Search Algorithms with Randomized Variable Neighborhoods for Solving Series of Clustering Problems

Ilnar Nasyrov, Mikhail Gudyma, Lev Kazakovtsev, Dmitry Stashkov

Problem Statement

- **The k-means problem**

$$\arg\min F(X_1, \dots, X_k) = \sum_{i=1}^{N} min_{j \in \overline{\{1,k\}}} \left\| X_j - A_j \right\|^2$$

- **The p-medium problem**

$$\arg\min F(X_1, \dots, X_k) = \sum_{i=1}^{N} min_{j \in \overline{\{1,k\}}} L(X_j, A_j)$$

Algorithm 1. The k-means algorithm (KMEANS)

Given: data vectors $A_1 \ldots A_N$, k of the initial centers of clusters $X_1 \ldots X_k$.

- 1. For each center X_i, compile a cluster (set) of C_i data vectors for which this center is the closest one.
- 2. For each cluster C_i, calculate the new value of the center X_i (i.e., to solve the Weber problem).
- 3. Repeat from step 1 if steps 1 and 2 have led to any changes in at least one center value.

Algorithm 2. GA with a greedy heuristic crossingover procedure and a real alphabet

Given: A set $V = (A_1, \dots, A_N) \in \mathcal{R}^d$, a number of cluster *p*, population size N_{POP}.

Loop

1. Create N_{POP} coordinate sets $X_1, \dots, X_{N_p}$: $X_i \subset \mathcal{R}^d$, $|X_k| = p$ $k = \overline{1, N_p}$, resulting from the execution of the KMEANS procedure. Thus, a local minimum of the problem is achieved in each χ_i. Memorize the corresponding values of the objective function in variables $F_1, \dots, F_{N_p}$.
2. If the STOP conditions (time limitation) are reached, then STOP; the result is the set x_k^*, $k^* = \arg min_{k=\overline{1,N_{POP}}} F_k$.
3. Select randomly two “parent” sets x_{k_1} and x_{k_2}, $k_1, k_2 \in \{\overline{1, N_{POP}}\}$, $k_1 \neq k_2$. Starting Algorithm 3, get a "child" set of coordinates X_c in which the local minimum of the objective function is achieved. Store the value of the objective function in the variable F_c.

Algorithm 2. GA with a greedy heuristic crossingover procedure and a real alphabet

4. **If** $\exists k \in \overline{\{1, N_{POP}\}}: x_k = x_c$ **then** go to Step 2.
5. Select an index $k_{worst} = \arg min_{k=\overline{1,N_{POP}}} F_k$. If $F_{worst} < F_c$ then go to Step 2.
6. Select randomly two indices $k_1 \; and \; k_2$, $k_1 \neq k_2$;
   Select $k_{worst} = \arg min_{k=\overline{1,N_{POP}}} F_k$.
7. Swap the values $x_{k_{worst}} \; and \; x_c$, save $F_{k_{worst}} = F_c$.

End Loop

Algorithm 3. A greedy crossing-over procedure for Algorithm 2

Given: The set $V = (A_1, \dots, A_N) \in \mathcal{R}^d$, the number of cluster p, two "parent" sets of centers x_{k_1} and x_{k_2}, the values of σ_e and L_{min}.

1. Unify sets $x_c = x_{k_1} \cup x_{k_2}$. Run the KMEANS procedure for $|x_c|$ clusters, starting with the solution x_c . Save result to x_c.

Loop

2. **If** $|x_c| = p$ **then** start the KMEANS procedure from the initial solution x_c (i.e., assign x_c=KMEANS(x_c)) ; STOP the algorithm; the result is x_c.

   2.1 Calculate the distances from each of the data vectors to the nearest element of the set x_c :

   $d_i = min_{X \in x_c} L(X, A_i)$ $i = \overline{1, N}$. For each of the data vectors, determine the nearest center x_c:

   $C_i = min_{X \in x_c} L(X, A_i)$ $i = \overline{1, N}$. Calculate the distance from each of the data vectors to the second element (center) closest to it from the set x_c: $D_i = min_{Y \in (x_c \setminus \{c_i\})} L(X, A_i)$.

