatorial problem with many applications, in order to make it possible to use quantum annealing to solve it. The decision problem corresponding to the Cyclic Bandwidth is known to be NP-complete [23], and is usefull for important application areas like VLSI designs [5], data structure representations [33], code design [10], and interconnection networks for parallel computer systems [19].

The classical Cyclic Bandwidth problem is an optimization problem where the input is an undirected graph $G = (V, E)$ with $|V| = n$ vertices. The task is to find a labeling φ of the vertices V, such that each vertex v is assigned a unique value $\varphi(v) \in [1, n]$, that minimizes $B_c(G, \varphi) = \max_{(u,v)\in E(G)}\{\min\{|\varphi(u) - \varphi(v)|, n - |\varphi(u) - \varphi(v)|\}\}$, which represents the *cyclic bandwidth* of G.

On the one hand, modeling this problem in QUBO with Boolean variables (i.e., with only two possible values, 0 and 1) and quadratic polynomials is not straightforward and deriving the part of the QUBO matrix that corresponds to the problem constraint is even more unclear. On the other hand, Constraint Programming [34, 1] is a programming paradigm that enables one to declaratively model problems as Constraint Satisfaction Problems (CSPs) or Constrained Optimization Problems (COPs).

We thus propose to convert a CSP model for the cyclic bandwidth problem into a QUBO formulation with quadratic penalty expressions corresponding to the problem constraints. The direct formulation of the cyclic bandwidth problem in COP is straightforward and very close from the mathematical formulation of the problem. However, solving such derived instances directly is not efficient in the Constraint Programming paradigm: most of the constraints are in the objective function, and thus, constraint propagation is rather inefficient. We then derive a CSP model for determining the satisfiability, and obviously a labelling with a cyclic distance less than or equal to a given k. This model is based on *table constraints*, i.e., constraint in extension studied initially in [2, 3] and more recently as table constraints, see for example [21], or [37] for a complete overview. This satisfiability algorithm is completed with two different algorithms for optimization: an incremental algorithm for minimization, and another algorithm based on dichotomy and an intrinsic property of the problem. It is then easier to derive from those formulations the QUBO model of the cyclic bandwidth model and the QUBO matrix representing the objective function and the penalty expressions corresponding to the constraints. We also give some upper bounds on the number of logical qubits and inter-qubit connections necessary to encode

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

some problem instances to discuss which kind of graphs could be treated with current quantum computing systems such as the D-Wave Advantage [27].

This paper is organized as follows. We introduce constrained optimization in QUBO in Section 2. Then, in Section 3, we give the problem and its CSP formulation. From these models, we propose a quadratic model for the cyclic bandwidth problem in Section 4. We then give the overall QUBO matrix representing the constraints and objective function of the model in Section 5. A short conclusion ends the paper.

2 Constrained Optimization in QUBO

Consider n Boolean variables $x_1, ..., x_n$, a Quadratic Unconstrained Binary Optimization (QUBO) problem consists in minimizing an *objective function* defined by a quadratic expression over $x_1, ..., x_n$: $\sum_{i \leq j} q_{ij} x_i x_j$

A QUBO problem can therefore be represented by a vector x of n Boolean decision variables and a square $n \times n$ matrix Q with coefficients q_{ij}. Then the minimization problem can be written as:
minimize $y = x^t Q x$, where x^t is the transpose of x.

Observe that this quadratic formulation also includes a linear part, corresponding to the matrix diagonal, because $x_i^2 = x_i$ for Boolean variables (x_i).

As pioneered by [25], simple combinatorial problems are easy to express directly as Ising or QUBO models, but more complex problems, especially problems involving *constraints* such as Constrained Optimization Problems (COP) or Constraint Satisfaction Problems (CSP), are more difficult to model in QUBO.

As developed in the Constraint Programming paradigm [34], Constraint expressions can be introduced in QUBO models as *penalties* in the objective function to minimize, that is, as quadratic expressions whose value is minimal when the constraint is satisfied. An easy way to formulate such a penalty for a given constraint is to create a quadratic expression which has value 0 if the constraint is satisfied and a positive value if the constraint is not satisfied, representing somehow the degree of violation of the constraint. However, in the general case, minimal values of penalty expressions are not bound by zero and can be negative.

Consider for instance the pseudo-Boolean constraint $x + y + z = 1$. It can be represented by the penalty expression $-x - y - z + 2xy + 2xz + 2yz$. This expression will have minimal value -1 when a single Boolean variable has value 1 and will have a higher value otherwise. Observe that this constraint enforces that among the three variables x, y and z one and only one will be set to 1. This quadratic expression corresponding to a pseudo-Boolean constraint can be used, for instance to model the well-known graph-coloring problem in order to state that one node shall be colored with a single color (e.g., among three possible values). Such a constraint is called *one-hot encoding* in QUBO and widely used for modeling combinatorial problems.

