Systems and Computing
Volume 2, Issue 1, 1-17, 2026.
https://doi.org/10.64409/sycom.v2.i1.22

Research Article Open Access

Avoiding Local Minima in Overlap-Based Clustering by Random Swap

Fei Wang, Le Li, Pasi Fränti
School of Computing, University of Eastern Finland, Joensuu, Finland

Corresponding author: Pasi Fränti (franti@cs.uef.fi)

Manuscript Review Record:
Submitted:
August 22, 2025

Accepted:
October 26, 2025

Published:
January 15, 2026

Cite This:
Fei Wang, Le Li, & Pasti Fränti. "Avoiding local minima in overlap-based clustering by random swap". *Systems and Computing*. Volume 2, Issue 1, 1-17, 2026. https://doi.org/10.64409/sycom.v2.i1.22

Abstract- Context: Cluster overlap has been recently introduced as an alternative objective function to the traditional sum-of-squared error (SSE) function used in k-means. It estimates the overlap by counting how many points are shared between the neighbor clusters. The overlap k-means was shown to provide more stable centroid locations than SSE, but at the cost of losing the dynamics of the k-means algorithm. **Objective:** In this paper, we address this drawback by adding a random swap step to the algorithm and introducing a new Overlap random swap. **Method:** It consists of three steps: (1) selective replacement of the centroids, (2) standard cluster assignment step, (3) weighted centroid calculation giving less weight to the overlapping points. **Results:** Experimental results confirm that ORS achieves competitive clustering accuracy, stable centroid locations, and improved boundary separation. The clustering accuracy (ACC) is about 95% and the centroid index (CI) is about 0.1. The algorithm converges within 3000 swap iterations. **Conclusions:** These results indicate that ORS helps avoid local minima and maintains the robustness of overlap-based clustering.

Keywords- Clustering, Overlap k-means, Cluster overlap, Random swap.

Acknowledgement

This work was part of the Ministry of Education and Culture's Doctoral Education Pilot under Decision No. VN/3137/2024-OKM-6 (The Finnish Doctoral Program Network in Artificial Intelligence, AI-DOC).

1. Introduction

Clustering partitions data into c clusters by minimizing a clustering objective function, typically the sum of squared errors (SSE). Overlap K-means [1] introduces a new objective function based on cluster overlap, which measures how many data points are shared between cluster neighborhoods. This overlap-based formulation avoids the redundancy between within-cluster and between-cluster and better captures ambiguity at cluster boundaries. It improves robustness and centroid stability but comes at the cost of reduced adaptability, as centroids tend to become fixed early, limiting the algorithm's ability to explore better configurations.

As illustrated in Figure 1, the pulling effect of distant points in K-means allows one of the centroids to relocate toward the cluster initially lacking representation. In contrast, Overlap K-means reduces the influence of border points by assigning them lower weights, thereby stabilizing centroid positions but also limiting the algorithm's flexibility.

The Random Swap algorithm (RS) was first proposed in [2] to address problems through a combination of global search and local optimization. It solves the cluster structure through a sequence of centroid swaps (global search) and fine-tuning using K-means (local search). This significantly improves traditional K-means by dramatically reducing the probability of getting trapped in suboptimal local solutions. The expected time complexity for centroids to converge to their correct location was shown to be $O(nk^2)$, i.e., linear dependency on the data size and quadratic on the number of clusters [3].

To restore the dynamic search capability lost in Overlap K-means, we integrate an RS mechanism that periodically replaces centroids with randomly chosen data points. This global perturbation helps to escape poor local optima. When combined with Overlap K-means, the resulting Overlap Random Swap algorithm regains the dynamic behavior while maintaining the local robustness of the overlap formulation.

The rest of the paper is organized as follows. We first discuss the related work in Section 2 and formulate our proposed method in Section 3. We then present the theoretical analysis in Section 4. The experimental setup is defined in Section 5, and the results are presented in Section 6. Conclusions are drawn in Section 7.

2. Algorithms minimizing SSE

Let $dist(x_i, x_j)$ be the distance function between data points x_i and x_j. Clustering is represented by the partition index $p(i)$ of the cluster to which the object x_i is assigned. The clusters are represented by their centroids c_j. The most common objective function is to minimize the sum of squared error between the data points and their cluster centroid:

$$SSE = \sum_{i=1}^{n} \text{dist}\,(x_i, c_{p(i)})^2 \quad (1)$$

K-means [4-6] and many of its variants [7-11] are based on this objective. SSE implicitly maximizes between-cluster variance, making the two measures mathematically dependent. As such, they do not offer complementary information, and optimizing one is sufficient [1]. In K-means, clustering is susceptible to the quality of the initial centroid selection. Poor initialization can lead to a partition being trapped in a local optimum. Moreover, the presence of noisy or ambiguous data points can significantly impact the clustering results. Noisy points and outliers, which deviate substantially from the main data distribution, can disproportionately influence the position of the centroids, leading to increased SSE and less meaningful clusters.

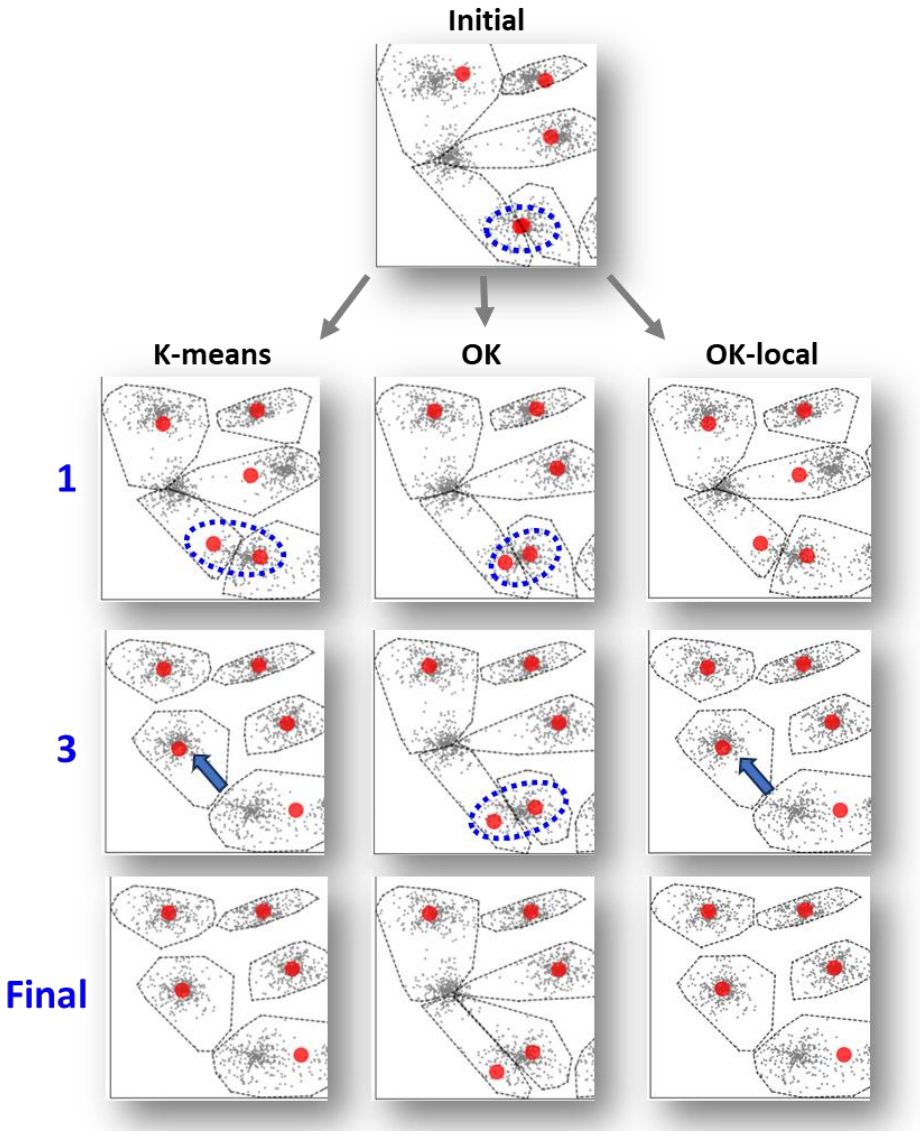

Figure 1. Snapshots of the bottom left corner of S1 after the 1st iteration, 3rd iteration, and the final solution [1]. Overlap k-means (middle) provides stable centroid locations as the two centroids refuse to move out of the cluster. The side-effect of this stability is that the centroids fail to move across clusters, resulting in a weaker exploration of the search space.

