Pesquisa Operacional (2013) 33(1): 105-121
© 2013 Brazilian Operations Research Society
Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope

CLUSTERING SEARCH

Alexandre César Muniz de Oliveira[1], Antonio Augusto Chaves[2]
and Luiz Antonio Nogueira Lorena[3*]

Received September 10, 2012 / Accepted February 5, 2013

ABSTRACT. This paper presents the Clustering Search (CS) as a new hybrid metaheuristic, which works in conjunction with other metaheuristics, managing the implementation of local search algorithms for optimization problems. Usually the local search is costly and should be used only in promising regions of the search space. The CS assists in the discovery of these regions by dividing the search space into clusters. The CS and its applications are reviewed and a case study for a problem of capacitated clustering is presented.

Keywords: hybrid metaheuristic, Clustering Search, optimization.

1 INTRODUCTION

Clustering Search (CS) has been proposed as a generic way of combining search metaheuristics with clustering aiming to detect promising search areas before applying local search procedures [7]. The possibility of employing any metaheuristic and to apply it to combinatorial and continuous optimization problems make CS as a flexible framework for building hybrid metaheuristics.

The main idea is to identify promising areas of the search space by generating solutions through a metaheuristic and clustering them into groups that are further explored with local search heuristics [8].

A more precise definition of CS is given in [13]: a hybrid method that aims to combine metaheuristics and local search in which the search is intensified only in areas of the search space that deserve special attention.

CS introduces intelligence and priority to the choice of solutions to apply, generally costly, local searches, instead of applying random or elitist choices. Therefore it is expected an improvement in the convergence process associated with a decrease in computational effort as a consequence of a more rational employment of the computational costly heuristics.

*Corresponding author
[1] Universidade Federal do Maranhão – UFMA. E-mail: acmo@deinf.ufma.br
[2] Universidade Federal de São Paulo – UNIFESP. E-mail: antonio.chaves@unifesp.br
[3] Instituto Nacional de Pesquisas Espaciais – INPE. E-mail: lorena@lac.inpe.br

This paper presents an overview and recent findings of Clustering Search, besides a case study for the Capacitated Centered Clustering Problem (CCCP), reviewing two early CS approaches, that employs different two metaheuristics, and comparing the computational results against a new approach, based on Iterated Local Search.

The remainder of this paper is organized as follows. Related applications are summarized in Section 2. Section 3 describes the basic ideas, conceptual components and recent features of CS. Section 4 examine three different CS approaches for Capacitated Centered Clustering Problem and new computational results are presented for problem instances found in the Literature. At last, the findings and conclusions are summarized in Section 5.

2 CS RELATED APPLICATIONS

CS was early proposed as a hybrid evolutionary algorithm, aware of detecting promising search area based on clustering. The original ideas behind the Evolutionary Clustering Search (ECS), proposed by Oliveira & Lorena [4], was validated by a well-succeeded application to unconstrained numerical optimization. The ECS was implemented by a steady-state genetic algorithm hybridized by local search mechanism based on Hooke and Jeeves direct search. In the computational experiments, the method was compared against other approaches, taken from the literature, including the well-known Genocop III and the OptQuest Callable Library.

A combinatorial version of the CS was later applied by Oliveira & Lorena [6] to instances of sequencing problems that arise in scenarios involving arrangement of a set of client orders, gates in VLSI circuits, cutting patterns, etc. Problem-specific evolutionary operator were designed to deal with solutions represented by permutations, as block-order crossover (BOX) and 2-swap mutation, as well as a local search procedure based on 2-Opt improvement moves. The results obtained by CS were comparable to the best found in the literature [7].

Nagano *et al.* [5] and Ribeiro Filho et al. [9] present an CS applied to the Permutation Flow Shop Scheduling problem with the objective of minimizing total flow time. The computational results show superiority for a set of test problems, regarding the solution quality.

Ribeiro Filho *et al.* [32] apply the CS to the m-machine No-wait Flow Shop Scheduling problem. In a no-wait flow shop, the operation of each job has to be processed without interruptions between consecutive machines, *i.e.*, when necessary, the start of a job on a given machine must be delayed so that the completion of the operation coincides with the beginning of the operation on the following machine. The computational results compare CS with the best known algorithms in the literature showing the superiority of the evolutionary hybrid regarding the solution quality.

Chaves *et al.* [8] present a CS applied to the Capacitated p-median Problem (CPMP). The CPMP considers a set of n points, each of them with a known demand, the objective consists of finding p medians and assign each point to exactly one median such that the total distance of assigned points to their corresponding medians is minimized, and the a capacity limit on the medians may not be exceeded. The computational results show that the CS found the best known solutions in 24 of 26 instances.

Biajoli & Lorena [10] apply the CS to a Traveling Tournament Problem (TTP). The TTP is an optimization problem that represents some types of sports timetabling, where the objective is to minimize the total distance traveled by the teams. The CS is combined with the metaheuristic Variable Neighborhood Search (VNS). The computational results consider benchmark problems available in literature and real benchmark problems, *e.g.* Brazilian Soccer Championship.

Chaves & Lorena [11] considers the CS with GRASP and VNS for the Prize Collecting Traveling Salesman Problem (PCTSP). In this problem a salesman collects a prize in each city visited and pays a penalty for each city not visited, considering travel costs between the cities. The objective is to minimize the sum of travel costs and penalties paid, while including in the tour enough cities to collect a minimum prize, defined *a priori*. The CS was compared with CPLEX and found the optimal solutions for instances up to 60 cities and better solutions for instances with 100 cities, in which CPLEX did not get to close the gap between lower and upper bounds.

Correa *et al.* [12] present an CS with GRASP for the probabilistic maximal covering location-allocation problem with the objective of maximizing the population that has a facility within a maximum travel distance or time. The computational results show that the CS got better results than others heuristics of the literature. The optimal values were found for some instances of 30-node and 818-node networks.

