Received February 14, 2019, accepted March 10, 2019, date of publication March 25, 2019, date of current version April 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2907161

Multi-Hub Location Heuristic for Alert Routing

MAREK ŠIMON, DIRGOVÁ LUPTÁKOVÁ IVETA, LADISLAV HURAJ, AND JIŘÍ POSPÍCHAL
Department of Applied Informatics and Mathematics, University of Ss. Cyril and Methodius in Trnava, 917 01 Trnava, Slovakia

Corresponding author: Ladislav Huraj (ladislav.huraj@ucm.sk)

This work was supported in part by the Optimization of Network Security by Computational Intelligence under Grant VEGA 1/0145/18, and in part by the Algorithm of Collective Intelligence: Interdisciplinary Study of Swarming Behavior in Bats under Grant APVV-17-0116.

ABSTRACT Trigger warning and other delay-sensitive applications for environmental, healthcare, and industrial or military surveillance and monitoring are usually based on networks. Surveillance nodes (like in the Internet of Things or wireless sensor networks) send the data to selected nodes (hubs) that forward the alert or alarm further to the next system level. The maximum length of the shortest paths from the nodes to their nearest hub should be optimized to minimize the maximum necessary number of hops required to route a warning. For k hubs, this requirement is expressed as k-source minimization of the maximum vertex eccentricity problem, i.e. minimization of d in a d-hop dominating set of a given cardinality k. In this contribution, several heuristic algorithms selecting the initial set H of hubs (i.e. the dominating set) are combined with our greedy and k-means-like approaches adapted from Euclidean to graph (i.e. geodesic) distance. The presented algorithms were tested on static geometric network models, standardly used for models of Wi-Fi networks. The best results were produced by the k-means-like algorithm with initialization provided by the centers of communities found by edge-betweenness community detection. When compared with, e.g., random selection of hub locations, our method can cut the worst case number of hops by half, and when compared with a classical k center approach, our improvement was more than 50%.

INDEX TERMS Clustering methods, communication networks, heuristic algorithms, optimization.

I. INTRODUCTION

Environmental and health monitoring, military surveillance and process control in the industry are all based on networks. Typically, these networks consist of an array of cheaper sensor nodes or other detection devices, which are not connected to a decision center directly but through intermediates or hubs. In most cases, the sensor nodes send their data to a hub through other sensor nodes, without being connected to the closest hubs directly. The same is true for wireless sensor networks or other wireless communication networks, such as cellular systems and/or mobile ad-hoc networks. As a part of risk analysis and reduction [17], alerts for critical situations may include a gas leak or fire, both at home or in a mine, as well as pollution or traffic alarms for accidents, medical emergencies, or incoming missiles in military applications. At the same time, since such events are rare both in time and in the number of nodes involved, such cases implicate dense networks and sparse communications. An emergency or high priority traffic is typically caused by threshold detection, where the measured data signifies an event triggering an alarm. In such a case, an optimal system should deliver the alert to the next level as soon as possible, including from the most eccentric nodes. The warning sent by the shortest route to the nearest hub should go through the minimum number of hops, even for the nodes farthest from the nearest hubs.

Selection of the set of nodes which shall serve as hubs in the network is therefore crucial to minimize the warning lag time. The current approach assumes knowledge of the whole graph/network.

A similar problem was studied by Amis *et al.* [2] and later by others [7], [10]. However, their heuristics tried to select the minimal set of hubs, when the maximum number of hops was specified. The problem studied in our paper is somewhat different, as the number of hubs is prescribed and the heuristics should select them to get the smallest maximum number of hops. Both problems are NP complete, even though for trees and interval graphs there exist linear time algorithms [5], [27].

Achieving the minimum of the average number of hops to the nearest hub was a typical task for IoT networks [21]. A similar technique was used to minimize the maximum latency between particular centroid and its nodes for multi-controllers in a Software Defined Network [25], but in this

The associate editor coordinating the review of this manuscript and approving it for publication was Jun Huang.

work the authors also minimized the sum of the shortest path distance from all nodes in the cluster to the hub.

An approach related to k-means clustering, but applied for community detection in networks, was used by [8], but they employed the ratio of the connecting edges and the size of the cluster to decide to which hub/cluster a node should be attached. This is different from the k-means-like clustering approach in this paper where a node is attached to the nearest hub (with the shortest length of the path to the hub). Even direct k-means clustering employing Euclidean distance has been used as a basis for cluster head selection [26]. Diverse applications of similar approaches include for example EEG signal analysis [14], [22].

Among other techniques to initialize the positions of hubs before the first iterations, the present paper employs the edge betweenness clustering. Although sink betweenness centrality was used in [37], their algorithm was concerned with optimization of other criteria and constraints.

In graph theory, a requirement to optimize the worst case among shortest paths to the closest hubs is expressed as k-source minimization of the maximum vertex eccentricity problem, i.e. minimization of d in a d hop dominating set of a given cardinality k. Although even the distributed algorithms exist to find k-hop dominating sets, the sets have to be connected, or the algorithm uses a number of additional technology related information [38].

Practically the same problem is referred to in graph theory as the k center problem, which has a long history of heuristic approaches [18], [20]. However, unlike the problem studied here, the k center problem is typically studied for graphs with weighted edges (i.e. the length of the path is computed as the sum of weights of its edges). The cutting edge approaches use stochastic metaheuristics like tabu search or simulated annealing [13], thus paying too much CPU time for slight improvements of a solution.

Selection of cluster heads, aiming for similar results as k-hop dominating sets, can be solved also by nature based metaheuristics like ant colony optimization [1], particle swarm optimization [36], firefly optimization [23], multi-objective evolutionary based algorithm [31], differential evolution [35], or compressive sensing [11]. However, proper evolutionary machine learning requires a number of trials, which is larger than in the typical heuristics in the orders of the magnitude.

A problem related to the one present in this paper is also the Influence maximization problem (IMP), where the task is to select k nodes in a network that can spread information further towards the maximum of the remaining nodes [28]. The difference is that, unlike in our problem, the nodes typically represent people in a social network and the probability that everyone will send the information further is not 100 per cent for IMP.

Many other approaches optimizing the ad-hoc or IoT networks exist [6], [9], [21], [33], [34], but they are mostly concerned with multicriteria optimization satisfying many constraints. A typical requirement is optimization of energy consumption and an ability to adapt to dynamic changes of the network topology over time. Another typical optimization goal is to provide k-coverage, where every location is covered by at least k different sensor nodes. Such optimization requires changing the node coordinates, while our approach requires selection of a subset of nodes, when their coordinates are already given. Routing is also often optimized, but in our case, the knowledge of shortest routes from a node to a nearest hub is already taken for granted, not assuming sudden dynamical changes. The approach in this paper also does not contain any technology-related constraints.