For modeling the CYCLIC BANDWIDTH problem, we need to consider a more complex constraint: the *AllDifferent/Permutation* constraint. To enforce that n

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

integer variables assigned to values in $\{1,\ldots,n\}$ form a permutation, we need to use a simple property of permutations stating that each value $k \in \{1,\ldots,n\}$ is assigned exactly once. This is known as a *two-way one-hot* in the QA literature. In the Constraint Programming community, such a *permutation constraint* is a special case of the *all-different* constraint, which has been the subject of a large literature and various solving techniques [15, 18].

Consider n integer variables $x_i \in \{1,\ldots,n\}$, and n^2 Boolean variables x_{ij} encoding each x_i in such a way that $x_{ij} = 1$ if x_i has value i and value 0 otherwise (*one-hot* encoding). The constraint that $(x_1,\ldots,x_n)$ forms a permutation of $\{1,\ldots,n\}$ can be encoded in QUBO by $2 \times n$ pseudo-Boolean constraints:

- n constraints corresponding to each of the n variables x_i stating that it can have only one value k
- n constraints for each of the n values k stating that it can be assigned to only one variable x_i.

Each constraint is a one-hot constraint of the form $\sum x_{ij} = 1$, with different index sets, which is equivalent to $(\sum x_{ij} - 1)^2 = 0$ as the x_{ij} are Boolean variables. Consider now the quadratic expression $(\sum x_{ij} - 1)^2$. This expression will have value zero when the permutation constraint is satisfied and a positive value otherwise; it can thus be used as a penalty corresponding to the constraint that should be added to the QUBO objective function in order to have the constraint satisfied for every minimal solution of the QUBO problem. Therefore the penalty for the permutation constraint is given by adding together all the $2 \times n$ penalty expressions corresponding to the basic one-hot constraints and simplifying the quadratic expression gives the following definition [11, 12]:

$$P_{\text{perm}} = \sum_{i=1}^{n} \sum_{j<j'} x_{ij}x_{ij'} \; + \; \sum_{j=1}^{n} \sum_{i<i'} x_{ij}x_{i'j} \; - \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij}$$

3 The cyclic bandwidth problem

3.1 The problem

The Cyclic Bandwidth problem is a graph labeling problem whose complexity has been established as NP-hard [23]. Let $G(V,E)$ be a finite undirected graph (called the *guest graph*) of order n (i.e., with $|V| = n$), and $C_n(V',E')$ be a cycle graph (called the *host graph*) with $|V'| = n$. An embedding of G in C_n is an injection $\phi : V \to V'$. The *cyclic distance* d_c between two vertices $(u,v) \in E$ (i.e., two vertices of V linked by an edge of E) is defined by:

$$d_c(u,v) = min\{|\phi(u) - \phi(v)|, n - |\phi(u) - \phi(v)|\} \tag{1}$$

The cyclic bandwidth of an embedding $\phi : V \to V'$ is the maximum distance between two vertices linked by an edge:

$$B_c(G,\phi) = max_{(u,v)\in E}\{d_c(u,v)\} \tag{2}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

The Cyclic Bandwidth problem consists in finding an embedding ϕ^* among the set $\mathcal{E}$ of embeddings from G to C_n such that $B_c(G, \phi^*)$ is minimum, i.e.,

$$B_C^*(G) = min_{\phi \in \mathcal{E}}\{B_c(G, \phi)\} \tag{3}$$

3.2 Cyclic bandwidth as a Constrained Optimization Problem

Constraint programming (CP) [34] is a paradigm for addressing combinatorial problems by employing various techniques derived from artificial intelligence, computer science, and operations research. In CP, users specify "what" the problem is rather than "how" to solve it. This approach allows users to declaratively state the problem: programming corresponds to modeling, and a program is a model. A model represents a problem; together with data they represent a problem instance (or just an instance) which is then solved by a solver. In our case, the model represents the cyclic bandwidth problem, the model plus a given graph are an instance of the cyclic bandwidth problem. A solver is one or some algorithms for solving problem instances. A solver is generic, i.e., it is not specialized for a given problem: a solver is just limited by the constraints and variables it accepts (most of the time, bounded integers with arithmetic or symbolic constraints). Classical constraint solvers in the CP approach are based on constraint propagation: the solver reduces the search space by repeatedly applying filtering functions to remove values of domains that cannot participate in any solution; this phase is interleaved with decision phases that split the search space into two, leading to a kind of branch and bound for optimization problems.

Consequently, a problem is formulated as either a constraint satisfaction problem (CSP) or a constrained optimization problem (COP). A CSP is characterized by decision variables, each with a domain of possible values, and constraints that define the relationships between these variables. A COP extends a CSP by including an objective function that needs to be optimized.

