A HYBRID HEURISTIC FOR THE OVERLAPPING CLUSTER EDITING PROBLEM

Guilherme Oliveira Chagas
Instituto Nacional de Pesquisas Espaciais
Av. dos Astronautas, 1758, 12245-970, São Jóse dos Campos, São Paulo, Brasil.
guilherme.chagas@inpe.br

Luiz Antonio Nogueira Lorena
Instituto Nacional de Pesquisas Espaciais
Av. dos Astronautas, 1758, 12245-970, São Jóse dos Campos, São Paulo, Brasil.
luiz.lorena@inpe.br

Rafael Duarte Coelho dos Santos
Instituto Nacional de Pesquisas Espaciais
Av. dos Astronautas, 1758, 12245-970, São Jóse dos Campos, São Paulo, Brasil.
rafael.santos@inpe.br

ABSTRACT

Clustering problems are common in several fields of science. In the graph theory context, a clustering can be made by transforming a graph into a disjoint union of complete subgraphs (clusters) by performing changes to the edge set. This problem is known as the cluster editing problem. However, there are situations when clusters may overlap, i.e., they can share vertices. There are few practical studies regarding the overlapping cluster editing problem in the literature. Then in this paper we propose a hybrid heuristic for the overlapping cluster editing problem. Our hybrid heuristic is based on coupling two metaheuristics, to generate solutions for the cluster editing problem, and a mixed-integer linear program, also introduced in this work, that is solved by using the cluster editing solutions as input. Experimental results show that the proposed hybrid heuristic is able to find good-quality solutions.

KEYWORDS. Cluster editing, overlapping clustering, hybrid heuristic.

OC - Otimização Combinatória.

https://proceedings.science/p/85411?lang=pt-br

1. Introduction

Clustering problems are found in several science areas such as bioinformatics [Ben-Dor et al., 1999; Sharan et al., 2003; Rahmann et al., 2007; Böcker et al., 2009], image processing [Wu and Leahy, 1993], computer vision, multimedia data analysis, facility location problems, data compression, marketing, pattern recognition and machine learning [Bansal et al., 2004; Shamir et al., 2004; Demaine et al., 2006; Aggarwal, 2013]. On an overview, considering a set of elements, the main objective of a clustering problem is to partition them into subsets according to the values of a given metric. Thus, elements belonging to the same subset have metric values more similar than elements belonging to different subsets. These subsets are also known as clusters.

Several methods and algorithms have been proposed over the years to perform data clustering and there is no method or technique that can be used to solve all clustering problems optimally [Xu and Wunsch II, 2005]. However, graph theory is a widely used approach to model these problems and obtain good quality clusterings [Shamir et al., 2004; Guo et al., 2009]. From a graph theoretic point of view, given an unweighted graph, a clustering problem can be modeled by considering elements as vertices and making them adjacent if the similarity value between the elements is larger (or smaller) than a threshold. Then, a cluster can be interpreted as vertices that are highly connected, that is, a dense subgraph or a complete subgraph (clique).

In this context, a clustering can be obtained by adding and deleting edges of an input graph so that it becomes a disjoint union of cliques. This is a well-known problem in combinatorial optimization referred to as *cluster editing problem*, also known as *correlation clustering problem* [Bansal et al., 2004]. The cluster editing problem is, probably, the most studied edge modification problem [Fellows et al., 2011] and has applications, mainly, in gene expression [Ben-Dor et al., 1999; Matsuda et al., 1999; Chesler et al., 2005; Jiang and Pei, 2009]. Transforming an input graph into a disjoint union of cliques by the minimum number of edge modifications is a NP-hard problem which was proved, independently, by Delvaux and Horsten [2004], Shamir et al. [2004] and Bansal et al. [2004]. Then, several heuristics, approximation algorithms and theoretical studies have been carried out in the literature for this problem. For a briefly survey of practical and theoretical results on this subject, see Il'ev et al. [2016].

In some problems, however, there are situations when clusters may overlap, i.e., they can share vertices. For example, in social networks users can be assigned to more than one cluster [Bonchi et al., 2013]. As Fellows et al. [2011] explain, the concept of the cluster editing problem fails to model these problems where clusters may overlap and has been criticized in the literature [Dehne et al., 2006; Maiza et al., 2016]. Therefore, it is necessary to relax the definition of the cluster editing problem that allows overlapping between clusters.

Overlapping clustering applications can be found in many areas. For example, Andersen et al. [2012] applied the definition of overlapping cluster editing in distributed computing. Benelallam et al. [2016] used overlapping clustering concept applied to distributed model transformations. Also, Maiza et al. [2016] and Pérez-Suárez et al. [2013] cite other areas where the overlapping cluster editing problem is important, such as image and video processing. In addition, variations of the overlapping cluster editing problem were presented by Damaschke [2010], Fellows et al. [2011] e Bonchi et al. [2013].

As mentioned previously, the cluster editing problem is hard. Thus, exact methods are only practical in instances with few vertices. For larger ones, heuristics are used to generate solutions at a reasonable computational cost. However, the solution quality is not guaranteed. Hence, hybrid heuristics, also known as matheuristics [Maniezzo et al., 2009], are alternatives to produce good-quality solutions with reasonable computation cost. Hybrid heuristics are formed by coupling exact methods and metaheuristics and have been used successfully in current combinatorial optimization research [Pereira et al., 2015; Fonseca et al., 2016; Wang et al., 2017].

There are few practical studies regarding the overlapping cluster editing problem in the literature. To the best of our knowledge, there are no heuristics or hybrid heuristics for this problem.

https://proceedings.science/p/85411?lang=pt-br

Then, in this paper, we propose a hybrid heuristic for the overlapping cluster editing problem. Our hybrid heuristic is based on coupling two metaheuristics, to generate solutions for the cluster editing problem, and a mixed-integer linear program, also introduced in this work, that is solved by using the cluster editing solutions as input.

The remainder of this work is organized as follows. The necessary notation and problems definitions are presented in Section 2. The proposed hybrid heuristic is detailed in Section 3. Section 4 shows the tests results. Our concluding remarks and discussing future work are presented in Section 5.

2. Mathematical notation

Let $G = (V, E)$ be a simple, undirected and unweighted graph, where V is the set of vertices, E is the set of edges, $n = |V|$ and $m = |E|$. Two vertices $v, u \in V$ are adjacent if, and only if, $(v, u) \in E$. A graph G is *complete* if, and only if, $\forall v \in V$ and $\forall u \in V$, where $v \neq u$, $(v, u) \in E$. In complete a graph, $m = \frac{n \cdot (n-1)}{2}$. A subgraph of G *induced* by a subset of vertices $U \subseteq V$ is a graph $G_U = (U, E_U)$, where $\forall v \in U$ and $\forall u \in U$, $(v, u) \in E_U$ if, and only if, $(v, u) \in E$. A subset of vertices $U \subseteq V$ is a clique if the subgraph of G induced by U, G_U, is complete.