Chaves *et al.* [13] apply the CS to the Assembly Line Worker Assignment and Balancing Problem (ALWABP). The ALWABP consists of assigning tasks to workstations, which are arranged in a predefined order, so that the precedence constraints are satisfied and some give measure of effectiveness is optimized. The CS got the best-known solution in 314 of 320 instances, and defined new best solutions for 306 instances.

Chaves & Lorena [14, 16] present two approaches of CS using Simulated Annealing and Genetic Algorithm, respectively, to the Capacitated Centered Clustering Problem. Both approaches have found good results, showing that the CS may be used with different metaheuristics.

Oliveira *et al.* [17] considers the CS with Simulated Annealing for Berth Allocation Problem (BAP). The BAP consists in allocating ships to positions of mooring using the maximum space of the quay and minimizing service time. The decisions to be made are concerning the position and time the ship should moor. Computational results are compared against recent methods found in the literature.

Ribeiro *et al.* [33] present a CS to the workover rig routing problem that is a variant of the vehicle routing problem with time windows. The objective is to minimize the total lost production in the operations of onshore oil fields. The computational results show that CS is a good heuristic for large instances. The authors illustrate a typical clustering process provided by CS, show active clusters, poor clusters, and promising clusters.

Due to the growing popularity of CS as effective and robust optimizer, researchers have been applying the CS to solve optimization problems arising in several fields. The literature on CS applications is summarized in Table 1.

Table 1 – Summary of applications of CS to optimization problems.

Areas and Problems	Metaheuristics used	Reference
Continuous Optimization		
Unconstrained Numerical Optimization	Genetic Algorithm	[4, 28]
Scheduling Problems		
Pattern Sequencing Problem	Genetic Algorithm	[6, 7, 28]
Permutation Flow Shop Scheduling Problem	Genetic Algorithm	[5, 9]
M-Machine No-Wait Flow Shop Scheduling Problem	Genetic Algorithm	[32]
Assembly Line Worker Assignment and Balancing Problem	Iterated Local Search; Variable Neighborhood Search; Simulated Annealing; Genetic Algorithm	[13, 29]
Location Problems		
Capacitated p-Median Problem	Simulated Annealing; Genetic Algorithm; Iterated Local Search; Variable Neighborhood Search	[8, 29]
Probabilistic Maximal Covering Location-Allocation Problem	GRASP	[12]
Capacitated Centered Clustering Problem	Genetic Algorithm; Simulated Annealing; Iterated Local Search; Variable Neighborhood Search	[14, 16, 29]
Hub Location Problem	Genetic Algorithm; Simulated Annealing/Tabu Search	[31]
Capacitated Hub Location Problem	Simulated Annealing	[34]
Routing Problems		
Traveling Tournament Problem	Variable Neighborhood Search	[10]
Prize Collecting Traveling Salesman Problem	GRASP/Variable Neighborhood Search; Genetic Algorithm; Simulated Annealing; Iterated Local Search	[11, 29]
Berth Allocation Problem	Simulated Annealing	[17]
Dial-a-Ride Problem	Simulated Annealing	[30]
Workover Rig Routing Problem	Simulated Annealing	[33]

3 CLUSTERING SEARCH FOUNDATIONS

Clustering Search (CS) employs clustering for detecting promising areas on the search space. It is particularly interesting to find out such areas as soon as possible to change the search strategy over them. An area can be seen as a search subspace defined by a neighborhood relationship in metaheuristic coding space.

The CS attempts to locate promising search areas by framing them by clusters. A cluster is defined by a *center*, c, that is generally, initialized at random and, posteriorly, it tends to progressively slip along really promising points in the search space.

The number of clusters $\mathcal{NC}$ can be fixed *a priori* [8] or dynamically determined according to the width of the search areas being explorated by the metaheuristic [4]. In the later case, clusters can be created in a way that all candidate solutions are covered by, at least, a cluster. By the other hand, inactive clusters, *i.e.*, clusters not covering any solutions may be eliminated.

The coverage is determined by a *distance metric* that computes the similarity between a given solution and the cluster center and must consider the problem nature. For example, in unconstrained continuous optimization, the similarity has been defined regarding the Euclidean distance [4]. In combinatorial optimization, the similarity can be defined as the number of movements needed to change a solution into the cluster center [7].

3.1 Component general guidelines

CS can be splitted off in four conceptually independent parts: a) the search metaheuristic (SM), b) the iterative clustering (IC) component, c) the analyzer module (AM), and d) the local searcher (LS). Figure 1 brings its conceptual design and Figure 2 presents the pseudo-code of CS.

The SM component can be implemented by any optimization algorithm that generates diversified solutions of the search space. It must work as a full-time solution generator, exploring the search space by manipulating a set of solutions, according to its specific search strategy.

IC component aims to gather similar solutions into groups, maintaining a representative cluster center for them. A *distance metric*, Δ, must be defined, *a priori*, allowing a similarity measure for the clustering process.

AM component examines each cluster, in regular intervals, indicating a probable promising cluster. A *cluster density*, also named *volume*, δ_j, is a measure that indicates the activity level inside the cluster j. For simplicity, δ_j can count the number of solutions generated by SM and grouped into c_j. Whenever δ_j reaches a certain *threshold* λ, meaning that some information template becomes predominantly generated by SM, such cluster must be better investigated to accelerate the convergence process on it. Clusters with lower δ_j can be eliminated or perturbed, as part of a mechanism that allows creating posteriorly other clusters, keeping framed the most active of them [7]. The cluster elimination does not affect the set of solutions in SM. Only the cluster is considered irrelevant for the process.

At last, the LS component is an internal searcher module that provides the exploitation of a supposed promising search area, framed by a cluster.

