Algorithms and Data Structures for Large Scale Geographic Information Systems: A Performance Analysis

João Valença
CISUC, DEI, University of Coimbra
Coimbra, Portugal
valenca@student.dei.uc.pt

Luís Paquete
CISUC, DEI, University of Coimbra
Coimbra, Portugal
paquete@dei.uc.pt

Pedro Reino
Smartgeo Solutions
Lisboa, Portugal
pedro.reino@smartgeo.pt

Carlos Caçador
Smartgeo Solutions
Lisboa, Portugal
carlos.cacador@smartgeo.pt

Abstract

This project aims to develop a GIS application operating through a Web platform in order to allow for a low cost and simplified integration, management and manipulation of georeferenced information. In particular, the goal is to develop a way to efficiently extract a subset of a collection of geographic points of interest whilst keeping the representativeness of the whole set. This problem of finding such representation is recast as a discrete optimisation problem. The approaches covered in this work include exact algorithms for finding minimum coverage subsets, as well as heuristic approaches for finding good approximations in real time.

Keywords: Geographic Clustering, k-Center, Coverage, Branch-and-Bound, Set Covering

1   Introduction

One obstacle when representing large amounts of geographic data is that the number of points to display can be overwhelming for a human, as well as computationally intensive to render for a machine. As such, there is a need to develop and implement a viable way to reliably calculate and display a subset of geographic points, whilst keeping a degree of representativeness of the larger set.

The purpose of this project is to develop a real-time algorithmic framework that can analyse geographic data provided by a geographic information system infrastructure. More precisely, the developed algorithm has to be able to aggregate and select geographic points according to a given set of criteria in real time. The chosen subset should keep a measure of representativeness of the larger set.

Figure 1 shows a representative subset of the set of cities, towns and villages in Sweden. The points were taken from a dataset available in [4].

Figure 1: The Sweden dataset (left) and a representative subset (right)
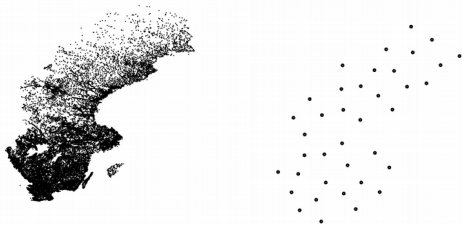
Problem Definition

Representativeness consists of finding a subset of points in a larger set that meets some representation quality. The subset chosen should be able to keep some properties of the original set, such as density, or distribution.

In this work, representativeness is understood as finding the subset that minimises the maximum distance between the the not chosen points, and their closest counterparts within the chosen set. This notion of representativeness is known in discrete optimisation as *coverage*. The problem can be approached by finding the subset of cardinality *k* that minimises the value of coverage, known as the *k-center* problem [3]. Another approach is to establish an arbitrary minimum distance and minimise the number of points needed to cover all the remaining points within that distance. This approach can be recast as a set cover problem.

In order to address these problems, several algorithms were developed and implemented. Optimal approaches to the problems include integer linear programming and branch-and-bound algorithms. However, the inherent overhead in these algorithms makes them unsuitable in real-time applications. For achieving an acceptable approximation to the optimal solution within a reasonable time frame of what is expected in a real-time web application, heuristic methods need to be used.

3   Data Structure

The application displays a rectangular window, showing a cut of a geographical region containing a small set of points that is representative of all the points of interest within that region. The algorithm chosen must be able to choose a representative set of points within the window quickly, as well as be able to recompute a new set points for a new window, resulting from panning or zooming the display window over the region.

The algorithms tested include exact algorithms for finding the minimum coverage subset for a given cardinality. These are comprised of two branch-and-bound algorithms: a naïve incremental approach and a geometric incremental approach that makes use of the properties of Delaunay triangulations in order to speed up point location queries via the Greedy Routing algorithm [1].