Two sets A and B are *disjoint* sets if $A \cap B = \emptyset$. The *symmetric difference* of two sets A and B is given by $A \Delta B = \{(A - B) \cup (B - A)\}$. The *Jaccard coefficient*, between two sets A e B, is defined by $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$. Then, two sets A and B have all the elements in common when $J(A, B) = 1$. If $J(A, B) = 0$, then A and B have no elements in common, i.e., $A \cap B = \emptyset$.

A graph G is a *cluster graph* if G is a disjoint union of cliques [Shamir et al., 2004]. In this work, a cluster C is a vertex subset of G, that is, $C \subseteq V$. Note that a cluster is not necessarily a clique. A clustering is a vertex partitioning $\mathcal{C} = \{C_1, C_2, \ldots, C_l\}$ such that, for $1 \leq i \leq l$, $C_i \subseteq V$, $C_i \neq \emptyset$, and $\bigcup_{i=1}^{l} C_i = V$. A clustering $\mathcal{C}$ is disjoint if, and only if, $\forall C_i \in \mathcal{C}$ and $\forall C_j \in \mathcal{C}$, with $C_i \neq C_j$, $C_i \cap C_j = \emptyset$. $\mathcal{C}$ is an overlapping clustering if, and only if, $\exists C_i \in \mathcal{C}$ and $\exists C_j \in \mathcal{C}$, with $C_i \neq C_j$, such that $C_i \cap C_j \neq \emptyset$. Given a vertex $v \in V$ and a clustering $\mathcal{C}$, the clusters set containing the vertex v is defined by $\ell_{\mathcal{C}}(v) = \{C_i \mid v \in C_i, C_i \in \mathcal{C}\}$. In addition, $|\ell_{\mathcal{C}}(v)| = 1$ if $\mathcal{C}$ is a disjoint clustering and $|\ell_{\mathcal{C}}(v)| \geq 1$ if $\mathcal{C}$ is an overlapping clustering.

Let E_V be the set of all possible edges of a graph $G = (V, E)$. The cluster editing problem aims at finding an edge subset F, such that $F \subseteq E_V$, so that the graph $G' = (V, E \Delta F)$ is a disjoint union of cliques. The subset F is denominated as edge edition set. In the minimization version of the cluster editing problem, it is necessary to find the smallest edge edition set.

Given a graph G and a clustering $\mathcal{C}$, the cost of a cluster editing solution is computed as presented by Equation 1 [Charikar et al., 2005]. In this equation, variables x_{ij}, for $1 \leq i < j \leq n$, are equal to one when vertices i and j belong to different clusters and x_{ij} is equal to zero when i and j belong to the same cluster.

$$K_{ce}(G, \mathcal{C}) = \sum_{i<j,\ (i,j)\in E} x_{ij} + \sum_{i<j,\ (i,j)\notin E} (1 - x_{ij}). \tag{1}$$

In the overlapping cluster editing problem we have to find an edge edition set F such that the vertices of the input graph G are partitioned into cliques. Note that, unlike the cluster editing problem, in the overlapping cluster editing problem the cliques are not necessarily disjoint. In other words, cliques can share vertices. Then, the overlapping cluster editing solution cost computing need to be modified. As a result, $x_{ij} = 1$ if $\ell_C(i) \cap \ell_C(j) = \emptyset$ and $x_{ij} = 0$ if $\ell_C(i) \cap \ell_C(j) \neq \emptyset$.

3. Hybrid heuristic

The hybrid heuristic proposed in this paper can be divided into three steps. Firstly, the metaheuristics *Biased Random-Key Genetic Algorithm* (BRKGA) [Gonçalves and Resende, 2011] and *Simulated Annealing* (SA) [Kirkpatrick et al., 1983] are used to generate a set of cluster editing solutions, without overlapping, of the input graph. Subsequently, all clusters belonging to the

https://proceedings.science/p/85411?lang=pt-br

solutions set are grouped together to form a new set. Then this set is used as input by CPLEX [IBM Corporation, 2017] to solve a mixed-integer linear programming model, which is described in Subsection 3.1. The main reason for using the clusters set from metaheuristics is to provide a good-quality input clusters to solve the mixed-integer linear programming formulation. An overlapping cluster editing solution is obtained with the resolution of this formulation.

The cluster editing solutions are obtained through the BRKGA and SA metaheuristics execution. For this, a number h_{sol} of solutions is passed as parameter to the hybrid heuristic. Hence, $\frac{h_{sol}}{2}$ solutions are selected from BRKGA execution and $\frac{h_{sol}}{2}$ solutions are selected from SA execution. In all experimental tests performed in this work we used $h_{sol} = 100$.

A pseudocode of the hybrid heuristic is shown in Algorithm 1. The BRKGA and SA metaheuristics are executed at lines 2 and 3. Then, at line 4, the clusters set is formed from metaheuristics solutions. Then, at line 5, CPLEX solves the mixed-integer linear program using the clusters set as input. The function $cplex\_solve$ returns an overlapping cluster editing solution.

Algorithm 1: Hybrid heuristic.

input : graph $G = (V, E)$; mixed-integer liner programming formulation $model$; BRKGA number of generations $gen_{\max}$; BRKGA population size p; BRKGA elite population size p_e; BRKGA mutant population size p_m; BRKGA elite allele inheritance probability ρ_e; SA initial temperature t_i; SA final temperature t_f; SA cooling rate α; SA Metropolis algorithm step size $sa_{\max}$.

output: found solution s;

1 **begin**
2 $hist_{sol} \leftarrow brkga(G, gen_{\max}, p, p_e, p_m, \rho_e)$;
3 $hist_{sol} \leftarrow hist_{sol} \cup sa(G, t_i, t_f, \alpha, sa_{\max})$;
4 $clusters \leftarrow get\_clusters(hist_{sol})$;
5 $s \leftarrow cplex\_solve(G, model, clusters)$;
6 **return** s;
7 **end**

This section is subdivided as follows. In the next subsection, we present the proposed mixed-integer linear programming model. Details about the BRKGA and SA implementation are presented in Subsection 3.2.

3.1. Mixed-integer linear programming model

The mixed-integer linear programming model proposed in this work to find overlapping cluster editing solutions is shown in the formulation 3. Given a cluster set $S = \{C_1, C_2, \ldots, C_N\}$ of the vertices V of an input graph G, in the proposed model, the objective is to produce an overlapping clustering $\mathcal{C} \subseteq S$, where $|\mathcal{C}| = r$ and $\bigcup_{C \in \mathcal{C}} C = V$. The $\mathcal{C}$ set is composed by r *clusters* with the best costs and, depending on the established criteria, have more or less overlaps with each other cluster belonging to $\mathcal{C}$.