3.2 Overview of the clustering process

The iterative clustering can be seen as a learning process about an external environment (search space), from where new sampled knowledge is acquired, resulting in changes the previously

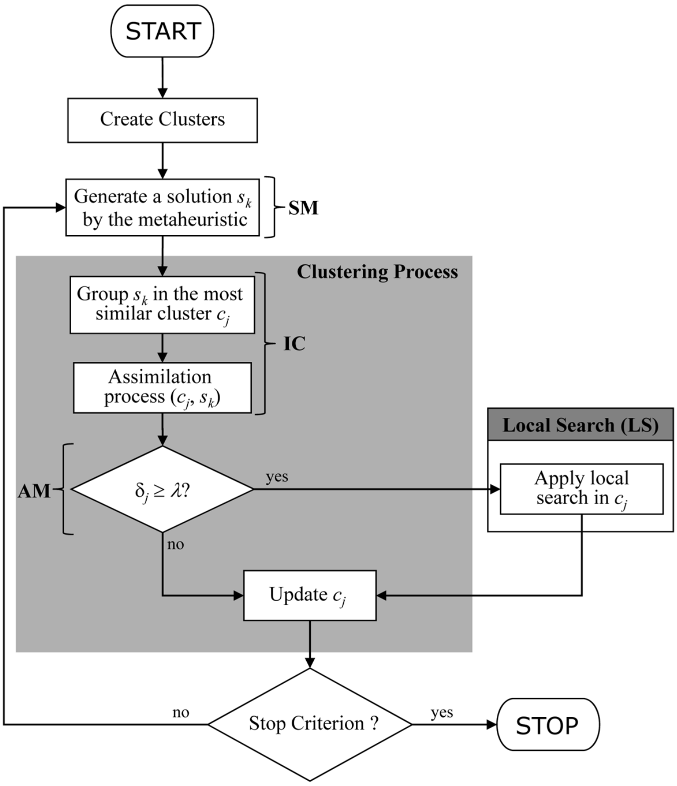
START
Create Clusters
Generate a solution s_k by the metaheuristic
SM
Clustering Process
Group s_k in the most similar cluster c_j
IC
Assimilation process (c_j, s_k)
AM
$\delta_j \geq \lambda$?
yes
no
Local Search (LS)
Apply local search in c_j
Update c_j
no
Stop Criterion ?
yes
STOP

Figure 1 – CS components.

```
algorithm CS
    create the initial clusters of CS
    { metaheuristic }
    while termination condition not satisfied do
        generate a solution ($s_k$) by metaheuristic
        { clustering process }
        find the most similiar cluster $c_j$ to the solution $s_k$
        insert $s_k$ into $c_j$ ( $\delta_j \leftarrow \delta_j + 1$ )
        $c_j \leftarrow$ Assimilation ($c_j$, $s_k$)
        if $\delta_j \geq \lambda$ then
            $\delta_j \leftarrow 0$
            { local search }
            find the best neighborhood of $c_j$
        end if
        update the center $c_j$
    end while
end algorithm
```

Figure 2 – CS pseudo-code.

learned beliefs [26]. The IC is the CS's core, working as a classifier, keeping grouped only relevant information, and driving search intensification in the promising search areas.

Solutions s_k generated by the SM are passed to the IC that attempts to group as known information, according to Δ. If the information is considered sufficiently new, it is kept as a center in a new cluster, c_{new}. Otherwise, redundant information activates the closest center c_i (cluster center that minimizes $\Delta(s_k, c_{j=1,2,\cdots,})$), causing some kind of perturbation on it. This perturbation means an *assimilation process*, in which the previously learned knowledge (center of the cluster) is updated by the received information. Cluster creation, *i.e.*, variable number of clusters, is not always implemented [8, 13, 16].

The assimilation process is applied over the closest center c_i, considering the new generated solution s_k. The general assimilation form is [28]:

$$c_i' = c_i \oplus \beta(s_k \ominus c_i) \tag{1}$$

where $\oplus$ and $\ominus$ are abstract operations over c_i and s_k meaning, respectively, addition and subtraction of solutions. The operation $(s_k \ominus c_i)$ means the vector of differences between each one of the n variables compounding the solutions s_k and c_i, considering the distance metric. A certain percentage β of the vector is the update step for c_i, giving c_i'. According to β, the assimilation can assume different forms. The three types of assimilation are shown in Figure 3.

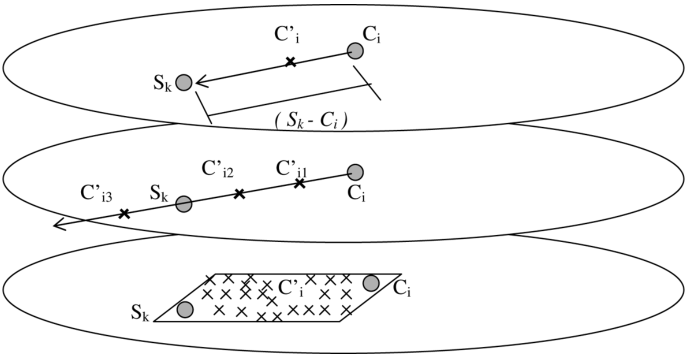

Figure 3 – Simple, path and crossover assimilations, respectively.

In simple assimilation, $\beta \in [0, 1]$ is a constant parameter, meaning a deterministic move of c_i in the direction of s_k. Only one internal point is generated more or less closer to c_i, depending on β, to be evaluated afterwards. The greater β, the less conservative the move is. This type of assimilation can be employed only with real-coded variables, where percentage of intervals can be applied to [4].

Despite its name, crossover assimilation is not necessarily associated with an evolutionary operator. In a general way, it means any random operation between two candidate solutions (parents), giving other ones (offsprings), similarly as a crossover operation in EAs. In this assimilation, β is an $n-$dimensional random vector and c_i' can assume a random point inside the hyper plane containing s_k and c_i.

Simple and crossover assimilations generate only one internal point to be evaluated afterwards. Path assimilation, instead, can generate several internal points or even external ones, holding the best evaluated one to be the new center. It seems to be advantageous, but clearly costly. These exploratory moves are commonly referred in path relinking theory [27].

