Graph Clustering Methods in Distributed Socio-Technical Systems

Zeljko Stojanov[1]

[1]University of Novi Sad, Technical Faculty "Mihajlo Pupin", Zrenjanin, Serbia

Abstract

Socio-technical systems are characterized by dynamic relations between human actors, organizations, and technical components, requiring sophisticated mathematical and engineering methods for their modelling, analysis, and optimization. Graphs have been recognized as an efficient and powerful abstraction for modelling and analysing these systems, while graph clustering plays an important role in analysing and optimizing their structure and characteristics. A method for graph clustering based on attribute similarity of neighbouring nodes is introduced in this paper. The method contains two algorithms, the first one for checking similarity between neighbouring nodes and the second one for merging similar neighbouring nodes into clusters. The implementation of the method for determining the minimal number of air pollution measurement stations dispersed in urban city environments is presented to illustrate a possible scenario of use. Implementation is based on simulation of the method in a simulation environment that creates initial scenarios with measurement stations presented as graph nodes, create data sets for measurement stations, and calculates the minimal number of measurement stations by using the presented graph clustering method.

Keywords

graph clustering, similarity measure, socio-technical systems, air pollution measurement

1. Introduction

Socio-technical systems (STS) are complex and dynamic systems characterized by relationships and interplay between humans, organizational structures, and technical elements, aimed at achieving specific objectives and behaviours [1, 2, 3]. In general, these systems have complex topologies and very heterogeneous structures [4], the understanding of which requires dual focus and diverse and broad knowledge of both technical and social disciplines [5, 6]. These systems emerge in diverse domains such as healthcare, education, transportation, logistics, social networks, cyber security, and collaborative software development. Modelling such systems includes capturing their structural and behavioural aspects. Graph-based models have emerged and are accepted as a powerful abstraction for representing complex networks in STSs [7, 8, 9], where nodes denote both human and technical entities, while edges denote interactions or relations between them. Graph-based models can effectively represent the complexity of systems, enabling easy and efficient analysis by applying various mathematical methods [10].

7^{th} International Workshop on Information, Computation, and Control Systems for Distributed Environments (ICCS-DE 2025), July 7–11, 2025, Irkutsk, Russia

zeljko.stojanov@uns.ac.rs (Z. Stojanov)

0000-0001-6930-5337 (Z. Stojanov)

ICCS-DE 2025 Workshop Proceedings (iccs-de.icc.ru)

DOI: 10.47350/ICCS-DE.2025.01

Clusters in the context of graphs are sets of nodes that have stronger relationships with each other within a cluster than with nodes in other clusters [11, 12]. Within the context of complex STSs, graph clustering methods play a crucial role in uncovering the latent structures of socio-technical networks and in grouping existing entities into specific clusters to improve system performance and scalability, and to enhance resilience [11, 13]. Graph clustering enables detection of functional or social communities and groups, communication bottlenecks and potential structural vulnerabilities. The flexibility of graph clustering methods enables their adaptation to weighted, directed, dynamic, or multilayer networks, making them suitable for analysing a wide range of STSs [14, 15].

The objective of the presented research is to present a method for graph clustering by checking similarity between neighbouring nodes in a sequence of discrete time moments (observations). The nodes that satisfy the similarity criteria for observed period are merged into clusters, leading to the minimal graph structure derived from the initial graph. For the validation of the method, a simulation environment that generates simulation scenarios with initial graphs and air pollution datasets is created. The implementation of the proposed graph clustering method in the case of determining minimal number of air pollution measurement stations in urban environments is presented for the city of Zrenjanin in Serbia.

2. Graph Clustering in Socio-Technical Systems

Clustering is a process of grouping data in an unsupervised manner based on a similarity measure by analysing underlying structure of complex data in a given heterogeneous context [16, 17, 18]. The result of clustering are clusters in which are highly correlated data. Graph clustering relates to grouping nodes in clusters by considering some similarity measure for nodes and edges in the graph [11]. Since the field of graph clustering is very attractive for modern socio-technical systems and new methods and algorithms are emerging [19], the aim of this review is only to gain insight into the field, and not to provide a complete overview of existing methods.

In general, there are two approaches to graph clustering: (1) Classifying nodes into clusters based on the node similarity measures for all nodes in the graph, and (2) Computing a fitness measure over the set of possible clusters to inform the decision on how good or useful clustering is.

For this research, the first group of approaches is of interest because the method proposed in the next section can be classified as node similarity-based clustering method. For this group of methods, the need for grouping nodes in the same cluster is directly related to their similarity. Instead of a similarity measure, a distance measure can be used, which suggests where the cluster boundary should be located, so that including more outside nodes would not drastically increase the intra-cluster distances [11]. Choosing the most suitable similarity or distance measure depends on the implementation domain and stated clustering task, and for that purpose may be used node or edge properties. A variety of graph clustering methods based on similarity measures have been proposed, like a graph-theoretic clustering method based on the Highly Connected Subgraphs (HCS) algorithm that identifies clusters as highly connected subgraphs [20], a new clustering algorithm based on graph connectivity that builds a similarity graph from

data and identifies clusters as connected subgraphs [21], a multi-similarity spectral clustering method for community detection in dynamic networks that creates similarity matrices per time slice based on attributes and topology of the network [22], and a spectral clustering method based on local similarity derived from shared neighbours [23].