In the formulation 3, with the binary variables y_i, for $1 \leq i \leq N$, it is defined which C_i clusters belong, or not, to the final overlapping cluster editing solution. Also, there is a cost d_i associated with each cluster C_i that represents how good this cluster is. The d_i values are given by the Equation 2.

$$d_i = \frac{E_{C_i}^{in}}{E_{C_i}^{\max}} - \frac{E_{C_i}^{out}}{|C_i| \cdot (|V| - |C_i|)}. \tag{2}$$

In the Equation 2, $E_{C_i}^{\max}$ is the maximum number of edges between C_i vertices, that is, $E_{C_i}^{\max} = \frac{|C_i| \cdot (|C_i| - 1)}{2}$. In addition, $E_{C_i}^{in}$ is the number of edges that connect vertices belonging to C_i

https://proceedings.science/p/85411?lang=pt-br

and $E_{C_i}^{out}$ is the number of edges connecting a vertex from C_i and a vertex that does not belong to C_i. Moreover, the maximum number of edges between vertices from C_i and vertices not belonging to it is given by $|C_i| \cdot (|V| - |C_i|)$.

$$\max \sum_{i=1}^{N} (d_i \cdot y_i - u_i) \tag{3a}$$

subject to

$$\sum_{j=1}^{N} \left| \frac{|C_i \cap C_j|}{|C_i \cup C_j|} - z_i \right| \cdot (y_i + y_j - 1) \leq u_i,\ i = 1, 2, \ldots, N, \tag{3b}$$

$$\sum_{i=1}^{N} y_i = r, \tag{3c}$$

$$\sum_{i=1}^{N} a_{ji} \cdot y_i \geq b,\ j = 1, 2, \ldots, n, \tag{3d}$$

$$y_i \in \{0, 1\}, u_i \in \mathbb{R},\ i = 1, 2, \ldots, N. \tag{3e}$$

Since the object function 3a must be maximized, the lowest values of the real variables u_i are obtained. This is because the u_i variables, in this function, have negative coefficients. With these variables, clusters with the smallest differences between the Jaccard coefficient, related to the other clusters, and the overlapping control parameters z_i are selected. Constraint 3b controls, with $z_i \in [0, 1]$, the overlaps between clusters. The closer z_i parameters values are to one, the greater the overlaps between clusters. The closer z_i parameters values are to zero, the smaller the overlaps between the clusters. The reason is that the overlap between a pair of clusters C_i e C_j is quantified by means of the Jaccard coefficient. Therefore, if clusters C_i e C_j have maximum overlap, that is, $C_i = C_j$, then $J(C_i, C_j) = 1$. If clusters C_i and C_j have no overlap, that is, $C_i \cap C_j = \emptyset$, then $J(C_i, C_j) = 0$. Thus, if $z_i = 1$, variable u_i will have the lowest value when $J(C_i, C_j) = 1$. On the other hand, if $z_i = 0$, variable u_i will have the lowest value when $J(C_i, C_j) = 0$.

In the constraint 3c is ensured that exactly r clusters are selected. It is guaranteed by constraint 3d that each graph vertex belongs to at least b clusters. In this constraint, for $1 \leq i \leq N$ and $1 \leq j \leq n$, $a_{ji} = 1$ if vertex j belongs to cluster C_i and $a_{ji} = 0$ otherwise. Also, constraint 3e defines variables y_i as binaries and u_i as reals ones.

For all experimental tests performed in this paper, we used $b = 1$ and r as the average number of clusters of the solutions set generated by metaheuristics. Furthermore, as shown in Section 4, we used two fixed values for the overlapping control parameters: $z_i = 0$ and $z_i = 1$, for all $1 \leq i \leq N$.

3.2. Metaheuristics

The BRKGA and SA metaheuristics were implemented to produce a set of solutions of the cluster editing problem. This set is used as input to solve the model presented in the Subsection 3.1. The SA metaheuristic was used because it is a classic and well-know metaheuristic. The BRKGA metaheuristic was implemented because it is relatively recent one and was successfully used in a variation of the overlapping cluster editing problem [Andrade et al., 2014].

In the BRKGA metaheuristic a solution is represented by means of a random-key array (chromosome). In this work we used chromosomes with $n + 1$ positions (alleles), where $n = |V|$. The last position $(n + 1)$ of each chromosome represents the maximum number of clusters that the decoded solution has. The n first chromosome positions represent the cluster in which each vertex belongs. The chromosome decoding step starts at allele $n + 1$ to determine the maximum number of clusters of the solution. For this, an upper bound, defined a priori, is utilized for the maximum

https://proceedings.science/p/85411?lang=pt-br

number of clusters (max_{clst}). Let A be an array with $n + 1$ positions which represents a BRKGA chromosome. The number of clusters in a decoded solution is given by $l = \lceil max_{clst} \times A[n + 1] \rceil$. In all tests carried out in this paper we used $max_{clst} = 200$. This value was empirically defined. Subsequently, the first n alleles are decoded to determine which of l clusters each vertex will belong to. This decoding is executed regarding l and is given by $k = \lceil l \times A[i] \rceil$, with $1 \leq i \leq n$ and $1 \leq k \leq l$, where C_k is the cluster which has vertex i. The BRKGA decoding step was parallelized.

A solution produced by the SA metaheuristic is an array A with n positions. Each position i of the array A, with $1 \leq i \leq n$, represents the cluster that the ith vertex belongs to. The SA initial solution is randomly generated. This solution is created by assigning to each position of array A a random value in the integer interval $[1, l]$.

The BRKGA and SA parameters values used in this work are presented in Table 1. Specifically, CALIBRA software [Adenso-Díaz and Laguna, 2006] was executed to obtain these values. The CALIBRA tests were performed in 13 instances of Bastos et al. [2016] with sizes ranging between 25 vertices and 100 vertices.

Table 1: BRKGA and SA parameters values used in the experimental tests carried out in this work. These values were obtained by CALIBRA software [Adenso-Díaz and Laguna, 2006].

	BRKGA					**SA**			
Parameter	gen	p	p_e	p_m	ρ_e	t_{init}	t_{final}	$sa_{\max}$	α
Value	696	820	0.19·p	0.23·p	60%	750	10^{-6}	750	0.98

In Table 1 are shown, in relation to the BRKGA metaheuristic, the number of generations (gen), the size of population (p), the elite population proportion (p_e), the mutant population proportion (p_m) and the elite allele inheritance probability (ρ_e). Still, in relation to SA the metaheuristic, the values of the initial temperature (t_{init}), final temperature (t_{final}), the step size of Metropolis algorithm ($sa_{\max}$) and the cooling rate (α) are also presented.