In this assimilation, β is a $\eta-$dimensional vector of constant and evenly spaced parameters, used to generate η samples taken in the path connecting c_i and s_k. Since each sample is evaluated by the objective function, the path assimilation itself is an intensification mechanism inside the clusters. The new center c_i' is given by the best evaluated solution sampled in the path.

Table 2 summarizes the variables, the parameters and the problem-specific decisions that need to be set on CS. The better parameters for a given application of the CS can only be set by experimentation.

Table 2 – Summary of variables, parameters and problem-specific decision of CS.

Symbol	Type	Description
c_j	variable	cluster center
δ_j	variable	density or volume of the cluster
λ	parameter	threshold of promising cluster
$\mathcal{NC}$	parameter	number of clusters
Δ	problem-specific decision	distance metric

4 CS: A CASE STUDY

A classical location problem is the Capacitated p-Median Problem (CPMP) [1], which have various applications in many practical situations. It can be described as follows: given a set of n points (customers), each of them with a known demand, the problem consists of finding p medians and assigning each point to exactly one median such that the total distance of assigned points to their corresponding medians is minimized, and the capacity limit on the medians may not be exceeded.

The Capacitated Centered Clustering Problem (CCCP) [2] is a generalization of the CPMP, which can be viewed as the problem of defining a set of p groups with limited capacity and minimum dissimilarity among the formed groups, in which each group has a centroid located at the geometric center of its points and covers all demands of a set of n points. The main difference to the CPMP is that groups are centered at the "average" of their points' co-ordinates, while for the CPMP the groups are centered by their medians.

The evaluation of the objective function for the CCCP solution has a high computational cost as the centroids are unknown *a priori*. So, in this paper we solved the CPMP with the metaheuristics and the others components of CS solved the CCCP based on the CPMP solution.

Therefore, a solution of the CCCP is represented by two vectors as a solution of the CPMP. The first vector represents the point/median assignments and the second vector represents the chosen medians. The CS performs movements on these structure and the centroids are found later, calculating the average of the points' coordinates of each group defined by the medians. Figure 4 shows an example of one solution with ten points and three groups.

Medians	3	5	7

Point	1	2	3	4	5	6	7	8	9	10
Median	5	3	3	7	5	7	7	5	3	7

Figure 4 – An example of the solution representation.

The objective function of CCCP (equation 2) is the total sum of distances between each centroid and its allocated points. In this algorithm, we set a penalty for the solutions that not satisfy the capacity constraints. Let T_j be the sum of distances for the group j and f_j be the infactibility measure; ϖ be the multiplier for the value f_j.

$$f(s) = \sum_{j=1}^{p} \left(T_j + \varpi * f_j\right) \tag{2}$$

The distance metric Δ is the number of points assigned to different medians between two solutions. So, the distance increases when there are a large number of allocations to different medians between the solutions.

4.1 Metaheuristics

In this paper, we present a review of two approaches to the CS using different metaheuristics to generate solutions to the clustering process: Genetic Algorithm (Chaves & Lorena [16]) and Simulated Annealing (Chaves & Lorena [13]). And a new approach to the CS using the Iterated Local Search.

In these metaheuristics, initial solutions are constructed randomly choosing p medians and allocating points to the closest median (respecting the capacity of the medians).

4.1.1 Genetic Algorithm

The Genetic Algorithm (GA) [18] employs evolutionary operators as selection, crossover and mutation. The population size was set to 100 solutions, randomly initialized at first.

The solutions, during the reproductive phase, are selected from the population and recombined, producing offspring, which comprise the next generation. Parents are randomly selected from the population using the tournament method, which favors best individuals.

Crossover operation partially exchanges information between two selected solutions. We implemented the uniform crossover [21], in which a crossover mask (the same length as the solution structure) is created at random and the parity of the bits in the mask indicate which parent will supply the offspring with which attributes. For each attribute the parent who contributes its attribute to the offspring is chosen randomly with equal probability. The mutation operator changes the median of a randomly selected point by other random median. The probabilities of crossover (p_c) and mutation (p_m) are set to 95% and 5% respectively. The termination condition of GA is the number of generations, that was defined as 1000.

4.1.2 Simulated Annealing

The Simulated Annealing (SA) algorithm [19] starts from a random initial solution, following the traditional SA algorithm schema. Given a temperature T, the algorithm randomly selects one of the moves to a neighborhood and computes the variation of the objective function. If it improves the current solution the move is accepted, otherwise there is a probability of acceptance that is lower in low temperatures.

Four different moves have been defined to compose distinct neighborhoods from a solution s, named N^1, N^2, N^3 and N^4. N^1 is obtained swapping the allocation of two points of different medians. N^2 is obtained swapping a median with an assigned point to it. N^3 is obtained dropping a point allocated to a median and allocating it to another median. Finally N^4 is obtained swapping a median with any other non median point.

The control parameters of the procedure are the rate of cooling or decrement factor (α), the number of iterations for a fixed temperature ($SA_{\max}$) and the initial temperature (T_0). In this paper, we use $\alpha = 0.95$, $SA_{\max} = 1000$ and $T_0 = 1000000$.

4.1.3 Iterated Local Search

The Iterated Local Search (ILS) [20] consists in the iterative application of a local search procedure to starting solutions that are obtained by the previous local optimum through a solution perturbation.

The ILS starts from a random initial solution. In order to escape from local optima and to explore new regions of the search space, ILS applies perturbations to the current solution. We used the four moves defined in SA that are randomly applied in our ILS procedure. The strength of the perturbation is the number of solution components (points) that are modified, and it is randomly defined in each iteration. A percentage φ ($\varphi \in [0.25, 0.60]$) of the number of points are altered by the perturbation.