Many studies enhance the K-means algorithm by incorporating global search to escape poor local minima. Early work applied genetic algorithms (GA) to encode candidate centroid sets and evolve them with crossover and mutation under clustering fitness functions [8]. The Randomized Local Search and Random Swap (RS) [2–3] alternate global swaps of centroids with local refinement (K-means iterations), dramatically reducing the likelihood of being trapped in bad local optima.

Several other strategies reshape the search landscape: K-means* gradually transforms data to ease optimization [9]; Breathing K-means alternates expansion–contraction ("breathing") to shake off local traps [10]; Simulated Annealing introduces temperature-controlled acceptance of uphill moves with tailored perturbations for clustering [11]; and Recombinator K-means uses k-means++ seeding inside an evolutionary recombination framework to retain high-quality

building blocks [12]. Compared with other methods, RS remains parameter-light, evaluating single-swap candidates and accepting only moves that improve objectives, which is an idea our method leverages.

To decouple within- and between-cluster effects and better capture boundary ambiguity, Overlap K-means introduces an overlap-based objective that estimates how many points are shared between neighboring clusters [1]. By giving smaller weights to high-overlap (border) points in centroid updates, OK produces more stable centroids and sharper boundaries than pure SSE optimization. Its localized variant (OK-local) further strengthens robustness by computing overlap in local neighborhoods. However, the same stability that protects against boundary noise reduces global exploration. Once centroids become locally stable, OK has difficulty relocating them.

3. Overlap-based Clustering by Random Swap

This section introduces the *Overlap Random Swap* (ORS) algorithm, which combines Overlap K-means with Random Swap to improve clustering performance. We begin by reviewing the Overlap K-means method.

3.1 Overlap K-means

Overlap K-means introduces the concept of overlap to measure the uncertainty in cluster assignment for individual data points. Rather than depending exclusively on global centroid positions, the overlap metric is calculated using local neighborhood information, which enhances the algorithm's resilience to noise and reduces sensitivity to boundary ambiguities in complex cluster structures.

The localized overlap value of a point x_i is defined as:

$$O_L(x_i) = \frac{Meanshift(x_i)}{dist(x_i, \overline{NN}(x_i))} \tag{2}$$

where the denominator measures the distance to the closest external point, with $\overline{NN}(x_i)$ representing the nearest neighbor of x_i in another cluster. In addition, *Meanshift*(x_i) represents the mean-shift distance within the same cluster, which is calculated as:

$$Meanshift(x_i) = \frac{\sum_{y \in \underline{KNN}(x_i)} dist(x_i, y)}{k} \tag{3}$$

where $\underline{KNN}(x_i)$ is a set of k nearest neighbors of x_i in the same cluster.

This overlap serves as an inverse weight in centroid updates. That is, points with higher overlap (likely near borders) exert less influence on the cluster center. The weighted centroid update rule is:

$$c_j = \frac{\sum_{p(i)=j} w_i \cdot x_i}{\sum_{p(i)=j} w_i} \tag{4}$$

$$w_i = exp[-(O_L(x_i)/\gamma)^2] \tag{5}$$

where γ is a constant bandwidth parameter.

By doing so, Overlap K-means reduces the impact of noisy or ambiguous points, shifting centroids toward more stable regions.

3.2 Random Swap

Random Swap enhances K-means by integrating global search through centroid replacement with local K-means optimization, significantly reducing the risk of suboptimal clustering solutions.

At its core, RS periodically perturbs the solution by replacing one centroid with a randomly selected data point. Unlike traditional K-means updates, which only adjust centroids based on their assigned members, RS introduces external candidates from the entire dataset. This allows the algorithm to discover previously uncovered clusters or escape redundant centroid allocations.

Each swap is evaluated with a lightweight reassignment and update step, and it is accepted only if it reduces the SSE. This controlled mechanism adds global exploration to the otherwise local optimization process without incurring a full re-computation.

RS removes one existing cluster and creates a new one in a different part of the data space. This is accomplished by selecting a randomly chosen prototype c_s and replacing it with a randomly selected data vector x_i:

$$c_s \leftarrow x_i \qquad \text{where} \quad s = rand(1,k), \quad i = rand(1,N) \tag{6}$$

This generates a global change in the clustering structure. An alternative implementation of the swap would be first to choose an existing cluster randomly, and then select a random data vector within this cluster. The first approach is used here for simplicity, but the second approach provides useful analytical insights for studying the swap operation.

After the swap, local repartition is performed to update the partition. This local repartition, while not obligatory (as the solution will be tuned by k-means afterwards), serves to accelerate the process. First, vectors of the removed cluster are repartitioned to nearby clusters by comparing distances to all other prototypes (including the new cluster) and selecting the nearest:

$$p_i \leftarrow arg \min_{1 \le j \le k} d\,(x_i, c_j)^2 \;\; \forall\, i \mid p_i = j \tag{7}$$

Second, the new cluster is formed by attracting vectors from nearby clusters by calculating the distance of all vectors to the new prototype. If this distance is smaller than the distance to the prototype of the current cluster, the vector joins the new cluster:

$$p_i \leftarrow arg \min_{k=j \vee k=p_i} d\,(x_i, c_k)^2 \forall i \in [1,N] \tag{8}$$

The resulting solution is then refined by applying two iterations of K-means to adjust the cluster boundaries locally. This procedure follows a trial-and-error strategy, where a candidate solution is accepted only if it leads to an improvement in the objective function.

Figure 2 illustrates typical K-means failures, such as a single centroid spanning two distinct clusters or multiple centroids being wasted on a single cluster, both of which stem from poor initialization. Figure 3 shows that Overlap K-means sharpens boundaries and yields more stable centroid locations through local overlap, yet it still inherits K-means' global suboptimality, with the correct configuration requiring more than two accepted centroid swaps. Figure 4 demonstrates that RS can overcome these limitations through repeated centroid replacement and global search, achieving near-optimal separation.