Insight into recent literature revealed the use of graph clustering in a variety of domains, such as the reengineering of software architecture [24], software defect prediction [25], smart city resource management [26], cyber-physical power grids [27], or community detection in social networks [28]. Due to the importance of graph clustering in many scientific fields, and its use in a large variety of human life domains, some recent review articles present more detailed reviews of graph clustering methods, such as a review of novel clustering methods [17], a review of fast clustering methods [18], or a review of attributed graphs in which both nodes and edges have attributes describing them [29].

3. Experience

This paper presents a method for minimizing the number of nodes in a graph by merging neighbouring nodes that possess similar characteristics, and model implementation in a simulation environment for minimizing the number of air pollution measurement stations in urban environment. The model involves comparing a predefined set of attributes among adjacent nodes, and when the similarity between them exceeds a specified threshold, the nodes are assigned to the same cluster.

In this section, a graph clustering method and its implementation in a study aimed at determining the minimal number of air pollution measurement stations in urban environments [30] are described. The focus on the article [30] is on detailed presentation of the the implementation of the method in case of the city of Zrenjanin in Serbia, while the focus of this paper is on a detailed presentation of the graph clustering method, especially on algorithms for checking node similarity and merging similar nodes into clusters.

3.1. A Method for Graph Clustering Based on Attribute Similarity of Neighbouring Nodes

The method constructs a clustered graph by determining similarity between neighbouring vertices (nodes) in an unweighted and undirected graph defined as $G = (V, E)$, where V is a set of M vertices and E is a set of edges.

To simplify comparison and merging of neighbouring nodes, a graph is presented as a set of adjacency lists associated to nodes. Adjacency list for node v is defined as $Adj(v) = \{u \in V | \{v, u\} \in E\}$.

Description of the method assumes setting up the basis of the scenario for presenting and manipulating unweighted and undirected graph in N observations in discrete time moments (observations). These basis includes:

- The graph contains M nodes v_i, where $i = 1, ..., M$.
- Each node v_i is presented with a set of K attributes a_h, where $h = 1, ..., K$. The values of each attribute a_h are in the set $[min_h, max_h]$.

- For each node, N observations in discrete time moments of each attribute are considered, so we have a_{hip}, where $p = 1, ..., N$. The naming convention is that for each attribute, the attribute index is first listed, then the node index, and finally the observation index.

The method execution is divided in two phases. In the first phase, similarity between neighbouring nodes is checked and a similarity matrix S is constructed for the graph. In the second phase, merging of similar nodes into clusters is based on values in the similarity matrix S and node similarity threshold T.

3.1.1. Checking the Similarity of Nodes

The similarity check is based on comparing the corresponding attributes of all nodes (e.g., the $h-$th attributes of nodes are compared with each other). For this similarity check, the similarity threshold of the $h-$th attribute is defined as T_h. The corresponding attributes are similar if their absolute difference is less than T_h.

For each individual observation of the graph at a discrete observation moment p, a threshold of nodes similarity K_T is defined based on the number of similar attributes, where $1 < K_T < K$. Nodes are similar if they have at least K_T similar attributes at the observed moment p.

To observe the similarity of nodes at the level of a selected observation moment p, a notation for the similarity of nodes i and j at the observation moment os_{ijp} (*observation similarity*) is introduced. The total similarity of two nodes after comparing in all observation moments is the value s_{ij} and it is entered into the similarity matrix S of the graph G. Pseudo algorithm for creating the similarity matrix S for graph G is presented in Algorithm 1.

By executing Algorithm 1, an absolute similarity matrix S for the graph G is constructed. Each element s_{ij} of the matrix S belongs to the set $\{0, 1, ..., N\}$. The higher the value of s_{ij}, the more similar the nodes v_i and v_j are.

3.1.2. Merging Similar Nodes into Clusters

Merging similar neighbouring nodes in a cluster is based on checking how values from absolute similarity matrix S relate to the overall similarity threshold T. To simplify notation and further calculations, the following terms are introduced:

- Relative similarity matrix RS with elements $\sigma_{ij} \in [0, 1]$, which are calculated as $\sigma_{ij} = \frac{s_{ij}}{N}$.
- Relative threshold $\tau = \frac{T}{N}$, where $\tau \in [0, 1]$. The greater the similarity of nodes, the closer the τ value is to 1.

Further, for the algorithm of merging similar neighbouring nodes into a cluster based on attribute similarity, it is necessary to introduce the concept of *one-dimensional normalized attributes value* A_i for each node v_i. For this purpose, a maximum value $A_{hi} = \max_p A_{hip}$ is introduced for each attribute a_h for node v_i for all N observations.