Similarly to models used by Amis and many others [2], our network is modeled as a graph $G = (V, E)$ and the positions of all nodes in the two-dimensional plane are given. Two nodes are always connected by an edge when their Euclidean distance is less or equal to a given transmission radius r. This is called unit disk graph or geometric network; it is an established family of graphs typically representing networks commonly used for modeling ad-hoc wireless networks [15], [19], [24].

In this paper, we describe new and adapted heuristics for the initial selection of a set of hubs among all nodes and merge them in a combination with a greedy and k-means-like heuristic. Our aim is to minimize the longest of a minimum length path from a node to its nearest hub. We then investigate the effectiveness of this new approach on generated models of networks. Histograms of worst paths lengths after a given number of iterations provide the results of the tests and are compared for all the combinations of initializations and heuristics, together with graphs showing their convergence in time. Our paper thus addresses the improvement of the worst-case scenario of the warning lag time.

II. HEURISTICS

A. ECCENTRICITY BASED MULTI-HOP CLUSTERING

Taking into consideration the goal of optimization, stated in the introduction, the objective function defined in details includes assumption that the network (graph) is composed of a set of nodes, hereafter denoted as $V = \{v_1, v_2, \ldots, v_n\}$, and all of the nodes are capable of taking the role of a hub. With the proposed mechanism, each node v_i transfers its data to its selected hub instead of transmitting the data directly to an Internet server so that the number of Internet connections required for the network equals the given number of hubs. The objective of the proposed algorithms is to find the set $H \subseteq V$ with cardinality $|H| = k$ of the selected hubs h_i, $i = 1, 2, \ldots, k$, minimizing the maximum hop count between a node v_j and its closest (in hop counts, i.e. graph/geodetic distance) hub for all nodes. When dist(a,b) equals a hop count in the shortest path between nodes a and b, i.e. graph distance of those nodes, we define member nodes $\forall v_j \in M_i \subseteq V$ that belong to a hub h_i as those, whose graph distance to any other hub is longer or equal to the graph distance d(v_j, h_i) to the hub h_i.

$$M_i = \left\{\forall v_j \in V \mid d\left(v_j, h_i\right) \leq d\left(v_j, h_m\right), \forall h_m \in H\right\} \quad (1)$$

for which holds $\bigcup_{i=1}^{k} M_i = V$, but a node v_j can be a member node of two different hubs h_m and h_n, if $d\left(v_j, h_m\right) = d\left(v_j, h_n\right), M_m \cap M_n \neq \emptyset$.

The objective function for a network $G(V,E)$ with the selected hub set $H \subseteq V$ to be minimized is formulated as follows:

$$d_{max}(G, H) = \min_{\forall H \subseteq V, |H|=k} \max_{\forall i, 1 \leq i \leq k, \forall v_j \in M_i} d\left(v_j, h_i\right) \quad (2)$$

The member nodes $\forall v \in M_i$ that belong to a hub h_i can be considered as a cluster. However, since nodes on the outskirts of the cluster can have equal geodetic distance to two or more clusters, the defined clusters are not necessarily disjunctive.

B. INITIALIZATION METHODS OF THE SET OF HUBS

An iterative optimization procedure usually requires an initial starting point in the search space. In order to start with a reasonable initial selection of hubs from the set of nodes V, our inspiration was taken from the initialization of the classical k-means clustering algorithm.

One of the simplest methods used is choosing the centers, i.e. hubs, randomly from the data points [29]. There is always a danger of choosing outliers or points that are too close to each other, but this is usually rectified by multiple runs. In the approach adapted for networks here, the hubs are chosen uniformly at random from the set of nodes *V*. In further description, this initialization method shall be denoted as *Random*.

In Random Partitions [16], each point is assigned to one of the k clusters uniformly at random. Since centroids computed as the initial means tend to be placed near the center of the dataset, in this paper this method was emulated by calculating eccentricity (the maximum graph distance between the selected vertex and any other vertex) of all vertices. Then the first k vertices with the smallest eccentricity were selected as hubs. In further description, this initialization method is denoted as *Centers*.

The k-means++ method [3] chooses the first center randomly from data points and the i-th ($i \in \{2, 3, \ldots, k\}$) center x' is chosen at random using a weighted probability distribution where a point x' is chosen with probability proportional to $D(x')^2$. The $D(x')$ denotes the distance from a point x' to the nearest of previously selected centers. The method in this paper is only adapted by using a geodetic (i.e. graph) distance denoting the length of the path in the number of edges instead of the Euclidean distance used in the standard k-means++ method. This initialization method is denoted as *k-means++ like*.

Also, the community detection algorithms [30] were used for the initial selection of hubs. First, the communities were found, and then in each community the node with the smallest eccentricity within the selected community was selected as a hub. Standard parameterization of the methods was used, as in G. Csardi and T. Nepusz in the igraph package of the R language. All popular community detection methods were considered, but only the hierarchical ones enable a user to enter the number of communities as an input parameter. This requirement has ruled out Louvain, label propagation and spinglass methods. Implementation of the Leading eigenvector method was rejected due to numerical difficulties (could not handle too few required communities) and the walktrap method was excluded due to excessive differences between the sizes of the communities it has found. This left only *edge betweenness* and *fast greedy* approaches, which were used for the hub set initialization.

C. GREEDY MULTI-HOP ECCENTRICITY OPTIMIZATION HEURISTIC

The greedy approach to optimization of the objective function (2) is presented in the Algorithm 1. Given the initial set of hubs, the Algorithm 1 finds the hub with the most distant member node (i.e. no other hub is closer to this node). Then the algorithm moves the hub one step on the path towards this distant node, and if the maximum distance given by the equation (2) did not increase, the new hub position is accepted, otherwise the hub returns to its previous position. These moves of the hubs are iteratively repeated a given number of times.

