Les Cahiers du GERAD ISSN: 0711–2440

Less is more approach for balanced minimum sum-of-squares clustering

L.R. Costa, D. Aloise,
N. Mladenović

G–2016–80

October 2016

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Less is more approach for balanced minimum sum-of-squares clustering

Leandro R. Costa [a]
Daniel Aloise [b]
Nenad Mladenović [c]

[a] *Universidade Federal do Rio Grande do Norte, Natal, Brazil*

[b] *GERAD & Universidade Federal do Rio Grande do Norte, Natal, Brazil*

[c] *GERAD & Mathematical Institute, Serbian Academy of Science and Arts, Belgrade, Serbia*

daniel.aloise@gerad.ca
nenad.mladenovic@gerad.ca

October 2016

Les Cahiers du GERAD
G–2016–80

Abstract: Clustering addresses the problem of finding homogeneous and well-separated subsets, called clusters, from a set of given data points. In addition to the points themselves, in many applications, there may exist constraints regarding the size of the clusters to be found. Particularly in balanced clustering, these constraints impose that the entities be equally spread among the different clusters. In this work, we present a basic variable neighborhood search heuristic for balanced minimum sum-of-squares clustering, following the recently proposed "less is more" approach, presented in *Information Sciences* 326, 160–171, (2016). Computational experiments show that the proposed algorithm outperforms the current state-of-the-art algorithm for the problem.

Keywords: Balanced clustering, minimum sum-of-squares, optimization

Acknowledgments: This work was partially supported by CNPq-Brazil grants 308887/2014-0 and 400350/2014-9; and CAPES-Brazil.

1 Introduction

Minimum sum-of-squares clustering (MSSC) is a central problem in cluster analysis. Given a set of n data points $P = \{p_1, p_2, \ldots, p_n\}$ in a given Euclidean space $\mathbb{R}^s$, it addresses the problem of finding k clusters thereby minimizing the summation of squared distances from each data point to the center of the cluster it belongs to.

The problem can be expressed as:

$$\begin{aligned} &\min_{x,y} \quad \sum_{i=1}^{n}\sum_{j=1}^{k} x_{ij}\|p_i - y_j\|^2 \\ &\text{subject to} \\ &\sum_{j=1}^{k} x_{ij} = 1, \quad \forall i = 1,\ldots,n \\ &x_{ij} \in \{0,1\}, \quad \forall i = 1,\ldots,n; \forall j = 1,\ldots,k. \end{aligned} \tag{1}$$

The decision variables $y_j \in \mathbb{R}^s$ for $j = 1,\ldots,k$ represent the location of the k cluster centers whereas decision variables x_{ij} express the assignment of the data point p_i to cluster j. The number of data points n is supposed to be larger than k, otherwise the problem is trivially solved by locating the cluster centers at the data point locations.

Among the several MSSC properties, we can highlight the following three:

(i) It minimizes intra-cluster variance while maximizes inter-cluster distances [29];

(ii) Given the centroids, each data point is assigned to its closest centroid, due to local optimality;

(iii) Given the assignments, the cluster centers are located in the centroids of the data points assigned to the cluster, due to first order optimality conditions. Consequently, MSSC can be treated as a combinatorial optimization problem on variables x.

The problem is NP-hard in the plane for general k [24]. When the dimension is in the instance, the problem is NP-hard already for two clusters [3]. The most popular heuristic for MSSC is indeed the k-means algorithm [14, 23]. From an initial partition, k-means alternately applies properties (ii) and (iii) above until stability is attained.

In some settings, it might be desirable that the number of data points be equally spread among the different clusters. Examples can be found in cloud computing [30], image segmentation [16], and the composition of working groups [11], where the authors consider balance constraints in the composition of MBA student teams. There are many ways of searching for balance in clustering [6], which vary in the way that balance is considered over the homogeneity and/or heterogeneity of the clusters to be found. In its strict sense, balanced clustering is achieved by adding the following constraints in (1).

$$\sum_{i=1}^{n} x_{ij} = \frac{n}{k} \quad \forall j = 1,\ldots,k, \tag{2}$$

whenever n is a multiple of k, otherwise there will exist (n mod k) clusters of size $\lceil \frac{n}{k} \rceil$ and k - (n mod k) clusters of size $\lfloor \frac{n}{k} \rfloor$. Model (1)+(2), called henceforth *balanced MSSC*, is equivalent to the minimum weighted perfect matching problem for $n/k = 2$, and consequently, can be solved in time $O(n^3)$. A NP-hardness proof for the particular case of two equal-sized clusters can be found in [8].