Further, normalized maxim value for each attribute a_h for node v_i is defined as

$$norA_{hi} = \frac{A_{hi}}{max_h}.$$

Algorithm 1 Create the similarity matrix S for graph G

Input: G — Initial graph
Input: M — Number of nodes
Input: K — Number of node attributes
Input: T_h — Similarity threshold of the h/th attribute
Input: K_T — Number of similar attributes for two nodes
Input: N — Number of observations
Output: S — Similarity matrix for graph G

```
{Check all nodes in graph G}
for i = 1 to M do
  for j = 1 to M do
    s_ij ← 0
    {Check all N observations}
    for p = 1 to N do
      os_{i,j,p} ← 0
      {Check all K attributes of observed nodes}
      for h = 1 to K do
        if |a_{h,i,p} − a_{h,j,p}| < T_h then
          os_{i,j,p} ← os_{i,j,p} + 1
        end if
      end for
      {Update the value for observed nodes in similarity matrix}
      if os_{i,j,p} ≥ K_T then
        s_ij ← s_ij + 1
      end if
    end for
  end for
end for
```

Finally, *one-dimensional normalized attributes value* is determined as a function $A : V \rightarrow [0, 1]$. For a node v_i it is defined as

$$A(v_i) = \frac{1}{K} \sum_h norA_{hi}.$$

Based on the the values $A(v_i)$ for initial set of nodes V, an ordered list of nodes is created

$$\overline{V} = (\overline{v}_1, \overline{v}_2, ..., \overline{v}_M), A(\overline{v}_i) \geq A(\overline{v}_{i+1})$$

Since the list of nodes $\overline{V}$ contains initial set of nodes but in new order (with decreasing $A(v_i)$ values), new adjacency lists are created to reflect new ordering of nodes. New adjacency lists are $Adj(\overline{v}_i), i = 1, ..., M$.

Created list of nodes $\overline{V}$ and adjacency lists $Adj(\overline{v}_i)$ are used in the algorithm for merging nodes into clusters by starting from the node with the highest $A(v_i)$, which is $\overline{v}_1$. Algorithm for merging similar neighbouring nodes in clusters is presented in Algorithm 2.

For storing information about clustered graph after merging nodes into clusters, an empty *Clustered Graph* set $CG = \{\}$ is created. The elements of the set CG will be calculated as ordered pairs in the form $(C, Adj(C))$, where C denotes a cluster in the clustered graph, while $Adj(C)$ denotes edges from the cluster C to neighbouring clusters.

Algorithm 2 Create clustered graph by merging similar neighbouring nodes

Input: $\overline{V} = \{\overline{v}_1, \overline{v}_2, \ldots, \overline{v}_M\}$, $Adj(\overline{v}_i)$ — List of nodes and their adjacency lists
Input: $RS = [\sigma_{i,j}]$ — Relative similarity matrix
Input: τ — Relative nodes similarity threshold
Output: CG — Clustered graph
$CG \leftarrow \emptyset$
$Active \leftarrow \overline{v}_1$
{*Pass through the list of nodes starting from the node with the highest* $A(v_i)$}
while $NEXT(Active)$ exists **do**
  $C(Active) \leftarrow \{Active\}$
  for each $\overline{v}_j \in \overline{V} \setminus \{Active\}$ **do**
    Remove $Active$ from $Adj(\overline{v}_j)$
  end for
  {*Check a list of adjacent nodes for the active node*}
  for all $u \in Adj(Active)$ **do**
    if $\sigma_{Active,u} \geq \tau$ **then**
      {*Add adjacency list of node* u *to the adjacency list of the active node*}
      $Adj(Active) \leftarrow Adj(Active) \cup Adj(u)$
      {*Add node* u *to the cluster of the active node*}
      $C(Active) \leftarrow C(Active) \cup \{u\}$
      Remove u from $\overline{V}$
      for each $\overline{v}_j \in \overline{V} \setminus \{Active\}$ **do**
        Remove u from $Adj(\overline{v}_j)$
      end for
    end if
  end for
  {*Add the cluster of the active node to the clustered graph*}
  $CG \leftarrow CG \cup \{(C(Active), Adj(Active))\}$
  $temp \leftarrow Active$
  $Active \leftarrow NEXT(Active)$
  Delete $temp$
end while

3.2. Method Implementation

Presented graph clustering method is implemented in a study aimed at determining the minimum number of air pollution measurement stations in urban city environment [30]. The implementation assumes covering the entire urban area of the city with a large number of