Algorithm 1 Greedy Multi-hop Eccentricity Optimization

Input: $G(V,E)$ (network determined by a set of V nodes and E edges)
k (required number of hubs to be selected)
MaxIters (limit of iterations)

Output: $H = h_1, h_2, \ldots, h_k$ (set of k hubs as a subset of V)

step 1: Select initial set $H = \{h_1, h_2, \ldots, h_k\}$ of k hubs among nodes by a chosen method; $iter \leftarrow 1$.

step 2: Distribute all the vertices $v(v \in V)$ to one of the k clusters using $v \in cluster_i$, *if* $d(v,h_i) \leq d(v,h_j)$, $\forall j \in \{1,2,\ldots,k\}$
where $d(u, v)$ represents the shortest path between nodes u and v.

step 3: Make a copy of hub set $H' \leftarrow H$
among all hubs and their clusters find a vertex v with the maximum distance from its hub, $\forall i,j \in \{1,2,\cdots,k\}$ and $\forall v \in cluster_i, \forall u \in cluster_j$, $d(v,h_i) \geq d(u,h_j)$
find the second node h_i* on a shortest path from h_i to v and update the hub set $H' = \{h'_1, h'_2, \ldots, h'_k\}$ by replacing $h'_i \leftarrow h^*_i$

step 4: Distribute all the vertices $v(v \in V)$ to one of the k clusters H' using $v \in cluster_i$ ', *if* $d(v, h^*_i) \leq d(v, h^*_j)$, $\forall j \in \{1,2,\ldots,k\}$

step 5: find $d_{max}(G, H')$
If $d_{max}(G, H') \leq d_{max}(G, H)$ then replace the hub set by a new one $H \leftarrow H'$ together with corresponding clusters;
$iter \leftarrow iter+1$

step 6: Repeat steps 2-5 while $iter \leq MaxIters$

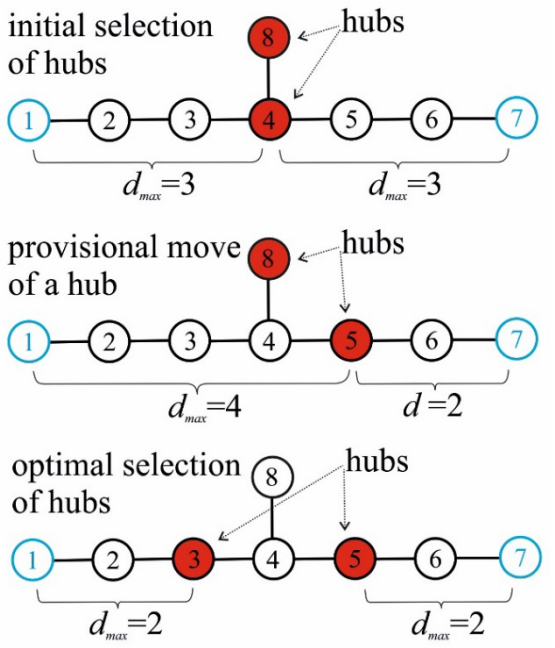

FIGURE 1. Illustration of failure of Algorithm 1, which unsuccessfully tries to move one of the hubs from the initial selection of hubs. The move would increase d_{max}, so the algorithm cannot achieve the optimum selection of the hubs shown at the bottom diagram.

One can stop the algorithm, when no improvement is achieved, but usually, there are several hubs in the same maximum distance from some nodes in their clusters. Moreover, one hub can move towards one of several equal distant member nodes and even when both the hub and the node are selected, this couple can be connected by several shortest paths. A random selection between the mentioned choices may bypass a local barrier in the objective function. This is also the reason, why one cannot assure convergence. The algorithm can be improved by avoiding the already tested solutions, e.g. by employing some equivalent of a tabu search, but it was not studied here, since the most likely improvement should come from repeated starts with different initial hub selection and with permutation of node indices.

An illustration of a potential failure of Algorithm 1 is shown in Figure 1. The upper part shows a simplified network with 8 nodes, from which nodes 4 and 8 were selected as the initial positions of its two hubs and marked by red discs. The d_{max} of this selection is 3, because to send a message from the hub 4 either to node 1 or 7 requires traversing three edges. Algorithm 1 then randomly selects node 7 as one of the most distant from any hub, and tentatively tries to move the nearest hub from node 4 one step towards node 7, which moves the hub from node 4 to node 5. This is shown in the middle of Figure 1. However, the d_{max} of the new selection is 4, because a message from the hub at node 5 has to traverse four edges before arriving at node 1. As the d_{max} increases, the move is considered unsuccessful and the hubs return to the previous positions. The optimal selection of hubs shown at the bottom of the Figure 1 cannot be achieved by Algorithm 1 from the current initial selection. Prevention of d_{max} increase would require a concurrent move of both hubs, which Algorithm 1 does not allow. While repeated runs with different initial selections can avoid being stuck in this particular local optimum, other similar obstacles can easily prevent Algorithm 1 from achieving the global optimum.

D. K-MEANS-LIKE MULTI-HOP ECCENTRICITY OPTIMIZATION HEURISTIC

Since most greedy algorithms tend to get stuck in local optima when applied to NP-complete problems, another approach was designed, based on a popular k-means clustering algorithm. However, classical k-means clustering can be applied only to data points, not to nodes or vertices of networks or graphs. Nevertheless, while data points in k-means clustering are attached to the closest centroid, the data points can be swapped in the algorithm with nodes and the Euclidean distance of data points towards the centroid can be replaced by the graph (geodesic) distance of nodes from the hubs. The major difference is that the probability of having more data points within the same distance from their closest centroids is so negligible that it is typically ignored, while in graphs or networks the corresponding occurrence is quite likely.

Algorithm 2 k-Means-Like Multi-hop Eccentricity optimization

Input: $G(V, E)$ (network determined by a set of V nodes and E edges)
k (required number of hubs to be selected)
MaxIters (limit of iterations)
Output: $H = \{h_1, h_2, \ldots, h_k\}$ (set of k hubs as a subset of V)
step 1: Select initial set $H = \{h_1, h_2, \ldots, h_k\}$ of k hubs among nodes by a chosen method; iter $\leftarrow$ 1.
step 2: Distribute all the vertices $v(v \in V)$ to their cluster(s) from H using $v \in cluster_i$, *if* $d\,(v,h_i) \leq d\,(v,h_j)$,$\forall j \in \{1,2, \ldots, k\}$
where $d\,(u, v)$ represents the shortest path between node u and v.
step 3: For each $cluster_i, \forall i \in \{1, 2, \ldots, k\}$find a vertex v_i with the minimum eccentricity within the subgraph defined by the cluster vertices, which differs from hubs $h_1, h_2, \ldots, h_{i-1}$ already selected in this step and update the hub set $H = \{h_1, h_2, \ldots, h_k\}$ by replacing $h_i \leftarrow v_i$
step 4: $iter \leftarrow iter+1$; $H \leftarrow H'$
step 5: Repeat steps 2-4 while $iter \leq MaxIters$

