

Solving the Minimum Sum Coloring Problem: Alternative Models, Exact Solvers, and Metaheuristics

Yu Du,^{a,*} Fred Glover,^b Gary Kochenberger,^b Rick Hennig,^b Haibo Wang,^c Amit Hulanadageri^b

^aBusiness School, University of Colorado Denver, Denver, Colorado 80102; ^bEntanglement, Inc., Boulder, Colorado 80305; ^cSanchez School of Business, Texas A&M International University, Laredo, Texas 78040

*Corresponding author

Contact: yu.du@ucdenver.edu,  <https://orcid.org/0000-0001-6286-4814> (YD); fredwglover@yahoo.com (FG); gary@entanglement.ai (GK); rick@entanglement.ai (RH); hwang@tamiu.edu,  <https://orcid.org/0000-0002-8580-829X> (HW); amithm92@gmail.com (AH)

Received: November 6, 2022

Revised: July 23, 2023; January 14, 2024;
April 16, 2024

Accepted: May 3, 2024

Published Online in Articles in Advance:
May 30, 2024

<https://doi.org/10.1287/ijoc.2022.0334>

Copyright: © 2024 INFORMS

Abstract. The minimum sum coloring problem (MSCP), a well-known NP-hard (nondeterministic polynomial time) problem with important practical applications, has been the subject of several papers in recent years. Because of the computational challenge posed by these problems, most solution methods employed are metaheuristics designed to find high-quality solutions with no guarantee of optimality. Exact methods (like Gurobi) and metaheuristic solvers have greatly improved in recent years, enabling high-quality and often optimal solutions to be found to a growing set of MSCPs. Alternative model forms can have a significant impact on the success of exact and heuristic methods in such settings, often providing enhanced performance compared with traditional model forms. In this paper, we introduce several alternative models for MSCP, including the quadratic unconstrained binary problem plus (QUBO-Plus) model for solving problems with constraints that are not folded into the objective function of the basic quadratic unconstrained binary problem (QUBO) model. We provide a computational study using a standard set of test problems from the literature that compares the general purpose exact solver from Gurobi with the leading QUBO metaheuristic solver NGQ and a special solver called Q-Card that belongs to the QUBO-Plus class. Our results highlight the effectiveness of the QUBO and QUBO-Plus models when solved with these metaheuristic solvers on this test bed, showing that the QUBO-Plus solver Q-Card provides the best performance for finding high-quality solutions to these important problems.

History: Accepted by Pascal Van Hentenryck, Area Editor for Computational Modeling: Methods & Analysis.

Supplemental Material: The software that supports the findings of this study is available within the paper and its Supplemental Information (<https://pubsonline.informs.org/doi/suppl/10.1287/ijoc.2022.0334>) as well as from the IJOC GitHub software repository (<https://github.com/INFORMSJoC/2022.0334>). The complete IJOC Software and Data Repository is available at <https://informsjoc.github.io/>.

Keywords: minimum sum coloring problem • metaheuristics • combinatorial optimization • quadratic unconstrained binary optimization • constraint programming

1. Introduction

The standard vertex coloring problem (SVCP) is to find a proper K -coloring that consists of assigning at most K different colors to the vertices of a general graph G such that adjacent vertices receive different colors. The classic graph coloring problem seeks to find a legal coloring with the minimum number of colors (i.e., where K is the chromatic number). Metaheuristic methods for solving large-scale SVCPs have been proposed by Malaguti et al. (2008) and Verma et al. (2015). The minimum sum coloring problem (MSCP) at the focus of this paper is a well-known variation of the SVCP that aims to determine a proper K -coloring while minimizing the sum of the colors assigned to the vertices (Jin et al. 2017). Here and throughout the rest of the paper, the colors of a K -coloring are taken from the set $\{1, 2 \dots K\}$. This problem was introduced by Supowit (1987) and Kubicka and Schwenk (1989), and it has a wide array of applications in areas such as scheduling, resource allocation, and Very-large-scale integration design, as observed in Bar-Noy et al. (1998), Kochenberger et al. (2004), and Malafiejski (2004).

In common with the standard vertex coloring problem, MSCP is known to be NP (nondeterministic polynomial time) hard, and because of the inability of exact solvers to deal with these problems effectively (Sager and Lin 1991), most computational studies of MSCP are carried out using metaheuristic methods (Mehrotra and Trick 1996, Gualandi and Malucelli 2011).

Kochenberger et al. (2005) (and later, Wang et al. 2012, Silva et al. 2020) reformulated the standard linear model for MSCP as a quadratic unconstrained binary problem (QUBO) of the form

$$\text{QUBO: optimize } x_0 = x^t Q x,$$

where x is a vector of binary variables and Q is a symmetric matrix of coefficients. Following the standard QUBO modeling approach (see Glover et al. 2019), these papers recast the traditional graph coloring formulation, which is a linear, constrained 0/1 optimization problem, into an equivalent QUBO problem by folding all problem constraints, via quadratic penalties, into the Q matrix. This approach has proven to be successful, even as problem size scales upward.

In this paper, we also consider the QUBO model along with some extensions called QUBO-Plus models. The unifying nature of the QUBO model for modeling and solving many combinatorial problems, known for many years, has recently gained additional prominence because of the emergence of quantum computers. Emphasis is being placed on QUBO models because they are mathematically equivalent to the Ising spin glass problem in physics. Accordingly, quantum annealers, digital annealers, and other quantum and quantum-inspired computers have sought to solve QUBO models both for their unifying nature and for their connection to Ising problems.

We note, however, that applications arise where there is a computational advantage to retain some constraints separate from those incorporated into Q . We refer to such “constrained” QUBOs as QUBO-Plus models (Glover et al. 2020, 2022a, b). This class, which we describe more fully in the next section, augments the standard QUBO representation by introducing important constraints separately from the pure QUBO model, enabling them to be treated by algorithms designed to explicitly accommodate constraints.

This model separation and specialized algorithm approach makes it possible to solve these problems more efficiently and effectively than by attempting to create a “transformed” QUBO model that folds the constraints into the Q matrix. Pursuing this theme, we describe several QUBO-Plus models for the minimum sum coloring problem, demonstrating that QUBO-Plus models have a significant impact on the success of exact and heuristic methods with computational and conceptual advantages in such settings. We observe that Zheng et al. (2014) use a semidefinite program approach for quadratic programs with cardinality constraints and that Xia et al. (2019) solve nonconvex quadratic programs via linear integer programming. However, they do not provide a useful framework for our present application.

Our study also investigates the use of a QUBO-Plus metaheuristic solver called Q-Card, which efficiently handles the QUBO problem with a cardinality constraint without folding the constraint into the objective function via a penalty.