We select the Location-Allocation and Interchange-Transfer heuristics as the local search of ILS. This heuristics performs a descent from a solution s' until it reaches a local minimum ($\hat{s}'$). Further details of these heuristics are presented in Section 4.3.

The acceptance criterion is biased towards diversification of the search since the best solution resulting from the local search phase is accepted if it improves the local minimum encountered so far or with a probability of 5%. And, the condition used to stop the algorithm was the maximal number of iterations, which was defined as 5000 iterations.

4.2 Clustering Process

Initially, we define the number of clusters $\mathcal{NC}$ as 20. The cluster centers should be generated in order to represent different regions of search space. This paper uses a greed method based on maximum diversity to create the initial centers, which generate a large set with $n(n >> \mathcal{NC})$ random solutions and a subset is selected with $\mathcal{NC}$ solutions that have the longest distance among themselves [16].

At each iteration of CS, one solution s_k is grouped into the closest cluster C_j; that is, the cluster that minimizes the distance between the solution and the cluster center. The volume δ_j is increased in one unit and the center c_j should be updated with new attributes of s_k (assimilation process).

The assimilation process uses the path-relinking method [25]. The procedure starts by computing the symmetric difference between the center c_j and the solution s_k; *i.e.*, the set of moves needed to reach s_k from c_j. A path of solutions is generated, linking c_j and s_k. At each step, the procedure examines all moves m from the current solution s and selects the one that results in the best-cost solution, applying the best move to solution s. The set of available moves is updated. The procedure terminates when 30% of the solutions in the path have been analyzed. The new center c_j is the best solution along this path. In this paper, one move is to swap one median of c_j by one median of s_k, changing the allocation of the points regarding the new median.

After performing the path-relinking, we must conduct an analysis of the volume δ_j, verifying if this cluster can be considered promising. A cluster becomes promising when its volume reaches the threshold λ ($\delta_j \geq \lambda$). The value of λ was define as 15 in this application.

Then, if the volume δ_j reached λ and the local search has been obtained success in this cluster, an exploitation is applied in center c_j by local search heuristics.

Finally, if there are clusters with lower δ or clusters in which local search often have been unsuccessful, instead of eliminating them, we apply a perturbation in the cluster center allowing it to move to another region of the search space. This random perturbation is performed by swapping 50% of allocations point/median.

4.3 Local Search

The VND method is implemented as local search of CS, intensifying the search in neighborhood of a promising cluster C_j.

The VND uses two heuristics: Location-Allocation and Interchange-Transfer [24]. This heuristics are applied using the solution for CPMP, and for each new generated solution is calculated the coordinates of the centroids and the value of the objective function of CCCP.

The Location-Allocation heuristic is based on the observation that the center c_j defines p groups, corresponding to the p medians and their allocated points. The center c_j can be improved by searching for a new median inside each cluster, swapping the current median by a non-median point and reallocating. We consider two steps for reallocating the points. The first one is to examine the points that were allocated to the current median and reallocate to the closest one. The second step is to examine the points assigned to the others medians and calculate the saving of moving them to the new one, if it improves the solution the point is reallocated to the new median.

The Interchange-Transfer heuristic tries a further improvement to the center c_j, which consists of two movements: interchange the allocation of two points and transfer one point from a median to another. All the possible movements are analyzed and the best one is performed.

If some heuristic obtains a better solution, VND returns to the first heuristic and the search continues from the better solution; otherwise, it changes to the next heuristic. The VND stopping condition is that are no more improvements can be made to the incumbent solution. The center c_j is updated if the new solution is better than the previous one.

4.4 Computational Results

The CS was coded in C and the computational tests executed on a 3 GHz Pentium 4. Four problem sets are used in this tests: two sets introduced by [24], that contains 6 instances named *sjc* and 5 instances named *p3038*, and two sets introduced by [2], that contains 7 instances named *ta* and 7 instances named *doni*.

Table 3 presents the results for the three approaches of CS. The entries in the table are the best solution (*best-sol*), the percentage deviation from the best solution found (*dev*) in 20 runs, the percentage deviation from the best-known solution (*gap*), the average running time to find the best solution (*best-time*) and the average running time (*time*) in seconds. The *imp* column shows the percentage of improvement of the CS's on the metaheuristics. The values in boldface show the best objective function value for each instance.

The results show the efficacy of the CS's. We observed that all approaches of CS had similar behavior, both in terms of solution quality and in running time. The CS found the best-known solution in the literature for 21 of the 25 instances tested.

The CS algorithm was very robust, producing low deviations (the average deviation, column *dev*, found by CS's were close to zero). And, the computational times of CS were competitive, finding very good solutions within few seconds for the smaller instances and in a reasonable time for the larger instances. We also note that the CS converges quickly to the best solution.

We can observe that SA, GA and ILS without the clustering process (column *imp*) gave solutions in poorer quality than CS. The best solutions found by CS were about 18,5%, 17,3% and 12,3% better than the best solutions found by the respective methods.

Table 4 reports a comparison with other methods of literature. Negreiros & Palhano [2] presented a two-phase heuristic using a Variable Neighborhood Search (VNS). Palhano *et al.* [22] developed a new polynomial constructive method (called CKMedians). And, Pereira & Senne [23] applied the column generation method in the CCCP.

Table 3 – CCCP: Results of the CS.