In this paper, we propose a Variable Neighborhood Heuristic (VNS) [18] based on a single neighborhood for balanced MSSC clustering. The heuristic follows the precepts of the recent "less is more" approach in metaheuristic design [26]. The paper is organized as follows. The next section introduces the fundamental elements of VNS as a combinatorial optimization method. Section 4 presents our VNS algorithm for balanced MSSC. Computational experiments are reported in Section 5, where our algorithm is compared with the current state-of-the-art balanced k-means method of Malinen and Fränti [25]. Finally, the conclusions are presented in Section 5.

2 VNS fundamentals

Variable Neighborhood Search is a stochastic search method aimed to find optimal or near-optimal solutions of global optimization problems. It is classified as a metaheuristic procedure, i.e., a framework that help other heuristics to scape from local optimality. VNS has been sucessfully applied to several NP-hard clustering problems [2, 7, 17, 20, 28].

Balanced MSSC belongs to the class of combinatorial optimization problems, formally defined without loss of generality as a minimization problem as follows. Let $Z = \{1, \ldots, z\}$ be a finite set and let $c = (c_1, \ldots, c_z)$ be a z-vector. For $F \subseteq Z$, define $c(F) = \sum_{j \in F} c_j$. Suppose we are given a collection of subsets $\mathcal{F}$ of Z. The *combinatorial optimization* problem is

$$\min\{c(F) : F \in \mathcal{F}\}. \tag{3}$$

Given a combinatorial optimization problem $C = (Z, \mathcal{F}, c)$, we define, for each feasible solution $F \in \mathcal{F}$, its characteristic vector $\chi^F \in \{0,1\}^Z$ such that $\chi^F_j = 1$ if $j \in F$, and $\chi^F_j = 0$ otherwise. Thus, C can be seen as minimizing over a polytope, i.e.,

$$\min\{c^T x | x \in conv\{\chi^F \in \{0,1\}^Z | F \in \mathcal{F}\}\}. \tag{4}$$

A local minimum x^* of (4) is such that

$$c^T x^* \leq c^T x, \forall x \in \mathcal{N}(x^*), \tag{5}$$

where $\mathcal{N}(x)$ denotes the *neighborhood* of x. In combinatorial optimization problems, the neighbors of a solution x can be obtained in many different ways, for example, by complementing one or two of the x components. VNS systematically operates over several neighborhoods of an optimization problem in order to achieve global optimality. The rationale behind VNS is that a local minimum w.r.t. one neighborhood is not necessarily so for another neighborhood. Consequently, changing neighborhoods might be effective on scaping from local minima. Of course, a global minimum is a local minimum regardless of the neighborhood under consideration.

VNS is a single trajectory metaheuristic like simulated annealing and tabu search [9], which means that a single solution x is kept during the whole execution of the algorithm. The main difference to the other metaheuristic frameworks is that improvements are searched in increasingly wider neighborhoods of x. First, a random neighbor solution x' is obtained in the current neighborhood $\mathcal{N}_t(x)$, and then a local descent method is performed from x' leading to another local minimum x''. If x'' is worse than (or equal to) x, it is ignored and the algorithm iterates from a new random neighbor x' obtained in a more distant neighborhood of x, i.e., $\mathcal{N}_{t+1}(x)$. Otherwise, x'' replaces x, and the algorithm resumes in the closest neighborhood of the new best solution, i.e., $\mathcal{N}_1(x)$. Whenever all neighboords have been explored without success, VNS restarts again from the closest neighborhood until a stopping condition (e.g. maximum CPU time) is met. Indeed, VNS favors the exploration of the closest neighborhoods of x. This is based on the observation that, for many problems, local minima with respect to one or several neighborhoods are relatively close to each other.

The basic steps of VNS are given in Algorithm 1.

In principle, VNS can use different neighborhood structures both in its diversification and intensification steps (see e.g. [4, 27]). However, recent studies have demonstrated that VNS heuristics based on a *unique* neighborhood structure might be very effective [26, 10], leading to the so-called "less is more" approach (LIMA) on metaheuristic design. We now turn to our implementation of a LIMA-VNS heuristic for balanced MSSC.