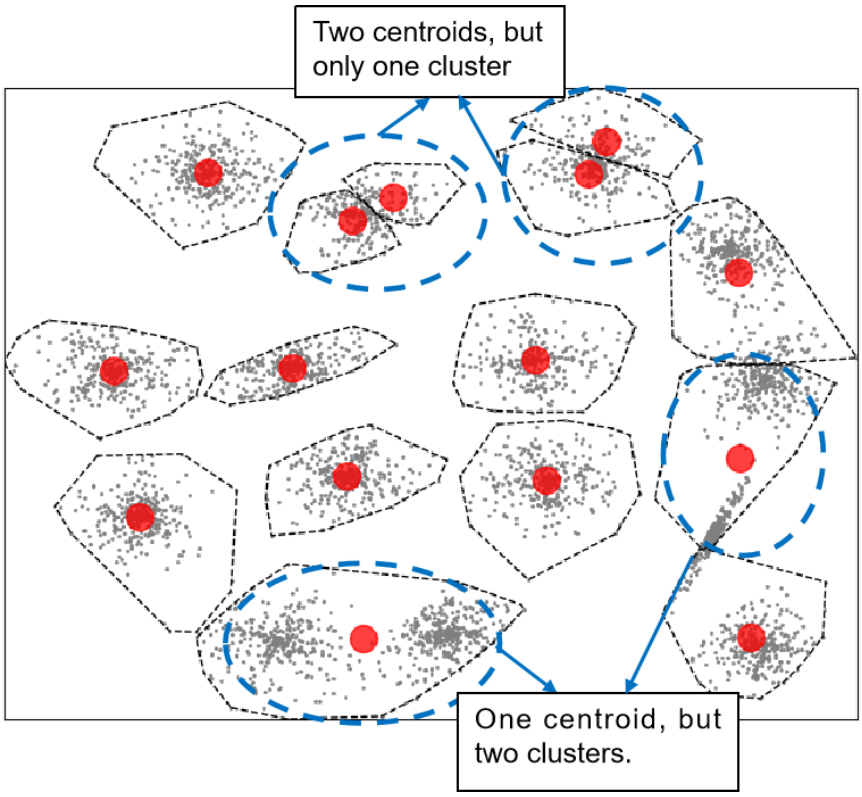

Figure 2. Typical failures of K-means algorithm. Two successful centroid swaps lead to the correct solution.

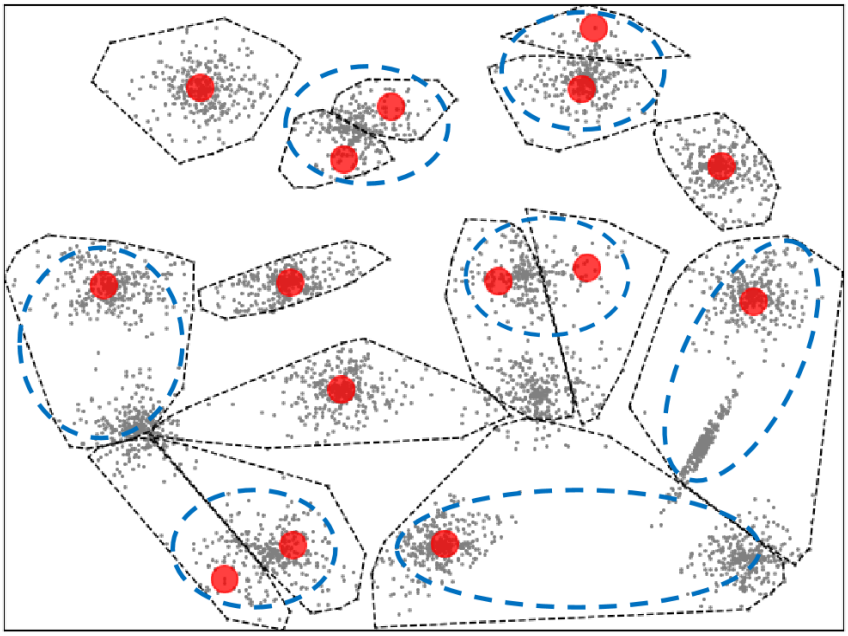

Figure 3. Overlap K-means yields more stable centroid locations than K-means, although the correct solution is achieved only after more than two accepted centroid swaps.

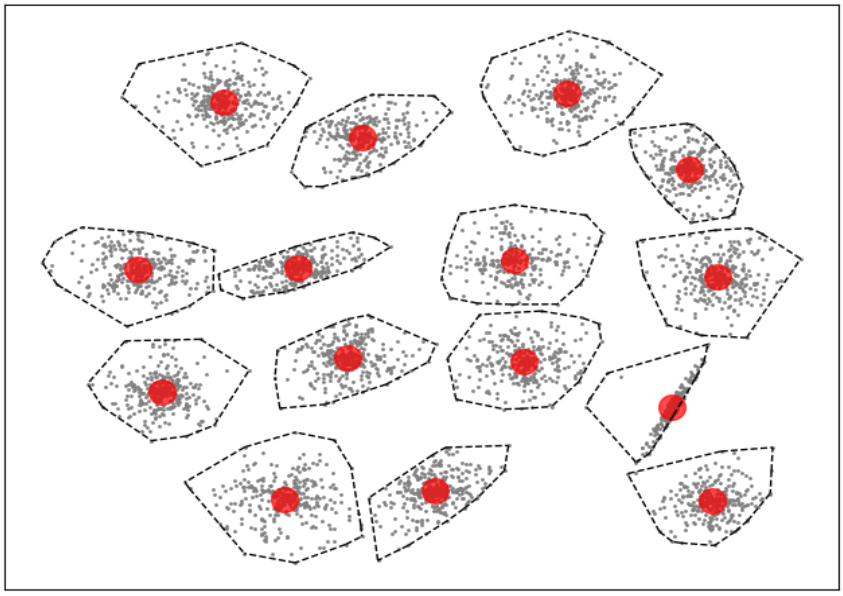

Figure 4. Random Swap after 5000 swaps achieves clean separation of clusters.

Based on these observations, we propose to integrate the RS strategy into the Overlap K-means framework by combining their strengths to escape from the local minima.

3.3 Overlap random swap

Overlap K-means is designed to improve K-means by reducing the influence of boundary points through overlap-based weighting. This approach aims to make centroid updates more stable and clustering results more robust, particularly in datasets with noise or overlapping structures. However, while the overlap K-means leads to greater stability of the centroids, it also reduces global exploration and retains the core limitations of K-means. It depends on randomly initialized centroids, which makes it sensitive to initialization. Once a centroid becomes locally stable, even if it is incorrectly placed, the algorithm has difficulty relocating it to a more appropriate position. As a result, the algorithm can easily become trapped in a poor local solution, similar to the behavior of K-means.

To overcome this limitation, we incorporate the Random Swap mechanism. It introduces global changes by randomly replacing one of the centroids with a data point from the dataset. This helps the algorithm escape suboptimal configurations to explore better clustering results.

Building on this idea, the ORS algorithm integrates local structure-aware optimization through Overlap K-means with a global exploration strategy via Random Swap. It begins by randomly selecting c points from the dataset as initial centroids and assigning data points to the nearest centroid for an initial partition.

Instead of SSE, the goal of ORS is to minimize the *overlap-weighted sum of square error* (SSE_o) of the dataset:

$$SSE_o = \sum_{i=1}^{n} w_i \, \|x_i - c_{p(i)}\|^2 \tag{9}$$

This ensures consistency with the overlap-based objective. The weighting preserves the robustness of Overlap K-means while providing a smooth, continuous criterion for evaluating swap candidates.

In each iteration, ORS performs the following steps: (1) a Random Swap operation, where one centroid is randomly replaced by a randomly selected data point; (2) two iterations of Overlap K-means using the O2 overlap criterion, in which overlap values are computed for each data point and used as weights to update the centroids, reducing the influence of boundary points; and (3) a test-and-selection step that accepts the swap only if it decreases the overall SSE, otherwise reverting to the previous state. This integrated approach of local optimization and global exploration effectively addresses both the initialization sensitivity and the local optima problem of Overlap K-means, ultimately discovering superior clustering structures through multiple iterations. The pseudocode of ORS is summarized below.