To determine the relative advantages of alternative models for the MSCP, our paper is organized as follows. In Section 2, we illustrate how to transform the MSCP into QUBO and QUBO-Plus models. Section 3 uses a simple example to illustrate the formulation of the alternative models. The state-of-the-art metaheuristic solver NGQ and the QUBO-Plus solver Q-Card are introduced in Section 4. Computational results are reported in Section 5, including comparisons with different models on exact and proposed metaheuristic solvers, demonstrating the superior performance obtained by using the QUBO-Plus model together with the Q-Card solver. The paper concludes with some observations in Section 6.

The contributions of this paper include the following:

- alternative models for both the traditional and QUBO models for the MSCP with explicit examples of model construction;
- computational experience highlighting the limitations of exact solvers, like Gurobi, for solving MSCP instances as problem size grows;
- computational comparisons clearly establishing the QUBO-Plus model with a cardinality constraint in conjunction with the new metaheuristic solver, Q-Card, as an extremely effective approach for solving large-scale instances of MSCP;
- empirical demonstration that the quadratic QUBO-Plus model we present for MSCP clearly outperforms the traditional linear model in terms of both solution quality and time to best as problem size grows; and
- new benchmark solutions for five large-scale MSCP instances: DSJC500-5, DSJC1000.1, flat1000_50, flat1000_660, and flat1000_76.

2. Problem Statement and Alternative Models

We consider the graph $G = (V, E)$ with n vertices and K possible colors. An assignment of colors to vertices is called a *proper coloring* if each vertex is assigned a color and if adjacent vertices have different colors. The

minimum sum coloring problem seeks to find a proper K -coloring such that the sum of all the colors over all vertices is minimized.

The standard model for MSCP (Sen et al. 1992) is a large 0/1 linear model that takes the following form, where the binary variable $x_{ik} = 1$ if color k is assigned to vertex i :

MSCP:

$$\begin{aligned} & \text{Minimize } f(x) = \sum_{i=1}^n \sum_{k=1}^K kx_{ik} \\ & \text{st.} \\ & \quad \text{(a) } \sum_{k=1}^K x_{ik} = 1 \quad \forall i \in (1, \dots, n) \\ & \quad \text{(b) } x_{ik} + x_{jk} \leq 1 \quad \forall (i, j) \in E, k \in (1, \dots, K) \\ & \quad \text{(c) } x_{ik} \text{ binary} \end{aligned} \tag{1}$$

The constraints of (a) ensure that each node gets a color, and the constraints of (b) require adjacent nodes to have different colors. The minimum value of the objective function, denoted by Σ , is the chromatic sum of the graph. MSCP is a computationally challenging problem that has been the focus of substantial effort over the past several decades devoted to developing various approximation models and algorithms (see, e.g., Benlic and Hao 2012, Wu and Hao 2012, Jin et al. 2014, Jin and Hao 2016, Wu et al. 2018).

As observed in Kochenberger et al. (2004) and as illustrated in Kochenberger et al. (2005), a linear model, like MSCP, can be recast into an equivalent quadratic model by adding quadratic penalties to the objective function as an alternative to explicitly enforcing the constraints in the traditional manner. For the constraints of (a), the penalties take the form

$$\text{Pen (a)} = P \left(\sum_{i=1}^n \left(\sum_{k=1}^K x_{ik} - 1 \right)^2 \right),$$

where P is a positive scalar penalty. For the constraints of (b), the penalties take the form

$$\text{Pen (b)} = P \left(\sum_{(i,j) \in E} \sum_{k=1}^K x_{ik} x_{jk} \right),$$

where again, P is a positive scalar penalty. These penalties, to be added to the original objective function to create a new objective function, will be either zero (corresponding to feasible solutions) or positive (corresponding to infeasible solutions). Consequently, an optimization process that attempts to minimize the new objective function will be drawn to feasible solutions to avoid the penalties associated with being infeasible, and any solution in which all penalties equal zero is a feasible solution to the original problem. It may be noted that QUBO penalty methods for incorporating constraints into the objective function differ from Lagrangian methods in several important respects. Lagrangian methods in general only yield a lower bound on the optimum objective function value for a minimization problem and an upper bound on the optimum objective function value for a maximization problem, and the solution to the Lagrangian is not assured to be feasible for the original problem. By contrast, with appropriate penalty weights, the penalty function we use to transform the (constrained) QUBO-Plus problem into the (unconstrained) QUBO problem causes the optimum QUBO-Plus objective function value to equal the optimum QUBO objective function value, and the optimal solutions for the QUBO problem are also optimal (and hence, feasible) for the QUBO-Plus problem.

Representing the original constraints by penalties in this fashion creates an alternative equivalent model in the form of a QUBO model, such as in the study by Wang et al. (2012). Such an approach has proven to be very effective in a wide variety of problems settings as documented in the survey papers of Kochenberger et al. (2004) and Glover et al. (2019).

When employing such penalties, the option exists to treat some constraints in the traditional manner (without the use of penalties). Such a “mixed” procedure produces a partial recasting into a QUBO model, giving rise to what we term a “QUBO-Plus” model. Both options are considered in the alternative models considered here.

In what follows, we introduce four alternative models for the MSCP, each solved by the exact method of Gurobi. In addition, Models 2 and 4 are solved by specially designed heuristic solvers. All four models have the

same decision variables but differ in the nature of their objective functions (linear versus quadratic) and in the number of constraints.

- Model 1. The standard linear model from the literature as represented by Equation (1) previously given. This model can be a large linear 0/1 optimization problem with many constraints depending on the number of nodes and the edge density.

- Model 2. This alternative formulation results from a full recasting of the traditional model into a QUBO model, where both sets of constraints, (a) and (b), are represented by penalties, yielding a pure QUBO model of the form

$$\text{Minimize } x^t Q x + C, \quad (2)$$

where the C , which equals $n \times P$, arises from the transformation process in each problem instance. Model 2 is exactly the QUBO model tested in the paper by Wang et al. (2012).

- Model 3. This model results from a partial recasting, where the adjacency constraints of (b) are represented by penalties, but the node assignment constraints of (a) are retained in their original, classical role. With the quadratic penalties included in the objective function, this problem takes the form

$$\begin{aligned} &\text{Minimize } x^t Q x \\ &\text{st.} \\ &\sum_{k=1}^K x_{ik} = 1 \text{ for all } i = 1, 2, \dots, n, \end{aligned} \quad (3)$$

where Q is a square, symmetric matrix incorporating the original objective function data from (Equation (1)) on the main diagonal and the adjacency data from constraint (b) in the off-diagonal elements.

