A system for planning street cleaning services by clustering

Pasquale Lepore, Gianmarco Ialeggio
Development section
Officine Sostenibili società benefit srl
Benevento, Italy
pasquale.lepore@officinesostenibili.com,
gianmarco.ialeggio@officinesostenibili.com

Francesco Cosimo Mazzitelli
Dept. of Engineering
University of Sannio
Benevento, Italy
f.mazzitelli@studenti.unisannio.it

Eugenio Zimeo
Dept. of Engineering
University of Sannio
Benevento, Italy
zimeo@unisannio.it

***Abstract*—This extended abstract presents an automated approach to optimize street cleaning operations in urban environments. Traditional manual methods for sizing and managing street cleaning services are inefficient and costly. The study proposes a methodology that combines clustering algorithms and domain-specific technologies to automate the planning process. Leveraging graph theory, the approach guarantees the adjacency of the grouped areas respecting the threshold constraints ensuring correct sizing for the deployment of cleaning teams.**

***Index Terms*—graphs, clustering, smart waste management, service architecture**

I. Introduction

Urban hygiene is a critical component of public health and the overall quality of life in cities. The effective management of street cleaning services, which involves the removal of solid waste, debris, and other pollutants from road surfaces, is essential for maintaining a clean, healthy, and visually appealing urban environment. Traditionally, the sizing and management of street cleaning services have been computed using manual methods and heuristic approaches based on human experience and intuition. These traditional methods are often inefficient, resulting in time-consuming processes and higher operational costs. The manual approach requires the cartographer to first identify the urban center and the countryside with a preliminary analysis based on evaluating the relationship between the area and the users residing in it. After this initial subdivision, the aggregation begins, starting from the street or census section with the smallest area-user ratio of the urban center. The cartographer must take into account the number of users that can be served by a fixed number of teams. This operation is performed to group adjacent streets or census sections not exceeding the threshold value representing users. Every time the threshold is reached a new cluster is created representing the work area of a team.

The industries' digitalization and advances in computer engineering offer not only infrastructures but also a set of algorithms capable of managing and automating these processes. Clustering techniques have emerged as a promising tool for enhancing the planning, computation time, and efficiency of street cleaning operations. The proposed analysis combines clustering algorithms and domain-specific technologies to create a general-purpose solution that can be adapted to various sizes and complexities of urban street networks. The paper discusses the results obtained from case studies in different urban contexts, highlighting the benefits and potential of this innovative approach in the urban hygiene field.

II. Methodology

The proposed solution aims to automate the steps made by the cartographer during the aggregation process. This operation must be performed to guarantee the adjacency of grouped areas and the respect of a threshold constraint. This constraint refers to the maximum number of users to be served in each macro-section and is computed based on the availability of vehicles. The problem was modeled using graph theory, which allowed us to build one representing the geometry of the area of interest. The resulting graph is characterized by nodes representing census sections and streets and arcs representing adjacency and intersections. The node attributes were extracted directly from the shapefile, the location of the nodes instead, was calculated by extracting the centroid of the polygon. The adjacency condition was computed with a check on the intersections of the polygons, identifying shapes which have at least one point in common.

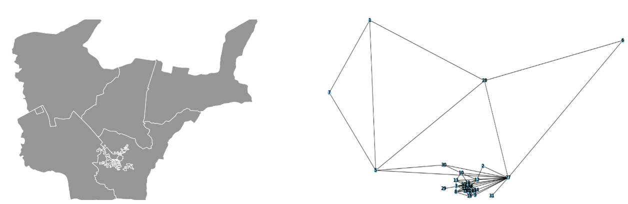

Fig. 1. Shape to graph conversion example

After the graph construction, the starting node is identified as the one with the lowest section-area/user ratio. The graph exploring algorithm for threshold clustering is characterized as follows:

- Insert the starting node in the queue.
- If the queue is not empty extract a node and perform a check on the cluster-users sum in order to assert that such value is less than the threshold.

- Mark the node as visited and insert all the neighbors in the queue.
- Insert the node into the cluster and repeat.

If the number of clusters required for the correct subdivision of the shape is known spectral clustering is performed. This cluster algorithm is based on identifying communities of nodes in a graph based on the edges connecting them.

III. Architecture Description

As illustrated in Figure 2, shapefiles are initially transmitted from QGIS, through a Python plugin. The developed plugin includes a user-friendly interface that facilitates the retrieval of a shapefile from the local filesystem. Once a shapefile is selected, the plugin employs GeoPandas, a Python library used to convert the shapefile into a GeoJSON format, essential for the subsequent transmission process.