Algorithm: ORS

ORS $(X, c, k, \gamma) \rightarrow \{p_i\}$

Input: X, c, k, γ=1

Output: $\{p_i\}$, $1 \leq i \leq n$, $p_i \in \{1, \ldots, c\}$

Select c points randomly from X as centroids c_j, $1 \leq j \leq c$
Initial partition according to the initial centroids

Repeat (for a maximum number of swaps 5000)

Step 1: Replacement of Centroids

(1) Randomly select one centroid c_j and a point x_r in X;
(2) Replace $c_j \leftarrow x_r$;

Step 2: Run Overlap K-means for two iterations

(1) Assign each x_i to its nearest centroid;
(2) Compute the overlap value $O_2(x_i)$ according to (1);
(3) Update each centroid using weighted average according to (3).

Step 3: Test and Selection

According to (9), if SSE_o decreases, accept the swap; otherwise, revert.

In the current implementation, ORS uses a fixed maximum number of swap attempts as its stopping criterion. Possible heuristics include monitoring the relative improvement in SSE over a fixed number of iterations and stopping when it falls below a certain threshold, for example, 5%. However, such criteria risk premature termination because the last successful swap may require many additional iterations [3], especially on unbalanced datasets. Previous analyses of RS [1] also provided theoretical convergence estimates that could be adapted in practice, but the stopping criterion remains an open problem.

4. Theoretical Analysis

While Overlap K-means improves boundary handling, it loses the dynamic exploration capability and is prone to getting trapped in local optima. To understand why Random Swap provides an effective solution, we need to examine the theoretical foundations of both approaches and their combination.

4.1 The Local Optimization Trap in Overlap K-means

Overlap K-means operates by minimizing the total overlap. Consider the current centroid configuration $S = \{c_1, c_2, \ldots, c_n\}$ with corresponding total overlap O(S). The algorithm becomes trapped in a local optimum S* when no further improvement is possible through standard weighted centroid updates:

$$O(S^*) \leq O(S') \quad \forall S' \in N(S^*) \tag{10}$$

where N(S*) represents the set of configurations reachable through the overlap-weighted update rule in Eq. (5).

The issue lies in the constrained optimization space of Overlap K-means. Each centroid c_j can only move within a restricted region determined by its assigned data points and their overlap values. While this region is restricted in each iteration, it changes dynamically as cluster assignments and overlap values are updated. Specifically, the weighted centroid update confines each c_j to the weighted convex hull of its cluster members:

$$c_j \in ConvexHull(\{x_i / O_2(x_i) : P(i) = j\}) \tag{11}$$

This mathematical constraint creates optimization "valleys" from which the algorithm cannot escape through local moves alone. When initialization places centroids in suboptimal positions, or when cluster boundaries are complex, Overlap K-means may settle in local minima.

4.2 Random Swap as a Global Exploration Mechanism

Random Swap enhances optimization by introducing centroid replacements, allowing the algorithm to escape the limitations of gradual local adjustments and explore new regions of the solution space.

In the early versions of Random Swap, both single-swap and multiple-swap iterations were tested. However, experiments reported in Tabu Search [13] and Randomized Local Search [2] showed that performing multiple swaps per iteration did not improve the results and, in fact, increased computational cost. The single-swap strategy [3] was therefore adopted as it provided comparable or better performance with simpler implementation.

The mechanism works by expanding the effective neighborhood of any configuration S. Through a single swap operation, the algorithm can reach any configuration in the set:

$$S_{RS}(S) = \{c_1, \ldots, c_{j-1}, x_r, c_{j+1}, \ldots, c_k\} : j \in \{1, \ldots, k\}, x_r \in X\} \tag{12}$$

This represents a dramatic expansion in exploration capability. While Overlap K-means's neighborhood is constrained to local adjustments, RS provides access to $k \times n$ distinct configurations in a single operation, where n is the total number of data points.

A key strength of this approach is its evaluation-based perturbation mechanism. Unlike random noise that might degrade solution quality, RS evaluates each centroid replacement. Only swaps that reduce the total SSE_o are accepted, ensuring that global exploration never compromises the quality of the current solution.

4.3 A combination of local and global search

ORS integrates local optimization with global exploration to achieve better clustering performance. Each iteration uses a strict accept-or-reject policy, where a swap is accepted only if it reduces the objective function.

To examine how ORS improves centroid stability over Overlap K-means, we track its convergence on the S4 dataset, characterized by strong cluster overlap. Figure 5 plots SSE_o across 5000 swap attempts; only 19 of them are accepted, marked by green dashed ticks. The most significant improvements occur during the first dozen accepted swaps, after which the curve stabilizes.

Figure 6 visualizes cluster evolution at three key points, showing how overlapping boundaries are progressively clarified. These results confirm that ORS maintains exploration while ensuring steady refinement once centroids stabilize.

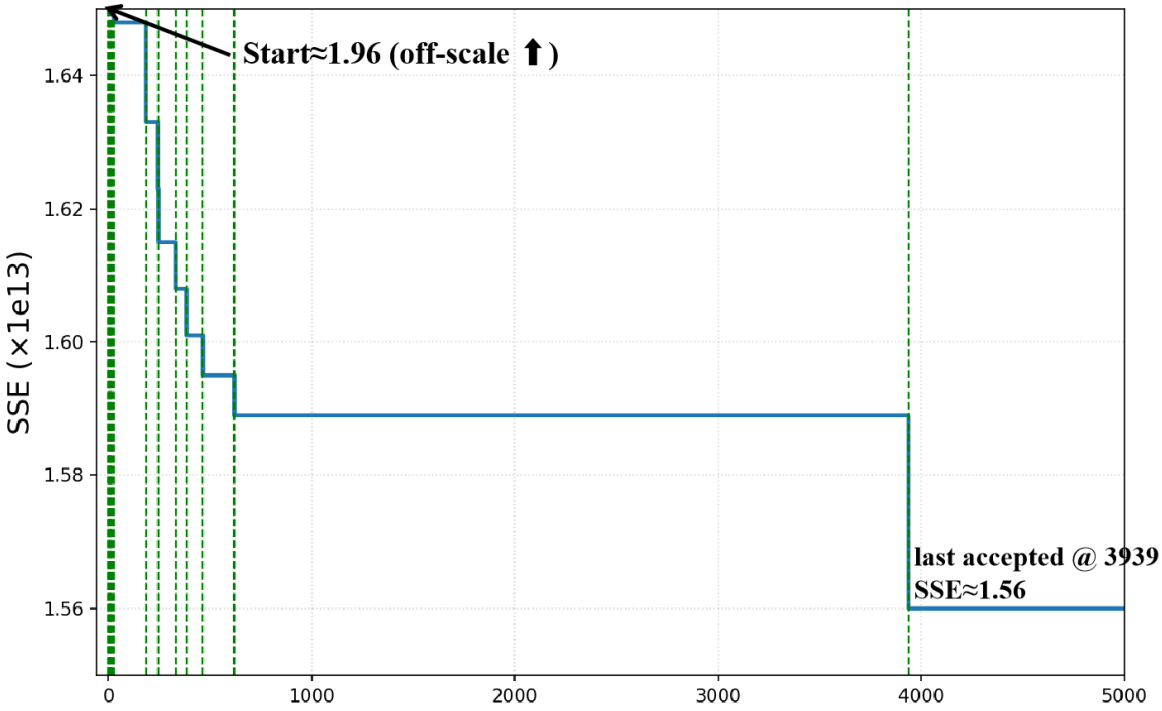

Figure 5. The development of SSE_o as a function of the swap attempts (maximum = 5000) on the S4 dataset (high overlap). Green dashed ticks mark the 19 accepted swaps (at attempts 1, 2, 3, 4, 8, 11, 12, 13, 16, 23, 24, 182, 243, 246, 329, 384, 464, 620, and 3939).