Note that the Q matrices for Model 2 and Model 3 have the same size but differ in that all the constraint information is incorporated into the Q matrix for Model 2, whereas only the adjacency constraints are represented in the Q matrix for Model 3.

- Model 4. This model is a cardinality constrained version of the problem taking the form

$$\begin{aligned} &\text{Minimize } x^t Q x \\ &\text{st.} \\ &\sum_{i=1}^n \sum_{k=1}^K x_{ik} = n. \end{aligned} \quad (4)$$

The Q matrix for Model 4 is the same as the Q matrix for Model 2 except that the penalty P does not appear on the main diagonal, as illustrated in the next section. This is the only change from Model 2, except for the inclusion of the cardinality constraint (sum of the variables equals the number of nodes in the graph). For Model 4, the additive constant is zero (as in Model 3). We comment here that the purpose of the cardinality constraint is to ensure that the required number of the node assignments is made.

Note that Model 2 is a pure QUBO model, whereas Models 3 and 4 are examples of QUBO-Plus models. Although the four models are equivalent in that they have the same optimal solutions, they differ considerably in structure. Most notably, Model 1 is a linear model with generally many constraints depending on the number of edges in the graph. In contrast, Models 2, 3, and 4 are quadratic models with few, if any, constraints. All four models have the same decision variables.

An explicit example of all four models along with the construction of the Q matrices for Model 2, Model 3, and Model 4 is provided next in Section 3.

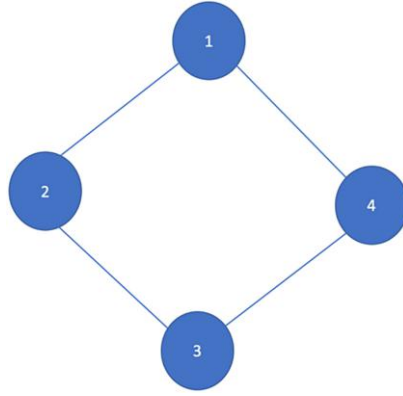
3. Example of All Four Models and Their Q Matrices

For purposes of illustration, we adopt the example used in the paper by Wang et al. (2012). Consider Figure 1, which is to be colored using two colors, with $n = 4$ and $K = 2$.

In these examples, we first present the standard linear model for MSCP followed by the three alternative quadratic models. To clearly demonstrate the development of the Q matrices, we present Model 3 immediately after Model 1 (before presenting Model 2 and Model 4) because of the relationship of the Q matrices for these models.

Model 2, the pure QUBO model, has a Q matrix populated with penalties from both of the constraints of (a) and (b), whereas the Q matrix for Model 3 has penalties derived solely from the adjacency constraints of (b). Thus, following Model 1, we first show the construction of the Q matrix for Model 3, and then, we demonstrate how it is modified to yield the further penalized Q matrix for Model 2. Given this, we demonstrate the construction of the Q matrix for Model 4, which results from that of Model 2 by simply removing the scalar penalty, P , from the main diagonal.

Figure 1. (Color online) An Example



3.1. Model 1

The classical linear MSCP model is

$$\begin{aligned}
 & \text{Minimize } f(x) = x_{11} + 2x_{12} + x_{21} + 2x_{22} + x_{31} + 2x_{32} + x_{41} + 2x_{42} \\
 & \text{st.} \\
 & \quad x_{11} + x_{12} = 1 \\
 & \quad x_{21} + x_{22} = 1 \\
 & \quad x_{31} + x_{32} = 1 \\
 & \quad x_{41} + x_{42} = 1 \\
 & \quad x_{11} + x_{21} \leq 1 \quad x_{12} + x_{22} \leq 1 \\
 & \quad x_{11} + x_{41} \leq 1 \quad x_{12} + x_{42} \leq 1 \\
 & \quad x_{21} + x_{31} \leq 1 \quad x_{22} + x_{32} \leq 1 \\
 & \quad x_{31} + x_{41} \leq 1 \quad x_{32} + x_{42} \leq 1 \\
 & \quad (x \text{ binary}).
 \end{aligned}$$

(As noted, Model 2 appears after Model 3.)

3.2. Model 3

Putting the original objective function and the adjacency constraints into the Q matrix, we have an equivalent constrained QUBO model of the form:

$$\begin{aligned}
 & \text{Minimize } x^t Q x \\
 & \text{st.} \\
 & \quad x_{11} + x_{12} = 1 \\
 & \quad x_{21} + x_{22} = 1 \\
 & \quad x_{31} + x_{32} = 1 \\
 & \quad x_{41} + x_{42} = 1 \\
 & \quad (x \text{ binary}),
 \end{aligned}$$

where the symmetric Q matrix is shown in Table 1.

3.3. Remarks

Remark 1. The position of the penalties in the Q matrix follows directly from the definition of Pen (b) given earlier. That is, the conventional constraint $x_{11} + x_{21} \leq 1$ is represented by the penalty $P x_{11} x_{21}$. The division by two is because of the symmetric nature of Q . Empty cells mean zero in the Q matrix.

Remark 2. Note that the coefficients of the original objective function appear on the main diagonal and that the nonzero off-diagonal elements represent the adjacency constraints.

Table 1. Q Matrix for Model 3

	x_{11}	x_{12}	x_{21}	x_{22}	x_{31}	x_{32}	x_{41}	x_{42}
x_{11}	1		$P/2$				$P/2$	
x_{12}		2		$P/2$				$P/2$
x_{21}	$P/2$		1		$P/2$			
x_{22}		$P/2$		2		$P/2$		
x_{31}			$P/2$		1		$P/2$	
x_{32}				$P/2$		2		$P/2$
x_{41}	$P/2$				$P/2$		1	
x_{42}		$P/2$				$P/2$		2

Remark 3. The scalar penalty P , usually taken to be a positive integer, is problem dependent. Generally, we want P to be large enough to ensure feasibility but no larger; otherwise, feasibility will dominate the original objective function. Surprisingly, small values often work just fine. However, feasibility requires that the quadratic terms be forced to zero, and this is always the ultimate test of whether P is large enough. For the MSCPs considered in this work, P equals the number of nodes in all models.

3.4. Model 2

Further populating the previous Q matrix with penalties representing the node assignment constraints (according to the definition of Pen (a) given earlier), we have the pure QUBO model:

$$\begin{aligned} &\text{Minimize } x^t Q x + C \\ &\text{st.} \\ &x \text{ binary,} \end{aligned}$$

where $C = n \times P$ and the Q matrix is now given in Table 2.