4. Results and analysis

In this section results of the hybrid heuristic tests are presented. All implementations were written in C++ language. For the resolution of models we used the IBM® ILOG® CPLEX® 12.8 [IBM Corporation, 2017]. All the computational tests were executed on a computer with Intel® Xeon® E5-2687W v2 CPU 3.40GHz × 8 processor with 25MiB *cache* memory and 62GiB of RAM. The operating system installed on this machine is Ubuntu 14.04.1 64*bits* with *kernel* 3.19.0-32-*generic*. In addition, in all CPLEX [IBM Corporation, 2017] executions were used $3h$ as time limit.

Two sets of instances were used to evaluate the hybrid heuristic. The first one consists of 112 instances generated by Bastos et al. [2016] of the cluster editing problem. These instances have sizes ranging between 21 vertices to 1000 vertices and can be obtained at `http://www2.ic.uff.br/~lbastos/`. The second one consists of 30 instances, with sizes ranging between 25 vertices to 1000 vertices, generated by Lancichinetti and Fortunato [2009] algorithm. Instances belonging to this second set have ground truth overlapping clustering solutions. These instances can be obtained at `http://www.lac.inpe.br/~rafael.santos/OCI/`. With this set the main objective is to verify if the hybrid heuristic is able to reproduce the original clustering. For this reason, we used the *FBCubed* Amigó et al. [2009] metric to evaluate the hybrid heuristic solutions in relation to the ground truth solution. The *FBCubed* metric, with values ranging in the real interval $[0, 1]$, is a supervised measure for evaluating overlapping clusterings. The closer to one is the *FBCubed* value, the better is the overlapping clustering relative to the ground truth. The closer to zero, the worse the clustering relative to the ground truth.

Table 2 shows results of the tests in Bastos et al. [2016] instances with up to 100 vertices. The costs, calculated by the equation 1, of the solutions generated by the BRKGA and SA metaheuristics and their execution times, in seconds, are shown. With regard to hybrid heuristic

https://proceedings.science/p/85411?lang=pt-br

results, in table 2, the solution costs, calculated by modifying the Equation 1, the execution time, in seconds, and the number of vertices belonging to more than one cluster (*ovlp*) are presented. Two versions of the hybrid heuristic were used with different overlapping control parameters values, one with $z_i = 0$ and another with $z_i = 1$. Furthermore, the optimal costs of some Bastos et al. [2016] instances with up to 100 vertices were obtained by CPLEX [IBM Corporation, 2017] solving the Charikar et al. [2005] linear integer programming model. The CPLEX [IBM Corporation, 2017] computational cost for solving this model are also shown.

Table 2: Tests results of hybrid heuristic, with $z_i = 0$ and $z_i = 1$, on Bastos et al. [2016] instances with sizes ranging from 21 vertices to 100 vertices. In addition, metaheuristics results are presented. Costs of Charikar et al. [2005] model solved by CPLEX [IBM Corporation, 2017] are also shown.