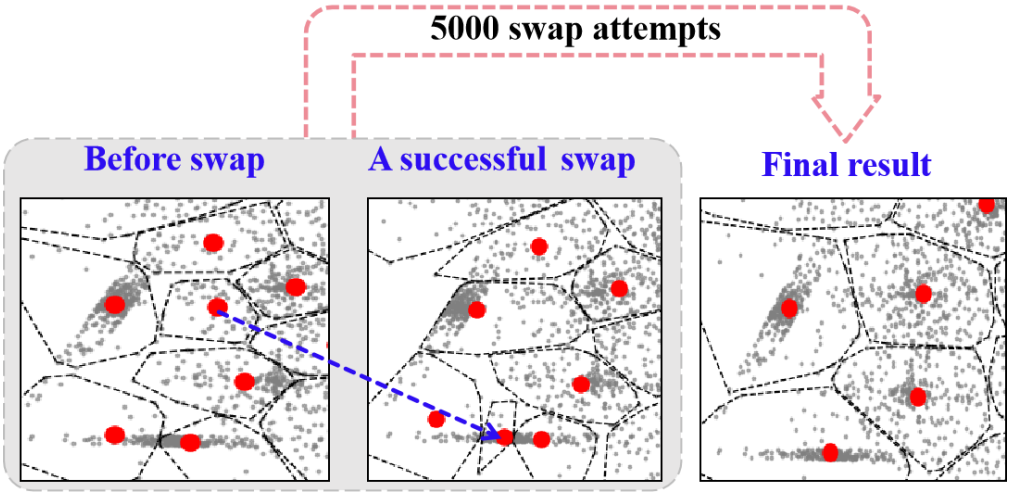

Figure 6. Cluster visualizations at three stages: initial, one accepted swap, and final.

They show the gradual correction of boundary points and the stabilization of centroids.

4.4 Complexity Analysis

The time complexity of the ORS is dominated by two main components within each iteration: the local 2-step clustering update and the evaluation of random swaps. In each iteration, ORS first performs a random swap by replacing one centroid with a randomly selected data point and then runs two iterations of Overlap K-means. The label assignment requires computing distances between each point and all k centroids, resulting in a cost of O(*nkd*).

The overlap computation, for each point, involves two nearest neighbor searches: one within its assigned cluster and another outside of it. These operations, along with overlap-based centroid updates, incur a cost of approximately O(*nd log n*) per iteration. Thus, each 2-step local update takes O(*nkd*+*ndlogn*) time. Since a maximum of S swaps bounds the outer loop, the overall time complexity is O(*S*·*nd*(*k*+*logn*)).

5. Experimental setup

The datasets used in the experiments are listed in Table 1 and illustrated in Figure 7. We selected them from the basic clustering benchmark [4, 14–20], covering a variety of cluster overlaps, shapes, densities, and real-world datasets.

Table 1. Datasets used in the experiments.

Dataset	Source	Acronym	N	d	k
S1	[4]		5000	2	15
S2	[4]		5000	2	15
S3	[4]		5000	2	15
S4	[4]		5000	2	15
A1	[18]		3000	2	20
A2	[18]		5250	2	35
A3	[18]		7500	2	50
Dim32	[4]	Dim	1024	32	16
Unbalance	[19]	Unb	6500	2	8
Flame	[16]	Fla	240	2	2
Aggregation	[17]	Agg	788	2	7
Birch1	[20]	B1	100000	2	100
Birch2	[20]	B2	100000	2	100
Locations 2012	[21]		13467	2	
Joensuu 2012	[21]		6014	2	

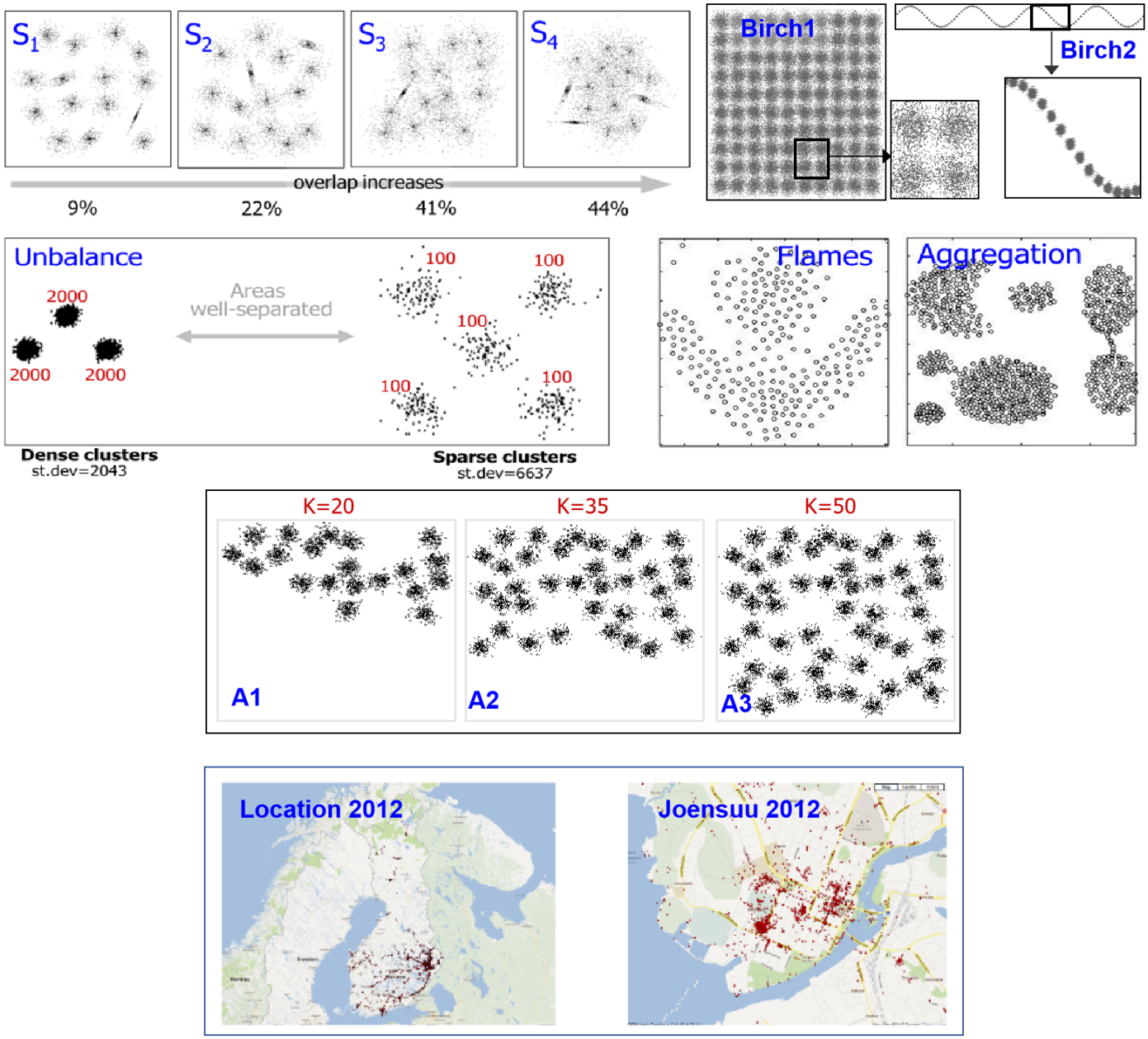

Figure 7. Visualization of the selected benchmark datasets with varying clustering challenges. The datasets differ in cluster overlap (S1–S4), regular structure (Birch1 and Birch2), density imbalance (Unbalance), shapes (Flame, Aggregation), and number of clusters (A1–A3), as well as two real-world datasets (Joensuu 2012 and Locations 2012).

