

Visualisation and Analysis of Geographic Information

Algorithms and Data Structures

João Valença
valenca@student.dei.uc.pt

Department of Informatics Engineering
University of Coimbra

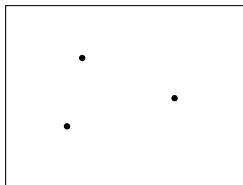
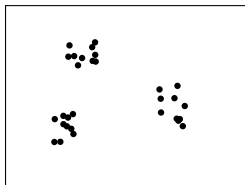
February 2, 2015

Motivation

- ▶ Geometric Information System Project
- ▶ Collaboration with Smartgeo
- ▶ Web application

Motivation

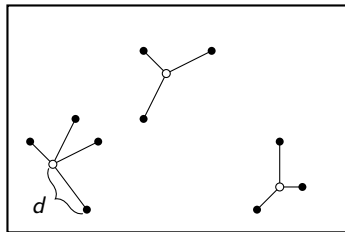
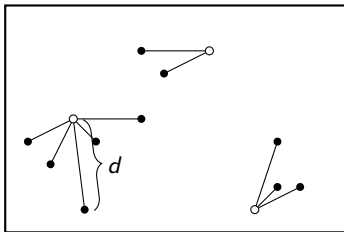
- ▶ Reduce visual information when displaying large numbers of geographic points
- ▶ Find a representative subset of a collection of geographic points



- ▶ The set of points is dynamic

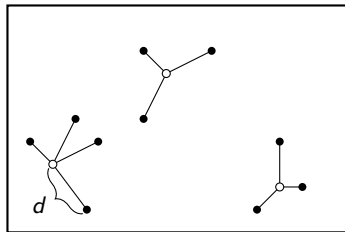
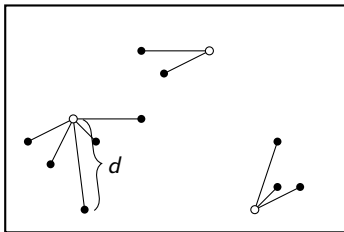
Coverage

► Minimising Coverage



Coverage

► Minimising Coverage



$$\min_{\substack{P \subseteq N \\ |P|=k}} \max_{n \in N} \min_{p \in P} \|p - n\|$$

Work Plan

- ▶ 1st 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.
- ▶ 2nd Semester
 - ▶ Development of heuristic approaches.
 - ▶ Experimental analysis of the algorithms.
 - ▶ Integration of the algorithms in the visualisation framework through web-mapping standards (WMS/WFS).
 - ▶ Comparison between different approaches using Open Street Map data.

Integer Linear Programming

$$\begin{array}{ll}\text{minimise} & D \\ \text{subject to} & \sum_{j=1}^N y_j = k \\ & \sum_{j=1}^N x_{ij} = 1 \quad i = 1, \dots, N \\ & \sum_{j=1}^N d_{ij} x_{ij} \leq D \quad i = 1, \dots, N \\ & x_{ij} \leq y_j \quad i = 1, \dots, N; j = 1, \dots, N \\ & x_{ij}, y_j \in \{0, 1\} \quad i = 1, \dots, N; j = 1, \dots, N\end{array}$$

Branch-and-bound

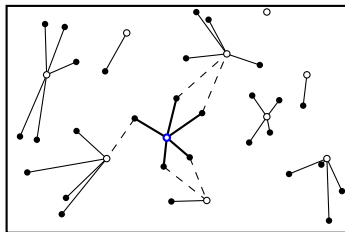
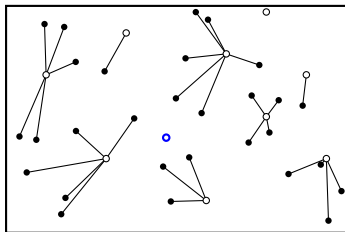
- ▶ Branching
 - ▶ Divide search space in a binary tree
 - ▶ At each step, decide if a point is a centroid or non-centroid
 - ▶ Update objective function accordingly

Branch-and-bound

- ▶ Branching
 - ▶ Divide search space in a binary tree
 - ▶ At each step, decide if a point is a centroid or non-centroid
 - ▶ Update objective function accordingly
- ▶ Bound
 - ▶ Assume best possible case
 - ▶ Prune tree