Algorithm 1 VNS

Select a set of neighborhoods $\mathcal{N}_t$, for $t = 1, \ldots, t_{max}$;
Find an initial solution x;
repeat
  $t \leftarrow 1$
  repeat
    Choose a random neighbor solution x' from $\mathcal{N}_t$;
    Apply a local descent method from x', obtaining x'';
    if $cost(x'') < cost(x)$ **then**
      $x \leftarrow x''$;
      $t \leftarrow 1$;
    else
      $t \leftarrow t + 1$;
    end if
  until $t = t_{max}$
until a stopping criterion is met

3 LIMA-VNS heuristic for balanced MSSC

Our VNS heuristic is based on the exploration of the *swap* neighborhood, which encompasses all the neighbor solutions obtained by swapping a point from a cluster with another one belonging to a different cluster. Figure 1 illustrates three possible neighbors of a balanced MSSC solution with two clusters in the swap neighborhood.

In order to efficiently explore the swap neighborhood, we reformulated the problem by means of the Huygens' theorem [13], which states that the sum of squared distances from all data points of a given cluster to its centroid is equal to the sum of squared distances between each pair of points of this cluster divided by its cardinality. Thus, the objective function of (1) can be rewritten as:

$$\sum_{j=1}^{k} \frac{\sum_{i=1}^{n-1} \sum_{\ell=i+1}^{n} \|p_i - p_\ell\|^2 x_{ij} x_{\ell j}}{\sum_{i=1}^{n} x_{ij}} \tag{6}$$

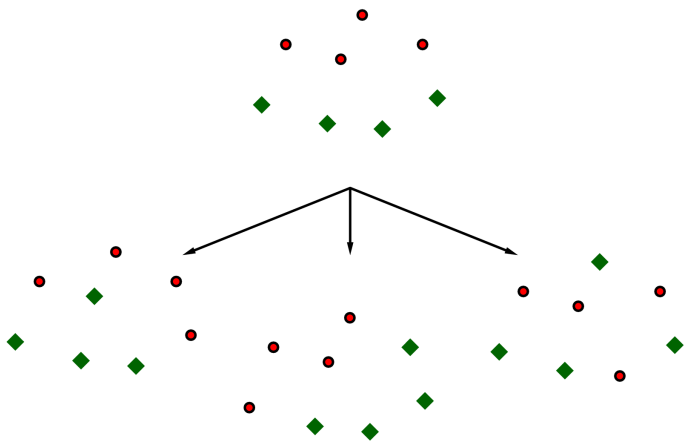

Figure 1: Possible neighbors of a balanced MSSC solution in the swap neighborhood

The expression (6) can be decomposed into two terms since the cardinality of the clusters in balanced MSSC are known a priori. Thus, (6) can be expressed as:

$$\frac{1}{\lceil \frac{n}{k} \rceil} \sum_{j \in \mu} \sum_{i=1}^{n-1} \sum_{\ell=i+1}^{n} \|p_i - p_\ell\|^2 x_{ij} x_{\ell j} + \frac{1}{\lfloor \frac{n}{k} \rfloor} \sum_{j \in \eta} \sum_{i=1}^{n-1} \sum_{\ell=i+1}^{n} \|p_i - p_\ell\|^2 x_{ij} x_{\ell j}, \tag{7}$$

where μ and η are the index set of clusters whose cardinality is $\lceil \frac{n}{k} \rceil$ and $\lfloor \frac{n}{k} \rfloor$, respectively.

We can notice that the cost of a cluster j^* is a quadratic term given by $x_{j^*} Q x_{j^*}^T$ divided by the cardinality of cluster j^* (i.e. a constant), where $x_{j^*} = (x_{1j^*}, \ldots, x_{nj^*})$ and $Q = (q_{ab})$ with $q_{ab} = \|p_a - p_b\|^2/2$. Consequently, the cost variation $\Delta_{\ell j^*}$ incurred by the addition of point p_ℓ to cluster j^* is given by:

$$\Delta_{\ell j^*} = \frac{2}{size(j^*)} \sum_{i=1}^{n} q_{\ell i} x_{ij^*}, \tag{8}$$

where $size(j^*)$ denotes the cardinality of cluster j^*. Likewise, $\Delta_{\ell j^*} = -2 \sum_{i=1}^{n} q_{\ell i} x_{ij^*} / size(j^*)$ when point p_ℓ is removed from cluster j^*. Thus, the variation in the cost of cluster j^*, from which a point $p_{\ell'}$ is removed and a point p_ℓ is added, is calculated by a closed expression given by $\Delta_{\ell j^*} - \Delta_{\ell' j^*}$. Since the cost of a swap neighbor can be computed in $O(1)$, the whole swap neighborhood of a given solution can be evaluated in $O(n^2)$ time.