We evaluate ORS against seven representative clustering algorithms covering the main methodological paradigms. This comparison includes initialization-sensitive, global optimization, density-based, and overlap-aware clustering strategies to demonstrate the performance of our method.

K-means (KM) and K-means++ (KM++) [21] as baseline centroid-based clustering, Random Swap (RS) [3] for global optimization clustering method, Density Peaks (DensP) [22], and DBSCAN [23] for density-based approaches, Overlap K-means (OK) [1] and its localized variant (OK-local) [1] for overlap-aware methods. We run all algorithms 100 times and report the average results, except for Birch datasets, which are repeated only 10 times due to their higher processing times. Clustering results are measured by clustering accuracy (ACC) [24] and centroid index (CI) [25].

6. Results

The results are summarized in Tables 2 and 3, which report clustering performance in terms of ACC and CI, respectively.

Methods such as RS and DensP achieved perfect clustering (CI=0) on most datasets. KM++ also performed well on specific datasets like Dim32 and Unbalance. In contrast, standard K-means consistently failed to achieve optimal results, with an average CI of 3.9, which demonstrates its sensitivity to initialization.

The comparison between KM, OK, and OK-local reveals that while OK introduces a more flexible boundary model, it does not necessarily improve overall performance. In fact, OK-local achieves better results than OK by applying localized overlap weighting, but both methods still inherit the local optimization limitations of K-means. The average CI for OK and OK-local were 5.3 and 3.9, respectively.

In contrast, ORS consistently achieved robust clustering performance (CI=0.1) across all datasets we selected. This demonstrates its ability to overcome both initialization sensitivity and local optima that limit traditional and Overlap-based K-means variants. Its average accuracy (ACC = 95%) also surpassed other methods, making it one of the most robust performers in the evaluation.

Table 2. Clustering results measured by clustering accuracy (ACC)↑ in %.

	S1	S2	S3	S4	A1	A2	A3	Dim	Unb	Fla	Agg	B1	B2	Av
KM	87	90	80	74	90	87	86	82	60	84	82	87	84	83
KM++	98	94	83	78	97	98	96	100	100	84	83	95	98	93
RS	99	97	86	80	100	100	100	100	100	84	83	97	100	94
DensP	99	97	85	79	98	99	99	95	100	100	86	100	100	95
OK	83	82	74	71	84	82	84	75	58	67	73	83	78	77
OK-local	86	89	79	76	87	86	88	77	60	84	83	88	84	82
DBSCAN	95	77	56	58	88	84	85	91	98	88	87	78	68	81
ORS	99	98	86	80	100	100	100	100	100	84	86	98	100	95

Table 3. Clustering results measured by centroid index (CI)↓.

	S1	S2	S3	S4	A1	A2	A3	Dim	Unb	Fla	Agg	B1	B2	Av
KM	2.0	1.1	1.2	1.3	2.0	4.2	6.5	2.9	3.5	0.0	0.7	8.8	16.1	3.9
KM++	0.2	0.5	0.6	0.4	0.6	0.7	1.7	0.0	0.0	0.0	0.7	3.3	1.5	0.8
RS	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.1
DensP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.1
OK	2.3	2.3	2.5	2.2	f2.9	6.0	7.5	4.0	4.0	0.5	1.8	12.5	20.0	5.3
OK-local	2.1	1.2	1.3	0.9	2.5	4.7	5.6	3.7	4.0	0.0	1.0	8.3	15.6	3.9
DBSCAN	11.0	9.0	18.0	22.0	12.0	20.0	28.0	0.0	1.0	1.0	2.0	11.0	5.0	10.8
ORS	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.1

To further validate the effectiveness of ORS on real-world data, we conducted experiments on two datasets: Joensuu 2012 and Locations 2012.

Since ground-truth cluster labels are unavailable for these datasets, the evaluation is based on SSE_o as an internal measure of clustering quality. As shown in Figures 8 and 9, each curve represents the mean over 10 independent runs, and the shaded regions indicate ±1 standard deviation, showing the variation across runs. ORS consistently achieved the lowest SSE_o values across different numbers of clusters, outperforming both OK and OK-local.

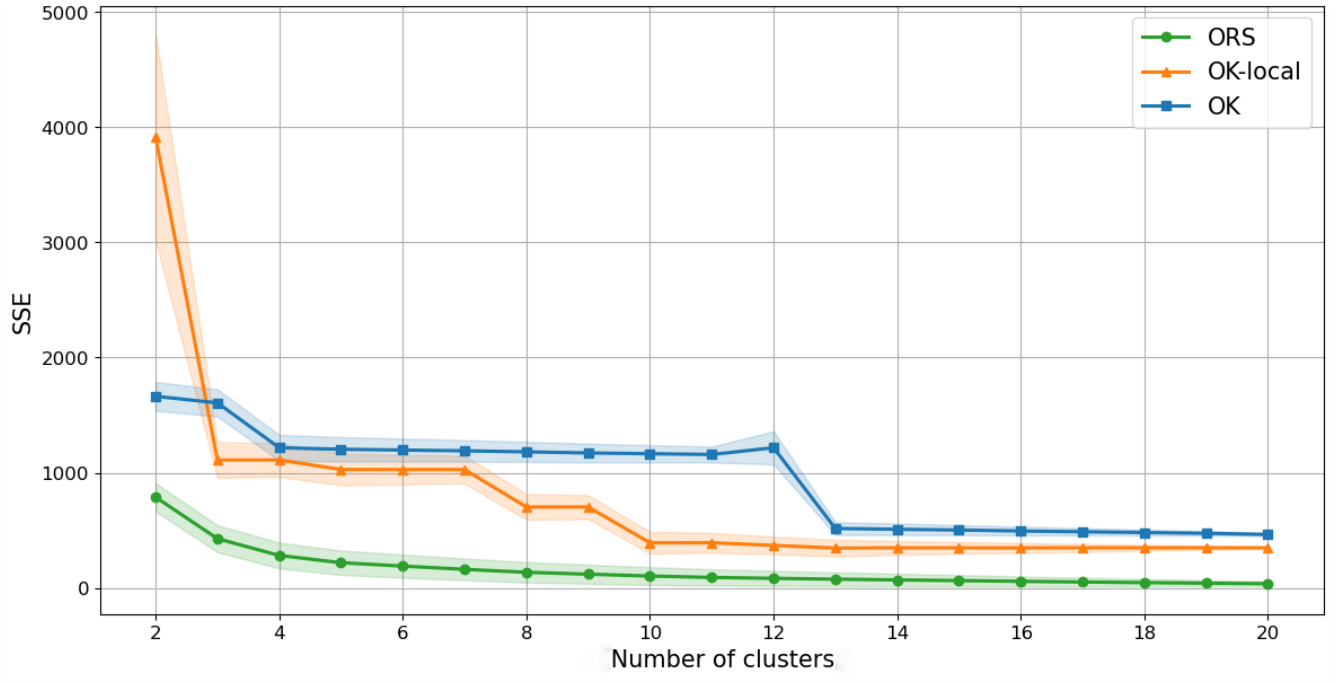

Figure 8. SSE_o of ORS, OK-local, and OK across different numbers of clusters on the Joensuu 2012 dataset.