Direct model The direct model is very close to the mathematical definition of the Cyclic Bandwidth problem. It is based on bounded integer variables (called finite domain decision variables) that are linked by some arithmetic linear constraints. A vertex v is represented by its number in $[1..n]$; by abuse of language, the number of v is also v, an integer in $[1..n]$. A label l_v for a vertex $v \in V$ is thus represented by a finite domain variable ranging from 1 to n:

$$\forall v \in [1..n], l_v \in [1..n] \tag{4}$$

In our model, a labeling ϕ is thus represented by a sequence $l_1, \ldots, l_n$ that is a permutation of $1, \ldots, n$. We denote by $\mathcal{S}_n$ the set of permutations of $1, \ldots, n$. The next constraint means that two vertices cannot have the same label:

$$\forall v, v' \in [1..n],\ v < v' \Rightarrow l_v \neq l_{v'} \tag{5}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

In terms of COP, these $n*(n+1)/2$ disequalities are equivalent to the global constraint[3] *AllDifferent* [15, 18]. Moreover, since there are n variables, each one having n candidate values, we have a permutation. The following constraint is thus semantically equivalent to the constraints in Formula (5):

$$\texttt{Permutation}(\{l_v \mid v \in V\}) \tag{6}$$

The cyclic bandwidth of the current labeling l is given by the following constraint:

$$B_c(G,l) = \max_{(u,v)\in E}\{d_c(l_u, l_v)\} \tag{7}$$

where $B_c(G,l)$ is a finite domain variable ranging from 1 to $n-1$. Note that d_c has been defined in Equation 1. As said before, the cyclic bandwidth must be minimized to obtain $l_1^*, \ldots, l_n^*$, one of the best cyclic labelings

$$B_C^*(G) = minimize_{(l_1,\ldots,l_n \in \mathcal{S}_n)}\ B_C(G,l) \tag{8}$$

where $B_C^*(G)$ is a finite domain variable ranging from 1 to $n-1$, and $\mathcal{S}_n$ is the set of all the permutations of $1, \ldots, n$. The direct model $\mathcal{M}_d^\star$ is thus:

$$\mathcal{M}_d^\star = (4) \wedge (6) \wedge (7) \wedge (8)$$

Since all of the constraints, except the permutation constraint, are in the objective function, this model does not provide enough information to efficiently reduce the search space.

Model with extensional constraints Now consider that we have a decision procedure $sat(G,k)$ able to return true if a labeling l such that $B_c(G,l) \leq k$ exists, otherwise false. A classic optimization algorithm (see Algorithm 1) for minimization is the incremental minimization: it consists of starting from a given lower bound k that can be initialized with $\lceil \Delta(G)/2 \rceil$ [24], where $\Delta(G)$ represents the maximum degree of G. If $sat(G,k)$ is satisfiable, then k is the minimum. Otherwise, k is incremented, and the algorithm iterates.
Hence, `optimization_inc(`$G, \lceil \frac{\Delta(G)}{2} \rceil$`)` returns $B_c^*(G)$.

However, it is obvious that if a cyclic labeling l exists with a cost of k, i.e., $B_c(G,l) = k$ then there is also one with a cost of $k+1$. Conversely, if there is no labeling with a cost of k, then none exists with a cost of $k-1$. Then, based on the above property and the satisfiability procedure, we present an efficient dichotomic optimization algorithm (see Algorithm 2). The lower and upper bounds, lb and ub, can be initialized with $\lceil \Delta(G)/2 \rceil$ and $\lceil n/2 \rceil$ respectively [24]. Hence, `optimization_dic(`G, lb, ub`)` returns $B_c^*(G)$.

Now, let's focus on the `sat(`G,k`)` function. We need to define a function able to decide whether there exists or not a labeling l such that $B_c(G,l) \leq k$.

[3] A global constraint has two advantages compared to the equivalent "simple" constraints: it is more expressive and the solver has some special more efficient algorithms to solve it.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

Algorithm 1 `optimization_inc`(G, k)

```
if sat(G, k) then
    return k
else
    k ← k + 1
    optimization_inc(G, k)
end if
```

Algorithm 2 `optimization_dic`(G, lb, ub)

```
k_best ← ub
while lb < ub do
    k ← (ub + lb) div 2
    if sat(G, k) then
        ub ← k
        k_best ← ub
    else
        lb ← k + 1
    end if
end while
return k
```

To this end, we consider finite domain extensional constraints (see, e.g., [3]), also known as table constraints [21]: a constraint is defined by enumerating the allowed tuples of constants satisfying it; this table can be seen as a kind of truth table of the constraint. Then, a tuple of variables satisfies the constraint if it is an element of this table. Now, let us consider $\mathcal{L}(k)$, the set of possible pairs of labels for pairs of vertices linked by an edge:

$$\mathcal{L}(k) = \{(\ell, \ell') \mid \ell, \ell' \in [1..n]^2, \ell \neq \ell' \Rightarrow \min\{|\ell - \ell'|, n - |\ell - \ell'|\} \leq k\}$$

The model is now given by:

- the same finite domain variables as for model $\mathcal{M}_d$;
- each label is unique, thus we keep the Permutation Constraint (6);
- distance d_c between the two vertices of an edge must be less than or equal to k, i.e.,

$$\forall (u, v) \in E,\ (l_u, l_v)\ in\ \mathcal{L}(k) \tag{9}$$

Thus, the finite domain extensional constraint satisfiability model is:

$$\mathcal{M}_{E,k} = (4) \wedge (6) \wedge (9) \tag{10}$$

Note that this model depends on k, the cyclic bandwidth value. When calling $\texttt{sat}(G, k)$, the $\mathcal{M}_{E,k}$ model is instanciated with graph G. Then, a classical propagation-based solver (such as Gecode [35] ou ACE [22]) can determine the satisfiability of the instance, and a labeling l respecting $B_c(G, l) \leq k$).