At first glance, the computation of the $\Delta_{\ell j}$ values in (8) would require $O(n)$ time for each $\ell = 1, \ldots, n$ and $j = 1, \ldots, k$. However, they can be updated in constant time. For example, if a point p_ℓ is added to cluster j^*, $\Delta_{\ell' j^*}$, for all $\ell' = 1, \ldots, n$ with $\ell \neq \ell'$, can be straightforwardly updated with

$$\Delta_{\ell' j^*} \leftarrow \Delta_{\ell' j^*} + \frac{2}{size(j^*)} q_{\ell j^*} x_{\ell' j^*}, \tag{9}$$

whereas

$$\Delta_{\ell' j^*} \leftarrow \Delta_{\ell' j^*} - \frac{2}{size(j^*)} q_{\ell j^*} x_{\ell' j^*}, \tag{10}$$

for all $\ell' = 1, \ldots, n$ with $\ell \neq \ell'$, shall point p_ℓ is removed from cluster j^*.

The random neighbors x' from the current solution x are selected in our VNS from nested swap neighborhoods $\mathcal{N}_t$. Thus, if $t = 2$, then x' is selected from the neighborhood composed by all solutions yielded from two random swaps in x. Likewise, if $t = 3$ then x' is obtained from three random swaps in x, and so on.

Our local descent method consists in evaluating from a given solution all the neighbors obtained by exchanging a pair of points between two clusters, then moving the search to the first improving neighbor found. This constrasts with typical gradient descent methods that choose the steepest descent direction for further search. In [19], the authors demonstrate via extensive computational experiments that the first improving direction should be preferred if the initial solution for the search is chosen at random.

Our heuristic relies on the exploration of a unique neighborhood in the search for the global minimum of balanced MSSC. The algorithm is simple and contrasts with sophisticated search methods found in the literature that combine elements from different methodologies (e.g. [12, 15, 21]). Such combination of ideas within the optimization search may sometimes hide the real reasons for the success or failure of a particular method.

4 Computational experiments

In order to assess the performance of our VNS-LIMA heuristic, we used 16 datasets from [5], which are described in Table 1. Computational experiments were performed on a Intel 2.5Ghz processor with 16GB of RAM memory. The algorithm was implemented in C++ [1] and compiled by gcc 4.4.

[1] Available upon request

Table 1: List of datasets

Dataset	n	s	k
Iris	150	4	3
Wine	178	13	3
Glass	214	10	7
Thyroid	215	5	3
Ionosphere	351	34	2
Libra	360	90	15
User knowledge	403	5	4
Body measurements	507	5	2
Water treatment plant	527	38	13
Breast cancer	569	30	2
Synthetic control	600	60	6
Vehicle	846	18	6
Vowel recognition	990	10	11
Yeast	1484	8	10
Multiple features	2000	240	7
Image segmentation	2310	19	7

VNS usually requires very few parameters to adjust. In particular, our VNS-LIMA heuristic has only one parameter: t_{max}. We set $t_{max} = 20$ following limited computational experiments.

Our method is compared against the state-of-the-art balanced k-means heuristic of Malinen and Fränti [25], implemented in Matlab and available at http://www2.uef.fi/en/sipu/data-and-software. Due to the balance constraints (2), the data points cannot be assigned to their closest centroids as done in the k-means algorithm. Instead, in the balanced k-means heuristic, the assignment of data points to cluster centroids is done by means of the Hungarian algorithm [22]. First a bipartite graph is constructed, consisting of n data points and n slots. The slots are then partitioned around the k centroids so that they form clusters of equal size (or different in at most one slot). The edge weight between a data point and a slot in the graph corresponds to the squared distance from the data point to the cluster centroid of that slot. The Hungarian algorithm is then executed to obtain the minimal weight pairing in time $O(n^3)$.

