Visualization and Analysis of Geographic Information

Algorithms and Data Structures

João Valença
valenca@student.dei.uc.pt

Departmento de Engenharia Informática
Universidade de Coimbra

October 24, 2013

Motivation

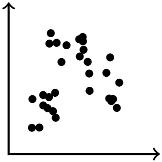
- Reduce visual information when displaying large numbers of geographic points

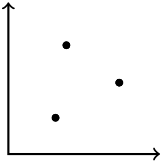
Motivation

- Reduce visual information when displaying large numbers of geographic points
- Find a representative subset of a collection of geographic points.

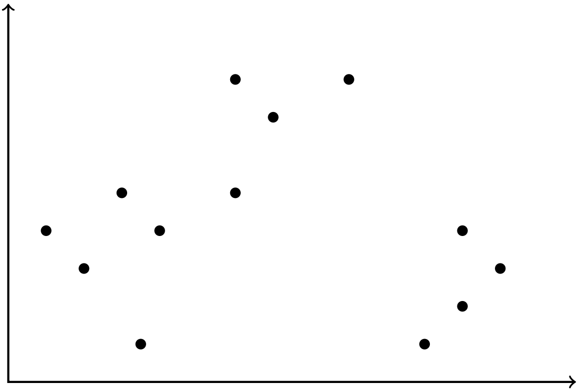
Motivation

- Reduce visual information when displaying large numbers of geographic points
- Find a representative subset of a collection of geographic points.



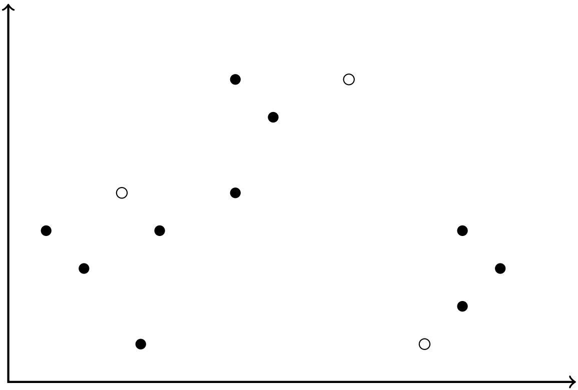


Problem



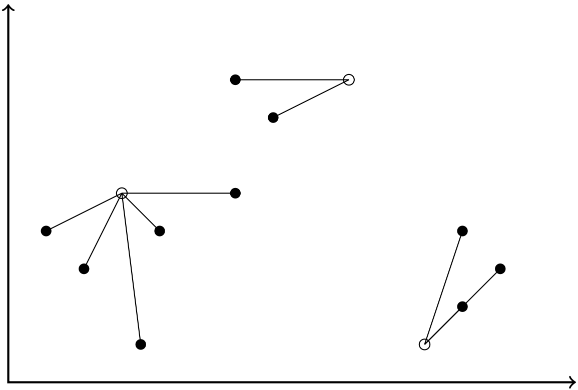
Given a set of points P and an integer $k \leq |P|$

Problem



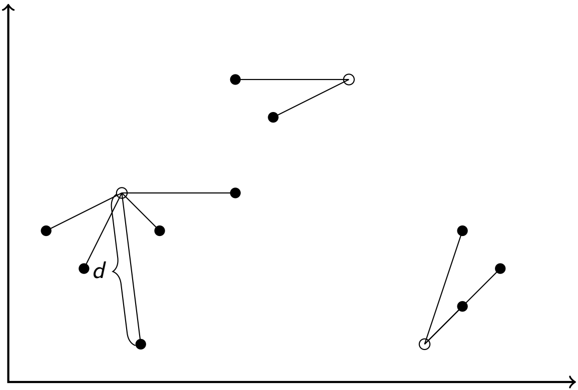
Find a subset of points $S \subseteq P, |P| = k$, that is "representative" of P

Problem



Consider the distance between every point in $P \setminus S$ and its closest point in S