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

4 The cyclic bandwidth problem as a QUBO model

Modeling the cyclic bandwidth problem as a quadratic unconstrained binary optimization (QUBO) problem is not intuitive. However, passing via the constraint formulation, and more especially the extensional constraint model decoupling satisfiability from optimization, makes it possible to formulate more easily a quadratic model equivalent to the $\mathcal{M}_{E,k}$ model. Let us first transform in a binary quadratic form the $\mathcal{M}_E$ model.

As we only consider Boolean variables, each variable l_u (u being a vertex, and l_u being the label of u) of the previous section is replaced by n variables $l_{u,1}, \ldots, l_{u,n}$ with the following semantics: $l_u = i$ iff $l_{u,i} = 1$ and for each $j \in [1..n], j \neq m \Rightarrow l_{u,j} = 0$.

We can enforce this semantic for each label l_u using the one-hot constraint:

$$\forall u \in [1..n], \sum_{i \in [1..n]} l_{u,i} \;=\; 1 \tag{11}$$

Then, each label i can be used only once:

$$\forall i \in [1..n], \sum_{u \in [1..n]} l_{u,i} \;=\; 1 \tag{12}$$

The Constraints (11) and (12) define a permutation of $1, \ldots, n$, see Section 2.

We finally have to formulate that the distances between the two labels of vertices of an edge are less than or equal to the given maximal distance k. Consider the $\mathcal{L}(k)$ set of couples are as defined before, and edges (u, v) defined by couples of vertices numbers. We can formulate the table constraint given by the couples in $\mathcal{L}(k)$ in terms of Boolean variables $l_{u,i}$ and $l_{v,j}$ as follows:

$$\forall (u, v) \in E, \sum_{(i,j) \notin \mathcal{L}(k)} l_{u,i} l_{v,j} \;=\; 0 \tag{13}$$

Then, the quadratic penalty corresponding to this table constraint can be easily derived from this set of pseudo-Boolean equations, as follows:

$$P_{\text{table}} = \sum_{(u,v) \in E} \left(\sum_{(i,j) \notin \mathcal{L}(k)} l_{u,i} l_{v,j} \right) \tag{14}$$

5 The QUBO matrix

We can now define the QUBO matrix, i.e., the matrix composed of the coefficients of quadratic monomial terms of the objective function to minimize.

First, we define the QUBO matrix $Q = Q_{\text{table}} + Q_{\text{perm}}$, where:

- Q_{table} is the matrix of penalties representing the quadratic expression P_{table} and corresponding to the table constraints; here, we have to minimize the sum appearing in Equation 14 in the QUBO model which is due to each of the table constraints (as defined in Constraint (9) for the CSP model).

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

– Q_{perm} is the matrix of penalties representing the quadratic expression P_{perm} given in Section 2 and corresponding to the permutation constraint, i.e., the set of Constraints in (11) and (12).

The matrix Q (and thus Q_{table} and Q_{perm}), is a square matrix of size n^2, n^2 being the number of Boolean variables in the vector of decision variables $L = (l_{1,1}, \ldots, l_{1,n}, \ldots, l_{n,1}, \ldots, l_{n,n})$. Solving the QUBO problem consists in minimizing L^tQL. In the case of the Cyclic Bandwidth Problem, $Q(l_{i,j}, l_{i',j'})$ can be seen as the penalty associated to giving the value j to the label l_i and the value j' to the label $l_{i'}$. The problem is thus satisfied if and only if $L^tQL = 0$.

5.1 Penalties

We now describe how the penalties are computed.

For the Permutation constraint, we have the penalty defined in [11, 12] and stated at the end of Section 2. Figure 1 shows an example for 3 initial variables with domains $\{1, 2, 3\}$, encoded by 9 Boolean variables. Note that on the matrix in Figure 1 we can see some diagonals of three 1 appearing, to forbid having twice the same label. We have also the repeated pattern of one-hot on the diagonal.

$$
Q_{perm} = \begin{array}{c} \\ l_{1,1} \\ l_{1,2} \\ l_{1,3} \\ l_{2,1} \\ l_{2,2} \\ l_{2,3} \\ l_{3,1} \\ l_{3,2} \\ l_{3,3} \end{array}
\begin{array}{c}
\begin{array}{ccccccccc} l_{1,1} & l_{1,2} & l_{1,3} & l_{2,1} & l_{2,2} & l_{2,3} & l_{3,1} & l_{3,2} & l_{3,3} \end{array} \\
\begin{pmatrix}
-1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
 & -1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
 & & -1 & 0 & 0 & 1 & 0 & 0 & 1 \\
 & & & -1 & 1 & 1 & 1 & 0 & 0 \\
 & & & & -1 & 1 & 0 & 1 & 0 \\
 & & & & & -1 & 0 & 0 & 1 \\
 & & & & & & -1 & 1 & 1 \\
 & & & & & & & -1 & 1 \\
 & & & & & & & & -1
\end{pmatrix}
\end{array}
$$

Fig. 1. Penalty matrix for a permutation constraint over 3 variables with 3 values. Note that the diagonals in red mean that two variables cannot have the same value. Note also that the matrices on the diagonal, in green (for variables $l_{1,-}$ and $l_{2,-}$) and blue (for $l_{3,-}$) means that a variable has one and only one value .