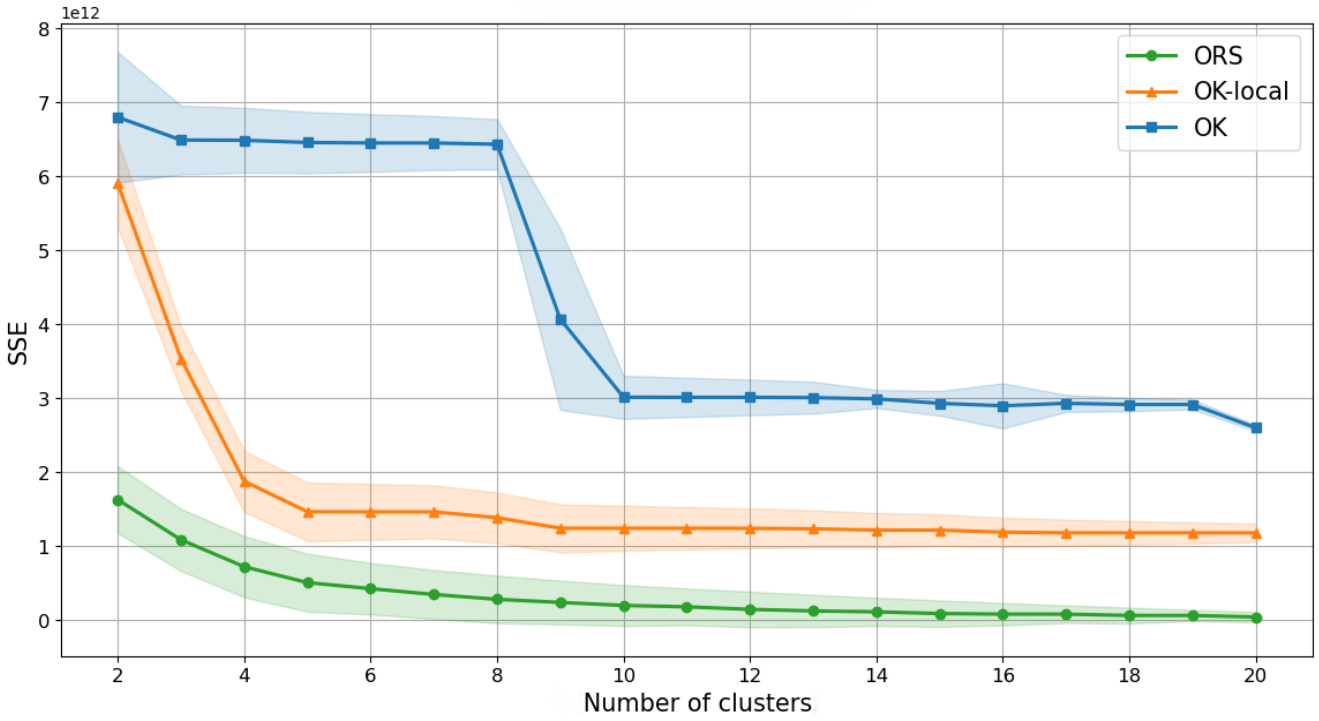

Figure 9. SSE_o of ORS, OK-local, and OK across different numbers of clusters on the Locations 2012 dataset.

Besides clustering performance, we also evaluated the empirical runtime scalability of ORS using the B1 dataset by subsampling $n \in \{1{,}000, \ldots, 100{,}000\}$ with an AMD R9-8940HX CPU and 32 GB RAM. As shown in Figure 10, the processing time grows nearly linearly with the number of points, from about 167 seconds at n = 1k to about 16560 seconds at n=100k.

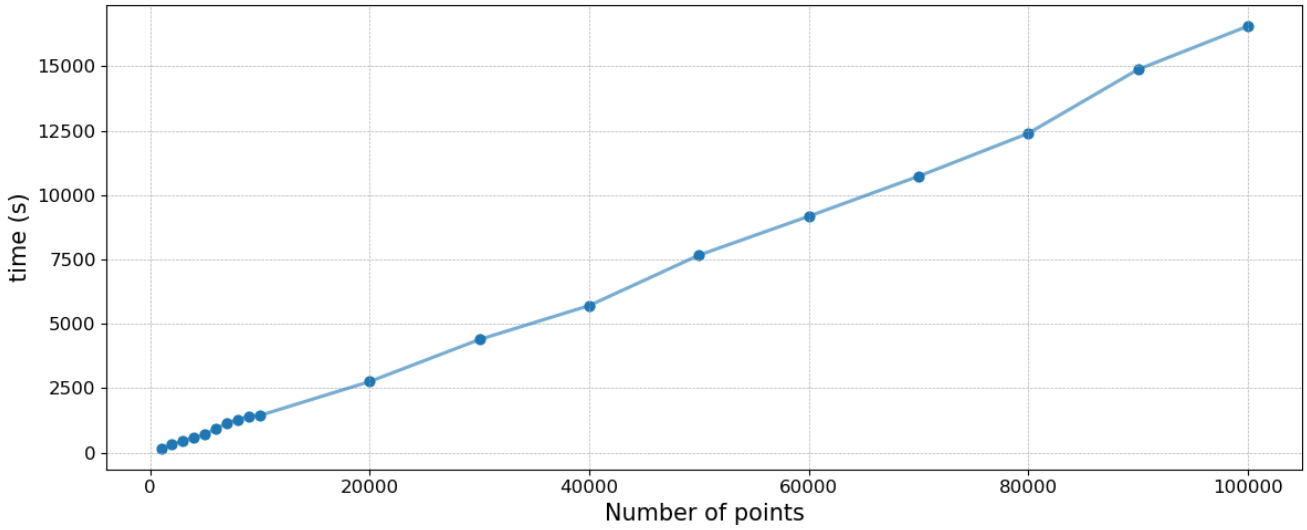

Figure 10. ORS runtime on subsets of Birch1.

An important empirical observation is that ORS converges well below the maximum limit of 5000 swap attempts across all tested datasets, with most requiring fewer than 2000 swaps. As shown in Figure 11, the number of swap attempts ranges from only 66 on Unb to 3939 on S4. Other datasets, such as S1 (1704), S2 (1201), B1 (2094), and Agg (1920), also stabilize far earlier than the maximum threshold. Here, *converge* refers to the point at which centroid positions no longer change significantly through further swap operations, rather than algorithmic convergence.