Because x_{ik} are binary variables, the relationship $x_{ik}^2 = x_{ik}$ holds and is exploited. Note that the penalties for the node assignment constraints modify the elements on the main diagonal and also show up in certain off-diagonal cells as indicated. The off-diagonal elements corresponding to the adjacency constraints are unchanged. The patterns in the Q matrices make it easy to develop these alternative models for MSCP. More general approaches for constructing Q matrices, suitable for other combinatorial optimization problems, can be found in the paper by Glover et al. (2020).

3.5. Model 4

As indicated in the previous section, this model is

$$\begin{aligned} &\text{Minimize } x^t Q x \\ &\text{st.} \\ &\sum_{i=1}^n \sum_{k=1}^K x_{ik} = n, \end{aligned}$$

with the Q matrix given in Table 3.

Here, the Q matrix is the same as that of Model 2 except that the scalar penalty P does not appear on the main diagonal. Because we are minimizing and because all elements of the Q matrix are nonnegative, a cardinality

Table 2. Q Matrix for Model 2

	x_{11}	x_{12}	x_{21}	x_{22}	x_{31}	x_{32}	x_{41}	x_{42}
x_{11}	$1 - p^a$	p^a	$P/2$				$P/2$	
x_{12}	p^a	$2 - p^a$		$P/2$				$P/2$
x_{21}	$P/2$		$1 - p^a$	p^a	$P/2$			
x_{22}		$P/2$	p^a	$2 - p^a$		$P/2$		
x_{31}			$P/2$		$1 - p^a$	p^a	$P/2$	
x_{32}				$P/2$	p^a	$2 - p^a$		$P/2$
x_{41}	$P/2$				$P/2$		$1 - p^a$	p^a
x_{42}		$P/2$				$P/2$	p^a	$2 - p^a$

^aNote that the penalties for the node assignment constraints modify the elements on the main diagonal and also show up in certain off-diagonal cells as indicated.

Table 3. Q Matrix for Model 4

	x_{11}	x_{12}	x_{21}	x_{22}	x_{31}	x_{32}	x_{41}	x_{42}
x_{11}	1	P^a	$P/2$				$P/2$	
x_{12}	P^a	2		$P/2$				$P/2$
x_{21}	$P/2$		1	P^a	$P/2$			
x_{22}		$P/2$	P^a	2		$P/2$		
x_{31}			$P/2$		1	P^a	$P/2$	
x_{32}				$P/2$	P^a	2		$P/2$
x_{41}	$P/2$				$P/2$		1	P^a
x_{42}		$P/2$				$P/2$	P^a	2

^aNote that the penalties for the node assignment constraints modify the elements on the main diagonal and also show up in certain off-diagonal cells as indicated.

constraint is appended to the model to ensure, given the penalty structure, that color assignments are made to the n nodes. (Else, the optimization would set all variables to zero.)

4. NGQ and Q-Card Solvers

The computational experience presented in Section 5 first gives a comparison of the performance of the four models based on the solver Gurobi, a commercial solver capable of optimizing modest-sized instances of all four models. These results are followed by further testing on Models 2 and 4 carried out using leading metaheuristic methods. Specifically, these additional results were obtained from Entanglement’s general purpose QUBO commercial solver, NGQ, for Model 2 and a specialized solver developed by the authors, called Q-Card, for the cardinality constrained formulation of Model 4 (Entanglement 2023).

NGQ is a quantum-inspired solver that combines different metaheuristic strategies, such as classical tabu search, diversification-driven tabu search, adaptive memory programming, scatter search, path relinking, crossover, and local search. This metaheuristic combination strategy has proven to be highly competitive in terms of both solution quality and computational efficiency relative to other approaches for solving the QUBO model.

With appropriate modifications, modern QUBO solvers, such as NGQ, can be customized to produce solutions that satisfy the explicit “Plus” constraints while optimizing the quadratic objective function. For such cases, we have produced the Q-Card solver to explicitly accommodate the constraints not represented by penalties while optimizing the quadratic objective function. The Q-Card solver is implemented in C++ using the C++ 17 standard. We compiled the code using GCC version 9.3.0. An abbreviated pseudocode for this solver is provided next.

Q-Card pseudocode is given in Algorithm 1.

Algorithm 1 (Pseudocode of Q-Card)

Input: Instances

Output: The best solution S_b

```

1:  $S_1, S_2, S_c, S_p \leftarrow \text{Init}()$ ;
2:  $S_b \leftarrow \text{saveBest}(S_1, S_2)$ ;
3:  $\text{generation} \leftarrow 0$ ;
4: while stop criterion is not satisfied do
5:   if  $S_1$  and  $S_2$  are similar then
6:      $S'_1, S'_2 \leftarrow \text{CrossoverExternal}(S_1, S_2)$ ; //refer to external link in Samorani et al. (2019)
7:   else if  $S'_1, S'_2 \leftarrow \text{CrossoverInternal}(S_1, S_2)$ ; //refer to internal link in Samorani et al. (2019)
8:   endif
9:    $S_1 \leftarrow \text{TabuSearch}(S'_1)$ ; //refer to the TPTS algorithm in Liu et al. (2020)
10:   $S_2 \leftarrow \text{TabuSearch}(S'_2)$ ;
11:   $S_c \leftarrow \text{saveBest}(S_1, S_2, S_c)$ ;
12:   $S_b \leftarrow \text{saveBest}(S_c, S_b)$ ;
13:  if  $\text{generation} \% \text{Iter} = 0$  then
14:     $S_1 \leftarrow S_p, S_p \leftarrow S_c$ 
15:     $S_c \leftarrow \text{Init}()$ ;
16:  end if
17:   $\text{generation} \leftarrow \text{generation} + 1$ 
18: end while
19: return  $S_b$ 

```

Note that in line 1, S_1 , S_2 , S_c , and S_p are initialized by selecting out n variables from $n \times k$ variables x_{ik} randomly. To elaborate the reason behind the operation of Step 1, note that S_c and S_p represent the best solution in the current generation and the previous generation, respectively. We initialize them as done in Step 1 to avoid adding conditions (such as if *generation* = 0, then xxx endif) in the loop. In fact, the initial values of S_c and S_p are usually much worse than the values of S_1 and S_2 , (which are optimized by the tabu search procedure). Thus, S_p can only affect S_1 and S_2 in *generation* 1. We adopt the procedure indicated in Step 1 because it increases the randomness of the algorithm. Also, in line 5, if there exist more than $0.8 \times k$ variables taking a value of one in both S_1 and S_2 , then S_1 and S_2 are considered similar as a basis for executing line 5. *Iter* in line 13 is a parameter to control the evolutionary algorithm, which can be set to five.