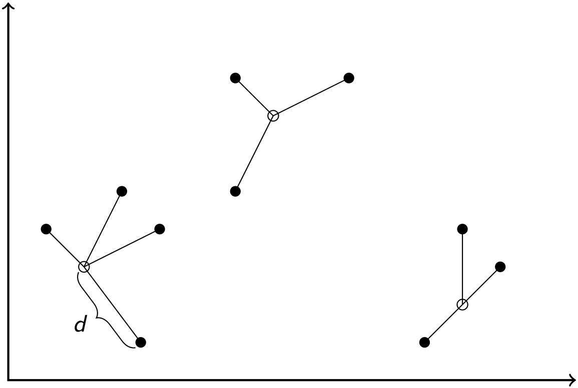
Problem



Consider the distance between every point in $P \setminus S$ and its closest point in S

k-centre problem: Find subset S that minimises the largest distance

Problem



Optimal solution

Work Plan

- 1^{st} Semester
  - Literature Review: Geographic Information Systems, OGC Standards WMS, WFS, Map Projections, algorithms and heuristics for clustering and facility-location problems.
  - Development of a Branch-and-Bound approach.
- 2^{nd} Semester
  - Development of heuristic approaches.
  - Experimental analysis of the algorithms.
  - Comparison between different approaches using Open Street Map data.
  - Integration of the algorithms in the visualisation framework through web-mapping standards (WMS/WFS).

Deterministic Approach

- Use a branch-and-bound approach.

Deterministic Approach

- Use a branch-and-bound approach.
- For each point, decide if it is a Centroid or not.

Deterministic Approach

- Use a branch-and-bound approach.
- For each point, decide if it is a Centroid or not.
- Incrementally update the value of the objective function.

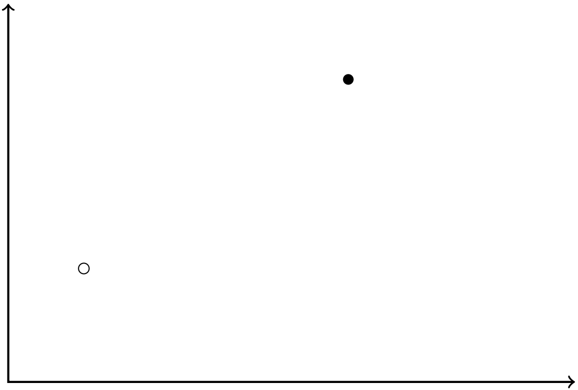
Deterministic Approach

- Use a branch-and-bound approach.
- For each point, decide if it is a Centroid or not.
- Incrementally update the value of the objective function.
- Use bounds to discard branches that do not contribute to the optimal solution.

Deterministic Approach

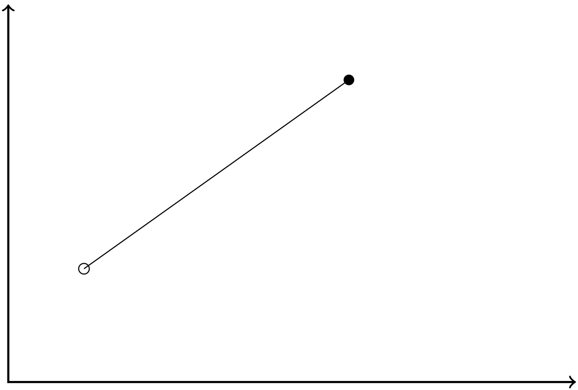
Add a point.

Deterministic Approach



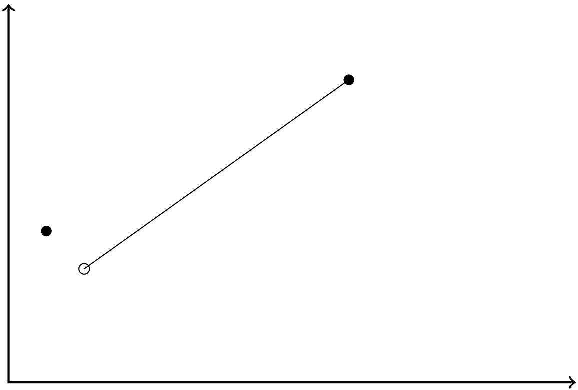
Add a non-centroid point (in black).