This algorithm, comparably to the original k-means clustering is also a greedy approach, which does not guarantee achieving an optimum. Moreover, it includes randomness, which was not an automatic part of the original k-means clustering. If several nodes have the same minimal eccentricity within the current cluster, the new hub is selected among them uniformly at random. This element of randomness prevents

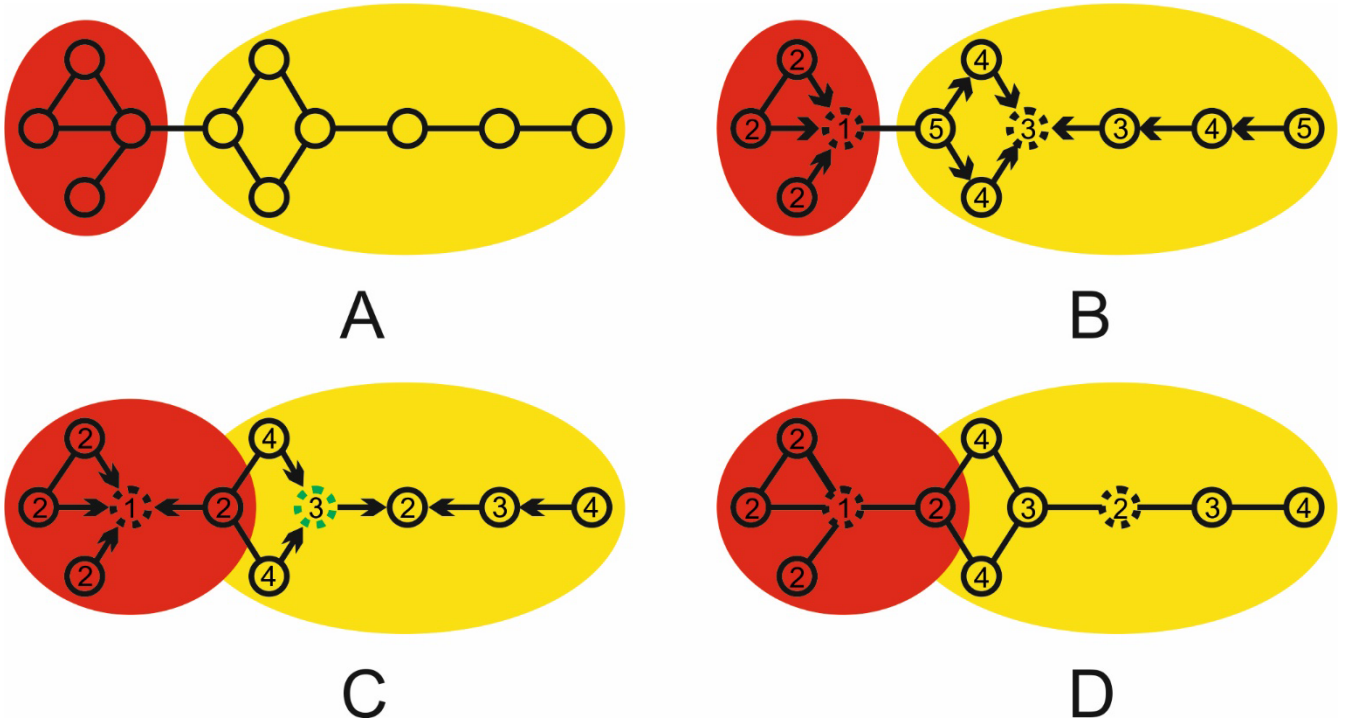

FIGURE 2. Illustration of the basic principles of the Algorithm 2. The network in (A) is divided into two parts by community detection algorithm. The circles in (B) contain eccentricities of the nodes for each subgraph apart, dotted circles correspond to the selected hubs with minimal eccentricities. The network in (C) is divided anew; each node belongs to the closest hub (with shortest path length). The new division of the network corresponds to new eccentricity values, and so in (D) are newly selected hubs with minimum eccentricities, marked by dotted circles.

automatic convergence; when no hub was replaced by a new one, the algorithm should not stop, as it would in classical k-means clustering. In the new algorithm, one must resort to stopping the algorithm by setting the maximum number of iterations.

Illustration of the principles of the Algorithm 2 with the initial set of hubs determined using edge betweenness community detection on a simple network is shown in Figure 2.

If we prescribe $k = 2$, it means that the algorithm should select two hubs in the network. First comes hierarchical community detection algorithm, in our case edge betweenness, with prescribed number of communities equal two. The result is shown in the diagram A of the Figure 2, where the network is divided into two parts, visualized by red and yellow backgrounds. The subgraphs corresponding to these parts or communities are then treated as separate graphs, and for each of their nodes, its eccentricity (path length to the farthest node in the subgraph) is found. This is shown by numbers within the node circles in the diagram B in Figure 2. Separately for both subgraphs, the arrows show directions of edges from each node towards the selected node with the minimum eccentricity. The nodes with the minimum eccentricity are marked by dotted circles. The first such node in the red background subgraph has an eccentricity 1, which means that it is connected directly by an edge with all the three remaining vertices of the subgraph. The second such a node in the yellow background subgraph has eccentricity 3. One has to travel though three edges to get to the rightmost node. There are actually two such nodes with the eccentricity three, and the leftmost is chosen randomly. These marked nodes compose the initial set H of k hubs from the step 1 in Algorithm 2.

Next step is to distribute all the vertices towards their clusters, in which the centers are the previously determined hubs. This new distribution of nodes into clusters marked by the red and yellow background would be the result of the step 2 of the Algorithm 2. It is shown in the diagram C of Figure 2, where the rightmost node of subgraph with the red background belongs in diagrams A and B to the yellow background subgraph. Since this node ''shifted its allegiance'', the eccentricities in the diagram C are slightly different from the diagram B. For example, the node, which shifted its allegiance, had originally the eccentricity 5 in the diagram B, since it took five-edge long path to get to the rightmost node. Now this node in diagram C belongs to the red subgraph and its eccentricity changed to two, since it takes two edges to travel to the leftmost ''red'' node. However, its graph distance towards the red hub is only one edge, while its graph distance towards the yellow hub is two.