Table 2 presents average and best results for 10 runs of the balanced k-means heuristic and the VNS-LIMA heuristic. Both heuristics start from the same initial random solutions, which are made distinct in each execution. The time spent by the balanced k-means heuristic in each run is used as stopping condition for the corresponding execution of the VNS-LIMA heuristic. Results are presented in the form of percentage deviations from the best solutions found for each data set in the 10 executions, which are given in the second column (*cost*) of the table. The third column (*time*) presents the average CPU times (in seconds) used in running both algorithms.

Table 2: Percentage deviations from the best solutions obtained by the balanced k-means and VNS-LIMA algorithms

Dataset	*cost*	*time*	bk-means		VNS-LIMA	
			Best	Avg.	Best	Avg.
Iris	8.136720e+01	0.42	**0.00**	**0.00**	**0.00**	**0.00**
Wine	3.767275e+06	0.60	**0.00**	0.80	**0.00**	**0.64**
Glass	5.047503e+02	2.54	0.02	1.73	**0.00**	**1.22**
Thyroid	3.441325e+04	1.45	**0.00**	2.99	**0.00**	**0.11**
Ionosphere	2.433977e+03	3.65	**0.00**	0.06	**0.00**	**0.03**
Libra	6.395595e+07	8.36	1.48	2.69	**0.00**	**0.17**
User knowledge	7.022118e+01	12.35	**0.00**	1.26	**0.00**	**0.54**
Body	1.136299e+05	10.51	**0.00**	0.08	**0.00**	**0.03**
Water treatment plant	7.746106e+09	24.04	0.06	**1.08**	**0.00**	1.28
Breast cancer	1.375244e+08	22.12	**0.00**	0.20	**0.00**	**0.09**
Synthetic control	1.001825e+06	16.98	**0.00**	0.58	**0.00**	**0.07**
Vehicle	2.894321e+06	45.66	**0.00**	0.36	**0.00**	**0.00**
Vowel recognition	1.965884e+03	148.36	**0.00**	0.60	0.20	**0.51**
Yeast	5.324765e+01	394.09	0.03	0.42	**0.00**	**0.19**
Multiple features	1.949141e+06	2233.19	0.02	0.43	**0.00**	**0.05**
Image segmentation	2.107874e+07	1827.98	0.59	1.51	**0.00**	**1.01**

We notice from the table that the best solutions are always obtained by the VNS-LIMA algorithm, except for dataset `Vowel recognition`. Regarding average solutions, the VNS-LIMA again obtains better (or equal) solutions than the balanced k-means heuristic, except for dataset `Water treatment plant`.

Due to the simplicity of the LIMA approach, it is not difficult to understand the reasons behind this last observation. It appears that the valleys in the `Water treatment plant` dataset are very deep since the VNS-LIMA algorithm converges very early to the final reported local minima. This fact makes the results obtained by the algorithm largely dependent on the initial solutions used, then recommending to increase the value of t_{max} for tests with that dataset. Table 3 shows that the results of the algorithm are in fact improved for $t_{max} = \lfloor n/2 \rfloor = 263$.

Table 3: Percentage deviations from the best solution obtained by the balanced k-means algorithm and the VNS-LIMA heuristic for $t_{max} = \lfloor n/2 \rfloor$

Dataset	*cost*	*time*	bk-means		VNS-LIMA	
			Best	Avg.	Best	Avg.
`Water treatment plant`	7.744471e+09	24.04	0.08	1.10	**0.00**	**0.98**

Indeed, as a stochastic optimization method, the VNS-LIMA algorithm can still obtain better results than the ones presented in Tables 2 and 3 if larger computing times are used.

We also compared the performance of the balanced k-means and VNS-LIMA heuristics over time. Two hundred runs of each heuristic were performed for datasets `Water treatment plant`, `Vehicle`, `Vowel recognition` and `Multiple features`. For each run $r = 1, \ldots, 200$, we collected the computation times t_r (in seconds) spent by each heuristic on finding a solution with cost smaller than (or equal) to a target solution value. For each dataset, the target solution value corresponds to the worst solution value obtained by running 200 times the balanced k-means heuristic. Following the methodology reported in [1], a probability $\Omega_r = (r - \frac{1}{2})/200$ is associated with the r-th smallest running time t_r. Figure 2 presents the plots of points $z_r = (\Omega_r, t_r)$, for $r = 1, \ldots, 200$, for datasets `Water treatment plant`, `Vehicle`, `Vowel recognition` and `Multiple features`. We observe that VNS-LIMA outperforms considerably the balanced k-means heuristic in all four datasets plotted in Figure 2, being able to obtain with much higher probability target solution values in much less computing times (remark the logarithmic scale in the x-axis). This behaviour is also observed for the rest of the datasets in Table 1.