Algorithm 3. A greedy crossing-over procedure for Algorithm 2

3. For each $X \in x_c$: calculate $\delta_X = F(x_c \backslash \{X\}) = \sum_{i:C_i=X}(D_i - d_i)$.

4.1 Calculate $n_\delta = \max\{[(|x_c| - p) * \sigma_e], 1\}$. Arrange the values δ_X and select a subset $x_{elim} = \{X_1, \dots, X_{n_\delta}\} \subset x_c$ of the n_δ points that correspond to the minimum values of δ_X.

4.2 For each $j \in \{\overline{2, |x_{elim}|}\}$ **If** $\exists k \in \{\overline{1, j-1}\}: L(X_j, X_k) < L_{min}$ then remove X_j from x_{elim}.

4.3 Assign $x_c = x_c \backslash x_{elim}$.

4.4 Redistribute data vectors between the centers closest to them $C_i = argmin_{X \in x_c} L(X, A_i)$ $i = \overline{1, N}$.

4.5 For each $X \in x_c$ **If** $\exists i \in \{\overline{1, N}\}: C_i = X$ and $C_i^* \neq X$, then recalculate the center of the X^*cluster $C_X^{clust} = \{A_i | C_i^* = X, i = \overline{1, N}\}$. Do assign $x_c = (x_c\{X^*\}) \cup \{X\}$.

End Loop

Algorithm 4. A genetic algorithm with a greedy heuristic and a partially unified solution for the p-median problem

Given: Population size N_{POP}, number of cluster *p*.

1. Generate (randomly or using the k-means ++ procedure) N_{POP} of various initial solutions $X_1, \dots, X_{N_{POP}} \subset \{\overline{1, N}\}, |x_i| = p, i = \overline{1, N_{POP}}$ which are the sets of indices of data vectors of power *p* used as initial solutions of the KMEANS algorithm. For each of the initial solutions, evaluate the value of the objective function $F_{fitness}(X)$, save the values of this function in variables $f_1, \dots, f_{N_{POP}}$.

Loop

2. **If** the STOP conditions are reached, **then** STOP. The solution is the solution x_{i^*} which corresponds to the smallest f_i value. Find the final solution, the KMEANS algorithm is launched (Algorithm 1).
3. Select randomly two indexes $k_1, k_2 \in \{\overline{1, N}\}$, $k_1 \neq k_2$.

Algorithm 4. A genetic algorithm with a greedy heuristic and a partially unified solution for the p-median problem

4.1 Using a random number generator, select an integer $r \in \{\overline{1,p}\}$.

4.2 From the set x_{k_2}, select randomly a subset $x^*_{k_2}$ of cardinality *r*.

4.3 Get an intermediate solution $x_c = x_{k_1} \cup x^*_{k_2}$.

5 **If** $\exists i \in \{\overline{1,N}\}: x_i = x_c$ **then** Go to Step 2.

6 **If** $|x_c| \leq p$, **then** go to Step 7.

7 Calculate $j^* = \arg min_{j \in x_c} F_{fitness}(X_c\{j\})$. Exclude j^* from x_c: x_c=$x_c \backslash \{j^*\}$.

Go to Step 5.

8 **If** $\exists i \in \{\overline{1,N_{POP}}\}: x_i = x_c$ **then** Go to Step 2.

9 Select an index $k_3 \in \{\overline{1,N_{POP}}\}$ using tournament selection: two indexes are randomly selected $k_4, k_5 \in \{\overline{1,N_{POP}}\}$ if $f_{k_4} > f_{k_5}$ then assign k_3=k_4 else k_3=k_5

10 Replace x_{k_3} and corresponding value of the objective function: $x_{k_3} = x_c$, $f_{k_3} = F_{fitness}(X_c)$

End Loop

Algorithm 5. A genetic algorithm with a greedy heuristic for solving a series of problems with the number of clusters $p \in \{2, p_{max}\}$

1. Initialization of a population of N_{POP} individuals. Each individual is a set of p_{max} centers (we denote them by X_i). Assign $F_{new,j} = +\infty$ for each $j \in \{1, N_{POP}\}$. Initialize the arrays of values of the objective function $F_k^* = +\infty$ and the best solutions $X_k^* = \{\}$ for each $k \in \{2, p_{max}\}$.