Instance	***n***	**CPLEX**		**BRKGA**		**SA**		**Hybrid heuristic ($z_i = 0$)**			**Hybrid heuristic ($z_i = 1$)**		
		cost	***t (s)***	**cost**	***t (s)***	**cost**	***t (s)***	**cost**	***ovlp***	***t (s)***	**cost**	***ovlp***	***t (s)***
cmpr_101_1_22	22	**8**	0.04	9	0.61	**8**	0.37	23	1	0.04	**8**	1	0.59
cmpr_101_1_25	25	**8**	0.05	9	1.95	**8**	0.41	24	1	0.09	10	1	0.63
cmpr_101_2_21	21	**15**	0.06	**15**	1.85	**15**	0.34	24	2	0.04	**15**	2	0.40
cmpr_101_2_25	25	**19**	0.15	**19**	1.08	**19**	0.42	34	0	0.08	**19**	2	0.52
cmpr_101_3_23	23	**21**	0.08	**21**	2.13	**21**	0.40	46	2	0.08	57	1	0.39
cmpr_101_3_25	25	**26**	0.09	**26**	0.48	**26**	0.45	42	1	0.08	101	4	0.49
cmpr_101_4_25	25	37	0.38	38	0.66	37	0.47	76	0	0.09	**34**	5	0.37
cmpr_101_5_25	25	44	1.19	46	0.61	44	0.48	68	0	0.05	**42**	11	0.35
cmpr_101_6_25	25	**65**	3.85	**65**	0.86	**65**	0.50	95	1	0.42	121	11	0.36
cmpr_101_7_25	25	**72**	4.61	75	0.66	**72**	0.46	133	0	0.09	73	7	0.32
cmpr_101_8_25	25	**80**	6.87	81	0.72	**80**	0.48	95	1	0.41	81	4	0.31
cmpr_101_9_25	25	**99**	10.39	105	0.68	100	0.52	137	0	0.38	121	13	0.08
cmpr_101_10_25	25	**91**	6.04	92	0.66	92	0.50	141	0	0.38	92	11	0.12
cmpr_102_1_24	24	20	0.08	20	0.66	20	0.40	29	1	0.05	**18**	1	0.50
cmpr_102_1_25	25	21	0.07	21	2.52	21	0.46	39	1	0.05	**20**	3	0.49
cmpr_102_2_21	21	**19**	0.08	**19**	0.64	**19**	0.35	28	2	0.04	**19**	6	0.45
cmpr_102_2_25	25	**24**	0.19	**24**	0.80	**24**	0.43	37	0	0.08	**24**	2	0.43
cmpr_102_3_25	25	**34**	0.47	**34**	0.58	**34**	0.45	74	2	0.07	39	3	0.49
cmpr_102_4_25	25	**45**	0.31	**45**	0.47	**45**	0.47	66	0	0.38	45	3	0.29
cmpr_102_5_25	25	**55**	1.60	**55**	0.50	**55**	0.49	71	0	0.08	56	4	0.29
cmpr_102_6_25	25	**77**	7.23	78	1.58	78	0.46	111	1	0.28	81	7	0.21
cmpr_102_7_25	25	**83**	8.23	85	0.60	**83**	0.46	105	0	0.28	87	4	0.49
cmpr_102_8_25	25	**80**	4.60	81	0.53	**80**	0.50	128	1	0.26	84	8	0.16
cmpr_102_9_25	25	**89**	18.24	91	0.50	**89**	0.55	135	0	0.36	114	13	0.13
cmpr_102_10_25	25	**102**	12.87	103	0.56	**102**	0.50	134	0	0.26	115	10	0.11
cmpr_105_1_45	45	33	1.18	34	2.61	33	1.19	162	1	0.16	**30**	6	2.04
cmpr_105_1_50	50	42	1.78	43	1.10	42	1.33	165	1	0.16	**38**	7	2.50
cmpr_105_2_50	50	88	5.14	94	0.91	90	1.33	138	1	0.16	**81**	6	1.52
cmpr_105_3_50	50	137	77.52	140	2.59	137	1.34	204	0	0.12	**134**	8	1.79
cmpr_105_4_50	50	209	7349.68	221	1.07	212	1.33	267	0	0.48	**206**	10	1.52
cmpr_105_5_50	50	-	-	245	0.95	**243**	1.40	291	2	0.44	475	13	1.63
cmpr_105_6_50	50	298	10696.43	382	0.92	303	1.40	355	0	0.60	**293**	10	1.25
cmpr_105_7_50	50	-	-	314	1.09	311	1.40	367	1	0.41	**304**	14	1.21
cmpr_105_8_50	50	-	-	416	1.32	408	1.47	459	2	0.50	**386**	13	1.06
cmpr_105_9_50	50	-	-	420	0.98	420	1.47	504	1	0.36	**402**	12	0.76
cmpr_105_10_50	50	-	-	**448**	1.08	449	1.53	604	2	0.71	464	34	0.48
cmpr_106_1_50	50	**48**	0.53	55	1.03	52	1.39	314	1	0.09	49	5	1.19
cmpr_106_2_50	50	111	1.34	116	0.84	112	1.37	189	0	0.44	**108**	6	1.31
cmpr_106_3_50	50	177	117.23	179	0.86	179	1.36	230	0	0.12	**174**	7	1.24
cmpr_106_4_50	50	230	2780.21	234	0.92	231	1.36	335	1	0.17	**223**	15	1.34
cmpr_106_5_50	50	-	-	304	1.30	292	1.41	322	1	0.36	**288**	26	1.13
cmpr_106_6_50	50	-	-	308	2.12	**307**	1.48	453	1	0.14	693	20	0.94
cmpr_106_7_50	50	-	-	392	0.84	386	1.50	439	0	0.60	**369**	20	0.98
cmpr_106_8_50	50	-	-	394	1.04	394	1.43	451	0	0.37	**381**	20	0.61
cmpr_106_9_50	50	-	-	431	1.31	430	1.49	536	1	0.41	**414**	26	0.54
cmpr_106_10_50	50	-	-	468	0.94	**459**	1.54	570	1	0.07	478	27	0.36
cmpr_109_1_98	98	186	47.45	198	2.73	193	4.72	579	2	0.37	**183**	7	7.75
cmpr_109_1_100	100	187	69.98	205	1.96	194	5.06	226	1	0.59	**184**	6	6.94
cmpr_109_2_100	100	-	-	453	1.83	442	4.98	487	1	0.39	**427**	16	6.33
cmpr_109_3_100	100	-	-	641	1.97	633	4.90	1354	2	0.32	**608**	12	5.43
cmpr_109_4_100	100	-	-	901	1.99	883	4.93	983	1	0.86	**862**	30	5.68
cmpr_109_5_100	100	-	-	1112	1.87	1083	5.05	1199	1	0.68	**1037**	24	4.70
cmpr_109_6_100	100	-	-	1313	1.85	1272	5.04	1333	1	0.77	**1234**	29	4.27
cmpr_109_7_100	100	-	-	1602	1.86	**1551**	5.03	2070	1	0.32	2371	30	3.58
cmpr_109_8_100	100	-	-	1759	1.84	1677	5.09	1708	2	0.60	**1664**	48	2.77
cmpr_109_9_100	100	-	-	1954	1.85	**1923**	5.22	1940	1	0.89	2071	47	1.66
cmpr_109_10_100	100	-	-	2121	1.83	**2088**	5.29	2410	4	0.19	2121	55	0.87
cmpr_110_1_100	100	**256**	4.05	380	1.83	345	4.99	782	3	0.46	276	54	4.24
cmpr_110_2_100	100	**468**	69.39	565	2.64	558	4.90	567	3	0.22	486	38	3.44
cmpr_110_3_100	100	**739**	2347.71	887	1.81	788	5.06	886	9	0.68	757	37	4.24
cmpr_110_4_100	100	-	-	1125	1.89	1026	4.99	1069	1	0.61	**987**	44	3.95
cmpr_110_5_100	100	-	-	1367	1.70	1265	5.01	1273	4	0.59	**1240**	20	4.12
cmpr_110_6_100	100	-	-	1517	2.46	1476	5.09	1628	3	0.85	**1444**	36	3.42
cmpr_110_7_100	100	-	-	1699	1.78	1666	5.03	1860	3	1.52	**1613**	21	3.20
cmpr_110_8_100	100	-	-	1850	1.95	1829	5.27	1989	6	2.68	**1715**	59	2.35
cmpr_110_9_100	100	-	-	1963	2.22	1946	5.24	2211	4	0.25	**1847**	67	1.37
cmpr_110_10_100	100	-	-	**2041**	1.77	2039	5.71	2311	5	0.15	2083	56	1.00
Best costs	-	25	-	12	-	24	-	0	-	-	**39**	-	-

Since the BRKGA and SA metaheuristics were only used to generate solutions that are used as input to solve the proposed model, it can be considered that these metaheuristics obtained

https://proceedings.science/p/85411?lang=pt-br

reasonable results regarding the costs of the cluster editing problem. For example, from 40 known optimal costs, obtained by CPLEX [IBM Corporation, 2017] in the resolution of the Charikar et al. [2005] model, the BRKGA and SA metaheuristics presented 12 and 25 optimal costs respectively.

Considering the hybrid heuristic results, it can observed that the best overlapping cluster editing cost where achieved with $z_i = 1$. This is because, as can be seen at columns "*ovlp*" of Table 2, when $z_i = 1$ the model 3 is solved by selecting the most overlapping clusters. Then, the number of vertices belonging to more than one cluster is greater than $z_i = 0$. Thus, clusterings generated with $z_i = 0$ have more edges connecting vertices from different clusters and, therefore, have a higher overlapping cluster editing cost.

In addition, it is observed that the hybrid heuristic obtained better costs in 39 of the 67 instances as compared to the metaheuristics and the optimal cluster editing costs. Also, it can be seen that, in 12 of the 40 instances in which optimal cluster editing costs are known, the hybrid heuristic achieved better costs than the optimal ones. This is because in an overlapping clustering vertices can belong to more than one cluster. Hence, there are fewer edges between vertices belonging to different clusters. Then allowing clusters to overlap may be a less costly alternative to the cluster editing problem.

Table 3 presents results of the hybrid heuristic tests in Bastos et al. [2016] instances with sizes ranging from 200 vertices to 1000 vertices. This table has the same structure as Table 2, except for the results of the Charikar et al. [2005] model resolution. Since $3h$ was used as CPLEX [IBM Corporation, 2017] maximum execution time, it was not possible to obtain cluster editing optimal solutions for Bastos et al. [2016] instances with more than 100 vertices.

Table 3: Tests results of hybrid heuristic, with $z_i = 0$ and $z_i = 1$, on Bastos et al. [2016] instances with sizes ranging from 200 vertices to 1000 vertices. In addition, metaheuristics results are presented.