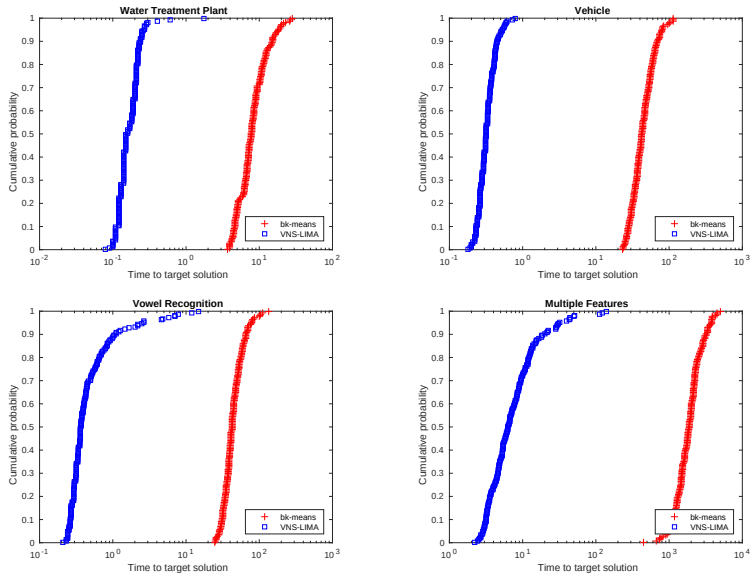

Figure 2: Time-to-target plots of heuristics balanced k-means and VNS-LIMA for instances `Water treatment plant`, `Vehicle`, `Vowel recognition` and `Multiple features`

5 Concluding remarks

Minimum sum-of-squares clustering is likely the most popular criterion in cluster analysis due to its mathematical properties for data in the Euclidean space. In some applications, balance is a desirable property for the clusters to be found. Among the many ways of promoting it, one of them is to striclty enforcing that the sizes of the clusters be equal (or differ in at most one unity). In this paper, we proposed a variable neighborhood search heuristic for balanced MSSC based only on systematic changes within nested swap neighborhoods. In summary, our results for benchmark datasets show that:

- For a fixed time horizon given by the CPU time spent by the balanced k-means heuristic to complete, the VNS-LIMA very often obtains better results than balanced k-means.
- The VNS-LIMA outperforms the balanced k-means heuristic, obtaining better results in less amount of time.
- The LIMA approach allows one to straighforwardly adjust his algorithm in order to improve its results.

Our VNS-LIMA methodology may also be adapted to size-constrained MSSC clustering for which the size of the clusters are predefined, but are not necessarily the same [31]. Finally, we expect that this paper could have showed the strenght of the LIMA approach when applied to algorithm design, making easier the interpretation of results because of the simplicity of the implemented codes.

References

[1] R. M. Aiex, M. G. Resende, C. C. Ribeiro, TTTplots: a perl program to create time-to-target plots, Optimization Letters 1 (4) (2007) 355–366.

[2] D. Aloise, G. Caporossi, P. Hansen, L. Liberti, S. Perron, M. Ruiz, Modularity maximization in networks by variable neighborhood search., Contemporry Mathematics–Graph Partitioning and Graph Clustering 588 (2013) 113–127.

[3] D. Aloise, A. Deshpande, P. Hansen, P. Popat, NP-hardness of Euclidean sum-of-squares clustering, Machine Learning 75 (2009) 245–249.

[4] D. J. Aloise, D. Aloise, C. T. Rocha, C. C. Ribeiro, J. C. Ribeiro Filho, L. S. Moura, Scheduling workover rigs for onshore oil production, Discrete Applied Mathematics 154 (5) (2006) 695–702.

[5] A. Asuncion, D. Newman, UCI Machine Learning Repository Http://www.ics.uci.edu/$\sim$mlearn/{MLR}epository.html.

[6] S. Basu, I. Davidson, K. Wagstaff, Constrained clustering: Advances in algorithms, theory, and applications, CRC Press, 2008.