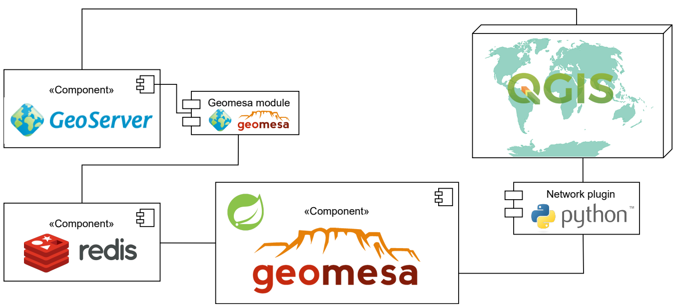

Fig. 2. System architecture

The converted GeoJSON file is then transmitted via a RESTful API to the endpoint of a Java application. This application, built using the Spring framework, performs advanced geodata manipulation operations leveraging GeoMesa, an open-source, Apache-licensed suite of tools designed to enable large-scale geospatial analysis on distributed computing systems. Upon receiving the GeoJSON data, the Java application extracts the necessary information. Depending on the type of operation required, a specific graph is created. Then the application starts clustering and groups the geospatial data into sets based on predefined criteria. This process has been described in detail in the previous section, outlining the methodologies and algorithms used. The processed data is then prepared for storage using the Redis Datastore, an open-source, in-memory data structure store. Redis is known for its versatility as a database, cache, and message broker. A SimpleFeature schema specific to the computed data is created to store the processed data. This schema is designed to accommodate mixed geometries, providing a more flexible and generic solution for storing geospatial data. The next step involves the integration with GeoServer, an open-source server that allows users to share, process, and modify geospatial data across the Internet. It supports the open standards of the Open Geospatial Consortium (OGC), making it compatible with a wide range of GIS applications and tools. By implementing the Web Map Service (WMS) standard, GeoServer can create maps in various output formats using OpenLayers, a free and open-source map library. Additionally, GeoServer adheres to the Web Feature Service (WFS) and Web Coverage Service (WCS) standards, which facilitate simplified sharing and manipulation of geospatial data. The GeoServer connector module, specifically designed for integration with Redis and GeoMesa automatically retrieves the data from the Redis datastore, allowing users to edit and share the data used to generate maps. Once the data has been retrieved and processed by GeoServer, it is sent back to QGIS via the WFS protocol. This return transmission ensures that the cartographer has access to the updated and processed data within their familiar QGIS environment allowing also to modify the output as needed.

IV. Results

We report an example with real data to show how the clustering is performed. The system was tasked with clustering roads for sweeping services, using a threshold value of 2000 meters, which represents the maximum road length that each team or vehicle can manage. Every road, represented in the shapefile as a LineString, is treated as a node and every adjacency or intersection as an edge.

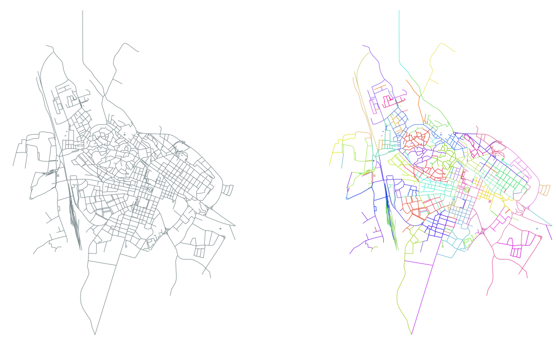

Fig. 3. Clustering of roads for the sweeping service using the threshold

Partition_ID	Length	Partition_ID	Length
0	1681,44	3	1736,54
1	1904,6	4	155,98
2	28,2	5	1925,19

The table reports the results of the first six clusters, some of these values fall significantly below the threshold. This is due to the territory's morphology and the uneven distribution of information on the graph. Consequently, small clusters are formed that cannot connect to others, either because they are isolated or because the other clusters have already reached the threshold.

V. Conclusion

This system represents a starting point in the field of urban hygiene bringing a significant speedup to the task of assigning teams to specific areas of the city not only facilitating the cartographer's work but also providing more accurate solutions. Experts have reported at least 30% increase in task speed, and the fact that clusters never exceed the user-defined threshold in any case.