For Constraints (13), i.e., the sums of quadratic monomials encoding the table constraint, we give penalties to couples not member of the table constraint. Thus, each sub-matrix $Q_{u,v}$ such that $(u, v) \in E$ is defined as shown in Figure 2 on the left: entries of the matrix are strictly greater than 0 if they correspond to an element which is not in the table, and equal to 0 otherwise. Sub-matrices corresponding to couples of variables that are not an edge are empty matrices.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

$$Q_{table} = \begin{array}{c} \\ l_{u,1} \\ l_{u,2} \\ l_{u,3} \end{array} \begin{array}{c} \begin{array}{ccc} l_{v,1} & l_{v,2} & l_{v,3} \end{array} \\ \begin{pmatrix} 1 & t(1,2) & t(1,3) \\ 0 & 1 & t(2,3) \\ 0 & 0 & 1 \end{pmatrix} \end{array} \qquad Q_{table} = \begin{array}{c} \\ l_{u',1} \\ l_{u',2} \\ l_{u',3} \end{array} \begin{array}{c} \begin{array}{ccc} l_{v',1} & l_{v',2} & l_{v',3} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

Fig. 2. Penalty sub-matrices considering variables with 3 candidate labels $\{1, 2, 3\}$. On the left, a penalty sub-matrix with $(u, v) \in E$ (and $u \neq v$ since we consider graphs without loops): the diagonal in red means that the two vertices cannot have the same label; for other entries, $t(i, j) = 1$ if $(i, j) \notin \mathcal{L}(k)$, 0 otherwise.
On the right: when $(u', v') \notin E$ the corresponding sub-matrix $Q_{u',v'}$ is empty.

5.2 Counting qubits and their connections

The QUBO matrix is the input representation for the quantum annealing computation that will be performed on quantum devices such as D-Wave systems or quantum-inspired dedicated hardware such as Fujitsu Digital Annealer. Each system has its own limitation in terms of numbers of (physical) qubits, qubit interconnections, and hardware graph/topology. Thus, it is important to evaluate the size of QUBO models in terms of logical qubits and their connections for the CYCLIC BANDWIDTH problem. Indeed, since we don't consider here the hardware graph, this will enable us to give, in the next section, a lower bound of the physical qubits required for their evaluation on current quantum hardware.

Consider a graph with n vertices, $|E|$ edges, and a cyclic labeling of value k. Let us first count the Boolean variables $l_{u,i}$ that we need to represent the n possible values of the n labels: hence, n^2 variables in total. Then, we can count $oc(l_{u,i})$, the number of occurrences of $l_{u,i}$ in the penalty matrices, i.e., connections with another variable.

For the permutation constraint: For the n $n \times n$-matrices composing the diagonal we have $n^2(n-1)/2$ strictly positive values (n matrices on the diagonal, each one with $n(n-1)/2$ values > 0). There are $n(n-1)/2$ $n \times n$-matrices in the upper triangular matrix without the diagonal. For each of these matrices, only the diagonal is composed of values strictly greater than 0 (n values per matrix). Thus, we have $n^2(n-1)/2$ non-zero entries in the upper triangular matrix. In total, the permutation constraint generates $n^2(n-1)$ penalties, i.e., entries of the Q_{perm} matrix that are strictly greater than 0.

All the $l_{i,j}$ variables are equivalent and penalized the same way. Hence, each variable $l_{i,j}$ is penalized: $2n^2(n-1)/n^2 = 2(n-1)$ times which is the total number of penalties divided by the number of lines, and multiplied by 2 since each entry of the matrix implies 2 variables. For example, for a permutation of 3 variables having a domain of size 3, this gives 4 penalties for each $l_{i,j}$ w.r.t. the permutation constraint.