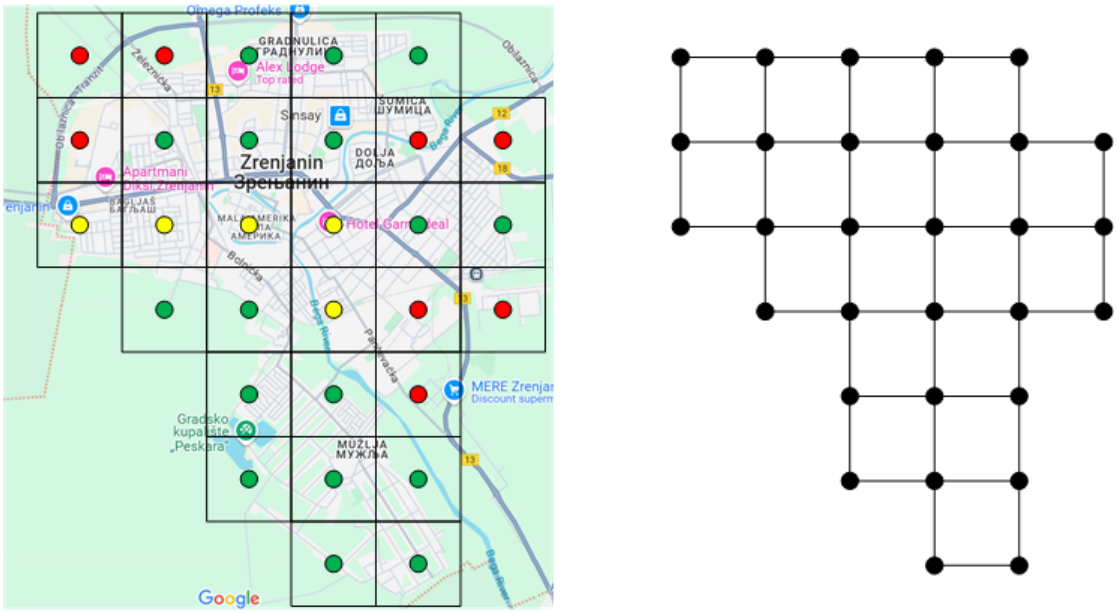

Figure 1: Sample graph with measurement stations for air pollution monitoring in urban settings

densely distributed air pollution measurement stations. The measurement stations are represented as nodes in a graph, each one covering a square region. The edges are constructed to connect neighbouring measurement stations (square regions). A typical scenario for creating a graph with nodes representing air pollution measurement stations placed in the city of Zrenjanin is presented in Figure 1. For the presented scenario measuring stations are 1000 meters apart.

The scenario for distributing measurement stations in the city area assumes considering the zone types in the city. Measurement stations in residential zones are marked with green circles, in commercial zones with yellow circles, and in industrial zones with red circles (see Figure 1). This fact does not affect the graph construction, but it affects the construction of air pollution data sets for specific zones.

The scenario for monitoring air pollution in the city includes the following pollutants: Nitrogen Dioxide (NO_2), Carbon Monoxide (CO), Sulfur Dioxide (SO_2), and Particulate Matter (PM10 and PM2.5). Typical values for residential, commercial and industrial zones in a city, taken from leading international air pollution monitoring organizations (European Environment Agency, World Health Organization) and the World Air Quality Index project are shown in Table 1. These values are used to generate random datasets for the measurement stations in corresponding zones.

3.2.1. Simulation Environment

Method implementation includes the design of a simulation environment for creating scenarios with measurement stations spread over the city area, and data sets for measurement stations based on their locations in different city zones. A simulation environment is created as a

Table 1
Typical values of air pollutants in different city zones used in creating random datasets for measurement stations in the simulation scenarios

	NO_2 ($\mu g/m^3$)	**CO (ppm)**	**SO_2 ($\mu g/m^3$)**	**PM10 ($\mu g/m^3$)**	**PM2.5 ($\mu g/m^3$)**
Range for residential zones	0–40	0–3	0–10	0–50	0–30
Range for commercial zones	20–60	2–4	4–20	30–70	20–50
Range for industrial zones	40–200	4–50	20–200	100–500	40–150
Average for residential zones	20	1.50	5	25	15
Average for commercial zones	40	3	12	50	35
Average for industrial zones	120	27	110	300	95
Max value for all zones	200	50	200	500	150
Min value for all zones	0	0	0	0	0

software solution implemented with Jakarta EE platform technologies [31]. The simulation environment enables the creation of the initial graph scenarios and datasets for measurement stations, and implementation of the presented graph clustering method for determining the minimal number of measurement stations that cover the whole city area. Datasets are created as MS Excel files by using Apache POI library [32], which is typical format for air pollution datasets. Data management and exchange between JEE software components is done by using XML [33]. The architecture of the created simulation environment is presented in Figure 2.

The two main modules of the simulation environments are *Scenario Generator* and *Scenario Optimizer*. The first one is used for generating scenarios for the simulation, while the second one analyses created initial scenario and calculates the minimal number of measurement stations for the city area by implementing the proposed graph clustering method. The *Utility* module contains auxiliary functionalities that enable the implementation of scenario creation and manipulation functionalities (graph manipulation utility, data processing utility, and method utility).

Scenario Generator uses external data, including local area maps, information about a city (spread of residential, commercial, and industrial zones), and typical values for air pollution in urban city areas for creating a scenario.

The measurement stations that remain in the graph actually represent clusters that include neighbouring measurement stations with similar characteristics of measured values. Each cluster is represented by one measuring station that has the most critical measured pollutant values. Each cluster is represented by one measuring station that has the most critical measured pollutant values.

3.2.2. Method Adaptation

Measurement stations distributed in the city area are treated as nodes in the graph. For each node, the attributes are measured values for five pollutants: Nitrogen Dioxide (NO_2), Carbon Monoxide (CO), Sulfur Dioxide (SO_2), and Particulate Matter (PM10 and PM2.5).