Algorithm 2 (Two-Phase Tabu Search (TPTS))

Input: Input solution S_i
Output: Output solution S_o
1: $S_o \leftarrow S_i$;
2: $S_b \leftarrow \text{saveBest}(S_1, S_2)$;
3: $\text{reset} \leftarrow 0$;
4: **while** $\text{reset} < \text{MaxIter}$ **do**
5: $S_o \leftarrow \text{Solution-based-TabuSearch}(S_i)$
6: $S_i \leftarrow \text{Attribute-based-TabuSearch}(S_o)$
7: $\text{reset} \leftarrow \text{reset} + 1$
8: **end while**
9: **return** S_o

For keeping the constraint in Model 4 satisfied, the Q-Card solver uses a constrained swap neighborhood structure. This swap move consists of exchanging two elements in $Z = \{x_{ij} \mid x_{ij} = 0\}$ and $O = \{x_{ij} \mid x_{ij} = 1\}$, respectively. Each element corresponds to a variable in Model 4. Then, the swap move will exchange values of one element in Z and one element in O . The elements to be swapped are chosen based on their objective function values $f(i) = \sum_{j \in O} x_{ij}$. Specifically, the smaller the value $f(i)$ of the element i in O (the greater the value $f(i)$ of element i in Z) is, the higher the probability of selecting that element. Moreover, to accelerate the search, the Q-Card solver restricts the neighborhood structure by candidate lists, which estimate the gain of removing or adding an element and select promising swap moves. The Q-Card solver additionally combines a solution-based tabu search and an attribute-based tabu search in a local search procedure to refine the search. Interested readers are referred to Liu et al. (2020) for details.

Although any QUBO-Plus model can be converted to an equivalent QUBO model by employing additional penalties, the structure of some constraints gives the Q-Card solver a significant computational advantage over QUBO solvers that are incapable of handling these constraints separately. This computational advantage is illustrated for the test problems considered here as seen in the results given in the following section, which show that the Q-Card solver is particularly valuable for solving larger problems.

5. Computational Testing

Except as noted otherwise, all results were obtained on a desktop personal computer with an Intel Core i7-7700k processor at 4.20 GHz and 32 GB of random access memory. Our computational testing of Models 1, 2, 3, and 4 was carried out on the well-known set of Center for Discrete Mathematics and Theoretical Computer Science problems used in the study by Wang et al. (2012) as well as some larger problems from Jin et al. (2017). All the implementations are available in the online appendix from Du et al. (2024).

Table 4 presents the problem characteristics for the problems used by Wang et al. (2012) and Jin et al. (2017). Columns 1–3 in Table 4 give the instance names along with the vertex number $|V|$ and edge number $|E|$ of the graphs. Column 4 in Table 4 presents the best upper bounds ub^* from the previous literature (Jin et al. 2017, Donne et al. 2021). We bolded the cases where the best upper bound ub^* is equal to the chromatic sum Σ . Columns 5 and 6 in Table 4 show the number of colors K to be used and the number of variables N in the resulting models.

5.1. Model Comparisons Using the Gurobi Exact Solver

Exact methods, like Gurobi Optimization, LLC (2022), have improved greatly in recent years, opening the door for possible application in problem settings typically reserved for metaheuristics methods. In this subsection, we report on the applicability of using Gurobi version 9.5 on the four alternative model forms for MSCP.

Table 4. Problem Characteristics

Instances	V	E	ub*	K	N
myciel3	11	20	21	6	66
myciel4	23	71	45	7	161
myciel5	47	236	93	8	376
myciel6	95	755	189	9	855
myciel7	191	2,360	381	10	1,910
anna	138	493	276	13	1,794
david	87	406	237	13	1,131
games120	120	638	443	10	1,200
huck	74	301	243	13	962
jean	80	254	217	12	960
miles250	128	387	325	10	1,280
miles500	128	1,170	705	22	2,816
queen5.5	25	160	75	7	175
queen6.6	36	290	138	9	324
queen7.7	49	476	196	9	441
queen8.8	64	728	291	10	640
DSJC125.1	125	736	326	8	1,000
DSJC125.5	125	3,891	1,012	22	2,750
DSJC125.9	125	6,961	2,503	50	6,250
DSJC250.1	250	3,218	970	12	3,000
DSJC250.5	250	15,668	3,210	35	8,750
DSJC250.9	250	27,897	8,277	80	20,000
DSJC500.1	500	12,458	2,836	16	8,000
DSJC500.5	500	62,624	10,886	56	28,000
DSJC1000.1	1,000	49,629	8,991	29	29,000
flat1000_50	1,000	245,000	25,500	52	52,000
flat1000_60	1,000	245,830	30,100	64	64,000
flat1000_76	1,000	246,708	37,164	84	84,000

Note. The cases where the best upper bound ub* is equal to the chromatic sum Σ are in bold.

The results presented in Table 5 were obtained from Gurobi version 9.5 using all default values, except with the presolve linearization option set to zero. Each problem was given a time limit of two hours (7,200 seconds), and for each problem instance, we report the “time to best” for each model as well of the percentage gap to ub* at 7,200 seconds. We reported the best solution we found within the time allowed, representing an upper bound in cases where we did not allow enough time to prove optimality. A solution is noted in Table 5 if it is the best in its row (i.e., compared with the other models in the table). For few of the problems, Gurobi failed to give a solution within the time limit, and hence, these are marked by —.

Table 5 gives a comparison of the performance of Gurobi on the linear model (Model 1) and its performance on the three alternative quadratic formulations for the 28 test problems considered. (Gurobi uses a different proprietary solver for the quadratic models than it uses for the linear model.) The first 19 problems (myciel3 to DSJC125.9) in Table 5 produce small instances of all models, and for these small problems, Model 1 dominates the quadratic models in terms of both solution quality and time to best. For these 19 problems, Model 3 was a close second in terms of solution quality, but it generally lagged behind the application of Model 1 in terms of time to best and gap performance.

For these same problems, the performance of Model 3 was followed by that of Model 4 and then, by Model 2, which turned in the worst results. All told, the numbers of times the four models were able to produce best results among others in the time limit of two hours were 19, 5, 17, and 8, respectively, for the 19 small instances. In terms of matching best-known solutions, the results were 17, 5, 13, and 8, respectively.

On the nine larger problems (DSJC250.1 to flat1000_76) in Table 5, Gurobi was unable to perform well using any of the models, and the relative gap in its performance between Model 1 and the quadratic models narrowed considerably. For these problems, both Model 1 and Model 3 using Gurobi were able to produce solutions for only three of the nine problems within the two-hour time limit. In a head-to-head comparison, Model 3 showed a slight edge over Model 1. Models 4 and 2, in that order, rounded out the comparisons.