Instance	***n***	**BRKGA**		**SA**		**Hybrid heuristic ($z_i = 0$)**			**Hybrid heuristic ($z_i = 1$)**		
		cost	***t* (s)**	**cost**	***t* (s)**	**cost**	***ovlp***	***t* (s)**	**cost**	***ovlp***	***t* (s)**
cmpr_113_1_200	200	857	3.25	844	17.79	1103	0	0.78	**808**	18	26.09
cmpr_113_2_200	200	1873	3.23	1850	17.88	2428	0	0.81	**1787**	25	23.32
cmpr_113_3_200	200	2841	3.22	**2813**	18.54	3167	3	0.68	3011	52	17.59
cmpr_113_4_200	200	3847	3.70	**3795**	18.11	4722	5	0.72	3969	41	17.03
cmpr_113_5_200	200	4822	3.36	**4736**	18.26	5308	2	0.93	4935	60	16.10
cmpr_113_6_200	200	5628	3.22	**5549**	18.31	6257	3	0.51	5970	35	13.33
cmpr_113_7_200	200	6699	3.39	**6618**	18.66	6674	3	4.43	6969	72	11.54
cmpr_113_8_200	200	7573	3.32	7428	18.81	7436	9	2.75	**7300**	80	10.99
cmpr_113_9_200	200	8381	3.24	8239	19.15	8230	6	3.24	**8111**	92	10.02
cmpr_113_10_200	200	8712	3.54	9071	19.22	9531	11	0.67	**8691**	180	3.25
cmpr_114_1_200	200	2044	3.37	1988	18.56	1724	42	1.10	**1420**	150	15.13
cmpr_114_2_200	200	2981	3.33	2785	18.50	2694	13	2.09	**2334**	117	14.85
cmpr_114_3_200	200	3746	3.36	3687	18.60	3633	10	2.19	**3343**	91	14.47
cmpr_114_4_200	200	4930	3.26	4714	18.50	4714	7	1.01	**4433**	72	14.18
cmpr_114_5_200	200	5635	3.27	**5463**	18.48	5486	4	3.42	5714	56	11.52
cmpr_114_6_200	200	6627	3.35	**6457**	18.80	6485	3	2.86	6622	81	13.51
cmpr_114_7_200	200	7310	3.39	7227	18.95	7309	10	1.10	**7035**	94	10.99
cmpr_114_8_200	200	8065	4.16	**7967**	18.99	8013	21	1.72	8116	91	11.99
cmpr_114_9_200	200	8560	3.44	8549	19.68	9353	9	0.68	**8359**	105	4.99
cmpr_114_10_200	200	**8907**	4.47	8991	19.55	9866	13	0.29	9704	200	2.47
cmpr_117_1_500	500	5987	9.96	**5769**	107.60	9802	15	1.79	6373	26	81.51
cmpr_117_2_500	500	11976	10.64	11787	108.47	**16240**	5	1.89	11685	52	71.59
cmpr_117_3_500	500	18279	9.80	18120	108.17	22383	32	1.63	**18016**	215	71.59
cmpr_117_4_500	500	24951	10.04	**24435**	108.24	26368	0	2.74	30050	279	56.43
cmpr_117_5_500	500	30597	10.67	**30481**	108.32	31479	9	6.72	32097	83	55.40
cmpr_117_6_500	500	36741	10.67	**36584**	108.95	38340	9	3.39	38202	352	51.63
cmpr_117_7_500	500	42846	10.57	42645	109.17	45164	9	1.81	**42373**	405	45.32
cmpr_117_8_500	500	49076	10.15	**48697**	110.54	48821	5	22.37	49879	205	42.94
cmpr_117_9_500	500	55014	10.50	**54634**	111.75	56385	32	5.63	55502	339	39.58
cmpr_117_10_500	500	**58618**	11.37	59909	114.31	62108	39	0.71	62108	500	8.23
cmpr_118_1_500	500	16066	10.08	16135	109.13	15710	41	10842.13	**13677**	330	63.11
cmpr_118_2_500	500	21238	10.40	20534	109.30	**20443**	23	402.31	21345	286	53.90
cmpr_118_3_500	500	27442	10.54	26958	109.55	34720	38	1.54	**26600**	321	58.96
cmpr_118_4_500	500	32937	10.42	**32587**	108.63	37309	26	1.67	33933	271	52.00
cmpr_118_5_500	500	37558	10.47	**37432**	109.39	37638	22	34.71	38139	324	54.60
cmpr_118_6_500	500	43208	10.16	**42897**	108.97	43116	17	113.94	43617	187	46.95
cmpr_118_7_500	500	47451	9.83	**47131**	110.13	50697	18	1.70	51888	190	42.69
cmpr_118_8_500	500	52275	10.50	**51897**	111.05	53692	20	3.03	52067	220	41.58
cmpr_118_9_500	500	56652	10.50	**56291**	110.79	60537	35	1.74	56441	142	36.36
cmpr_118_10_500	500	**58064**	11.38	59147	114.33	61817	37	0.48	61128	487	8.18
cmpr_121_1_1000	1000	25737	29.04	**24669**	423.82	31678	172	10805.48	38112	679	94.32
cmpr_121_2_1000	1000	50162	28.97	**49182**	427.97	55110	163	1132.57	78464	643	103.54
cmpr_121_3_1000	1000	74694	30.17	**73708**	416.90	79203	114	1582.11	77423	668	102.67
cmpr_121_4_1000	1000	98716	29.22	**97838**	428.89	103842	110	15.72	111923	704	104.89
cmpr_121_10_1000	1000	**238131**	35.16	245233	437.15	248313	68	1.03	248989	1000	14.82
Best costs	-	4	-	**24**	-	2	-	-	15	-	-

https://proceedings.science/p/85411?lang=pt-br

In Table 3, for the two versions of the hybrid heuristic, we observe the same behavior of the results presented in Table 2, that is, the hybrid heuristic obtained better results with $z_i = 1$. It is also observed that the SA metaheuristic achieved the best costs in 24 out of 45 instances. The hybrid heuristic, with $z_i = 1$, obtained the best costs in 15 out of 45 instances. Besides the Bastos et al. [2016] instances being randomly generated without cluster formations, another reason that may have influenced the hybrid heuristic results is the quality of solutions generated by metaheuristics. This is because the size of instances presented in Table 3. Since theses instances have more vertices than instances presented in Table 2, metaheuristics may take longer to converge. As a result the metaheuristics solutions set could contain bad clusters. In addition, the number of clusters utilized in a solution of model 3 is the average of clusters in the solutions set. Then, a solution from the hybrid heuristic could use more clusters than the metaheuristics and, therefore, the solution cost may be greater.

In Table 4 results of the hybrid heuristic tests on the 30 instances generated by Lancichinetti and Fortunato [2009] algorithm are shown. These instances have sizes ranging from 25 to 1000 vertices. In order to differentiate each instance, the number of edges (m) is presented. For each size, there are five instance that have graph density ranging from sparse to dense. In addition, the hybrid heuristic results of the supervised metric *FBCubed* are shown. All the 30 instances have ground truth overlapping clustering.