		CS-SA						CS-AG						CS-ILS					
Problem	Best-known	Best-sol	Dev	Gap	Best-time	Time	Imp	Best-sol	Dev	Gap	Best-time	Time	Imp	Best-sol	Dev	Gap	Best-time	Time	Imp
TA25	1251,44	**1251,44**	0,00	0,00	0,73	4,63	1,76	**1251,44**	0,00	0,00	0,68	2,16	1,76	**1251,44**	0,00	0,00	0,73	6,65	1,76
TA50	4474,52	**4474,52**	0,00	0,00	0,89	7,93	0,08	**4474,52**	0,00	0,00	0,99	5,52	0,00	**4474,52**	0,00	0,00	0,91	6,67	0,08
TA60	5356,58	**5356,58**	0,00	0,00	1,72	10,05	0,00	**5356,58**	0,00	0,00	1,05	9,13	0,00	**5356,58**	0,00	0,00	1,55	6,73	0,00
TA70	6240,67	**6240,67**	0,00	0,00	1,08	11,33	0,44	**6240,67**	0,00	0,00	0,77	8,78	0,44	**6240,67**	0,00	0,00	0,87	7,09	0,00
TA80	5515,46	5730,28	0,00	3,89	3,59	17,72	0,74	5730,28	0,00	3,89	2,59	22,77	0,79	5730,28	0,00	3,89	5,03	12,88	0,38
TA90	8899,05	9069,85	0,00	1,92	0,99	15,64	0,00	9069,85	0,00	1,92	1,26	22,11	0,70	9069,85	0,00	1,92	0,89	10,58	0,00
TA100	8102,04	**8102,04**	0,00	0,00	5,67	23,00	0,61	**8102,04**	0,00	0,00	11,24	48,59	1,08	**8102,04**	0,00	0,00	7,82	17,41	0,40
SJC1	17359,75	**17359,75**	0,01	0,00	5,53	24,96	0,00	**17359,75**	0,02	0,00	8,17	38,82	0,02	**17359,75**	0,01	0,00	7,88	19,28	0,17
SJC2	33181,65	**33181,65**	0,00	0,00	37,75	119,81	0,95	**33181,65**	0,00	0,00	40,37	179,42	0,43	**33181,65**	0,00	0,00	11,74	104,84	2,25
SJC3a	45354,38	**45354,38**	0,08	0,00	258,35	486,20	1,38	45358,23	0,06	0,01	509,66	1206,72	2,92	45359,09	0,11	0,01	135,98	388,32	2,91
SJC3b	40661,94	40663,44	0,06	0,00	209,16	540,90	1,87	**40661,94**	0,02	0,00	771,83	1433,80	4,08	**40661,94**	0,04	0,00	261,91	458,79	3,06
SJC4a	61931,60	**61931,60**	0,08	0,00	473,86	1249,87	3,95	**61931,60**	0,04	0,00	1092,97	3025,72	6,54	61944,86	0,05	0,02	401,17	893,54	3,32
SJC4b	52214,55	**52214,55**	0,16	0,00	617,73	1645,37	2,80	52227,60	0,06	0,02	1965,82	3995,20	6,75	52227,11	0,07	0,02	540,30	1199,82	4,84
p3038_600	128203,40	128974,03	0,40	0,60	5459,54	9991,66	51,74	128419,95	0,23	0,17	6137,67	9736,80	48,89	**128203,40**	0,51	0,00	3842,84	9436,96	51,26
p3038_700	116051,88	116158,34	0,69	0,09	9205,09	11675,07	53,96	116325,05	0,31	0,24	6848,52	11658,11	53,09	**116051,88**	0,66	0,00	5359,16	10915,27	51,70
p3038_800	106961,20	**106961,20**	1,12	0,00	9210,10	13368,57	57,01	107764,69	0,32	0,75	8335,36	13194,76	60,89	107209,82	0,66	0,23	5893,56	12810,04	52,16
p3038_900	99756,59	**99756,59**	0,75	0,00	11389,33	15049,56	59,14	99968,15	0,44	0,21	11726,17	15341,74	67,04	100133,85	0,65	0,38	5386,27	13931,29	57,59
p3038_1000	92706,38	93148,68	0,53	0,48	14184,79	18698,52	64,59	**92706,38**	1,22	0,00	10747,13	17128,41	77,40	92850,88	1,04	0,16	4474,16	15451,21	63,87
DONI1	3021,41	3022,26	0,25	0,03	34,39	76,22	10,70	3027,63	0,44	0,21	86,38	127,04	3,12	3026,87	0,30	0,18	70,41	105,91	0,77
DONI2	6080,70	6372,81	0,21	4,80	112,12	242,60	4,52	6373,26	0,02	4,81	309,77	687,48	0,34	6377,77	0,05	4,89	316,78	410,20	0,67
DONI3	8438,96	8446,08	0,62	0,08	203,42	543,19	13,67	**8438,96**	0,51	0,00	624,12	928,41	6,01	8470,18	0,67	0,37	711,60	924,60	0,43
DONI4	10854,48	**10854,48**	0,62	0,00	249,45	1011,08	19,98	10952,27	0,38	0,90	1069,07	2389,58	1,62	11094,62	0,88	2,21	1395,03	1681,27	0,00
DONI5	11134,94	**11134,94**	0,75	0,00	763,65	1617,99	17,90	11209,99	0,11	0,67	2175,04	3624,40	1,17	11182,87	0,19	0,43	2297,91	2730,88	1,36
DONI6	15722,67	15814,56	0,56	0,58	3326,34	8705,53	45,07	**15722,67**	0,34	0,00	6174,83	10316,55	22,29	15780,97	0,34	0,37	10005,59	14466,46	2,88
DONI7	18596,74	19038,91	0,47	2,38	10193,80	18926,76	49,56	**18596,74**	0,82	0,00	15860,55	26913,91	60,87	18623,59	1,99	0,14	20649,82	29751,42	5,43
Average			0,29	0,59	2637,96	4162,57	18,50		0,21	0,55	2980,08	4881,84	17,13		0,33	0,61	2471,20	4629,92	12,29

Table 4 – CCCP: Comparison of the results.