While the ''red'' hub in the diagram C of Figure 2 retains the minimum eccentricity, it is not true for the ''yellow'' hub, marked by the green dotted circle. It has still an eccentricity value three, but its neighbor to the right has the minimum eccentricity value two. Therefore, the ''yellow'' hub must be chosen anew, with the minimum eccentricity two, as shown in

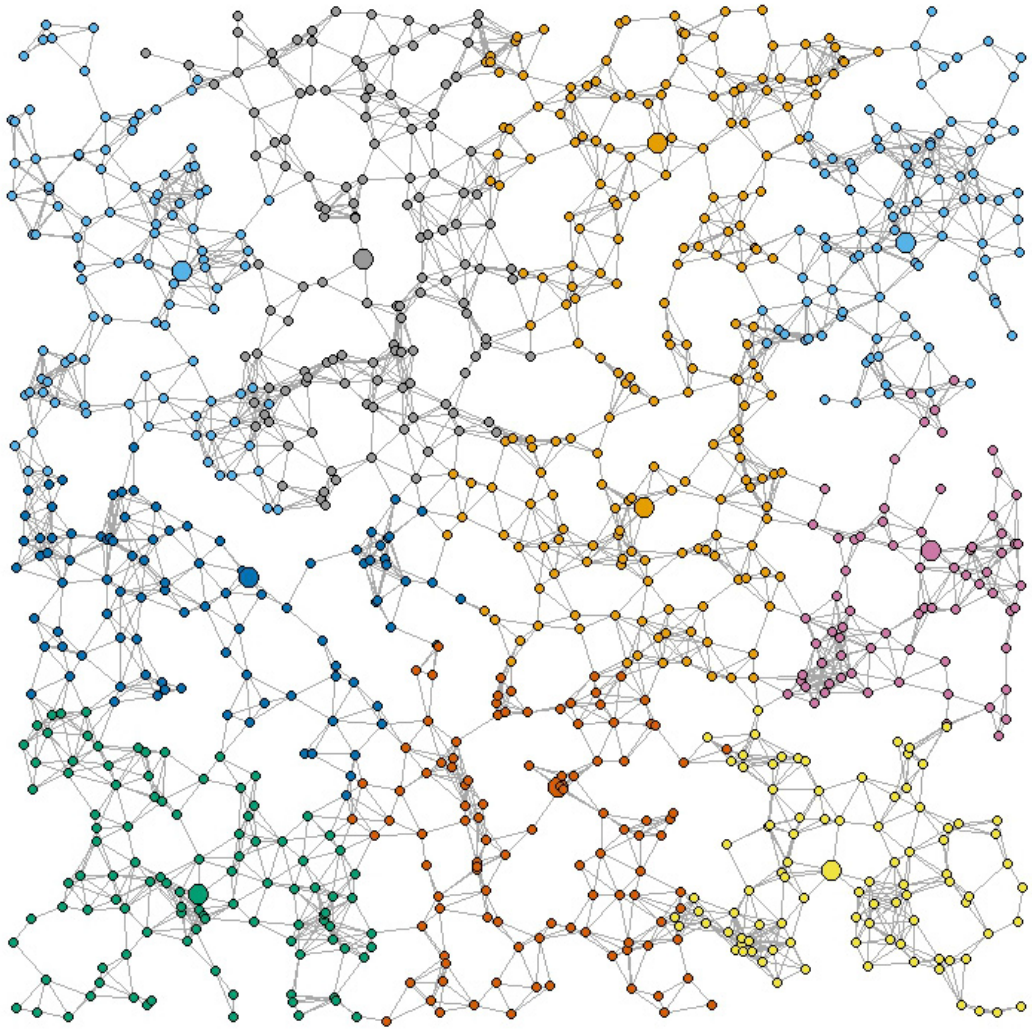

FIGURE 3. Illustrative result of the selection of hubs in a random geometric network on a unit square with a 1000 nodes, with r = 0.05 and with 10 selected hubs denoted by bigger circles. The clusters corresponding to the hubs are color differentiated.

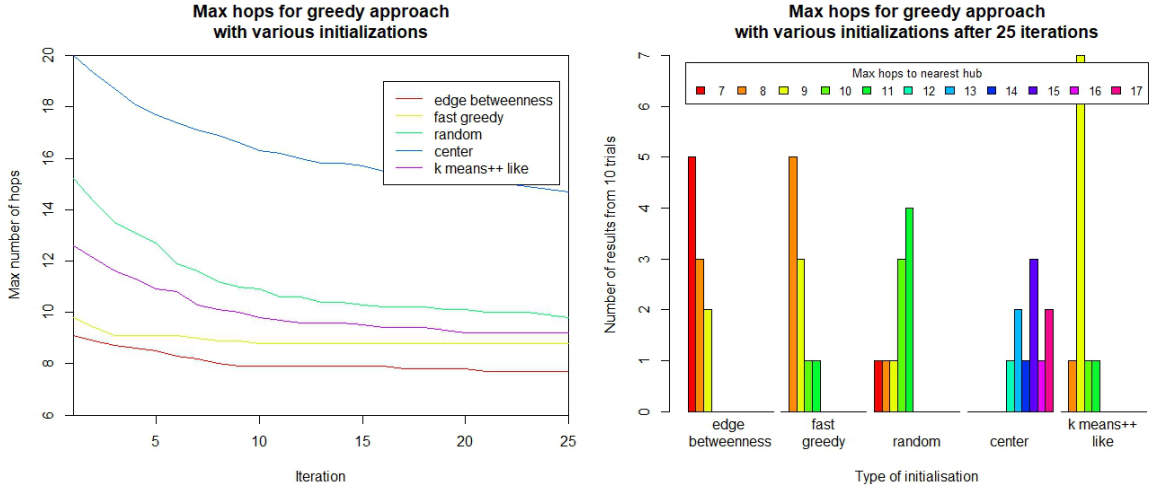

FIGURE 4. The first set of graphs shows averages of maximum number of hops depending on iterations of the greedy Algorithm 1 for all studied initializations of the algorithm. The following bar graph shows the quality of results after 25 iterations in more details.