A unit measurement at a measuring station i is labelled as m_{hip}. Unit measurement m_{hip} corresponds to node attribute a_{hip} in the presented clustering method. For the unit of measurement

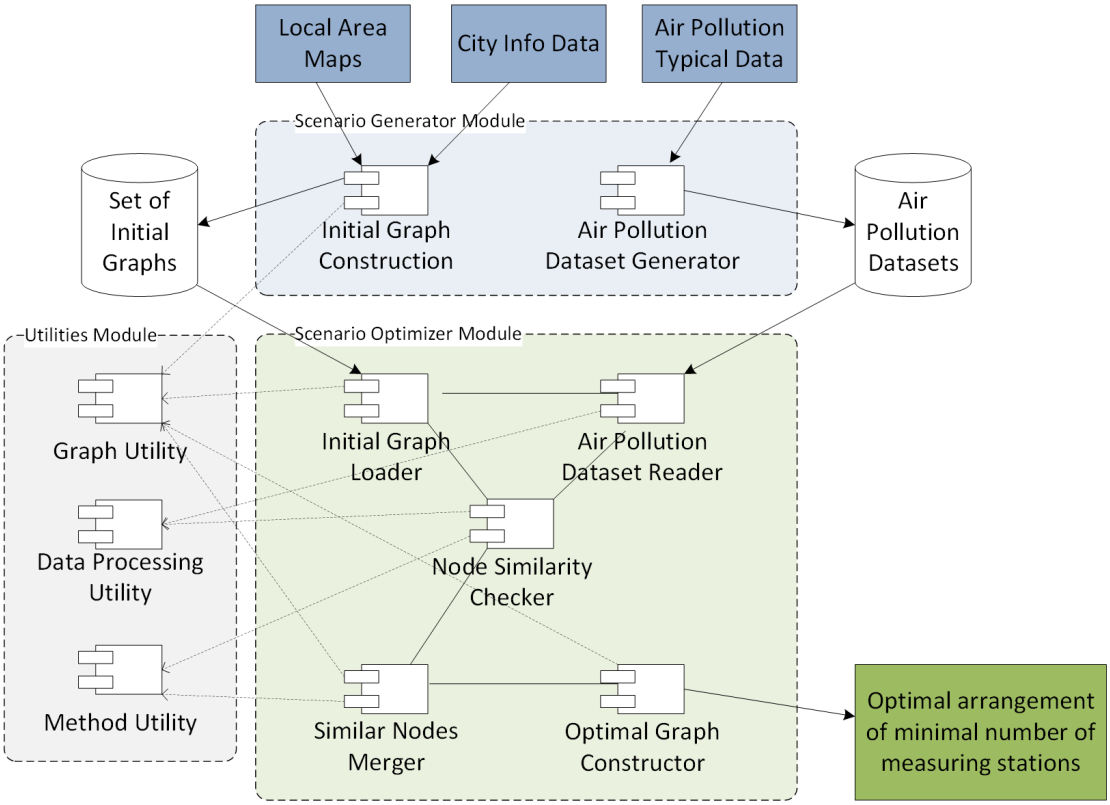

Figure 2: Simulation environment architecture

m_{hip} the indexes are:

- $h = 1, ..., K$ – index representing measurement values for pollutants at each node.
- $i = 1, ..., M$ – index representing the node in the set of nodes in the graph.
- $p = 1, ..., N$ – index representing the ordinal number of a measurement in a set all measurements.

Checking the similarity of two measured values (the same air pollutant at two neighbouring measurement stations), the similarity thresholds T_h are defined for each measured value (node attribute). For each discrete time moment, for all measurement values, the overall similarity threshold K_T is set (how many measured values should be similar on neighbouring nodes), where $1 < K_T \leq K$. The similarity checks for two measured values h of two neighbouring nodes i and j at time p is calculated as

$$|m_{hip} - m_{hjp}| < T_h \times Range_h, \quad \text{where } Range_h = \max_h - \min_h .$$

If at least K_T measured values of two neighbouring measurement stations at a given point p are similar (satisfy the similarity threshold for that value), overall similarity for these two

measurement stations in the similarity matrix S is increased by 1 ($s_{ij} = s_{ij} + 1$). Values T_h, $Range_h$, max_h and min_h are taken from Table 1 for all measured air pollutants.

After checking the similarity between neighbouring nodes for all N measurements (time moments), the similarity matrix S contains similarity measures for all neighbouring nodes in the range from 0 to N. For non-neighboring nodes similarity value is 0. Further, the relative similarity matrix RS with elements σ_{ij} and the relative threshold $\tau = \frac{T}{N}$ are calculated.

The following values were set in the simulation environment to check their influence on the similarity level between neighbouring nodes:

- Distance between measurement stations: $Dist \in \{500m, 1000m\}$
- Total number of measurements at each measurement station (simulation of one measurement per day) $N = 60$.
- $T_h \in \{10\%, 20\%\}$. Threshold T_h is the same for all pollutants to simplify the simulation.
- $K_T \in \{K - 1, K\}$, where $K = 5$
- $\tau \in \{0.8, 0.9, 1.0\}$

3.2.3. Results

Simulation was performed for two global scenarios. The first one has a 500m distance between measurement stations and 92 measurement stations, while the second one has a 1000m distance between measurement stations and 30 dispersed measurement stations. The total number of measurements N in both cases is 60. With proposed variations of parameters T_h, K_T and τ, 12 different sub-scenarios can be identified for both global scenarios, and for each scenario 10 simulations with 10 different datasets were performed.