We note that this performance pattern, where a linear model outperforms the equivalent quadratic alternative model on small- to medium-sized problems but lags behind in performance on larger problems, has been reported for other classes of problems as well. For instance, see the recent paper by Du et al. (2022).

Table 5. Computational Results of Model 1, Model 2, Model 3, and Model 4 Using Gurobi

Problem identification	Model 1			Model 2			Model 3			Model 4			ub*
	Solution	Time to best	Gap (%)	Solution	Time to best	Gap (%)	Solution	Time to best	Gap (%)	Solution	Time to best	Gap (%)	
myciel3	21 ^a	0	0	21 ^a	0	0	21 ^a	0	0	21 ^a	0	0	21
myciel4	45 ^a	0	0	45 ^a	0	0	45 ^a	0	0	45 ^a	5	0	45
myciel5	93 ^a	0	0	104	0	0	93 ^a	8	0	93 ^a	6,524	0	93
myciel6	189 ^a	27	0	215	0	13.76	189 ^a	13	0	215	188	13.76	189
myciel7	381 ^a	176	0	438	3	14.96	392	978	2.89	418	5,137	9.711286	381
anna	276 ^a	0	0	282	5,023	2.17	276 ^a	4	0	276 ^a	17	0	276
david	237 ^a	0	0	252	4,950	6.33	237 ^a	4	0	237 ^a	7	0	237
games120	443 ^a	0	0	449	323	1.35	443 ^a	708	0	446	195	0.68	443
huck	243 ^a	0	0	243 ^a	4,180	0	243 ^a	0	0	243 ^a	14	0	243
Jean	217 ^a	0	0	218	2,054	0.46	217 ^a	1	0	217 ^a	4	0	217
miles250	325 ^a	0	0	329	1,277	1.23	325 ^a	3	0	327	20	0.62	325
miles500	705 ^a	1	0	708	8	0.43	716	2,762	1.56	745	7,077	5.67	705
queen5.5	75 ^a	0	0	75 ^a	0	0	75 ^a	0	0	75 ^a	0	0	75
queen6.6	138 ^a	0	0	138 ^a	802	0	138 ^a	2	0	148	5,176	7.25	138
queen7.7	196 ^a	0	0	229	15	16.84	196 ^a	14	0	220	5,585	12.24	196
queen8.8	291 ^a	64	0	325	84	11.68	319	559	9.62	325	4,453	11.68	291
DSJC125.1	326 ^a	6,439	0	356	7,007	9.20	350	1,127	7.36	340	105	4.29	326
DSJC125.5	1,219	4,112	20.45	1,277	6,424	26.19	1,203 ^a	4,196	18.87	1,248	2,530	23.32	1,012
DSJC125.9	2,665 ^a	6,830	6.47	2,969	4,638	18.62	2,759	5,582	10.23	3,101	6,968	23.89	2,503
DSJC250.1	1,141	6,022	17.63				1,122 ^a	7,124	15.67	1,139	7,200	17.42	970
DSJC250.5							4,129 ^a	6,204	28.63	8,250	7,061	157.01	3,210
DSJC250.9													8,277
DSJC500.1	3,766	1,887	32.79				3,493 ^a	6,542	23.17	9,224	7,146	225.25	2,836
DSJC500.5													10,886
DSJC1000.1	12,928 ^a	891	—										8,991
flat1000_50													25,500
flat1000_60													30,100
flat1000_76													37,164

Note. For few of the problems, Gurobi failed to give a solution within the time limit, and hence, these are marked by —. The cases where best solution = Opt(Or ub*) are in bold.
^aThe solution is the best in its row (i.e., compared with the other models in the table).

In Section 5.2, we see results illustrating improved performance derived from Models 2 and 4, models that did not perform well in Table 5, obtained by applying leading metaheuristics rather than Gurobi.

5.2. Performance of Metaheuristic Methods

The results in Table 5 showcase the limited applicability that exact solvers, like Gurobi, have on MSCP instances as the problem size scales upward. The last five problem instances in our test bed have 28,000–84,000 binary variables, creating models that challenge the success of exact methods given reasonable run time limits. For such large-scale instances, modern metaheuristic methods, like NGQ and Q-Card, are better suited to deliver high-quality solutions within reasonable time restrictions. Such improved performance can be seen in Table 6.

For ease of comparison, we include the Gurobi exact solver results for Models 2 and 4 along with the metaheuristic results for these two models in Table 6. The earlier results for the standard linear model are also included to facilitate further comparisons. Because of the large size of the last five problems instances, a run time limit of 20 hours for each model/solver combination was imposed. As with Table 5, a time limit of two hours was imposed on the first 23 problems. Because NGQ and Q-Card are metaheuristic approaches, we report best solution, mean results plus standard deviations, and average time to the best in Table 6. We run 10 times for small instances and 5 times for the last large instance.

Again, we see that all the models and solvers did well on the first 19 problems. These problems are small and relatively easy to solve. The performance edge of the metaheuristic solvers is seen on the larger instances.

Looking at the two sets of results obtained from Model 4, Table 6 shows that the Q-Card results dominate the Gurobi results by a wide margin. To a lesser extent but still convincingly, the results shown for Model 2 using NGQ also dominate those for Model 2 using Gurobi. Both the Model 4 results using Q-Card and the Model 2 results using NGQ dominate the Model 1 results.

Looking across all columns, Table 6 reveals that Model 4 using Q-Card quickly found best solutions for 26 of the 28 problems, narrowly missing on the other 2 problems. Model 2 using NGQ also provided high-quality solutions for many problems but failed to match the Model 4 results using Q-Card on 12 instances, coming up considerably short on several problems. With one exception, Model 1 using Gurobi, even with the expanded run time limit of 20 hours, was unable to produce solutions for the larger instances.

Note that we recall that the best upper bound ub^* is reported in the literature on MSCPs. The best upper bounds are found by Breakout Local Search (Benlic and Hao 2012), Memetic Algorithm for Sum Coloring (Jin et al. 2014), MDS(5) +LS (Helmar and Chiarandini 2011), MA-MSCP (Memetic Algorithm for Minimum Sum Coloring Problem) (Moukrim et al. 2013), and hybrid evolutionary search algorithm (Jin and Hao 2016). Some results by using Model 4 with Q-Card deviate slightly from the best upper bounds reported in the literature. However, the Q-Card method is a general method for the QUBO-Plus model with a cardinality constraint, whereas the approaches that give the best solutions known are specialized methods for solving the MSCP.

Overall, we see that the quadratic models, particularly the QUBO-Plus model in this case, with suitable metaheuristic solvers can be successfully employed to solve large instances of MSCPs, greatly extending the size of problem instances capable of being solved in a reasonable amount of time.

