SDM'18 - Graph sketching-based Massive Data Clustering

Anne Morvan[12], Krzysztof Choromanski[3], Cédric Gouy-Pailler[1] and Jamal Atif[2]

[1]CEA, LIST, [2]Université Paris-Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, [3]Google Brain Robotics

Objectives

We present a new clustering algorithm DBMSTClu providing a solution to the following issues: 1) detecting arbitrary-shaped data clusters, 2) with no parameter, 3) in a space-efficient manner by working on a limited number of linear measurements, a *sketched* version of the dissimilarity data graph $\mathcal{G}$.

Steps of the method

1. The dissimilarity data graph $\mathcal{G}$ with N nodes is handled as a stream of edge weight updates and sketched in one pass into a compact structure with space cost $O(N \operatorname{polylog}(N))$, cf. method from [1] relying on ℓ_0-sampling principle [2].
2. From the graph sketch, an Approximate Minimum Spanning Tree (AMST) $\mathcal{T}$ is recovered containing $N-1$ weighted edges s.t. for all $i \in [N-1]$, weights $w_i \in (0, 1]$. An MST is good for expressing the underlying structure of a graph.
3. Without any parameter, DBMSTClu performs successive edge cuts in $\mathcal{T}$ which create new connected components that can be seen as clusters. At each iteration, a cut is chosen as the one maximizing a criterion named Density-Based Validity Index of a Clustering partition (DBCVI) based on *Dispersion* and *Separation* defined on each connected component.

Cluster Dispersion & Separation

- The **Dispersion** of cluster C_i (DISP) is defined as the maximum edge weight of C_i:

$$\forall i \in [K],\ \mathrm{DISP}(C_i) = \begin{cases} \max\limits_{e_j \in E(C_i)} w(e_j) \text{ if } |E(C_i)| \neq 0 \\ 0 \qquad\qquad \text{otherwise.} \end{cases}$$

- The **Separation** of cluster C_i (SEP) is defined as the minimum distance between nodes of C_i and nodes of other clusters $C_j, i \neq j, i, j \in [K]$. $Cuts(C_i)$ is the set of edges incident to C_i:

$$\forall i \in [K],\ \mathrm{SEP}(C_i) = \begin{cases} \min\limits_{e_j \in Cuts(C_i)} w(e_j) \text{ if } K \neq 1 \\ 1 \qquad\qquad \text{otherwise.} \end{cases}$$

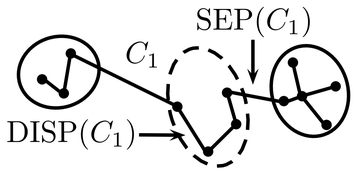

Figure 1: SEP and DISP for cluster C_1 in $\mathcal{T}$ ($N = 12$, $K = 3$).

Validity Index of Cluster & Clustering Partition

- The **Validity Index of cluster** C_i, $i \in [K]$ is defined as:

$$V_C(C_i) = \frac{\mathrm{SEP}(C_i) - \mathrm{DISP}(C_i)}{\max(\mathrm{SEP}(C_i), \mathrm{DISP}(C_i))}$$

- The **Density-Based Validity Index of a Clustering partition** $\Pi = \{C_1, \ldots, C_K\}$, DBCVI($\Pi$) is defined as the weighted average of the Validity Indices of all clusters in the partition.

$$\mathrm{DBCVI}(\Pi) = \Sigma_{i=1}^{K} \frac{|C_i|}{N} V_C(C_i)$$

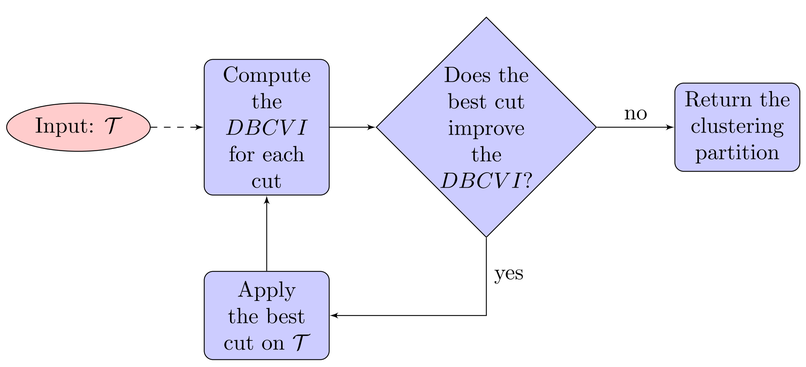

Figure 2: DBMSTClu algorithm

To whom correspondence should be adressed: anne.morvan@cea.fr. Anne is partly supported by the *Direction Générale de l'Armement* (French Ministry of Defense).

Tricks for a linear time implementation

1. For a performed cut in cluster C_i, $V_C(C_j)$ for any $j \neq i$ remain unchanged.
2. SEP and DISP exhibit some directional recurrence relationship in $\mathcal{T}$: knowing these values for a given cut, we can deduce them for a neighboring cut left and right (cf. Fig. 3) by a Double Depth-First Search.