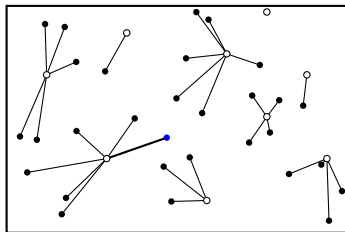
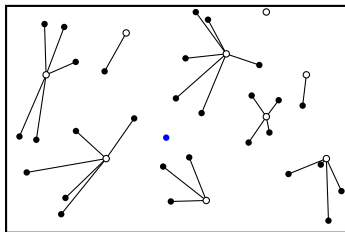
Branch-and-bound

- ▶ Inserting a Centroid
 - ▶ Search all non-centroids for assignment update
 - ▶ Smaller coverage value



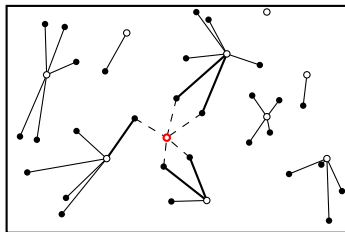
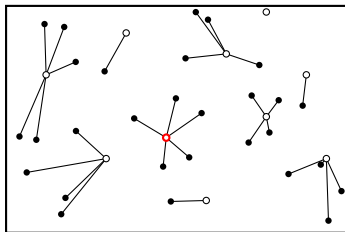
Branch-and-bound

- ▶ Inserting a Non-centroid
 - ▶ Search for closest centroid
 - ▶ Higher coverage value



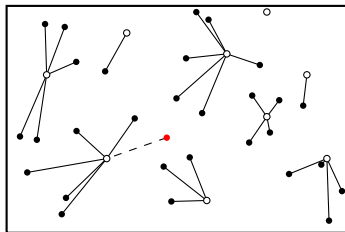
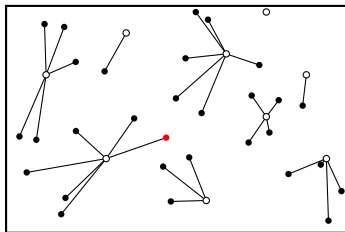
Branch-and-bound

- ▶ Removing a Centroid
 - ▶ Update all non-centroids
 - ▶ Higher coverage value



Branch-and-bound

- ▶ Removing a Non-centroid
 - ▶ Update objective function
 - ▶ Smaller coverage value

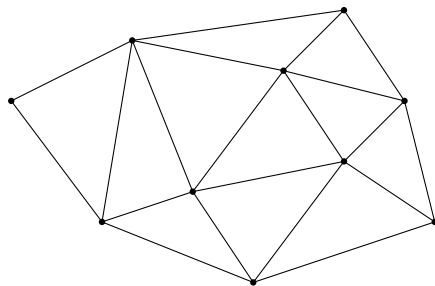


Geometric Approach

- ▶ Unnecessary number of calculations
 - ▶ Use geometric structures to speed objective function update

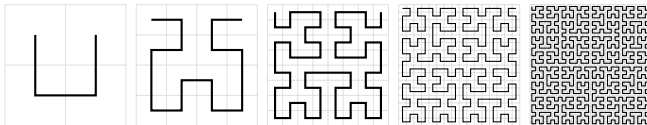
Geometric Approach

- ▶ Unnecessary number of calculations
 - ▶ Use geometric structures to speed objective function update
 - ▶ Delaunay triangulations



Geometric Approach

- ▶ Greedy Routing
- ▶ Use Hilbert curves to minimise distance between consecutive routing calls

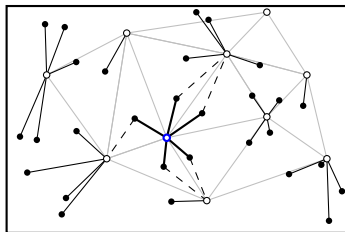
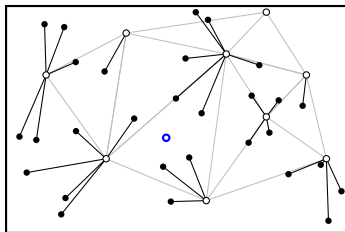


Geometric Approach

- ▶ Pre-process:
 - ▶ Initialize Delaunay Triangulation
 - ▶ Sort points by Hilbert curve

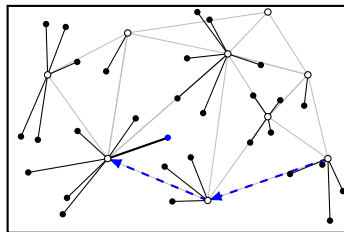
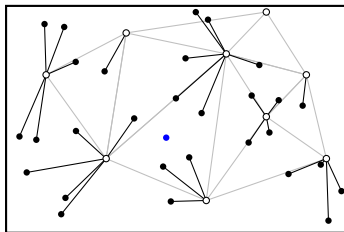
Geometric Approach