Loop

2. Randomly select $j_1, j_2 \in [1, N], j_1 \neq j_2$.
3. $X_{new} = X_{j_1} \cup X_{j_2}$
4. **Do while** $|X_{new}| > p_{max}$:

   4.1 Select an element j such that its exclusion gives the smallest increase in the objective function $j = \arg min_{i \in X_{new}} F(X_{new} \backslash \{j\})$

   4.2 $X_{new} = X_{new} \backslash \{j\}$

   End Do

Algorithm 5. A genetic algorithm with a greedy heuristic for solving a series of problems with the number of clusters

5. Assign $F_{new} = 0$, $X^* = X_{new}$.
6. **Do while** $|X_{new}| > 2$:

   6.1 Assign F_{new}=$F_{new} + f(X_{new})$; $k = |X_{new}|$; $F_k = f(X_{new})$; **if** $F_k < F_k^*$ **then** assign F_k^*=F_k;

   6.2 Follow steps 4.1 and 4.2 for X_{new}.

   End Do
7. Select j_3 using tournament selection based on the value of $F_{new,j}$. Assign $F_{j_3} = F_{new}$; $X_{j_3} = X^*$, $F_{new,j_3} = F_{new}$.
8. Check the STOP conditions.

End Loop

Algorithm 5. A genetic algorithm with a greedy heuristic for solving a series of problems with the number of clusters $p \in \{2, p_{max}\}$

1. The fitness function F_{new} differs from the objective function of the problem;
2. The objective functions for each value of p are summarized to form a new utility function;
3. This algorithm has an advantage only for solutions with the number of clusters close to p_{max} and gives worse results, in comparison with other methods, for tasks with the number of clusters significantly different from p_{max}.

Previous studies

In previous our papers proposed :

1. to stop the exclusion of clusters at approximately $p = p_{max}/3$, and to obtain other solutions, for example, by rerunning the algorithm with a lower p_{max} value*.
2. a simple modification of the serial algorithm, where the population size monotonically increases with the iteration number**.

* Gudyma, M.N., Kazakovtsev, L.A. Evolutionary serial algorithms with the dynamic and heterogeneous populations for the automatic grouping. Sistemyupravleniyaiinformacionnyetekhnologii. 2(68), 33-38 (2017)

** Kazakovtsev, L.A., Antamoshkin, A.N. and Fedosov, V.V.: Greedy heuristic algorithm for solving series of eee components classification problems. IOP Conf. Series: Materials Science and Engineering 122, Article ID 012011 (201

6), https://doi.org:10.1088/1757-899X/122/1/012011.

Algorithm 6. GA with a greedy heuristic and a dynamic population for solving a series of problems with $p \in \{2, p_{max}\}$

1. Initialization of the initial population of the initial size of $N_{popstart}$ individuals. Each individual is a set of p_{max} centers (we denote them by X_i). Assign $F_{new,j} = +\infty$ for each $j \in \{1, N_{popstart}\}$. Initialize the arrays of values of the objective function $F_k^* = +\infty$ and the best solutions $X_k^* = \{\}$ for each $k \in \{2, p_{max}\}$. $N_{iter} = 0$.

Loop

2. $N_{iter} = N_{iter} + 1; N_{pop} = \max\{N_{popstart}[\sqrt{1 + N_{iter}}] + 2\}$; If N_{pop} has changed then initialize the N_{pop} individual as in step 1. Randomly select $j_1, j_2 \in [1, N_{pop}], j_1 \neq j_2$.
3. $X_{new} = X_{j_1} \cup X_{j_2}$
4. **Do while** $|X_{new}| > p_{max}$:

   4.1 Select an element j such that its exclusion gives the smallest increase in the objective function $j = \arg min_{i \in X_{new}} F(X_{new} \backslash \{j\})$

   4.2 $X_{new} = X_{new} \backslash \{j\}$

   End Do

Algorithm 6. GA with a greedy heuristic and a dynamic population for solving a series of problems with $p \in \{2, p_{max}\}$

5. Assign $F_{new} = 0$, $X^* = X_{new}$.
6. **Do while** $|X_{new}| > 2$:

   6.1 Assign F_{new}=$F_{new} + f(X_{new})$; $k = |X_{new}|$; $F_k = f(X_{new})$; **if** $F_k < F_k^*$ **then** assign F_k^*=F_k;

   6.2 Perform steps 4.1 and 4.2 for X_{new}.

   End Do
7. Select j_3 using tournament substitution based on the value of $F_{new,j}$. Assign $F_{j_3} = F_{new}$; $X_{j_3} = X^*$, $F_{new,j_3} = F_{new}$.
8. Check the STOP conditions.