Deterministic Approach



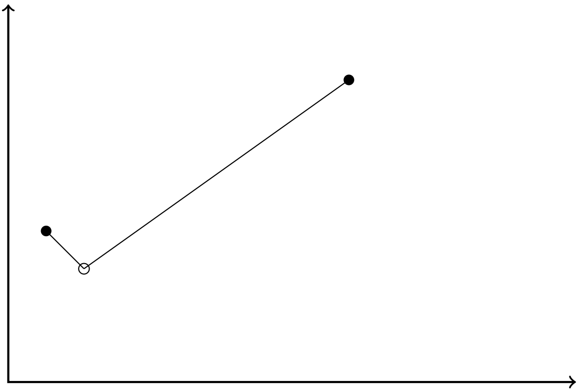
Assign each non-centroid to the closest centroid.

Deterministic Approach



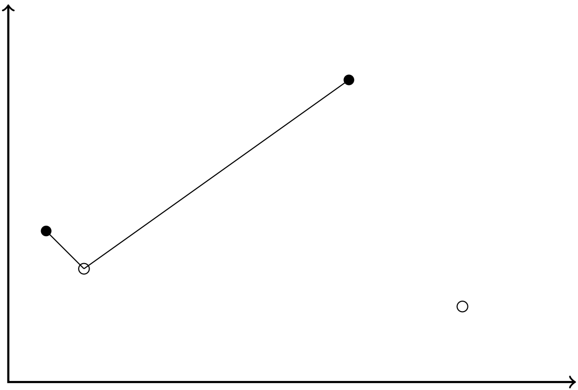
Adding non-centroid points.

Deterministic Approach



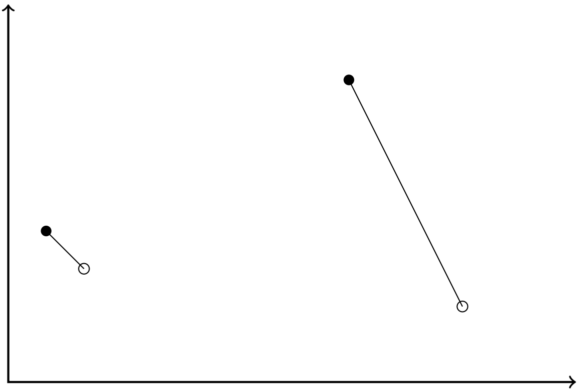
Assign each non-centroid to the closest centroid.

Deterministic Approach



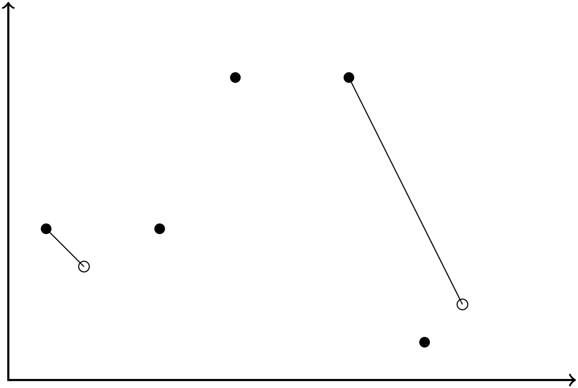
Adding a centroid.

Deterministic Approach



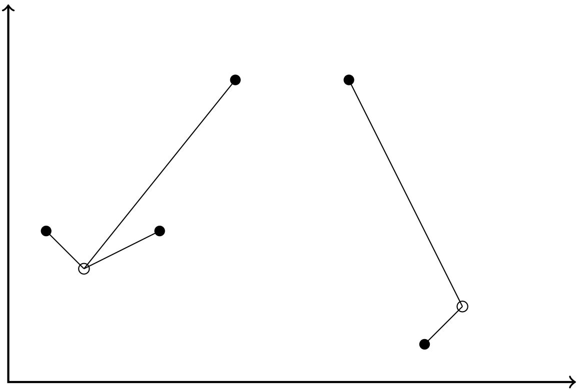
Assign each non-centroid to the closest centroid.

Deterministic Approach



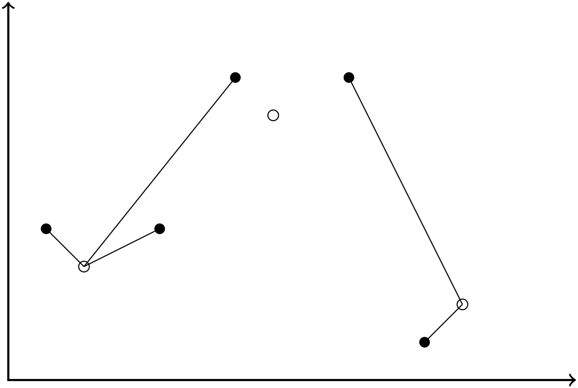
Adding more non-centroid points.