Figure 2: CPU time for various values of N

Figure 3: Number of points in the covering set with respect to the minimum distance.
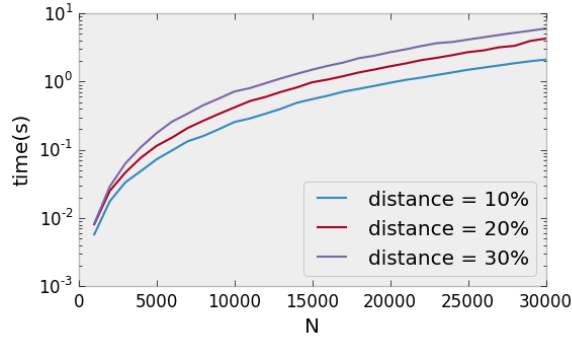
An exact integer linear programming approach is benchmarked as well. This project also includes a few heuristic approaches and approximation algorithm techniques to solve the representation problem more efficiently.

The algorithm serves as the middleware responsible for filtering the response of a GIS server to a Web Feature Service, or WFS request. Finally, a web application receives the response filtered by the algorithm and interacts with a human user.

The candidate algorithms were tested and benchmarked using data from the Open Street Map project. The project features large quantities of open source geographic data, as well as a versatile API for fetching data.

4 Experimental Results

The first approach solves the representation problem by finding the minimum the coverage subset with a fixed cardinality. The solution approaches include Branch-and-Bound algorithms, as well as Integer Linear Programming. The performance of these algorithms is too slow even for a small number of points, and takes minutes to solve very small instances of the problem, deeming these algorithms unusable in the context of a web application.

The second approach, i.e. minimising the number of points chosen given a fixed minimum distance, was solved using a greedy approximation algorithm to the set covering problem. This algorithm guarantees an approximation with at most $ln(N)$ points more than the optimal solution [2]. This algorithm yielded much faster times, whilst keeping an acceptable quality for the resulting sets.

Figures 2 and 3 plot some results obtained by running the algorithm on a square region with uniformly distributed points. The program was written in $C++$ and compiled using *gcc 4.9.2* on an Arch Linux Operating System (3.19.2-1-ARCH). The CPU used was an Intel i7 Dual-Core with 2Ghz, and 8Gb of 1600MHz memory.

Figure 2 shows the CPU-time in seconds taken by the algorithm with respect to the number of points. A minimum distance was fixed (as a percentage of the region size) for the different runs. For each run, the number of points was varied to plot the CPU time as a function of N.

Figure 3 shows the number of points K that the algorithm returned for the representative set with respect to the minimum distance. Each run has a different value for N. The minimum distance is varied between 10% to 45% of the dimension of the side of the square region. Smaller values of K represent better solutions.
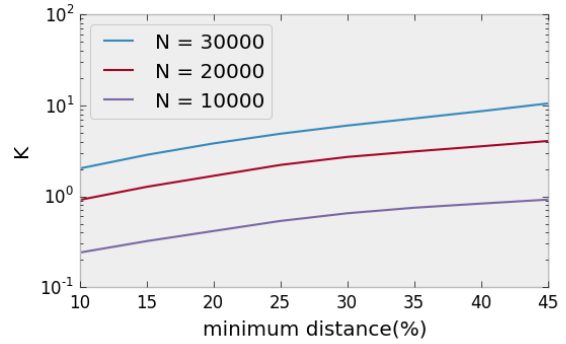
These results show that the algorithm can be used in the context of a web application. They may still be improved by implementing a more efficient range query search algorithm in the pre-processing stage of the approximation algorithm. Approaches such as heuristic algorithms and other approximation algorithms are also under investigation.

References

[1] P. Bose and P. Morin. Online routing in triangulations. *Algorithms and Computation*, LNCS, 1741:113–122,1999

[2] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3): 233–235, 1979.

[3] N. Megiddo and A. Tamir. New results on the complexity of p-centre problems. *SIAM Journal on Computing*, 12(4):751–758, 1983.

[4] Maths Department of the University of Waterloo. Travelling sales-man problem datasets - sweden. URL http://www.math.uwaterloo.ca/tsp/world/sw24978.tsp.

Acknowledgements