		Negreiros & Palhano[2]		Palhano *et al.* [22]		Pereira & Senne[23]		CS	
Problem	Best-known	Best-sol	Gap	Best-sol	Gap	Best-sol	Gap	Best-sol	Gap
TA25	1251,44	**1251,44**	0,00	–	–	1280,49	2,32	**1251,44**	0,00
TA50	4474,52	4476,12	0,04	–	–	**4474,52**	0,00	**4474,52**	0,00
TA60	5356,58	**5356,58**	0,00	–	–	5357,34	0,01	**5356,58**	0,00
TA70	6240,67	6241,55	0,01	–	–	**6240,67**	0,00	**6240,67**	0,00
TA80	5515,46	5730,28	3,89	–	–	**5515,46**	0,00	5730,28	3,89
TA90	8899,05	9103,21	2,29	–	–	**8899,05**	0,00	9069,85	1,92
TA100	8102,04	8122,67	0,25	–	–	8168,36	0,82	**8102,04**	0,00
SJC1	17359,75	17696,53	1,94	20341,34	17,18	17375,36	0,09	**17359,75**	0,00
SJC2	33181,65	33423,84	0,73	35211,99	6,12	33357,75	0,53	**33181,65**	0,00
SJC3a	45354,38	47985,29	5,80	50590,49	11,54	45379,69	0,06	**45354,38**	0,00
SJC3b	40661,94	–	–	–	–	41185,18	1,29	**40661,94**	0,00
SJC4a	61931,60	66689,96	7,68	69283,05	11,87	61969,06	0,06	**61931,60**	0,00
SJC4b	52214,55	–	–	–	–	52989,44	1,48	**52214,55**	0,00
p3038_600	128203,40	192024,83	49,78	135481,99	5,68	–	–	**128203,40**	0,00
p3038_700	116051,88	176731,07	52,29	123698,76	6,59	–	–	**116051,88**	0,00
p3038_800	106961,20	184502,38	72,49	117705,48	10,05	–	–	**106961,20**	0,00
p3038_900	99756,59	176781,51	77,21	111033,27	11,30	–	–	**99756,59**	0,00
p3038_1000	92706,38	159139,89	71,66	110049,78	18,71	–	–	**92706,38**	0,00
DONI1	3021,41	**3021,41**	0,00	3234,58	7,06	–	–	3022,26	0,03
DONI2	6080,70	**6080,70**	0,00	6692,71	10,06	–	–	6372,81	4,80
DONI3	8438,96	8769,05	3,91	9797,12	16,09	–	–	**8438,96**	0,00
DONI4	10854,48	11516,14	6,10	11594,07	6,81	–	–	**10854,48**	0,00
DONI5	11134,94	11635,18	4,49	11827,69	6,22	–	–	**11134,94**	0,00
DONI6	15722,67	18443,50	17,31	–	–	–	–	**15722,67**	0,00
DONI7	18596,74	23478,79	26,25	–	–	–	–	**18596,74**	0,00
average			**17,57**		**10,38**		**0,51**		**0,43**

The proposed method presents new best-known solutions for 17 instances. And for the others instances, the solutions obtained by CS were close to the best-known solutions. We can observe that the average deviation from the best-known solution (column *gap*) of the CS was close to zero (0.43%).

The CS found the best-known solutions for all instances of the *sjc* and *p3038* sets. For the instances in the *ta* set, the CS found the best-known solutions in 5 of 7 instances. For the last class of instances (*doni*), the CS found the best solutions in 5 of 7 instances.

5 CONCLUSIONS

This paper presents a new method of detecting promising search areas based on clustering: the Clustering Search (CS). In a general way, CS attempts to locate promising search areas by cluster of solutions. The clusters work as sliding windows, framing the search areas and giving a reference point (center) to problem-specific local search procedures. Furthermore, the cluster center itself is always updated by a permanent interaction with inner solutions, in a process called assimilation. Therefore, the process of detecting promising areas becomes an interesting alternative, preventing the indiscriminate application of local search methods.

In this paper, research work on CS has been surveyed. The advances in CS designs, CS for complicated optimization problems and some applications in real problems, have been covered. The CS is still in its early stage, although there have been a large number of publications.

A version of CS for Capacitated Centred Clustering Problem (CCCP) was presented as a case study. We use three metaheuristics to generate solutions to the CS (Genetic Algorithm, Simulated Annealing and Variable Neighborhood Search), showing that the CS can be used with different methods.

The computational results were examined from two viewpoints: the best metaheuristic for CS and comparison against other literature methods. The three approaches to CS had similar behavior, finding solutions with a small gap to the best known solution. Moreover, the CS's showed robust with a small deviation between the average solution and the best solution found in 20 runs. The computational time was also similar in the three approaches, finding good solutions in a reasonable time. In comparison against other literature methods, CS has achieved superior performance in terms of solution quality in most instances.

For further work, it is intended to build new algorithms based on CS, including other metaheuristics as Ant Colony System, Tabu Search and Large Neighborhood Search. And also exploring other aspects of CS, such as: parallelize their components and create a CS for multiobjective optimization problem.

ACKNOWLEDGMENTS

The authors acknowledge CNPq (Process 300692/2009-9, 470813/2010-5 and 558084/ 2009-5) for their financial support.

REFERENCES

[1] MULVEY J & BECK P. 1984. Solving capacitated clustering problems. *European Journal of Operational Research*, **18**(3): 339–348.

[2] NEGREIROS MJ & PALHANO AW. 2006. The capacitated centred clustering problem. *Computers and Operations Research*, **33**(6): 1639–1663.

[3] DIGALAKIS J & MARGARITIS K. 2002. An experimental study of benchmarking functions for genetic algorithms. *International Journal of Computer Mathematics*, **79**(4): 403–416.

[4] OLIVEIRA ACM & LORENA LAN. 2004. Detecting promising areas by evolutionary clustering search. In *Advances in Artificial Intelligence*. Bazzan ALC and Labidi S. (Eds.) Springer Lecture Notes in Artificial Intelligence Series, **3171**: 385–394.

[5] NAGANO MS, RIBEIRO FILHO & LORENA LAN. 2006. Metaheuristica Híbrida Algoritmo Genético-Clustering Search para Otimização em Sistemas de Produção Flow Shop Permutacional. *Learning and Nonlinear Models*, **4**(1): 32–42.