For the table constraints: For each edge $(u, v) \in E$, we have the following. Suppose i is the label of u, then $2k$ labels j can correspond to i to respect the given distance k. Thus, $n - 2k$ labels are penalized to minimize the objective function

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

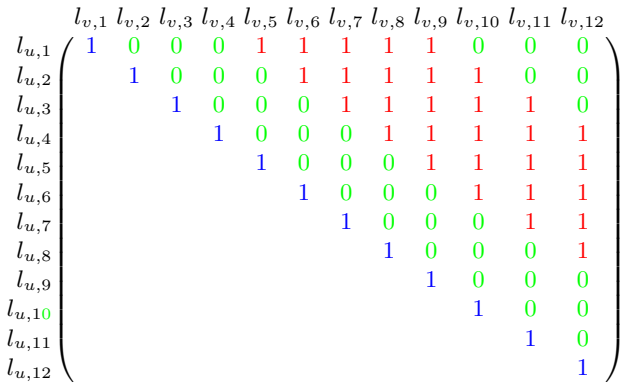

Fig. 3. Penalty matrix for a table constraint for an edge $(u, v) \in E$ with $k = 3$ and $n = 12$. Note that the width of the two diagonal stripes of 0 (in green) is k. The diagonal of 1 (in red) has a width of $n - 2k - 1$. Note also that when k is growing of an increment δ, the width of the two diagonal stripes of 0 (in green) is also growing of δ. Last, note that the main diagonal of 1 (in blue) means that the same label cannot be given to two vertices. This diagonal could be set to 0 since this constraint is taken in account by the permutation matrix.

of Equation (13). These penalties are all in the $n \times n$-submatrix corresponding to the $l_{u,i}$ and $l_{v,j}$, and they are distributed as shown in Figure 3.

All the penalty sub-matrices corresponding to edges from G are disjoint (i.e., each one fits to one u and one v), Thus, in total, a variable $l_{u,i}$ appears $oc(l_{u,i}) = d(u)(n - 2k)$ times for the objective function of Equation (14), with $d(u)$ being the degree of u (i.e., the number of edges connected to u). This number of occurrences can be made independent from u by bounding it as follows:

$$oc(l_{u,i}) \leq \Delta(G)(n - 2k)$$

with $\Delta(G)$ the maximum degree of the graph.

For the addition of the penalty matrices for the objective function and the permutation constraint. An easy counting consists in only summing occurrences of entries strictly greater than zero. This gives the following upper bound of occurrences of a variable $l_{u,i}$:

$$oc(l_{u,i}) \leq d(u)(n - 2k) + 2(n - 1)$$

Moreover, since $d(u) \leq \Delta(G)$

$$oc(l_{u,i}) \leq \Delta(G)(n - 2k) + 2(n - 1)$$

However, we can see on the penalty matrix of the permutation that the diagonals of 1, corresponding to a sub-matrix $l_{u,\_} \times l_{v,\_}$, is already in the penalty

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

matrix of the objective if and only if $(u, v) \in E$. This means that one non-negative entry is counted in the penalty matrix of the permutation and of the tables. We hence obtain that:

$$oc(l_{u,i}) \;=\; d(u)(n-2k) + 2(n-1) - d(u) \;=\; d(u)(n-1-2k) + 2(n-1)$$

This can be over-estimated for all vertices u by:

$$oc(l_{u,i}) \leq \Delta(G)(n-1-2k) + 2(n-1)$$

5.3 Practical usability

We will focus our analysis on the quantum annealing D-Wave Advantage system [27], which has a specific architecture and graph topology interconnecting the qubits: there is a total of 5640 qubits, and each qubit has 15 connections linking with others qubits according to the *Pegasus* topology [6]. This specification is useful to be able to evaluate the number of physical qubits needed for representing an instance, as inter-connections between qubits have to be realized either by physical connections ("couplers" in quantum device terminology) or by using extra qubits with a transformation called *minor embedding* [8, 9]. The graph topology is indeed a key limit for the size of the instances that can be implemented on quantum annealing devices. For example on the D-Wave Advantage, the theoretical maximal all-to-all connectivity that can be simulated is limited to 177 "logical" qubits [27], vs. 5640 "physical" qubits. Note that the minor embedding algorithm currently implemented on D-Wave systems is based on heuristics and thus is not guaranteed to perform the theoretical optimal embedding, i.e., the one with the minimal number of qubits and inter-qubits connections.

In the previous section, we gave the theoretic "dimensions" of an instance. We now illustrate this usability on the 113 Harwell-Boeing classical benchmark [26], extracted from the Harwell-Boeing sparse matrix collection[1]. Consider the instance `pores_1` which is rather small but also rather dense: it has 103 edges and $n = 30$ vertices, and the maximum degree is $\Delta(G) = 9$ (for 6 vertices). The minimal cyclic bandwidth is in [5..15] as explained in Section 3 when fixing the bounds of the optimization algorithm. This means that we need 900 logical qubits to represent the $l_{u,i}$ (QUBO variables), and each logical qubit has at most 247 connections for $k = 5$ and 49 connections for $k = 15$. Then, for $k = 15$, we need 4 physical qubits for each logical qubit (with a chain of physical qubits representing the same logical qubit), thus a total of 3600 qubits. Since we don't take into account the hardware graph, this is a lower bound that remains tractable on systems such as the D-Wave Advantage. To ensure feasibility, the upper bound – considering hardware constraints – must also be tractable.