Deterministic Approach



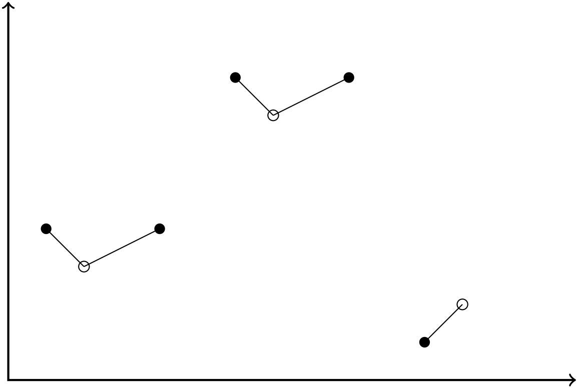
Assign each non-centroid to the closest centroid.

Deterministic Approach



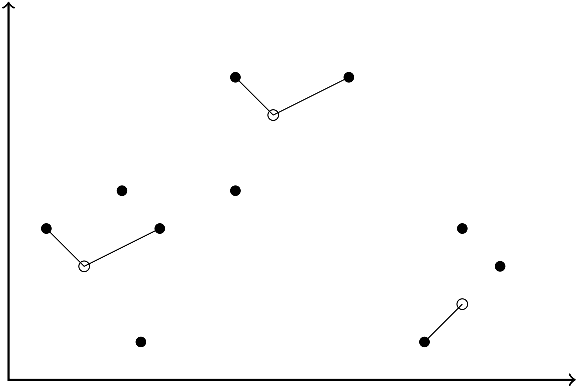
Adding another centroid point.

Deterministic Approach



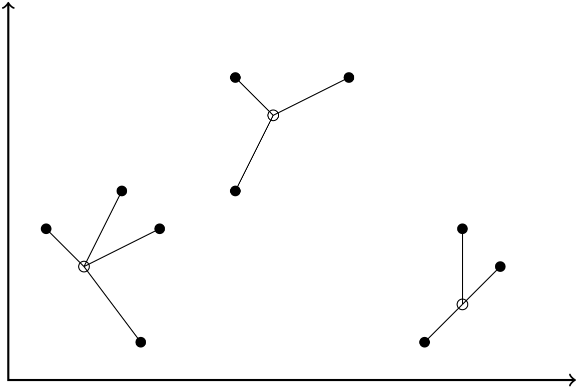
Assign each non-centroid to the closest centroid.

Deterministic Approach



Finishing the solution branch.

Deterministic Approach

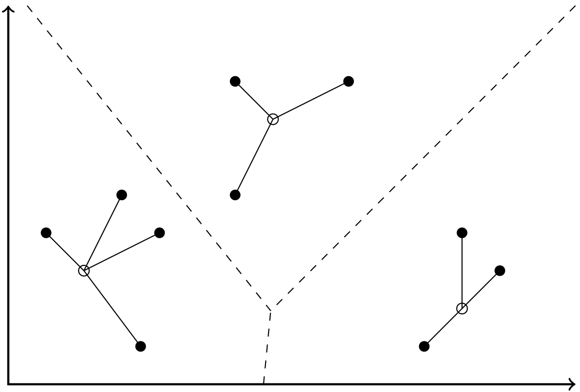


Finishing the solution branch.

Future Work

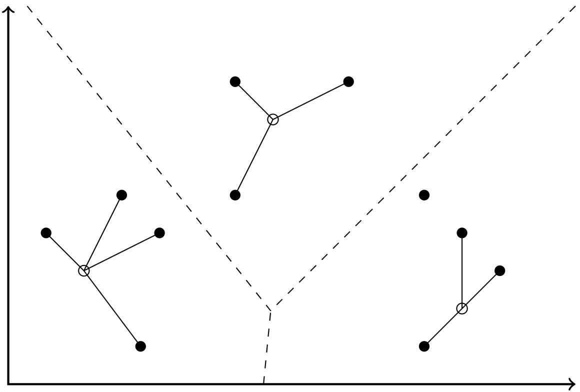
- Implement a planar location data structure to allow Voronoi diagrams to speed insertions and deletions of points to the solutions up from $\mathcal{O}(n)$ to $\mathcal{O}(\log n)$
- Explore bounds to cut the recursive tree.
- Explore heuristic approaches to generate acceptable non-optimal solutions for larger problems.
- Apply and benchmark approaches with real-life data.