End Loop

Algorithm 7. Genetic algorithm with a heterogeneous population for the p-median problem

1. Initialization of a population of N_{pop} individuals. Each individual is a set of a different number of centers (we denote them by X_i). Power $|X_i|$ randomly selected $|X_i| \in \{2, p_{max}\}$. Initialize the arrays of the best values of the objective function $F_k^* = +\infty$ and the best solutions $X_k^* = \{\}$ for each $k \in \{2, p_{max}\}$ and the arrays of values of the objective function of each individual for each number of clusters $F_{k,i} = +\infty$, $k \in \{2, p_{max}\}$, i $\in \{1, N_{pop}\}$.

Loop

2. Randomly select $j_1, j_2 \in [1, N_{pop}], j_1 \neq j_2$.
3. $X_{new} = X_{j_1} \cup X_{j_2}$
4. **Do while** $|X_{new}| > p_{max}$:

   4.1 Select an element j such that its exclusion gives the smallest increase in the objective function $j = \arg min_{i \in X_{new}} F(X_{new} \backslash \{j\})$

   4.2 $X_{new} = X_{new} \backslash \{j\}$

   End Do

Algorithm 7. Genetic algorithm with a heterogeneous population for the p-median problem

5. Choose $p_{child} \in \{2, p_{max}\}$ randomly. **If** $p_{child} > |X_{new}|$ **then** $p_{child} = |X_{new}|$;
6. Assign $f_{child,|X_{new}|} = f(X_{new})$;
7. **Do while** $|X_{new}| > p_{child}$:

   7.1 Select an element j such that its exclusion gives the smallest increase in the objective function $j = \arg min_{i \in X_{new}} f_{child}(X_{new} \backslash \{j\})$

   7.2 $X_{new} = X_{new} \backslash \{j\}$

   End Do

Algorithm 7. Genetic algorithm with a heterogeneous population for the p-median problem

8. **Do while** $|X_{new}| > 2$:

   8.1 Assign $f_{child,|X_{new}|} = f(X_{new})$; $k = |X_{new}|$; $f_{k,|X_{new}|} = f(X_{new})$;

   8.2 If $f_{k,|X_{new}|} < F_k^* = f_{k,|X_{new}|}$;

   8.3 Follow steps 4.1 and 4.2 for X_{new}

   End Do

9. Select $j_3 \in \{1, N_{pop}\}$ using tournament selection. Assign $X_{j_3} = X_{new}$; $f_{j_3,k} = f_{child}$
10. Check the shutdown conditions.

End Loop.

Algorithm 8. Serial local search algorithm with greedy heuristic

0. Initialization: Generate randomly $\chi_I = \{X_1, \ldots, X_l\}$ which is a set (population) of solutions $l \in \{\overline{2, K}\}$; select randomly K centers for χ_k;

For all l' from K-1 down to 2:

assign $\chi_{l'} = GREEDHEUR(\chi_{I+1})$;

assign $\chi_{I'}$ =KMEANS $(\chi_{l'})$;

End For.

Repeat

1. Choose randomly $k' \in \{\overline{2, K}\}$, while the probability of choice is equal $\frac{2k'(K-1)}{K+2}$, to increase the probability of large values of k'.
2. Generate randomly χ' with k' centroids.

Algorithm 8. Serial local search algorithm with greedy heuristic

3. Assign $\chi' = KMEANS\ (\chi')$.
4. Unify$\chi'' = \chi_{k'} \cup \chi'$.
5. Assign $\chi'' = KMEANS\ (\chi'')$.
6. **Do while** $|\chi''| > K$:

   6.1Assign χ''=GREEDHEAR (χ'');

   6.2 Assign $\chi'' = KMEANS\ (\chi'')$;

   End Do

Algorithm 8. Serial local search algorithm with greedy heuristic

7. **Do while** $|\chi''| > 2$:

   7.1 **If** $F\left(\chi_{|\chi''|}\right) > F(\chi'')$ **then**

   assign $\chi_{|\chi''|}$=GREEDHEAR (χ'');

   ϕssign $\chi'' = KMEANS\ (\chi'')$;

   End If

   End DO

8. **Until** the STOP condition is met

Algorithm 9. Procedure GREEDHEUR(χ)

Given: Initial solution $\chi = \{X_1, \dots, X_l\}$

1. Assign $k' = \arg min_{k=\overline{1,l}} F(\chi \backslash X_k)$
2. Return $\chi \backslash X_{k'}$

Algorithm 10. A search algorithm with randomized variable neighborhoods formed by application of greedy agglomerative procedures

0. Initialization: Generate randomly $\chi_I = \{X_1, \dots, X_l\}$ which is a set of solutions $l \in \{\overline{2, K}\}$; select randomly K centers for χ_k;

Assign χ_k= KMEANS(χ_k);

For all l' from K-1 to 2:

assign $\chi_{l'} = GREEDHEUR(\chi_{I+1})$;

assign $\chi_{I'}$ =KMEANS ($\chi_{l'}$);

End For.

