Hierarchical Density-Based Clustering using Incremental Similarity Search

Erich Schubert

Data Mining, TU Dortmund University, Dortmund, Germany, erich.schubert@tu-dortmund.de

Hierarchical clustering

Probably the earliest clustering method.
Already used in the 1960s for "numerical taxonomy".

Basic principle:

- Start with each point as its own cluster
- Merge the two closest clusters
- Repeat merging until everything is merged into a single cluster

This yields a dendrogram, a tree showing the merging process.

Distances of clusters can be computed in various ways:
e.g., single-linkage, complete, average, Ward's, **HDBSCAN***, …
But the classic approach has cubic $O(N^3)$ complexity, improved algorithms $O(N^2)$ $\Rightarrow$ this does not scale well.

Single-link Clustering with Incremental Neighbor Search

We can obtain edges in increasing-length without a pairwise distance matrix:

- For each point, start an **incremental neighbor search**
- Build a heap of all searchers, ordered by the distance to the nearest neighbor
- Repeat until only one cluster remains:
  - Extract the top searcher from the heap
  - Poll the searcher for the next candidate and reinsert to the heap
  - Merge candidate edge if not already connected

By omitting distance computations of already merged clusters, we hope for $O(N)$ searches in $O(\log N)$ time *on well-behaved data*.
But: in the *worst case*, we have to consider all $O(N^2)$ edges!

Extension to HDBSCAN*

HDBSCAN* clustering, like single-linkage, is based on a minimum spanning tree, but using a modified distance:

$$\text{reach-dist}(a,b) := \max\{d(a,b), \text{coreDist}(a), \text{coreDist}(b)\}$$

Where coreDist(a) is the distance to the minPts nearest neighbor.

Straightforward extension of HSSL to HDBSCAN* clustering:

- determine coreDist for each point
- for each point, add a candidate heap (by reach-dist)
- advance the searchers while their lower bounds are below the current best candidate (exploit $\text{reach-dist}(a,b) \geq d(a,b)$)
- skip already connected points (via union-find)
- add discovered points with reach-dist to the candidate heap

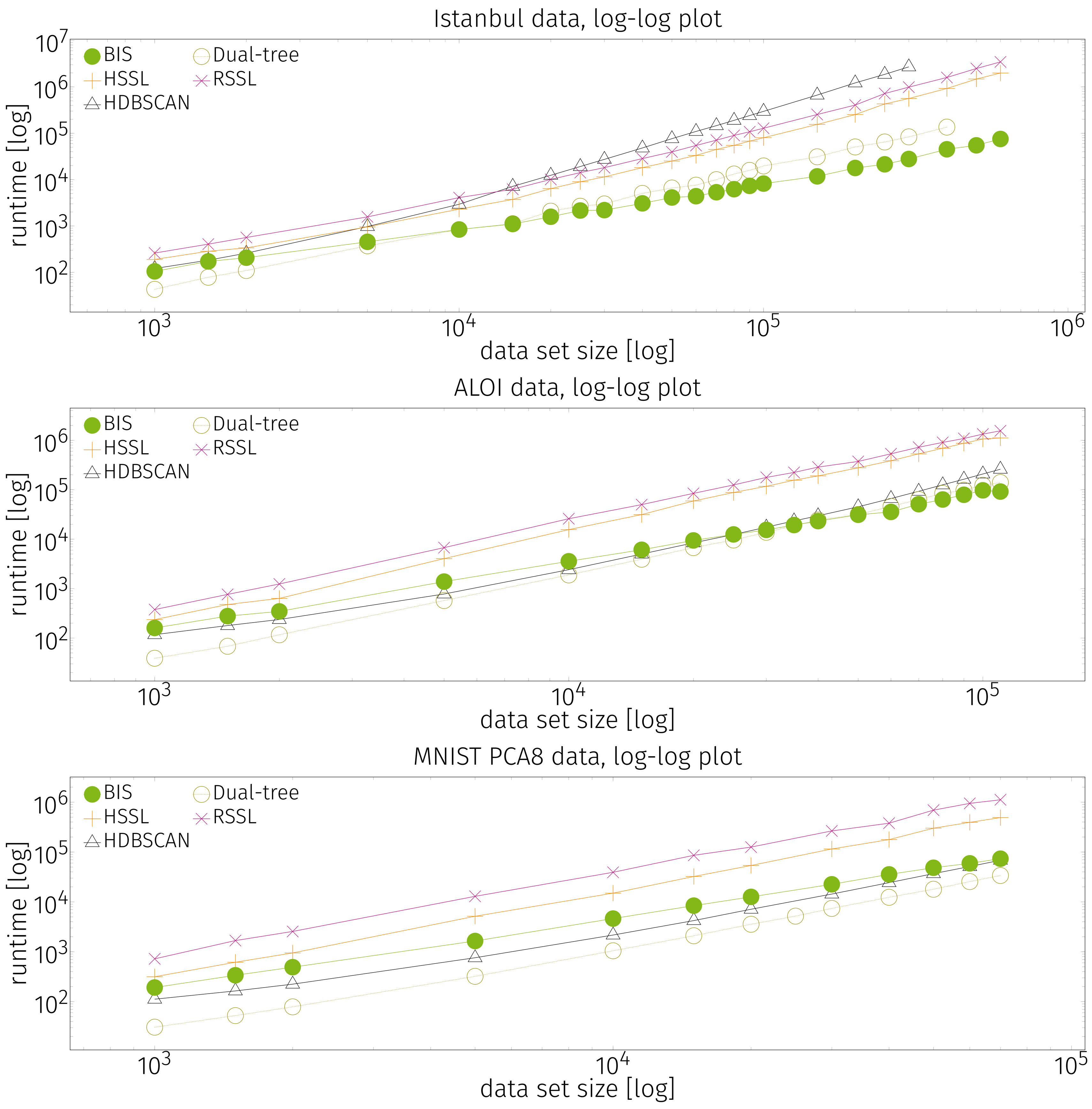
Reducing the Number of Searches

Finding the **longest edge** in the MST is the problem.

We evaluate three variants in the experiments:

- **Heap-of-Searchers Single-Linkage (HSSL):**
  a slight improvement of the basic algorithm using *bounds* to reduce computations
- **Restarting Search Single-Linkage (RSSL):**
  instead of preserving the full search state, only the distance and identity of the nearest candidate for each point is stored. To find the next candidate, search is *restared*, but skips over all results closer than the previous distance
- **Borůvka with Incremental Searchers (BIS):**
  merge every point with its nearest neighbor in each round,
  to reduce the number of iterations and
  identify the longest edge from the "smaller" end first

Experiments: HDBSCAN* Clustering



Unfortunately, only on some data sets we achieve the desired speedups, and the basic $O(N^2)$ method and dual-tree are still competitive on many high-dimensional data sets.

Erich Schubert

- Professor of Data Mining at TU Dortmund
- Unsupervised Machine Learning and AI
- Accelerated Similarity Search
- Cluster Analysis, Outlier Detection, Text Mining

18th International Conference on
Similarity Search and Applications
SISAP 2025, Reykjavik, Iceland