Future Work



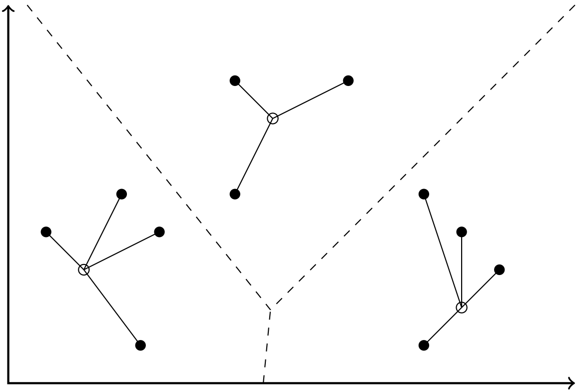
The Voronoi diagram partitions the space in cells.
Each cell contains all points whose closest centroid is the same.

Future Work



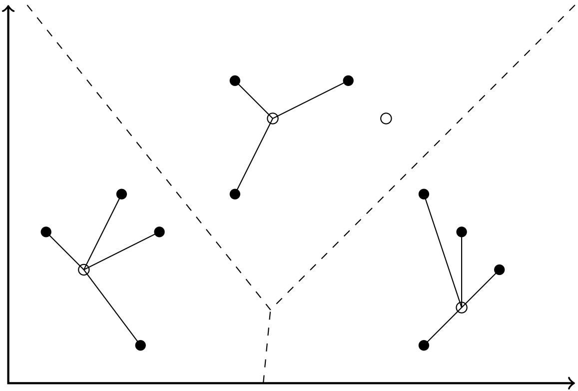
With the help of additional structures, insertion of a non-centroid point can be done in $\mathcal{O}(\log n)$

Future Work



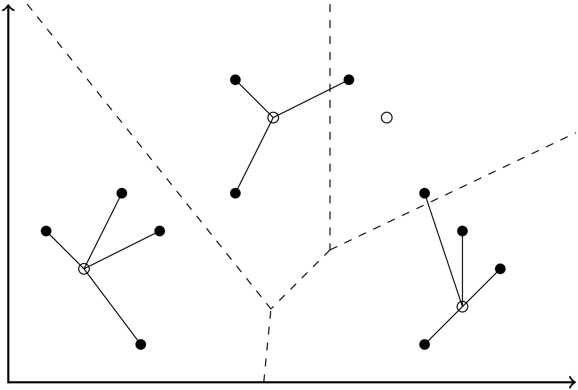
Adding n non-centroid points takes $\mathcal{O}(n \log n)$
Better than the naïve $\mathcal{O}(n^2)$

Future Work



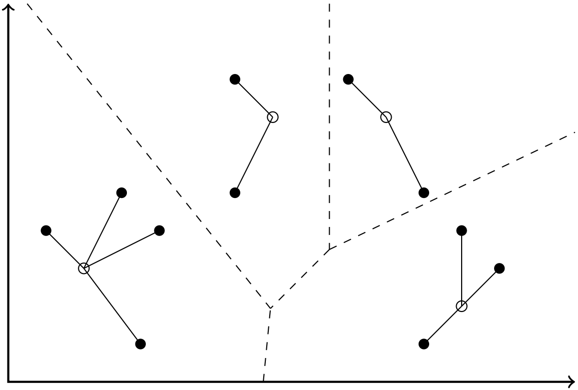
Using an incremental Voronoi algorithm, adding a centroid point also takes $\mathcal{O}(\log n)$, and creates another cell.

Future Work



Updating the solution only requires a check through each of the new cell's neighbours.

Future Work



This step can still take $\mathcal{O}(n^2)$ comparisons, but this occurrence is fairly rare.