However, 19 physical qubits are necessary for each logical qubit for $k = 5$, therefore requiring 17,100 qubits in total. Even without considering the hardware graph, we can determine that this instance is not tractable on the D-Wave Advantage or any other current quantum system.

[1] see, e.g., `https://math.nist.gov/MatrixMarket/collections/hb.html` and `https://sparse.tamu.edu/HB`

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

Although larger, instances such as `bcspwr01`, and paths, cycles, caterpillars, and small trees with less than 30 vertices from [31] could be tractable on the current D-Wave quantum devices because of very low densities.

6 Discussion and Conclusion

The use of quantum computers, and in particular quantum annealing, to solve concrete problems in the domain of combinatorial optimization currently raises tremendous interest. Quadratic Unconstrained Binary Optimization (QUBO) is the input formalism for such quantum devices based on quantum annealing and is thus the target language of choice for modeling optimization problems to be solved by quantum annealing. However, modeling an optimization problem in QUBO can be a tedious task, especially when complex constraints are involved as such constraints have to be replaced by quadratic penalty expressions to be integrated in the objective function.

Graph embedding, and more especially graph labeling and the cyclic bandwidth problems, are problems that can easily be modeled in the constraint programming paradigm, which consists in declaratively modeling problems as CSP (for satisfaction problems) or COP (for optimization problems), but this formulation is not always efficient. Rephrasing the cyclic bandwidth problem as a satisfaction problem together with an upper layer for optimization, a CSP model based on extensive constraints (table constraints) can be defined. This model is elegant and can be solved more efficiently than the direct model with standard constraint propagation-based solvers. Moreover, this model is a good basis for deriving the penalty matrix and define a QUBO model to be solved by quantum annealing. For practical considerations, we also count the number of logical qubits and of qubit inter-connections required in this model for a given graph instance. We can see that the smallest instances of the HB113 benchmarks seems attainable by the current D-Wave Advantage device. Quantum hardware is still in early development, and larger instances will be achievable as devices with more qubits and better qubit interconnections will be available in the coming years, for instance the D-Wave Advantage2 system with 4800 qubits and a 20-way qubit connectivity.

Various graph labeling problems are based on the optimization of a distance between labels, with as many labels as vertices: for example, the 2D bandwidth problem (e.g., [32]), the bandwidth problem [17] and the cyclic bandwidth problem, as in this paper. All these problems can be modeled with table constraints and permutation/alldifferent constraints, and therefore can also be formulated in QUBO by applying the technique and methodology presented in this paper.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

References

1. Apt, K.R.: Principles of Constraint Programming. Cambridge University Press (2003)
2. Apt, K.R., Monfroy, É.: Automatic generation of constraint propagation algorithms for small finite domains. In: Jaffar, J. (ed.) Principles and Practice of Constraint Programming - CP'99, 5th International Conference, Alexandria, Virginia, USA, October 11-14, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1713, pp. 58–72. Springer (1999). https://doi.org/10.1007/978-3-540-48085-3‘¨5
3. Apt, K.R., Monfroy, É.: Constraint programming viewed as rule-based programming. Theory Pract. Log. Program. **1**(6), 713–750 (2001). https://doi.org/10.1017/S1471068401000072
4. Aramon, M., Rosenberg, G., Valiante, E., Miyazawa, T., Tamura, H., Katzgraber, H.G.: Physics-inspired optimization for quadratic unconstrained problems using a digital annealer. Frontiers in Physics **7**, 48 (2019)
5. Bhatt, S.N., Leighton, F.T.: A framework for solving VLSI graph layout problems. J. Comput. Syst. Sci. **28**(2), 300–343 (1984). https://doi.org/10.1016/0022-0000(84)90071-0
6. Boothby, K., Bunyk, P., Raymond, J., Roy, A.: Next-generation topology of d-wave quantum processors (2020), arXiv:2003.00133(quant-ph)
7. Bunyk, P.I., Hoskinson, E.M., Johnson, M.W., Tolkacheva, E., Altomare, F., Berkley, A.J., Harris, R., Hilton, J.P., Lanting, T., Przybysz, A.J., Whittaker, J.: Architectural considerations in the design of a superconducting quantum annealing processor. IEEE Transactions on Applied Superconductivity **24**(4), 1–10 (2014)
8. Choi, V.: Minor-embedding in adiabatic quantum computation: I. the parameter setting problem. Quantum Information Processing **7**(5), 193–209 (2008)
9. Choi, V.: Minor-embedding in adiabatic quantum computation: II. minor-universal graph design. Quantum Information Processing **10**(3), 343–353 (2011)
10. Chung, F.: Selected Topics In Graph Theory, vol. 3, chap. Labelings of Graphs. ACADEMIC PRESS LIMITED (1988)
11. Codognet, P.: Constraint solving by quantum annealing. In: Silla, F., Marques, O. (eds.) ICPP Workshops 2021: 50th International Conference on Parallel Processing, USA, August 9-12, 2021. pp. 25:1–25:10. ACM (2021)
12. Codognet, P.: Comparing QUBO models for quantum annealing: integer encodings for permutation problems. Int. Trans. Oper. Res. **32**(1), 18–37 (2025). https://doi.org/10.1111/ITOR.13471
13. Farhi, E., Goldstone, J., Gutmann, S., Lapan, J., Lundgren, A., Preda, D.: A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. Science **292**(5516), 472–475 (2001)
14. Farhi, E., Goldstone, J., Gutmann, S., Sipser, M.: Quantum computation by adiabatic evolution (2000), `https://arxiv.org/abs/quant-ph/0001106`
15. Gent, I.P., Miguel, I., Nightingale, P.: Generalised arc consistency for the alldifferent constraint: An empirical survey. Artificial Intelligence **172**(18), 1973–2000 (2008)
16. Glover, F.W., Kochenberger, G.A., Du, Y.: Quantum bridge analytics I: a tutorial on formulating and using QUBO models. Annals of Operations Research **314**, 141–183 (2022)
17. Harper, L.H.: Optimal assignments of numbers to vertices. Journal of the Society for Industrial and Applied Mathematics **12**(1), 131–135 (1964)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12