6. Conclusions and Future Work

In this paper, we analyzed four different models for the NP-hard minimum sum coloring problem. All four models are equivalent, and the transformations among those models do not change the underlying computational complexity. In this analysis, we examine the application of three different solvers, an exact method and two metaheuristic methods, and we show how their performance varies depending on the model employed and the problem size.

Our study constitutes the first investigation of solving MSCP via QUBO-Plus models using the metaheuristic solver Q-Card, demonstrating how the MSCP can be recast into the QUBO and QUBO-Plus models and explaining how these models can, respectively, be solved with the NGQ and Q-Card algorithms without any tuning or adaptation to handle the special structure of MSCP. Our experimental evaluations on a set of 28 benchmark instances show that the Q-Card approach for the general QUBO-Plus model is able to reach solutions that are unmatched for larger problem instances by using the other methods and models.

This study also provides optimal solutions for 17 instances of smaller size via the Gurobi Mixed Integer Programming solver applied to an Linear programming model. Although Gurobi runs very rapidly on these test problems and it succeeds in solving to optimality, we find that it fails to obtain solutions that match the quality of those obtained by Q-Card in 11 of the 28 instances.

This demonstration that a QUBO-Plus model, with its special handling of special constraints, can offer significant advantages for solving larger problems motivates a future study devoted to developing and testing new

Table 6. Comparing Exact and Metaheuristic Solvers

Problem identification	Model 1 exact			Model 2 exact			Model 2 NGQ			Model 4 Gurobi exact			Model 4 Q-Card		
	Solution	Time to best		Solution	Time to best		Best solution	Average solution (std)	Average time to best	Solution	Time to best		Best solution	Average solution (std)	Average time to best
myciel3	21 ^a	0		21 ^a	0		21 ^a	21 (0) ^a	0.0002	21 ^a	0		21 ^a	21 (0) ^a	0.001
myciel4	45 ^a	0		45 ^a	228		45 ^a	45 (0) ^a	0.007	45 ^a	5		45 ^a	45 (0) ^a	0.0176
myciel5	93 ^a	0		93 ^a	0		93 ^a	93 (0) ^a	0.1278	93 ^a	6524		93 ^a	93 (0) ^a	0.1764
myciel6	189 ^a	27		189 ^a	0		189 ^a	190.8 (1.5491)	2.5114	215	188		189 ^a	189 (0) ^a	16.3272
myciel7	381 ^a	176		381 ^a	3		381 ^a	386.8 (8.9084)	6.9332	418	5137		382	382.4 (0.5163)	110.699
anna	276 ^a	0		276 ^a	5,023		276 ^a	276 (0) ^a	4.976	276 ^a	17		276 ^a	276 (0) ^a	88.4444
david	237 ^a	0		237 ^a	4,950		237 ^a	237 (0) ^a	6.6194	237 ^a	7		237 ^a	237 (0) ^a	31.4688
games120	443 ^a	0		443 ^a	323		443 ^a	446.4 (3.3730)	169.511	446	195		443 ^a	443 (0) ^a	100.081
huck	243 ^a	0		243 ^a	4,180		243 ^a	243 (0) ^a	0.137	243 ^a	14		243 ^a	243 (0) ^a	0.0008
Jean	217 ^a	0		217 ^a	2,054		217 ^a	217 (0) ^a	1.5828	217 ^a	4		217 ^a	217.4 (0.5163)	6.8466
miles250	325 ^a	0		329	1,277		326	326.8 (3.0594)	573.5232	327	20		325 ^a	326.6 (0.8432)	568.1484
miles500	705 ^a	1		708	8		711	712.4 (16.4600)	1.750	745	7,077		712	715.2 (2.8596)	126.6716
queen5.5	75 ^a	0		75 ^a	0		75 ^a	75 (0) ^a	0.0132	75 ^a	0		75 ^a	75 (0) ^a	0.0002
queen6.6	138 ^a	0		138 ^a	802		138 ^a	138 (0) ^a	1.3796	148	5,176		138 ^a	138 (0) ^a	0.1282
queen7.7	196 ^a	0		229	15		196 ^a	196 (0) ^a	26.4266	220	5,585		196 ^a	196 (0) ^a	0.018
queen8.8	291	64		325	84		291 ^a	291 (0) ^a	879.5624	325	4,453		291 ^a	291 (0) ^a	91.712
DSJC125.1	326	6,439		356	7,007		331	333.6 (3.3065)	847.4676	340	105		326 ^a	326.2 (0.4216)	290.9694
DSJC125.5	1,219	4,112		1,277	6,424		1,032	1,038.6 (9.1796)	5,030.024	1,248	2,530		1,022 ^a	1,030.8 (1.5491)	2,257.731
DSJC125.9	2,665	6,830		2,969	4,638		2,673	2,792.0 (57.1625)	3,709.4281	3,101	6,968		2,510 ^a	2,524.6 (2.0655)	3,911.9098
DSJC250.1	1,141	6,022		—	—		1,030	1,047.8 (12.7958)	1,480.23	1,139	7,200		985 ^a	995.6 (3.1692)	3,657.6842
DSJC250.5	—	—		—	—		3,447	3,812.6 (746.5221)	4,076.9	8,250	7,061		3,320 ^a	3,343.4 (16.5274)	2,909.0968
DSJC250.9	—	—		—	—		11,606	11,618.0 (88.8869)	1,394.6504	—	—		8,419 ^a	8,519.0 (16.0000)	3,837.1284
DSJC500.1	3,766	1,887		—	—		3,184	3,187.8 (42.8298)	4,944.042	9,224	7,146		2,976 ^a	2,998.6 (12.6947)	5,392.181
DSJC500.5(20 h)	—	—		—	—		17,129	17,350	11,383.99	—	—		11,590 ^a	11,890	18,433.919
DSJC1000.1(20 h)	12,646	3,9788		—	—		10,588	10,600	57,930.73	—	—		10,073 ^a	10,073	17,707.639
flat1000_50(20 h)	—	—		—	—		—	—	—	—	—		25,500 ^a	25,500	14,633.584
flat1000_60(20 h)	—	—		—	—		—	—	—	—	—		20,289 ^a	30,289	69,555.539
flat1000_76(20 h)	—	—		—	—		—	—	—	—	—		37,167 ^a	37,167	47,122.848

Notes. For few of the problems, Gurobi failed to give a solution within the time limit, and hence, these are marked by —, std, standard deviation. The cases where best solution = Opt(Or ub*) are in bold.

^aThe solution is the best in its row (i.e., compared with the other models in the table).

methods for additional QUBO-Plus models that involve alternative sets of constraints and embrace an expanded range of applications.