The construction of a graph with a minimum number of measurement stations by clustering the initial graph is shown in Figure 3. For illustration, an example scenario with 30 measuring stations with the distance of 500m, $T_h = 20$, $K_T = K - 1$, and $\tau = 0.8$ is shown. The resulting minimal graph contains 7 nodes representing the 7 resulting clusters, and represents the scenario with the highest minimization of 77%.

As a result of 10 simulations , the average number for the minimal number of measurement stations is rounded to an integer value. After this initial simulation (the 1st phase minimization), the five control simulations were performed (the 2nd phase minimization) with five created minimal graphs and five new datasets for air pollutants. The results of minimizing the number of measuring stations in percentages for scenarios with 1000m and 500m distances between measuring stations are shown in Tables 2 and 3 respectively. For labelling different scenarios in Tables the following format is used: $(Dist, T_h, K_T, \tau)$.

The results obtained from the simulation indicate the following:

- Greater minimization of the number of measurement stations is obtained when the initial number of stations is 30, i.e., when the initial distance between the measurement stations is larger. This means that increasing the initial number of measurement stations will not lead to a more efficient minimization of the number of stations.
- Simulation results after the control simulations for all scenarios are, in the majority of cases, the same or slightly different than the results after the initial simulation (insignificant minimization of the graph). These results confirm the correctness and efficiency of the presented graph clustering method.

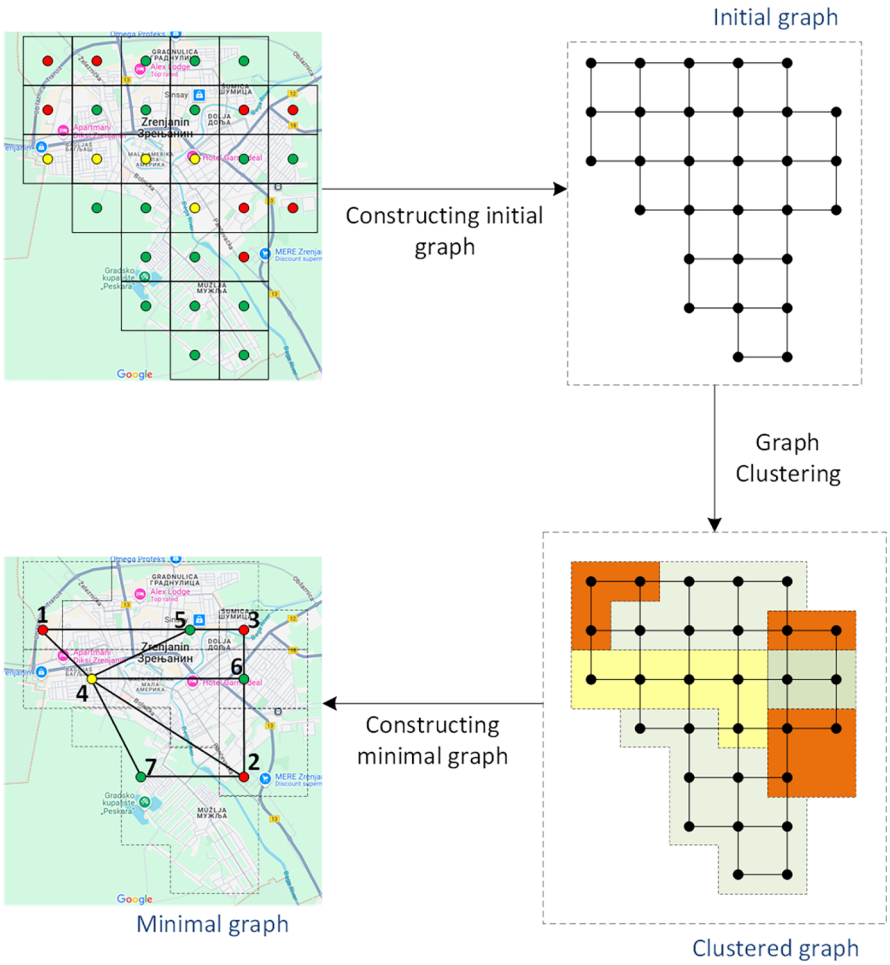

Figure 3: Creating a graph with a minimum number of measurement stations by clustering the initial graph

3.3. Discussion

The main advantages of the presented graph clustering method and presented simulation environment are their adaptability and extensibility.

Method adaptability is reflected in the selection of simulation parameters in the method, such as thresholds for determining the similarity of neighbouring nodes (attribute-related thresholds and node-related thresholds), the number of attributes per node, and the number of time points for observing nodes. In addition, different application scenarios may require adaptation of the similarity checking method at the level of individual or all node attributes, but also at the level of the nodes themselves (e.g., if we consider different social networks, professional or expert networks, distributed algorithms for complex engineering or scientific calculations, urban traffic and logistic networks).