[7] N. Belacel, P. Hansen, N. Mladenovic, Fuzzy J-means: a new heuristic for fuzzy clustering, Pattern Recognition 35 (10) (2002) 2193–2200.

[8] A. Bertoni, M. Goldwurm, J. Lin, F. Saccà, Size constrained distance clustering: Separation properties and some complexity results, Fundamenta Informaticae 115 (1) (2012) 125–139.

[9] I. BoussaïD, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Information Sciences 237 (2013) 82–117.

[10] J. Brimberg, N. Mladenović, R. Todosijević, D. Urošević, A basic variable neighborhood search heuristic for the uncapacitated multiple allocation p-hub center problem, Optimization Letters (2015) 1–15.

[11] J. Desrosiers, N. Mladenović, D. Villeneuve, Design of balanced MBA student teams, Journal of the Operational Research Society 56 (1) (2005) 60–66.

[12] A. Duarte, J. Sánchez-Oro, M. G. Resende, F. Glover, R. Martí, Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization, Information Sciences 296 (2015) 46–60.

[13] A. Edwards, L. Cavalli-Sforza, A method for cluster analysis, Biometrics 21 (1965) 362–375.

[14] E. Forgy, Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications, Biometrics 21 (1965) 768.

[15] G. Garai, B. Chaudhurii, A novel hybrid genetic algorithm with Tabu search for optimizing multi-dimensional functions and point pattern recognition, Information Sciences 221 (2013) 28–48.

[16] L. Hagen, A. B. Kahng, New spectral methods for ratio cut partitioning and clustering, IEEE Transactions on Computer-aided design of integrated circuits and systems, 11 (9) (1992) 1074–1085.

[17] P. Hansen, N. Mladenović, J-MEANS: a new local search heuristic for minimum sum of squares clustering, Pattern Recognition 34 (2001) 405–413.

[18] P. Hansen, N. Mladenović, Variable neighborhood search: principles and applications, European Journal of Operational Research 130 (2001) 449–467.

[19] P. Hansen, N. Mladenović, First vs. best improvement: An empirical study, Discrete Applied Mathematics 154 (5) (2006) 802–817.

[20] P. Hansen, M. Ruiz, D. Aloise, A VNS heuristic for escaping local extrema entrapment in normalized cut clustering, Pattern Recognition 45 (12) (2012) 4337–4345.

[21] Y. Jin, J.-K. Hao, Hybrid evolutionary search for the minimum sum coloring problem of graphs, Information Sciences 352 (2016) 15–34.

[22] H. W. Kuhn, The Hungarian method for the assignment problem, Naval research logistics quarterly 2 (1-2) (1955) 83–97.

[23] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of 5^{th} Berkeley Symposium on Mathematical Statistics and Probability, vol. 2, Berkeley, CA, 281–297, 1967.

[24] M. Mahajan, P. Nimbhorkar, K. Varadarajan, The planar k-means problem is NP-hard, Lecture Notes in Computer Science 5431 (2009) 274–285.

[25] M. I. Malinen, P. Fränti, Balanced k-means for clustering, in: Structural, Syntactic, and Statistical Pattern Recognition, Springer, 32–41, 2014.

[26] N. Mladenović, R. Todosijević, D. Urošević, Less is more: basic variable neighborhood search for minimum differential dispersion problem, Information Sciences 326 (2016) 160–171.

[27] C. C. Ribeiro, D. Aloise, T. F. Noronha, C. Rocha, S. Urrutia, An efficient implementation of a VNS/ILS heuristic for a real-life car sequencing problem, European Journal of Operational Research 191 (3) (2008) 596–611.

[28] É. Santi, D. Aloise, S. J. Blanchard, A model for clustering data from heterogeneous dissimilarities, European Journal of Operational Research 253 (3) (2016) 659–672.

[29] H. Späth, Cluster analysis algorithm for data reduction and classification of objects, John Wiley & sons, New York, 1980.

[30] W. Su, J. Hu, C. Lin, S. Shen, SLA-Aware Tenant Placement and Dynamic Resource Provision in SaaS, in: Web Services (ICWS), 2015 IEEE International Conference on, IEEE, 615–622, 2015.

[31] S. Zhu, D. Wang, T. Li, Data clustering with size constraints, Knowledge-Based Systems 23 (8) (2010) 883–889.