References

- Bar-Noy A, Bellareb M, Halldorsson MM, Shachnai H, Tamir T (1998) On chromatic sums and distributed resource allocation. *INFORMS J. Comput.* 140(2):183–202.
- Benlic U, Hao JK (2012) A study of breakout local search for the minimum sum coloring problem. Bui LT, Ong YS, Hoai NX, Ishibuchi H, Suganthan PN, eds. *Simulated Evolution and Learning. SEAL 2012, Lecture Notes in Computer Science*, vol. 7673 (Springer, Berlin), 128–137.
- Donne DD, Furini F, Malaguti E, Calvo RW (2021) A branch-and-price algorithm for the minimum sum coloring problem. *Discrete Appl. Math.* 303:39–56.
- Du Y, Glover F, Kochenberger G, Hennig R, Wang H, Hulandageri A (2024) Solving the minimum sum coloring problem: Alternative models, exact solvers, and metaheuristics. <https://doi.org/10.1287/ijoc.2022.0334.cd>, <https://github.com/INFORMSJoC/2022.0334>.
- Du Y, Kochenberger G, Glover F, Wang H, Lewis M, Xie WH, Tsuyuguchi T (2022) Solving clique partitioning problems: A comparison of models and commercial solvers. *Internat. J. Inform. Tech. Decision Making* 21(1):59–81.
- Entanglement (2023) Breaking the rules. Accessed July 20, 2022, <https://www.entanglement.com>.
- Glover F, Kochenberger G, Du Y (2019) Quantum bridge analytics I: A tutorial on formulating and using QUBO models. *4OR* 17:335–371.
- Glover F, Kochenberger G, Hennig R, Du Y (2022a) Quantum bridge analytics I: A tutorial on formulating and using QUBO models. *Ann. Oper. Res.* 314:141–183.
- Glover F, Kochenberger G, Ma M, Du Y (2020) Quantum bridge analytics II: QUBO-Plus, network optimization and combinatorial chaining for asset exchange. *4OR* 18:387–417.
- Glover F, Kochenberger G, Ma M, Du Y (2022b) Quantum bridge analytics II: Combinatorial chaining for asset exchange. *Ann. Oper. Res.* 314:185–212.
- Gualandi S, Malucelli F (2011) Exact solution of graph coloring problems via constraint programming and column generation. *INFORMS J. Comput.* 24(1):81–100.
- Gurobi Optimization, LLC (2022) Gurobi Optimizer Reference Manual. Accessed August 10, 2022, <https://www.gurobi.com>.
- Helmar A, Chiarandini M (2011) A local search heuristic for chromatic sum. *9th Metaheuristics Internat. Conf.*, (Udine, Italy).
- Jin Y, Hao JK (2016) Hybrid evolutionary search for the minimum sum coloring problem of graphs. *Inform. Sci.* 352–353:15–34.
- Jin Y, Hamiez JP, Hao JK (2017) Algorithms for the minimum sum coloring problem: A review. *Artificial Intelligence Rev.* 47:367–394.
- Jin Y, Hao JK, Hamiez JP (2014) A memetic algorithm for the minimum sum coloring problem. *Comput. Oper. Res.* 43(3):318–327.
- Kochenberger G, Glover F, Alidaee B, Rego C (2004) A unified modeling and solution framework for combinatorial optimization problems. *OR Spectrum* 26:237–250.
- Kochenberger G, Glover F, Alidaee B, Rego C (2005) An unconstrained quadratic binary programming approach to the vertex coloring problem. *Ann. Oper. Res.* 139(1):229–241.
- Kubicka E, Schwenk AJ (1989) An introduction to chromatic sums. *Proc. 17th Conf. ACM Annual Comput. Sci. Conf. CSC '89* (ACM, New York), 39–45.
- Liu X, Chen J, Wang M, Wang Y, Su Z, Lü Z (2020) A two-phase tabu search based evolutionary algorithm for the maximum diversity problem. *Discrete Optim.* 2020:100613.
- Malafiejski M (2004) Sum coloring of graphs. Kubale M, ed. *Graph Colorings*, Contemporary Mathematics No. 352 (American Mathematical Society, Providence, RI), 55–65.
- Malaguti E, Monaci M, Toth P (2008) A metaheuristic approach for the vertex coloring problem. *INFORMS J. Comput.* 20(2):302–316.
- Mehrotra A, Trick MA (1996) A column generation approach for graph coloring. *INFORMS J. Comput.* 8(4):344–354.
- Moukrim A, Sghiouer K, Lucet C, Li Y (2013) Upper and lower bounds for the minimum sum coloring problem. Working paper, Université de technologie de Compiègne, Compiègne, France.
- Sager TJ, Lin SJ (1991) A pruning procedure for exact graph coloring. *ORSA J. Comput.* 3(3):226–230.
- Samorani M, Wang Y, Lv Z (2019) Clustering-driven evolutionary algorithms: An application of path relinking to the quadratic unconstrained binary optimization problem. *J. Heuristics* 25(4):629–642.
- Sen A, Deng H, Guha S (1992) On a graph partition problem with application to VLSI layout. *Inform. Processing Lett.* 43(2):87–94.
- Silva C, Aguiar A, Lima PMV (2020) Mapping graph coloring to quantum annealing. *Quantum Machine Intelligence* 2:16.
- Supowit KJ (1987) Finding a maximum planar subset of a set of nets in a channel. *IEEE Trans. Comput. Aided Design Integrated Circuits Systems* 6(1):93–94.
- Verma A, Buchanan A, Butenko S (2015) Solving the maximum clique and vertex coloring problems on very large sparse networks. *INFORMS J. Comput.* 27(1):164–177.
- Wang Y, Hao JK, Glover F, Lu Z (2012) Solving the minimum sum coloring problem via binary quadratic programming. Preprint, submitted April 22, <https://arxiv.org/abs/1304.5876>.
- Wu Q, Hao JK (2012) An effective heuristic algorithm for sum coloring of graphs. *Comput. Oper. Res.* 39(7):1593–1600.
- Wu Q, Zhou Q, Jin Y, Hao JK (2018) Minimum sum coloring for large graphs with extraction and backward expansion search. *Appl. Soft Comput.* 62:1056–1065.
- Xia W, Vera JC, Zuluaga LF (2019) Globally solving nonconvex quadratic programs via linear integer programming techniques. *INFORMS J. Comput.* 32(1):40–56.
- Zheng X, Sun X, Li D (2014) Improving the performance of MIQP solvers for quadratic programs with cardinality and minimum threshold constraints: A semidefinite program approach. *INFORMS J. Comput.* 26(4):690–703.