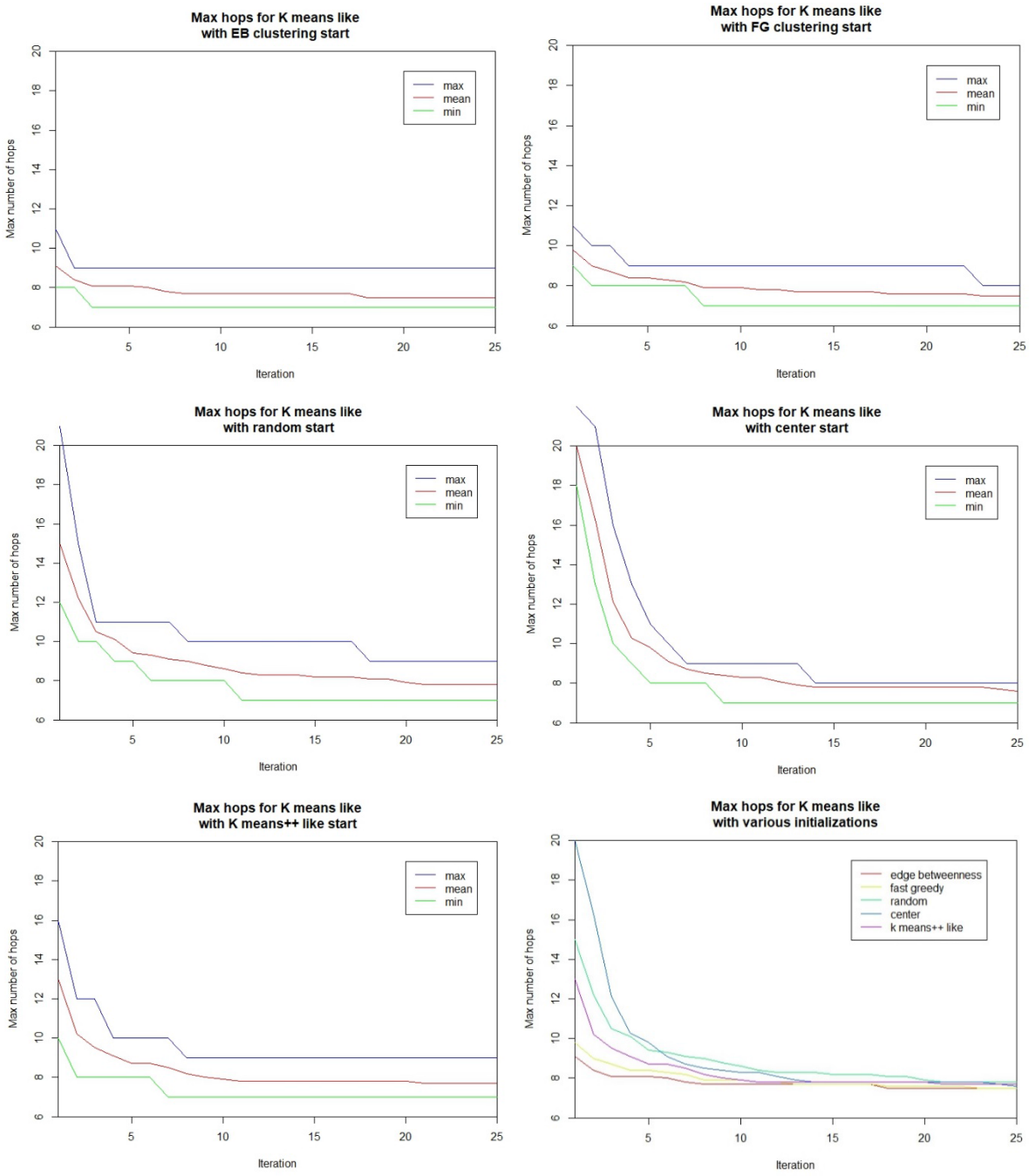

FIGURE 5. The first five graphs show averages, maximum and minimums of max number of hops depending on iterations of the *k*-means-like Algorithm 2 for all the studied initializations of the algorithm. The last graph compares the averages in one figure.

the diagram D. This corresponds to the step 3 of Algorithm 2. Further steps of the algorithm would just repeat the procedures which principle was shown in the diagrams C and D.

III. RESULTS AND DISCUSSION: TESTING THE HEURISTICS

In order to test the proposed heuristic strategies, the strategies were applied to 10 randomly generated connected geometric networks (all with one component), based on the distance of random points on a unit square. The number of nodes in each network was set to $|V| = 1000$. The radius within which the vertices were connected by an edge was set to $r = 0.05$ in agreement with [4]. For a required number of $k = 10$ hubs, we tried ten times the best approach (k-means-like algorithm with the initial selection of hubs by edge betweenness community detection) for each of the generated networks, each

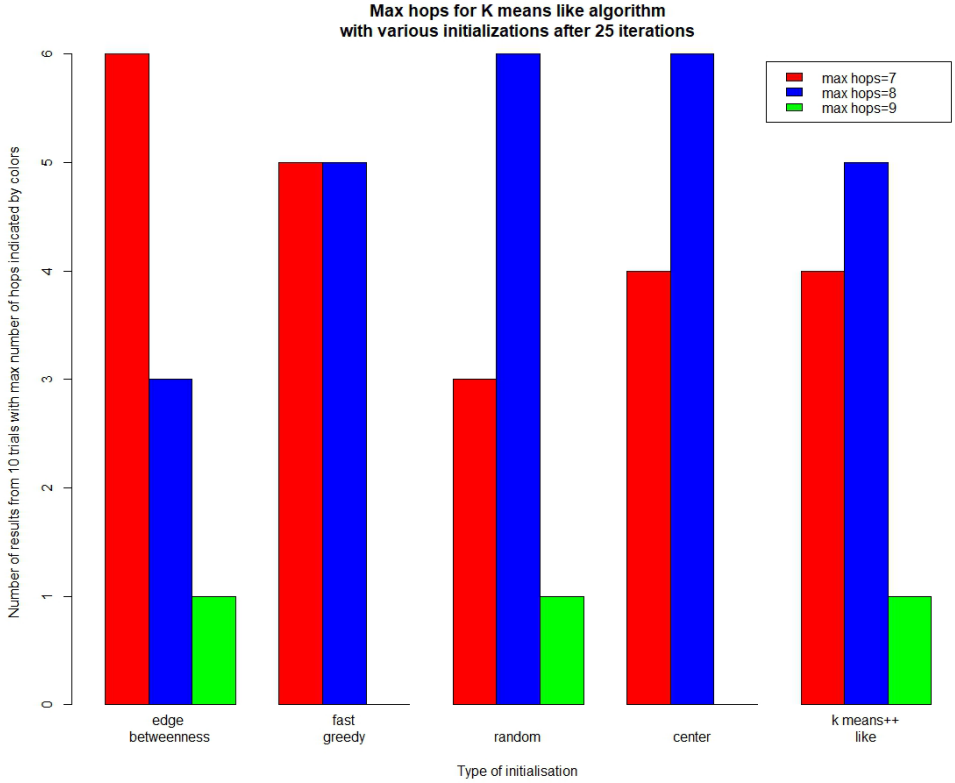

FIGURE 6. The bar graph shows the maximum number of hops after 25 iterations for the *k*-means-like algorithm for all its studied initializations.

time with different initial permutation of indices of nodes of the network. Edge betweenness clustering is a deterministic algorithm which for the same network provides the same value of maximum edge betweenness. However, when there are more edges with the same betweenness value, the algorithm depends on indexing, and for different permutations of node indices the algorithm may provide different results. In all tested networks, the minimum of maximum node graph distance to its closest hub was 7; however, it was not always achieved - not even by the best algorithm. An example of the result can be seen at Figure 3, where each cluster belonging to a different hub is colored differently (the nodes with equal distance to two hubs are randomly assigned to one of them in the figure). The hubs are denoted by bigger circles.