18. van Hoeve, W.J.: The alldifferent constraint: A survey. CoRR **cs.PL/0105015** (2001), https://arxiv.org/abs/cs/0105015
19. Hromkovič, J., Müller, V., Sýkora, O., Vrt'o, I.: On embedding interconnection networks into rings of processors. In: PARLE'92. Lecture Notes in Computer Science, vol. 605, pp. 53–62. Springer (1992)
20. Kadowaki, T., Nishimori, H.: Quantum annealing in the transverse Ising model. Physical Review E **58**, 5355–5363 (Nov 1998)
21. Lecoutre, C.: Optimization of simple tabular reduction for table constraints. In: Principles and Practice of Constraint Programming, 14th International Conference, CP 2008, Proceedings. LNCS, vol. 5202, pp. 128–143. Springer (2008)
22. Lecoutre, C.: Ace, a generic constraint solver (2023), https://arxiv.org/abs/2302.05405
23. Lin, Y.: The cyclic bandwidth problem. Systems Science and Mathematical Sciences **7** (01 1994)
24. Lin, Y.: Minimum bandwidth problem for embedding graphs in cycles. Networks **29**(3), 135–140 (1997)
25. Lucas, A.: Ising formulations of many NP problems. Frontiers in Physics **2** (2014)
26. Martí, R., Laguna, M., Glover, F., Campos, V.: Reducing the bandwidth of a sparse matrix with tabu search. European Journal of Operational Research **135**(2), 450–459 (2001)
27. McGeoch, C., Farré, P.: The Advantage system: Performance update (2021), Technical Report, D-Wave, 01-10-2021
28. McGeoch, C.C.: Adiabatic Quantum Computation and Quantum Annealing: Theory and Practice. Morgan & Claypool (2014)
29. Mohseni, N., McMahon, P., Byrnes, T.: Ising machines as hardware solvers of combinatorial optimization problems. Nature Reviews Physics **4**, 363–379 (2022)
30. Punnen, A.P.: The Quadratic Unconstrained Binary Optimization Problem. Springer International Publishing (2022)
31. Ren, J., Hao, J.K., Rodriguez-Tello, E., Li, L., He, K.: A new iterated local search algorithm for the cyclic bandwidth problem. Knowledge-Based Systems **203**, 106136 (2020)
32. Rodríguez-García, M.Á., Sánchez-Oro, J., Rodriguez-Tello, E., Monfroy, É., Duarte, A.: Two-dimensional bandwidth minimization problem: Exact and heuristic approaches. Knowl. Based Syst. **214**, 106651 (2021). https://doi.org/10.1016/J.KNOSYS.2020.106651
33. Rosenberg, A.L., Snyder, L.: Bounds on the costs of data encodings. Math. Syst. Theory **12**, 9–39 (1978). https://doi.org/10.1007/BF01776564
34. Rossi, F., van Beek, P., Walsh, T. (eds.): Handbook of Constraint Programming, Foundations of Artificial Intelligence, vol. 2. Elsevier (2006)
35. Schulte, C., Tack, G., Lagerkvist, M.Z.: Modeling and programming with Gecode. https://www.gecode.org/doc-latest/MPG.pdf (may 2019)
36. Tasseff, B., Albash, T., Morrell, Z., Vuffray, M., Lokhov, A., Misra, S., Coffrin, C.: On the emerging potential of quantum annealing hardware for combinatorial optimization. Journal of Heuristics (2024), published online, 02/08
37. Yap, R.H.C., Xia, W., Wang, R.: Generalized arc consistency algorithms for table constraints: A summary of algorithmic ideas. In: 34th AAAI Conference on Artificial Intelligence, NY, USA. pp. 13590–13597. AAAI Press (2020). https://doi.org/10.1609/AAAI.V34I09.7086
38. Yarkoni, S., Raponi, E., Bäck, T., Schmitt, S.: Quantum annealing for industry applications: introduction and review. Reports on Progress in Physics **85**(10), 104001 (2022)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_12