Method extensibility is reflected in the addition of new variables that can affect nodes' attributes

Table 2
Minimization in percentage of initial graphs with measurement stations distance of 1000m

Scenario	1st phase Minimization [%]	2nd phase Minimization [%]
S(1000,10,K-1,0.8)	70	70
S(1000,10,K-1,0.9)	53	57
S(1000,10,K-1,1.0)	3	3
S(1000,10,K,0.8)	60	60
S(1000,10,K,0.9)	43	43
S(1000,10,K,1.0)	0	0
S(1000,20,K-1,0.8)	77	77
S(1000,20,K-1,0.9)	63	63
S(1000,20,K-1,1.0)	10	10
S(1000,20,K,0.8)	53	57
S(1000,20,K,0.9)	47	47
S(1000,20,K,1.0)	3	3

Table 3
Minimization in percentage of initial graphs with measurement stations distance of 500m

Scenario	1st phase Minimization [%]	2nd phase Minimization [%]
S(500,10,K-1,0.8)	66	67
S(500,10,K-1,0.9)	48	48
S(500,10,K-1,1.0)	2	2
S(500,10,K,0.8)	51	51
S(500,10,K,0.9)	34	34
S(500,10,K,1.0)	0	0
S(500,20,K-1,0.8)	68	70
S(500,20,K-1,0.9)	55	55
S(500,20,K-1,1.0)	8	8
S(500,20,K,0.8)	54	54
S(500,20,K,0.9)	38	38
S(500,20,K,1.0)	4	4

in a way that nodes mutually affect each other, leading to the use of directed and weighted graphs (for example, modelling weather conditions in the presented implementation of the clustering method, where wind direction may influence air pollutants in different city zones).

Simulation environment extensibility relates to enabling the support for importing datasets from other sources and in other formats such as XML or JSON, which are commonly used in distributed software systems. This would enable testing the presented method with different datasets and potentially improving its characteristics.

And finally, the use of simulation environments for testing and fine-tuning specific methods is cost and time-effective, especially in the case of their implementation in complex distributed environments. In the presented implementation scenario, it is easier and cheaper to simulate and test measurements of air pollutants in the simulation environment than to install and maintain a set of measurement stations over a long time period to achieve the optimal or minimal number of measurement stations.

4. Conclusions

This paper presents a novel method for graph clustering based on node attribute similarity, together with a simulation environment for validating the method. In addition, method adaptation and implementation for determining the minimal number of air pollution measurement stations in urban settings is presented. Despite the stated advantages and benefits of the presented method and simulation environment, several future research directions are interesting for additional validation and improvements of the presented method.

The first research direction relates to comparing the proposed clustering method with other graph clustering methods on the same datasets. These comparisons will enable positioning this method in the area of graph clustering methods and help in finding potential method improvements.

The second research direction relates to developing more specific similarity measures based on the probability of potential states of nodes and their attributes over a long period, the semantics of the data stored in node attributes, or the correlation among data in node attributes.

And finally, the development and improvement of the simulation environment, and its evolution to a real system used in specific socio-technical systems (social networks, recommendation systems, logistics and transportation, natural eco systems), is an attractive direction, leading to empirical testing of the correctness and usability of the proposed method.

References

[1] G. H. Walker, N. A. Stanton, P. M. Salmon, D. P. Jenkins, A review of sociotechnical systems theory: a classic concept for new command and control paradigms, Theoretical Issues in Ergonomics Science 9 (2008) 479–499. doi:10.1080/14639220701635470.

[2] P. N. Edwards, S. J. Jackson, G. C. Bowker, C. P. Knobel, Understanding infrastructure: Dynamics, tensions, and design. Report of a Workshop on "History & Theory of Infrastructure: Lessons for New Scientific Cyberinfrastructures", resreport, Ann Arbor, MI, US, 2007.

[3] G. Baxter, I. Sommerville, Socio-technical systems: From design methods to systems engineering, Interacting with Computers 23 (2011) 4–17. doi:10.1016/j.intcom.2010.07.003.

[4] A. Barrat, M. Barthelemy, A. Vespignani, Dynamical Processes on Complex Networks, Cambridge University Press, Cambridge, UK, 2008.

[5] W. M. Fox, Sociotechnical system principles and guidelines: Past and present, The Journal of Applied Behavioral Science 31 (1995) 91–105. doi:10.1177/0021886395311009.

[6] Z. Stojanov, J. Stojanov, G. Jotanovic, D. Dobrilovic, Weighted networks in socio-technical systems: concepts and challenges, in: I. Bychkov, A. Tchernykh, A. Feoktistov (Eds.), Proceedings of the 2nd International Workshop on Information, Computation, and Control Systems for Distributed Environments (ICCS-DE 2020), volume 2638, CEUR Workshop Proceedings, Irkutsk, Russia, 2020, pp. 265–276.

[7] J. L. Gross, J. Yellen (Eds.), Handbook of graph theory, Discrete mathematics and its applications, CRC Press, Taylor & Francis Group, Boca Raton, FL, USA, 2003.