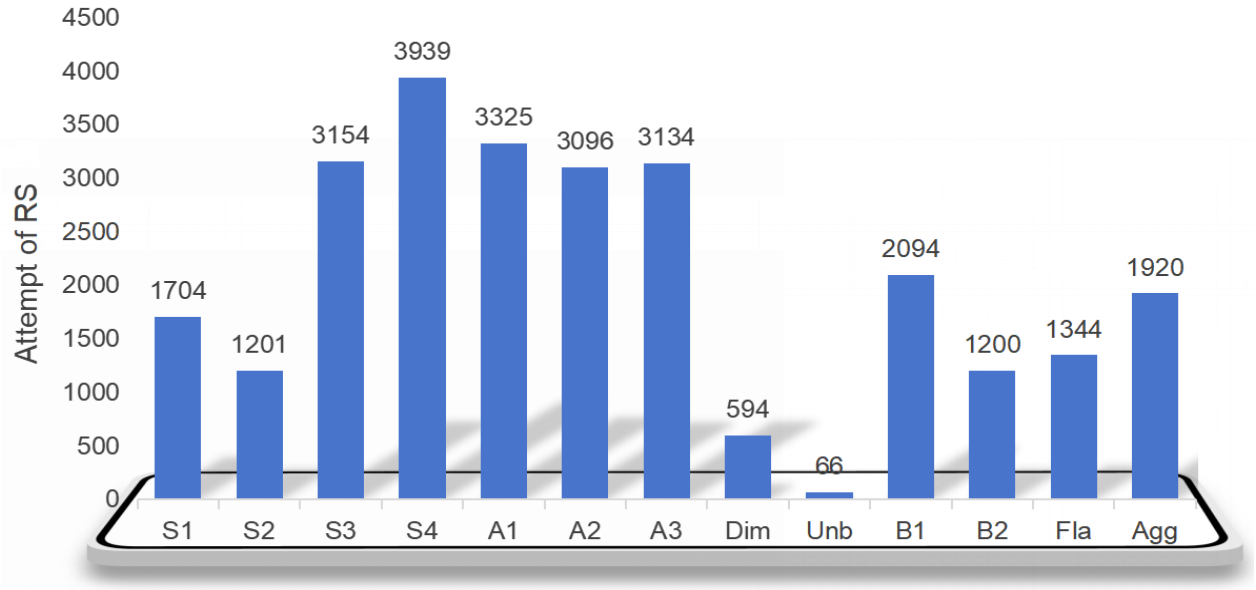

Figure 11. Number of swap attempts required for ORS to *converge* on each dataset.

7. Conclusions

This paper presented ORS, a clustering algorithm that combines the local optimization of Overlap K-means with the global exploration of Random Swap. Traditional K-means and its overlap variant often get trapped in local optima: the former due to sensitivity to initialization, and the latter due to reduced adaptability from overlap-based weighting. ORS addresses both issues by introducing controlled global swaps to escape poor local minima. Experimental results show that ORS yields clustering accuracy of about 95% (average CI = 0.1) across diverse datasets and typically converges within 3000 swaps or fewer. These properties make it both effective and practical for real-world clustering tasks.

Future work may extend ORS by developing adaptive or data-driven stopping criteria to improve robustness and efficiency. In addition, further analysis of its convergence properties could provide stronger theoretical guarantees.

References

[1] P. Fränti, C. Cariou, Q. Zhao, "Cluster overlap as objective function", *CMC-Computers, Materials & Continua*, 66534, 1-28, 2025. https://doi.org/10.32604/cmc.2025.066534

[2] P. Fränti , J. Kivijärvi, "Randomized local search algorithm for the clustering problem", *Pattern Analysis and Applications*, 3 (4), 358-369, 2000. https://doi.org/10.1007/s100440070007

[3] P. Fränti, "Efficiency of random swap clustering", *Journal of Big Data*, 5:13, 1-29, 2018. https://doi.org/10.1186/s40537-018-0122-y

[4] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets", *Applied Intelligence*, 48 (12), 4743-4759, 2018. https://doi.org/10.1007/s10489-018-1238-7

[5] E. Forgy, "Cluster analysis of multivariate data: efficiency vs. interpretability of classification", *Biometrics*, 21(3), 768–780, 1965.

[6] J. MacQueen, "Some methods for classification and analysis of multivariate observations", *Berkeley symposium on Mathematical Statistics and Probability*. 281-297, 1967.

[7] S.P. Lloyd, "Least squares quantization in PCM", *IEEE Trans. Information Theory*, 28 (2), 129–137, 1982. https://doi.org/10.1109/TIT.1982.1056489

[8] P. Fränti, J. Kivijärvi, T. Kaukoranta , O. Nevalainen, "Genetic algorithms for large scale clustering problems", *The Computer Journal*, 40 (9), 547-554, 1997. https://doi.org/10.1093/comjnl/40.9.547

[9] M.I. Malinen, R. Mariescu-Istodor, P. Fränti, "K-means*: clustering by gradual data transformation", *Pattern Recognition*, 47 (10), 3376-3386, 2014. http://dx.doi.org/10.1016/j.patcog.2014.03.034

[10] B. Fritzke, "Breathing k-means", *arXiv preprint arXiv:2006.15666*, 2020.

[11] J. Lee, D. Perkins, "A simulated annealing algorithm with a dual perturbation method for clustering", *Pattern Recognition*, 112, 107713, 2021. https://doi.org/10.1016/j.patcog.2020.107713

[12] C. Baldassi, "Recombinator K-means: an evolutionary algorithm that exploits k-means++ for recombination". *IEEE Trans. on Evolutionary Computation*, 26 (5), 991-1003, 2022. https://doi.org/10.1109/TEVC.2022.3144134

[13] P. Fränti, J. Kivijärvi, O. Nevalainen: "Tabu search algorithm for codebook generation in vector quantization", *Pattern Recognition*, 31 (8), 1139-1148, 1998. https://doi.org/10.1016/S0031-3203(97)00127-1

[14] S. Sieranoja, P. Fränti, "Fast and general density peaks clustering", *Pattern Recognition Letters*, 128, 551-558, 2019. https://doi.org/10.1016/j.patrec.2019.10.019

[15] L. Fu, E. Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data", *BMC bioinformatics*, 8(1), 3, 2007. https://doi.org/10.1186/1471-2105-8-3

[16] A. Gionis, H. Mannila, P. Tsaparas, "Clustering aggregation", *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1), 4, 2007. https://doi.org/10.1145/1217299.1217303

[17] I. Kärkkäinen, P. Fränti, "Dynamic local search algorithm for the clustering problem", *Research Reports*, A-2002-6, University of Joensuu, 2002.

[18] M. Rezaei, P. Fränti, "Set-matching measures for external cluster validity", *IEEE Trans. on Knowledge and Data Engineering*, 28 (8), 2173-2186, 2016. https://doi.org/10.1109/TKDE.2016.2551240

[19] T. Zhang, R. Ramakrishnan, M. Livny, "BIRCH: A new data clustering algorithm and its applications", *Data Mining and Knowledge Discovery*, 1 (2), 141-182, 1997. https://doi.org/10.1023/A:1009783824328

[20] P. Fränti, "Mopsi location-based service", *Applied Computing and Intelligence*, 4 (2), 209-233, 2024. https://doi.org/10.3934/aci.2024013

[21] D. Arthur, S. Vassilvitskii, "K-means++: the advantages of careful seeding", *ACM-SIAM Symp. on Discrete Algorithms (SODA'07)*, January 2007. https://10.5555/1283383.1283494

[22] A. Rodriguez, A. Laio, "Clustering by fast search and find of density peaks", *Science*, 344(6191), 1492-1496, 2014. https://doi.org/10.1126/science.124207

[23] E. Martin, H-P. Kriegel, J. Sander, X. Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise." *kdd*. 96 (34), 1996. https://doi.org/10.5555/3001460.3001507

[24] P. Fränti, S. Sieranoja, "Clustering accuracy", *Applied Computing and Intelligence*, 4 (1), 24-44, 2024. https://doi.org/10.3934/aci.2024003

[25] P. Fränti, M. Rezaei, Q. Zhao, "Centroid index: cluster level similarity measure", *Pattern Recognition*, 47 (9), 3034-3045, 2014. http://dx.doi.org/10.1016/j.patcog.2014.03.017