[6] OLIVEIRA ACM & LORENA LAN. 2006. Pattern Sequencing Problems by Clustering Search. *Springer Lecture Notes in Artificial Intelligence Series*, **4140**: 218–227.

[7] OLIVEIRA ACM & LORENA LAN. 2007. Hybrid Evolutionary Algorithms and Clustering Search. *Hybrid Evolutionary Systems: Studies in Computational Intelligence*, **75**: 81–102.

[8] CHAVES AA, CORREA FA & LORENA LAN. 2007. Clustering Search Heuristic for the Capacitated p-median Problem. *Springer Advances in Software Computing Series*, **44**: 136–143.

[9] RIBEIRO FILHO G, NAGANO MS & LORENA LAN. 2007. Evolutionary Clustering Search for Flowtime Minimization in Permutation Flow Shop. *Lecture Notes in Computer Science*, **4771**: 69–81.

[10] BIAJOLI FL & LORENA LAN. 2007. Clustering Search Approach for the Traveling Tournament Problem. *LNAI*, **4827**: 83–93.

[11] CHAVES AA & LORENA LAN. 2008. Hybrid Metaheuristic for the Prize Collecting Traveling Salesman Problem. *Lecture Notes in Computer Science*, **4972**: 123–134.

[12] CORREA FA, CHAVES AA & LORENA LAN. 2008. Hybrid heuristics for the probabilistic maximal covering location-allocation problem. *Operational Research: an International Journal*, **7**: 323–343.

[13] CHAVES AA, LORENA LAN & MIRALLES C. 2009. Hybrid metaheuristic for the Assembly Line Worker Assignment and Balancing Problem. *Lecture Notes in Computer Science*, **5818**: 1–14.

[14] CHAVES AA & LORENA LAN. 2010. Clustering Search Algorithm for the Capacitated Centered Clustering Problem. *Computers and Operations Research*, **37**(3): 552–558.

[15] COSTA T, OLIVEIRA ACM & LORENA LAN. 2010. Advances in Clustering Search. *Advances in Soft Computing*, **73**: 227–235.

[16] CHAVES AA & LORENA LAN. 2011. Hybrid Evolutionary Algorithm for the Capacitated Centered Clustering Problem. *Expert Systems with Applications*, **38**: 5013–5018.

[17] OLIVEIRA RM, MAURI GR & LORENA LAN. 2012. Clustering Search for the Berth Allocation Problem. *Expert Systems with Applications*, **39**: 5499–5505.

[18] HOLLAND JH. 1975. Adaptation in Natural and Artificial Systems. Technical Report. University of Michigan Press, Michigan, USA.

[19] KIRKPATRICK S, GELLAT DC & VECCHI MP. 1983. Optimization by simulated annealing. *Science*, **220**: 671–680.

[20] STÜTZLE T. 1999. Iterated local search for the quadratic assignment problem. Tese (Doutorado), TU Darmstadt.

[21] SYSWERDA G. 1989. Uniform crossover in genetic algorithms. In: Proceedings of International conference on genetic algorithms: 2–9.

[22] PALHANO AWC, NEGREIROS MJ & LAPORTE G. 2008. A constrained k-median procedure for the capacitated centered clustering problem. In: XIV CLAIO, Congreso Latino Ibero Americano de Investigación de Operaciones. CD-ROM.

[23] PEREIRA MA & SENNE ELF. 2008. A column generation method for the capacitated centred clustering problem. In: VI ALIO/EURO workshop on applied combinatorial optimization: 1–6.

[24] LORENA LAN & SENNE ELF. 2003. Local search heuristics for capacitated p-median problems. *Networks and Spatial Economics*, 3(4): 407–419.

[25] GLOVER F. 1996. Tabu search and adaptive memory programing. *Interfaces in computer science and operations research*: 1–75.

[26] YAGER RR. 1990. A model of participatory learning. *IEEE Transaction on Systems, Man and Cybernetics*, **20**(5): 1229–1234.

[27] GLOVER F, LAGUNA M & MARTÍ R. 2000. Fundamentals of scatter search and path relinking. *Control and Cybernetics* **39**: 653–684.

[28] OLIVEIRA ACM. 2004. Algoritmos evolutivos híbridos com detecção de regiões promissoras em espaços de busca contínuos e discretos. Tese de doutorado em Computação Aplicada, INPE, São José dos Campos, Brazil.

[29] CHAVES AA. 2009. Uma meta-heurística híbrida com busca por agrupamentos aplicada a problemas de otimização combinatória. Tese de doutorado em Computação Aplicada, INPE, São José dos Campos, Brazil.

[30] KAISER MS. 2009. Aplicação de metaheurística híbrida na resolução do problema dial-a-ride. Dissertação de mestrado em Engenharia de Transportes, COPPE, Universidade Federal do Rio de Janeiro, Brazil.

[31] ALMEIDA WG. 2009. Métodos Heurísticos para Problemas de Localização de Concentradores. Dissertação de Mestrado em Computação Aplicada, INPE, São José dos Campos, Brazil.

[32] RIBEIRO FILHO G, NAGANO MS & LORENA LAN. 2007. Hybrid Evolutionary Algorithm for Flowtime Minimisation in No-Wait Flowshop Scheduling. *MICAI 2007, LNAI 4827*, Springer-Verlag: 1099–1109.

[33] RIBEIRO GM, LAPORTE G & MAURI GR. 2011. A comparison of three metaheuristics for the workover rig routing problem. *European Journal of Operational Research*: <http://dx.doi.org/10.1016/j.ejor.2012.01.031>.

[34] ALMEIDA WG & SENNE ELF. 2010. Metaheurística Híbrida com busca por agrupamento aplicado ao problema de localização de hubs com restrições de capacidade. *XLII SBPO 2010*.