Table 4: Hybrid heuristic results, with $z_i = 0$ and $z_i = 1$, on instances generated by Lancichinetti and Fortunato [2009] algorithm. Results of metaheuristics are also presented.

n	*m*	BRKGA		SA		Hybrid heuristic ($z_i = 0$)				Hybrid heuristic ($z_i = 1$)			
		cost	*t (s)*	cost	*t (s)*	cost	*FBCubed*	*ovlp*	*t (s)*	cost	*FBCubed*	*ovlp*	*t (s)*
25	28	14	0.73	**13**	0.46	29	0.51	0	0.07	**13**	**0.70**	3	0.51
25	139	85	0.71	**84**	0.49	140	0.43	1	0.05	**84**	**0.62**	15	0.11
25	156	**90**	0.79	**90**	0.50	155	0.48	0	0.35	**90**	**0.67**	19	0.16
25	158	80	0.54	**75**	0.48	139	0.39	0	0.43	88	**0.62**	19	0.09
25	265	**35**	0.60	**35**	0.47	55	0.48	1	0.20	**35**	**0.74**	25	0.04
50	113	73	1.32	71	1.37	92	0.39	1	0.09	**65**	**0.48**	8	1.66
50	471	339	1.04	**331**	1.46	350	0.19	5	0.40	334	**0.24**	28	0.69
50	591	421	0.95	**381**	1.54	509	0.49	2	0.09	421	**0.61**	34	0.42
50	614	404	0.97	**367**	1.45	481	0.29	2	0.10	404	**0.64**	35	0.45
50	1039	**186**	1.37	**186**	1.55	**186**	0.52	2	0.07	**186**	**0.69**	50	0.14
100	264	187	1.83	174	4.92	787	0.27	4	0.29	**149**	**0.50**	9	7.00
100	1286	719	2.87	787	5.04	957	0.36	16	0.59	**711**	**0.41**	71	4.28
100	1703	1095	1.88	1123	5.19	1141	0.30	8	0.79	**1082**	**0.36**	70	3.20
100	2250	1559	1.76	1582	5.45	1542	0.39	19	0.68	**1493**	**0.51**	73	1.41
100	4121	**829**	1.94	**829**	5.54	**829**	0.25	3	0.04	**829**	**0.94**	100	0.14
200	450	344	3.42	316	17.81	1490	0.27	3	0.52	**239**	**0.57**	48	29.72
200	6330	4693	3.30	4390	18.87	**4264**	0.27	46	0.53	6852	**0.30**	117	11.69
200	7467	4564	3.51	4624	19.58	4897	0.38	78	1.32	**4049**	**0.42**	167	7.95
200	14051	**5849**	3.87	**5849**	21.00	**5849**	0.29	9	0.11	**5849**	**0.78**	200	0.22
200	16543	**3357**	3.39	4476	20.64	**3357**	0.20	8	0.10	**3357**	**0.86**	200	0.70
500	2440	2362	10.59	1977	107.69	3079	0.26	6	3.28	**1841**	**0.35**	119	71.05
500	21285	19692	10.58	**18348**	113.24	23303	0.08	75	1.96	**18348**	**0.10**	216	65.10
500	24177	22811	9.80	21044	112.52	21781	**0.13**	109	6.45	**19730**	0.08	341	64.34
500	33550	30793	10.54	27685	112.34	25772	0.16	89	15.40	**24256**	**0.19**	251	48.88
500	93255	**31495**	11.10	48172	124.93	**31495**	0.46	9	0.15	**31495**	**0.63**	500	0.42
1000	4996	6614	29.02	**4139**	421.07	13712	0.08	126	4.22	12202	**0.25**	574	102.73
1000	70857	69189	29.03	68377	433.11	80846	0.06	107	4.60	**67075**	**0.08**	754	101.28
1000	94399	91048	29.04	83608	427.27	88805	0.08	269	1229.49	**81594**	**0.10**	659	92.93
1000	129055	125019	28.98	107619	436.29	136973	0.05	88	4.55	**102190**	**0.07**	711	86.31
1000	306286	**193214**	33.44	**193214**	491.43	**193214**	0.20	12	0.21	**193214**	**0.63**	1000	0.73
Best costs	-	8	-	14	-	7	-	-	-	**24**	-	-	-

As can be seen in Table 4, the hybrid heuristic, with $z_i = 1$, obtained the best costs in 24 out of 30 instances. We also observe better costs, in relation to the tests in the Bastos et al. [2016] instances, of the solutions generated by the hybrid heuristic with $z_i = 0$. This is because these 30 instances generated by the Lancichinetti and Fortunato [2009] algorithm originally have overlapping *clusters*.

In relation to the results of the *FBCubed* metric, it is observed that the best results were obtained with $z_i = 1$. Also, the hybrid heuristic, with $z_i = 1$, obtained *FBCubed* values greater than 0.5 in 16 instances. With these values of the *FBCubed* metric, the generated solutions can be considered good-quality clusterings.

https://proceedings.science/p/85411?lang=pt-br

5. Conclusions

In this paper a hybrid heuristic for the overlapping cluster editing problem was introduced. This hybrid heuristic is based on coupling the BRKGA and SA metaheuristics to generate solutions for the cluster editing problem and the CPLEX [IBM Corporation, 2017] that uses these solutions as input to solve a mixed integer linear program, also proposed in this work. To best of our knowledge the proposed hybrid heuristic is the first heuristic for the overlapping cluster editing problem.

The proposed hybrid heuristic showed promising results in experimental tests carried out in this paper. In the Bastos et al. [2016] instances, the hybrid heuristic achieved better costs in 54 of the 112 instances. In the tests with the instances generated by Lancichinetti and Fortunato [2009] algorithm, the hybrid heuristic obtained better costs in 24 of the 30 instances. Furthermore, regarding the execution time, the hybrid heuristic showed low computational costs in all 112 instances of Bastos et al. [2016] and in all the 30 instances generated by Lancichinetti and Fortunato [2009] algorithm. Although improvements have yet to be made, the hybrid heuristic has proved to be promising.

For future work, some points of the hybrid heuristic should be improved. For example, the number of clusters to be used in an overlapping clustering solution and increase the variety of cluster editing solution set. To increase the variety of this set, other metaheuristics and other methods, such as the column generation method [Oliveira et al., 2017], can be implemented.

6. Acknowledgments

This work was undertaken with the support of the CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Coordination for the Improvement of Higher Education Personnel) and CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico (National Council for Scientific and Technological Development), process number 301836/2014-0.

References

Adenso-Díaz, B. and Laguna, M. (2006). Fine-tuning of algorithms using fractional experimental designs and local search. *Operations Research*, 54(1):99–114.