[8] K. K. Meng, D. Fengming, , T. E. Gum, Introduction to graph theory, World Scientific Publishing, Singapore, 2007.
[9] A. S. da Mata, Complex networks: a mini-review, Brazilian Journal of Physics 50 (2020) 658–672. doi:10.1007/s13538-020-00772-9.
[10] F. Hu, A. Mostashari, J. Xie (Eds.), Socio-Technical Networks: Science and Engineering Design, CRC Press, Taylor & Francis Group, Boca Raton, FL, USA, 2011.
[11] S. E. Schaeffer, Graph clustering, Computer Science Review 1 (2007) 27–64. doi:10.1016/j.cosrev.2007.05.001.
[12] A. Dasgupta, J. Leskovec, K. J. Lang, M. W. Mahoney, Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters, Internet Mathematics 6 (2009) 29–123. doi:10.1080/15427951.2009.10129177.
[13] M. Newman, Networks: An Introduction, Oxford University Press, New York, USA, 2010.
[14] Z. Yang, R. Algesheimer, C. J. Tessone, A comparative analysis of community detection algorithms on artificial networks, Scientific Reports 6 (2016) 30750. doi:10.1038/srep30750.
[15] T. Chakraborty, A. Dalmia, A. Mukherjee, N. Ganguly, Metrics for community analysis: A survey, ACM Computing Surveys 50 (2017). doi:10.1145/3091106.
[16] J. Kleinberg, E. Tardos, Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields, Journal of the ACM 49 (2002) 616–639. doi:10.1145/585265.585268.
[17] M. Hloch, M. Kubek, H. Unger, Graph-based Clustering Algorithms – A Review on Novel Approaches, Advances in Science, Technology and Engineering Systems Journal 6 (2021) 21–28. doi:10.25046/aj060403.
[18] J. Xue, L. Xing, Y. Wang, X. Fan, L. Kong, Q. Zhang, F. Nie, X. Li, A comprehensive survey of fast graph clustering, Vicinagearth 1 (2024) 7. doi:10.1007/s44336-024-00008-3.
[19] V. Estivill-Castro, Why so many clustering algorithms: a position paper, ACM SIGKDD Explorations Newsletter 4 (2002) 65–75. doi:10.1145/568574.568575.
[20] E. Hartuv, R. Shamir, A clustering algorithm based on graph connectivity, Information Processing Letters 76 (2000) 175–181. doi:10.1016/S0020-0190(00)00142-3.
[21] J. Wan, K. Zhang, Z. Guo, D. Miao, A new clustering algorithm based on connectivity, Applied Intelligence 53 (2023) 20272–20292. doi:10.1007/s10489-023-04543-2.
[22] X. Qin, W. Dai, P. Jiao, W. Wang, N. Yuan, A multi-similarity spectral clustering method for community detection in dynamic networks, Scientific Reports 6 (2016) 31454. doi:10.1038/srep31454.
[23] Z. Cao, H. Chen, X. Wang, Spectral clustering based on the local similarity measure of shared neighbors, ETRI Journal 44 (2022) 769–779. doi:10.4218/etrij.2021-0230.
[24] U. Desai, S. Bandyopadhyay, S. Tamilselvam, Graph neural network to dilute outliers for refactoring monolith application, Proceedings of the AAAI Conference on Artificial Intelligence 35 (2021) 72–80. doi:10.1609/aaai.v35i1.16079.
[25] X. Wang, L. Lu, Q. Tian, H. Lin, Ic-graf: An improved clustering with graph-embedding-based features for software defect prediction, IET Software 2024 (2024) 8027037. doi:10.1049/2024/8027037.
[26] A. A. Kapanski, R. V. Klyuev, A. E. Boltrushevich, S. N. Sorokova, E. A. Efremenkov, A. Y. Demin, N. V. Martyushev, Geospatial clustering in smart city resource management: An initial step in the optimisation of complex technical supply systems, Smart Cities 8 (2025).

doi:10.3390/smartcities8010014.

[27] N. Jacobs, S. Hossain-McKenzie, S. Sun, E. Payne, A. Summers, L. Al-Homoud, A. Layton, K. Davis, C. Goes, Leveraging graph clustering techniques for cyber-physical system analysis to enhance disturbance characterisation, IET Cyber-Physical Systems: Theory & Applications 9 (2024) 392–406. doi:https://doi.org/10.1049/cps2.12087.

[28] K. Georgiou, C. Makris, G. Pispirigos, A distributed hybrid community detection methodology for social networks, Algorithms 12 (2019). doi:10.3390/a12080175.

[29] C. Bothorel, J. D. Cruz, M. Magnani, B. Micenkova, Clustering attributed graphs: Models, measures and methods, Network Science 3 (2015) 408–444. doi:10.1017/nws.2015.9.

[30] Z. Stojanov, V. Brtka, G. Jotanovic, G. Jausevac, D. Perakovic, D. Dobrilovic, Graph clustering-based model for optimizing the number of air pollution measurement stations in urban environments, Mobile Networks and Applications (2025). doi:10.1007/s11036-025-02455-8.

[31] Eclipse Foundation AISBL, Jakarta EE | Cloud Native Enterprise Java | Java EE | The Eclipse Foundation, https://jakarta.ee/, 2025. Last accessed: 2025-05-28.

[32] The Apache Software Foundation, Apache POI™ – the Java API for Microsoft Documents, https://poi.apache.org/, 2025. Last accessed: 2025-05-28.

[33] Eclipse Foundation AISBL, Jakarta XML Web Services 4.0 | Jakarta EE | The Eclipse Foundation, https://jakarta.ee/specifications/xml-web-services/4.0/, 2025. Last accessed: 2025-05-28.