**Center-Based Clustering of Trajectories**

Kevin Buchin, Anne Driemel, Natasja van de L’Isle, André Nusser

**Problem Definition**

Given:

A set of polylines $T$ and $k, \ell \in \mathbb{N}$

Wanted:

Center polylines $C_0, C_1, \ldots, C_k$, such that

- they induce clusters $C_0 \cup C_1 \cup \cdots \cup C_k = T$
- each $C_i$ has at most $\ell$ nodes
- they minimize $\max_{i \in [1:k]} \ell_i C_i \cdot d_F(I, C_i)$

**Fréchet Distance**

- curve similarity measure
- has a notion of traversals
- especially good for spatio-temporal data

**Intuition**

Wanted:

Formal Definition

$$d_F(\pi, \sigma) := \min_{f \in T_\pi, g \in T_\sigma} \max_{t \in [0,1]} ||\pi_f(t) - \sigma_g(t)||$$

$\pi, \sigma$ = polygonal curves

$T_\pi = \text{monotone, continuous traversals of } \pi$

$T_\sigma = \text{monotone, continuous traversals of } \sigma$

**Algorithm**

1. Compute Initial Clustering
2. Improve Centers
3. Update Clusters
4. If the centers changed, GOTO 2

**Details**

1. use Gonzalez’ Algorithm
   1. choose arbitrary curve, compute its $\ell$-simplification, and call the result $C_1$
   2. for $i \in \{2, \ldots, k\}$: choose input curve that is farthest from $C_1, \ldots, C_{i-1}$, compute its $\ell$-simplification and call it $C_i$

2. naive: find the best $\ell$-simplification among the curves in each cluster
   - issue: dependent on input curves, and none of them might be suitable
   - we propose: Fréchet centering

3. match each curve to its closest center

**Implementation and Experiments**

- Implementation* in modern C++
- Uses recent advances for fast Fréchet distance computation
- Comparison to complete linkage with k-means center
- Experiments on two data sets:
  - Pigeon Data
  - Handwritten Characters

<table>
<thead>
<tr>
<th></th>
<th>time (s)</th>
<th>diam.</th>
<th>radius</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline approach</td>
<td>21.83</td>
<td>0.076</td>
<td>0.0104</td>
</tr>
<tr>
<td>our approach</td>
<td>2.04</td>
<td>0.076</td>
<td>0.0092</td>
</tr>
</tbody>
</table>

Table: Runtime comparison on the pigeon data set of our approach and the baseline approach.

The quality of our approach is slightly better on the two measures while being one order of magnitude faster.

*Code available at: https://gitlab.com/anusser/klcluster-sigspatial19