Aggarwal, C. C. (2013). An introduction to cluster analysis. In Aggarwal, C. C. and Reddy, C. K., editors, *Data Clustering Algorithms and Applications*, chapter 1, p. 1–27. CRC Press, Boca Raton, FL, USA.

Amigó, E., Gonzalo, J., Artiles, J., and Verdejo, F. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486.

Andersen, R., Gleich, D., and Mirrokni, V. (2012). Overlapping clusters for distributed computation. In *WSDM '12 Proceedings of the fifth ACM international conference on Web search and data mining*, p. 273–282, Seattle, USA. ACM.

Andrade, C. E., Resende, M. G. C., Karloff, H. J., and Miyakawa, F. K. (2014). Evolutionary algorithms for overlapping correlation clustering. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation - GECCO'14*, p. 405–412, Vancouver, Canada. ACM.

Bansal, N., Blum, A., and Chawla, S. (2004). Correlation clustering. *Machine Learning*, 56(1): 89–113.

Bastos, L., Ochi, L. S., Protti, F., Subramanian, A., Martins, I. C., and Pinheiro, R. G. S. (2016). Efficient algorithms for cluster editing. *Journal of Combinatorial Optimization*, 31(1):347–371.

Ben-Dor, A., Shamir, R., and Yakhimi, Z. (1999). Clustering gene expression patterns. *Journal of Computer Biology*, 6(3-4):281–297.

https://proceedings.science/p/85411?lang=pt-br

Benelallam, A., Cuadrado, J. S., and Cabot, J. (2016). Efficient model partitioning for distributed model transformations. In *International Conference on Software Language Engineering*, p. 226–238, Amsterdam, Netherlands. ACM SIGPLAN.

Böcker, S., Briesemeister, S., Bui, Q. B. A., and Truss, A. (2009). Going weighted: Parameterized algorithms for cluster editing. *Theoretical Computer Science*, 410(52):5467–5480.

Bonchi, F., Gioni, A., and Ukkonen, A. (2013). Overlapping correlation clutering. *Knowledge and Information Systems*, 35(1):1–32.

Charikar, M., Guruswami, V., and Wirth, A. (2005). Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383.

Chesler, E. J., Lu, L., Shou, S., Qu, Y., Gu, J., Wang, J., Hsu, H. C., Mountz, J. D., Baldwin, N. E., Langston, M. A., Threadgill, D. W., Manly, K. F., and Williams, R. W. (2005). Complex trait analysis of gene expression uncovers polygenic and pleiotropic networks that modulate nervous system function. *Nature Genetics*, 37(3):233–242.

Damaschke, P. (2010). Fixed-parameter enumerability of cluster editing and related problems. *Theory of Computing Systems*, 46(2):261–283.

Dehne, F., Langston, M. A., Luo, X., Pitre, S., Shaw, P., and Zhang, Y. (2006). *The Cluster Editing Problem: Implementations and Experiments*, p. 13–24. Springer Berlin Heidelberg, Berlin, Heidelberg.

Delvaux, S. and Horsten, L. (2004). On best transitive approximations to simple graphs. *Acta Informatica*, 40(9):637–655.

Demaine, E. D., Emanuel, D., Fiat, A., and Immorlica, N. (2006). Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187.

Fellows, M. R., Guo, J., Komusiewicz, C., Niedermeier, R., and Uhlmann, J. (2011). Graph-based data clustering with overlaps. *Discrete Optimization*, 8(1):2–17.

Fonseca, G. H. G., Santos, H. G., and Carrano, E. G. (2016). Integrating matheuristics and metaheuristics for timetabling. *Computers & Operation Research*, 74:108–117.

Gonçalves, J. F. and Resende, M. G. C. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525.

Guo, J., Komusiewicz, C., Niedermeier, R., and Uhlmann, J. (2009). *A More Relaxed Model for Graph-Based Data Clustering: s-Plex Editing*, p. 226–239. Springer, Berlin, Heidelberg. ISBN 978-3-642-02158-9.

IBM Corporation (2017). IBM ILOG CPLEX Optimization Studio V12.8.0 documentation.

Il'ev, V., Il'eva, S., and Kononov, A. (2016). *Short Survey on Graph Correlation Clustering with Minimization Criteria*, p. 25–36. Springer International Publishing, Cham. ISBN 978-3-319-44914-2.

Jiang, D. and Pei, J. (2009). Mining frequent cross-graph quasi-cliques. *ACM Transactions on Knowledge Discovery from Data*, 2(4):16:1–42.

Kirkpatrick, S., Gelatt, C. D., and Vecch, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

https://proceedings.science/p/85411?lang=pt-br

Lancichinetti, A. and Fortunato, S. (2009). Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80 (1):016118.

Maiza, M. I., N'Cir, C.-E. B., and Essoussi, N. (2016). Overlap regulation for additive overlapping clustering methods. In *10th International Conference on Research Challenges in Information Science*, p. 1–6, Grenoble, France. IEEE.

Maniezzo, V., Stützle, T., and Voß, S. (2009). *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*. Springer, New York, USA.

Matsuda, H., Ishihara, T., and Hashimoto, A. (1999). Classifying molecular sequences using a linkage graph with their pairwise similarities. *Theoretical Computer Science*, 210(2):305–325.

Oliveira, R. M., Chaves, A. A., and Lorena, L. A. N. (2017). A comparison of two hybrid methods for constrained clustering problems. *Applied Soft Computing Journal*, 54:256–266.

Pereira, M. A., Coelho, L. C., Lorena, L. A. N., and Souza, L. C. (2015). A hybrid method for the probabilistic maximal covering location-allocation problem. *Computers & Operations Research*, 57(Supplement C):51–59.

Pérez-Suárez, A., Martínez-Trinidad, J. F., Carrasco-Ochoa, J. A., and Medina-Pagola, J. E. (2013). Oclustr: a new graph-based algorithm for overlapping clustering. *Neurocomputing*, 121(9):234–247.

Rahmann, S., Wittkop, T., Baumbach, J., Martin, M., Truß, A., and Böcker, S. (2007). Exact and heuristic algorithms for weighted cluster editing. In *6th Annual International Conference on Computational Systems Bioinformatics*, p. 391–401, London, UK. Imperial College Press.

Shamir, R., Sharan, R., and Tsur, D. (2004). Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1-2):173–182.

Sharan, R., Maron-Katz, A., and Shamir, R. (2003). Click and expander: a system for clustering and visualizing gene expression data. *Bioinformatics*, 19(14):1787–1799.

Wang, K., Saho, Y., and Zhou, W. (2017). Matheuristic for a two-echelon capacitated vehicle routing problem with environmental considerations in city logistics service. *Transportation Research Part D*, 57:262–276.

Wu, Z. and Leahy, R. (1993). An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113.

Xu, R. and Wunsch II, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678.

https://proceedings.science/p/85411?lang=pt-br