- ▶ Inserting a Centroid
 - ▶ Insert centroid in triangulation
 - ▶ Search all non-centroids for assignment update
 - ▶ Smaller coverage value



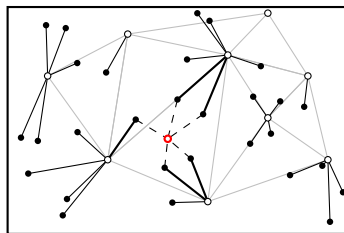
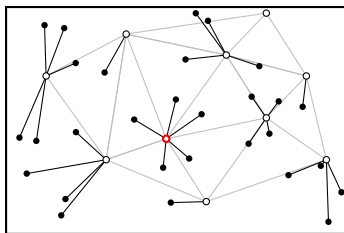
Geometric Approach

- ▶ Inserting a Non-centroid
 - ▶ Search for closest centroid using greedy routing
 - ▶ Higher coverage value



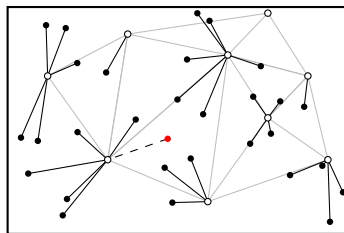
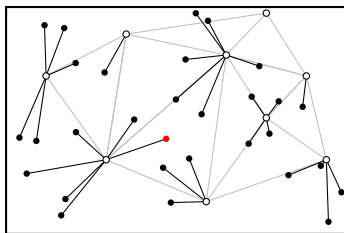
Geometric Approach

- ▶ Removing a Centroid
 - ▶ Revert assignment
 - ▶ Remove centroid from triangulation
 - ▶ Higher coverage value



Geometric Approach

- ▶ Removing a Non-centroid
 - ▶ Update objective function
 - ▶ Revert assignment
 - ▶ Smaller coverage value



Algorithm Comparison

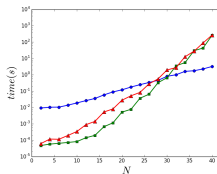
Algorithm	Insert		Remove	
	Centroid	Non-Centroid	Centroid	Non-Centroid
Naïve BB	$\Theta(N)$	$\Theta(K)$	$\Theta(N)$	$\mathcal{O}(1)$
Geometric BB Average Case*	$\mathcal{O}(\log K + N/K)$	$\mathcal{O}(\sqrt{K})$	$\mathcal{O}(N/K)$	$\mathcal{O}(1)$
Geometric BB Worst Case	$\mathcal{O}(K + N)$	$\mathcal{O}(K)$	$\mathcal{O}(N)$	$\mathcal{O}(1)$

Algorithm Comparison

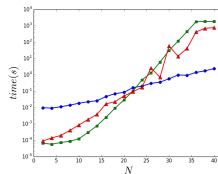
- ▶ Tests Performed
 - ▶ Effect of N
 - ▶ Change N
 - ▶ Keep proportional K
 - ▶ Effect of K
 - ▶ Fixed N
 - ▶ Change K

Algorithm Comparison

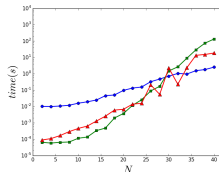
Effect of N



(a) $K = 0.25N$



(b) $K = 0.5N$

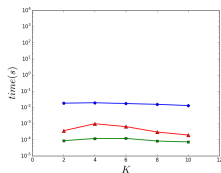


(c) $K = 0.75N$

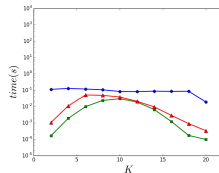
● – Integer Linear Programming ▲ – Delaunay Assisted B&B ■ – Naive B&B

Algorithm Comparison

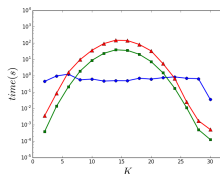
Effect of K



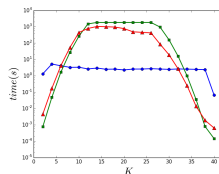
(a) $N = 10$



(b) $N = 20$



(c) $N = 30$



(d) $N = 40$

● – Integer Linear Programming ▲ – Delaunay Assisted B&B ■ – Naive B&B

Future Work

- ▶ Heuristic Approach
- ▶ Approximation Algorithms
- ▶ Adapt to dynamic set
- ▶ Integration with WFS standard
- ▶ Benchmark with real data