The results for the Algorithm 1 are shown at Figure 4. The first graph shows the dependence of average values of maximal number of hops to the closes hubs on iterations of the algorithm, calculated for the 10 randomly generated connected geometric networks. The results are color differentiated for different methods of initial selection of hubs. The lower the line is, the better the results are. The quality of results decreases from edge betweenness, fast greedy, k-means++ like, up to random and center as the worst. While all the methods seem to converge slowly, the edge betweenness begins already with an average close to 9 in the initial iterations, which is near the (probable) optimum 7. The bar graph on the right of Figure 4 shows in more details the situation after 25 iterations. For the edge betweenness (our best initialization method), half of the networks after 25 iterations converge to the (probable) optimum 7, the rest were one or two hops worse. The only other method which achieved this probable optimum at least once was a random selection of initial hubs, but the majority of these results were worse by four hops. Results for all initialization methods had a significant variance.

The results of Algorithm 2 are shown in Figure 5 in greater details. The first five pictures show averages, maximums, and minimums of the maximum number of hops depending on iterations of the k-means-like Algorithm 2, calculated for the same 10 randomly generated connected geometric networks that were used for testing the Algorithm 1. The last picture compares the averages of the maximum number of hops in one picture. Although after about 15 iterations there does not seem to be much difference between the initial selections of the hubs, the more detailed bar graph in Figure 6 shows clear differences in the quality of results after 25 iterations. The first two methods, based on community detection, clearly outperform the rest of the methods.

The Figure 7 shows that a range of hops is a reasonable indicator of the convergence of the algorithm, but it is still too imprecise to be used as a stopping criterion.

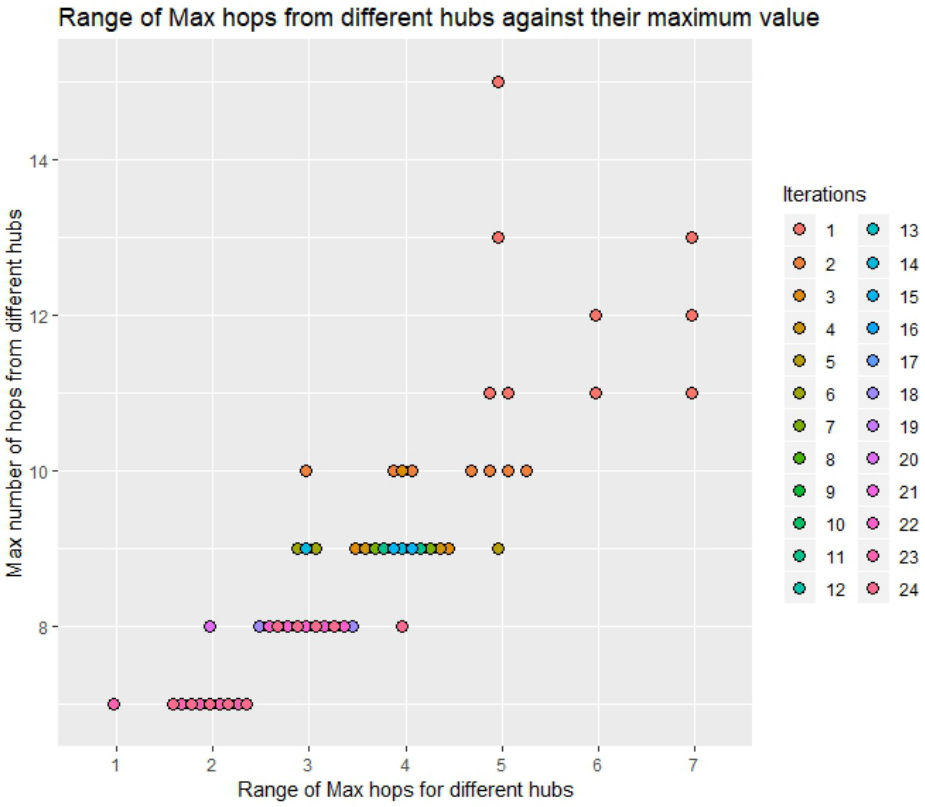

FIGURE 7. Maximum number of hops depending on the range of maximum number of hops from different centers with color-coded iterations.

In order to compare these results to the results of a classical k center heuristic, Gonzalez [18] approach was tried on the test networks. It provided an average maximum number of hops 10.7, which is more than 50 per cent worse than the achieved optimum by our multi-start approach. The Gonzalez algorithm tended to place most of the hubs near the edges of the unit square - within which the nodes were generated - which was far from ideal.

Furthermore, new sets of geometric networks (RGN) were generated with the radius $r = 0.06$ and 0.07. For each network of the set, max route length was calculated from Gonzalez [18] approach and was compared with multistart (10x) k-means-like approach with edge betweenness initialization. As Table 1 shows, Gonzalez approach provides about 50 percent worse results when compared to multistart k-means-like approach.

Every optimization approach includes at least some tuning parameters. However, in both the present approaches such parameters have mostly discrete values (e.g., the number of iterations) and categorical values (the used hub selection initialization method). Optimization of the performance greatly depends on the size of the studied network. Shortest paths from a single hub in unweighted networks can be determined by a by breadth-first search with complexity $O(|E| + |V|)$, which must be multiplied by the number of hubs k. If we set the number of iterations as a given constant, the complexity of the whole method is determined by the initialization method. For a large network, one would likely pick random, k-means++ like, or graph center initialization of hubs. It can be assumed that for a bit smaller networks, fast greedy $O(|E| + |V|\log^2|V|)$ provides better results, and for networks roughly up to 1000 vertices, edge betweenness $O(|V||E|\text{^}2)$ likely provides best results for a given CPU time. Since the optimal values largely depend on the size of networks, their type and their density, among other parameters, we refrained from a detailed running time analysis and tuning of parameters.