Assign O=1; //Current type of neighborhood

Assign StepsWOImprove = 0; // number of steps without improvement

Repeat

1. Choose randomly $k' \in \{\overline{2, K}\}$, while the probability of choice is equal $\frac{2k'(K-1)}{K+2}$, to increase the probability of large values of k'.
2. Generate randomly χ' with k' centers.

Algorithm 10. A search algorithm with randomized variable neighborhoods formed by application of greedy agglomerative procedures

3. Assign $\chi' = KMEANS\ (\chi')$.
4. **If** O=3 **then**

    4.1 Unify$\chi'' = \chi_{k'} \cup \chi'$; SerialGreed()

    4.2 **ElseIf** O=1 **then**

    4.3 For i=1 to k' do:

        4.3.1 Unify$\chi'' = \chi_{k'} \cup \{\chi'\}$; SerialGreed()

    End For.

Algorithm 10. A search algorithm with randomized variable neighborhoods formed by application of greedy agglomerative procedures

4.4 **Elself** O=2 **then**

4.4.1 **For** i=1 to k' **do**:

4.4.2 Generate random $r \sim U\{0,1\}$;

4.4.3 Assign $r = [r^2 * (k' - 1)] + 1$;

4.4.4 Choose randomly $\chi''' \in \chi'$;

4.4.5 Unify$\chi''' = \chi_{k'} \cup \chi'''$; SerialGreed()

End For.

5. **If** StepsWOImprove≥StepsMax **then**

5.1 Assign O=O+1;

5.2 **If** O>3 **then**

5.2.1 Assign O=1; StepsWOImprove=0

6. **Until** the stop condition is not met

Algorithm 11. Procedure SerialGreed ()

1. $\chi'' = \text{KMEANS}\ (\chi'')$.
2. StepsWOImprove= StepsWOImprove+1;
3. Do while $|\chi''| > K$:

   Assign $\chi'' = GREEDHEAR\ (\chi'')$

   Assign $\chi'' = KMEANS\ (\chi'')$
4. Do while $|\chi''| > 2$:

   If $F\left(\chi_{|\chi''|}\right) > F(\chi'')$ **then**

   $\chi_{|\chi''|} = \chi''$; StepsWOImprove=0

   Assign $\chi'' = GREEDHEAR\ (\chi'')$

   Assign $\chi'' = KMEANS\ (\chi'')$

Computational Experiments

Dataset,p,N	Par.	SVNGH	SLGH	HetPop	kmeans
2D522B p=10 N=3711	F(x)	11775.45	11775.68	11776.48	**11777.89**
	σ	2.471	0.095	1.879	**1.023**
2D522B p=15 N=3711	F(x)	9452.31	9452.29	9456.17	**9456.83**
	σ	3.932	0.911	2.621	**4.817**
140UD25 p=10 N=523	F(x)	974.19	974.12	974.51	**974.76**
	σ	0.587	0.319	0.370	**0.283**
140UD25 p=20 N=523	F(x)	658.16	658.87	661.23	**667.68**
	σ	0.687	0.523	1.081	**1.188**
1526TL1 p=10 N=1234	F(x)	3441.14	3441.58	3442.62	**3454.09**
	σ	1.547	1.258	2.492	**17.296**
Ion p=3 N=351	F(x)	8448.63	8449.47	8450.95	**8451.55**
	σ	2.476	2.569	2.784	**2.444**
Mopsi p=20 N=6014	F(x)	208.11	208.65	208.76	**209.98**
	σ	**0.793**	**0.987**	**1.473**	**0.836**

Conclusions

- Developed local search algorithms with greedy heuristic procedures and randomized neighborhoods allow us to obtain results simultaneously for a series of problems.
- These results are not inferior in accuracy to the previously developed serial algorithms and well-known algorithms for solving a single problem.
- At the same time, the new algorithm allows us to solve simultaneously a series of problems with a different number of clusters which is important if the number of clusters is unknown.