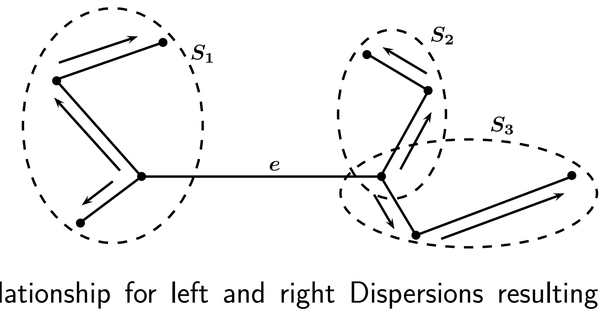

Figure 3: Recursive relationship for left and right Dispersions resulting from the cut of edge e: $\mathrm{DISP}_{left}(e) = \max(w(S_1))$, $\mathrm{DISP}_{right}(e) = \max(w(S_2), w(S_3))$ where $w(.)$ returns the edge weights of the subtree in parameter. Separation works analogically.

Experimental results

1. **Safety of the sketching:**

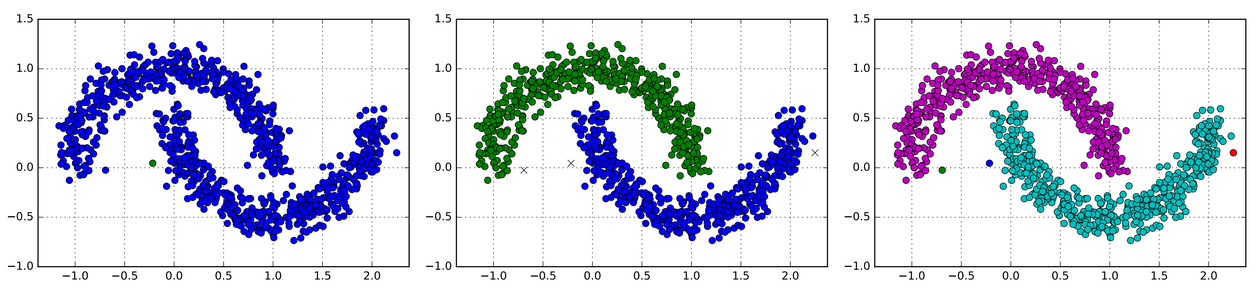

Figure 4: Noisy moons: SEMST, DBSCAN ($\epsilon = 0.15$, $minPts = 5$), DBMSTClu with AMST

2. **Scalability:** experiments within the Stochastic Block Model

$K \backslash N$	1000	10000	50000	100000	250000	500000	750000	1000000
5	0.34	2.96	14.37	28.91	73.04	148.85	218.11	292.25
20	0.95	8.73	43.71	88.51	223.18	449.37	669.29	889.88
100	4.36	40.25	201.76	398.41	995.42	2011.79	3015.61	4016.13
"100/5"	12.82	13.60	14.04	13.78	13.63	13.52	13.83	13.74

Table 1: DBMSTClu's execution time (in s) varying N and K (avg. on 5 runs).

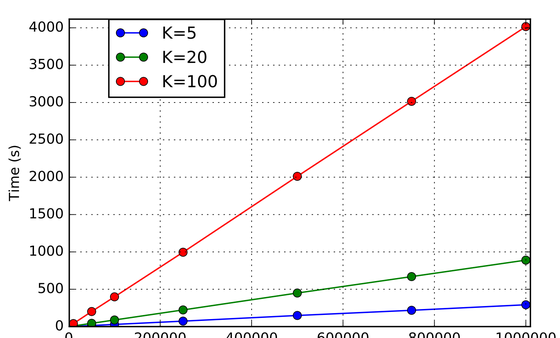

Figure 5: Visualization of Table 1 exhibiting linear property

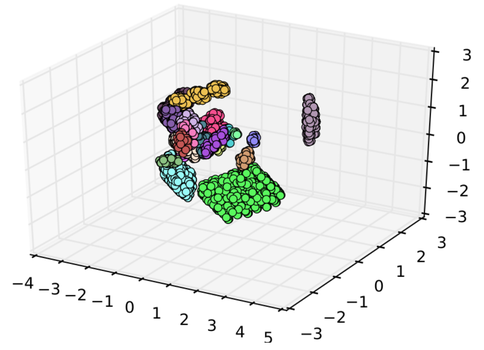

Figure 6: DBMSTClu applied on real dataset mushroom ($N = 8124$) detects 23 clusters in 3.36s while DBSCAN requires 9s.

Conclusion and perspectives

- We introduced a novel space-efficient density-based clustering algorithm working solely on an MST without any parameter. Its robustness has been assessed by using as input an approximate MST, retrieved from the dissimilarity graph sketch, rather than an exact one.
- Further work would be to 1) use DBMSTClu in privacy issues, 2) adapt both the MST recovery and DBMSTClu to the fully online setting by updating the current MST and clustering partition as new edge weight updates are seen.

[1] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 459–467, Philadelphia, PA, USA, 2012. Society for Industrial and Applied Mathematics.

[2] Graham Cormode and Donatella Firmani. A unifying framework for l_0-sampling algorithms. *Distributed and Parallel Databases*, 32(3):315–335, 2014. Special issue on Data Summarization on Big Data.