TABLE 1. Results of optimum routing by k-means-like algorithm with edge-betweenness, compared to Gonzalez [18] approach.

10 random geometric networks, 1000 nodes and *k=10*			
	***r*=0.05**	***r*=0.06**	***r*=0.07**
average $\|E\|$	3764.4	5395.8	7229.3
average node degree	7.5	10.8	14.5
average max route length by Gonzalez [18]	10.9	7.9	6
average max route length multistart k-means-like	7	5.1	4.1

The question remains whether the Algorithm 2 can be stopped early based on some more sophisticated criterion than maximum number of iterations. During the calculations, it has been noticed that either the converged algorithm tends to get the same number of maximum hops for all the hubs in the network, or the difference is at most one. When the number of iterations is low, the range of maximum number of hops for each hub can be quite wide. The number of iterations shown by a color code in Figure 7 also indicates that the progress during the iterations tends to move from the right upper corner to the left lower corner. Unfortunately, this criterion is not perfect; sometimes even the configuration with only two different values of maximum number of hops for different hubs and their clusters may not signify convergence to the optimum. A further run of the algorithm then can improve such results. Optimum results can be expected with early stopping of the iterations due to a low maximum range of hops for different hubs, and repeated runs with different permutation of indices of nodes.

IV. CONCLUSION

The best combination of initial hub selection by edge betweenness followed by k-means-like approach seems to provide very good results compared, for example, to the random selection of hubs: our method reduced the worst-case number of hops by about a half.

Both the greedy Algorithm 1, as well as the k-means-like Algorithm 2 were able to substantially improve the initial selection of hubs, whatever selection method was used. However, the greedy Algorithm 1 was more prone to get stuck in a local optimum, and most of the time it failed to achieve the (likely) assumed global optimum. Only for the initial selection of hubs using the edge betweenness community detection was the Algorithm 1 not stuck. Edge betweenness achieved for both, the Algorithm 1 as well as the Algorithm 2, the assumed optimum in at least half of the trials, and in the rest of the trials, the solution was at worst two hops longer than the assumed optimum. The initial selection also made a great difference in the initial iterations of Algorithm 2, but after ten iterations of the algorithm, the differences were nearly unsubstantial.

As the complexity of the algorithm is concerned, edge betweenness [30] for sparse graphs is $O(|V|^3)$ as is one iteration of the k-means-like algorithm. Although similarly to evolutionary heuristics the number of iterations of the algorithm has no natural bounds, one can still set it reasonably to a couple of tens of iterations, so that the complexity would remain unaffected. An alternative stopping method of the Algorithm 2 might be the reduction of the range of different maximum hop numbers for different hubs with their clusters to two. On the other hand, Algorithm 2 converges rather quickly, after a few tens of iterations there is usually not much point in continuing further. In such case, it would be much more effective to use multiple restarts, similar to common k-means clustering.

Even though Algorithm 2 is fast enough to work for slowly changing dynamic networks, in this contribution it was tested on a standard static model. Better adaptation to dynamic networks would require rewriting the Algorithm 2 as a distributed algorithm and such action would likely decrease the algorithm efficiency, similarly to results in [32].

Our k-means-like Algorithm 2 with the initial selection of hubs using edge betweenness community detection can be currently claimed as Pareto optimal, giving the most reasonable tradeoff between execution time and the maximum alert lag. Much faster approach (complexity $O(|V|k)$) is balanced by fifty per cent worse maximum alert lag in a classical k center greedy heuristic [18]. On the other hand, if one would require better results at the price of an increase in computational resources by a few orders of magnitude, simulated annealing like [13] or one of the evolutionary approaches might be better suited.

REFERENCES

[1] F. Aadil, K. B. Bajwa, S. Khan, N. M. Chaudary, and A. Akram, "CACONET: Ant colony optimization (ACO) based clustering algorithm for VANET," *PLoS ONE*, vol. 11, May 2016, Art. no. e0154080.
[2] A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh, "Max-min d-cluster formation in wireless ad hoc networks," in *Proc. Conf. Comput. Commun. 19th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, Mar. 2000, pp. 32–41.
[3] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.
[4] A. Bagchi, S. Galhotra, T. Mangla, and C. M. Pinotti, "Optimal radius for connectivity in duty-cycled wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 11, no. 2, 2015, Art. no. 36.
[5] S. C. Barman, M. Pal, and S. Mondal, "An optimal algorithm to find minimum k-hop dominating set of interval graphs," *Discrete Math., Algorithms Appl.* doi: 10.1142/S1793830919500162
[6] S. Basagni, "Distributed clustering for ad hoc networks," in *Proc. 4th Int. Symp. Parallel Archit., Algorithms, Netw.*, 1999, pp. 310–315.
[7] P. Basuchowdhuri and S. Majumder, "Finding influential nodes in social networks using minimum k-hop dominating set," in *Proc. Int. Conf. Appl. Algorithms.*, Cham, Switzerland: Springer, 2014, pp. 137–151.
[8] A. Bóta, M. Krész, and B. Zaválnij, "Adaptations of the k-means algorithm to community detection in parallel environments," in *Proc. 17th Int. Symp. Symbolic Numeric Algorithms Sci. Comput. (SYNASC)*, 2015, pp. 299–302.
[9] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proc. 4th Annu. ACM/IEEE Int. Conf. Mobile Comput. Netw.*, 1998, pp. 85–97.
[10] A. Campan, T. M. Truta, and M. Beckerich, "Approximation algorithms for d-hop dominating set problem," in *Proc. Int. Conf. Data Mining (DMIN)*, 2016, pp. 86–91.
[11] S. Chehrazad, H. S. Aghdasi, N. Shariati, and M. Abolhasan, "Addressing coverage problem in wireless sensor networks based on evolutionary algorithms," in *Proc. 23rd Asia–Pacific Conf. Commun. (APCC)*, 2017, pp. 1–5.
[12] X. Chen and J. Shen, "Maximum necessary hop count for packet routing in MANETs," in *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, J. Wu, Ed. Boca Raton, FL, USA: Auerbach Publications, 2005, pp. 60–69.
[13] J. G. Diaz, R. M. Mendez, J. S. Hernandez, and R. M. Mendez, "Local search algorithms for the vertex k-center problem," *IEEE Latin America Trans.*, vol. 16, no. 6, pp. 1765–1771, Jun. 2018.
[14] M. Diykh and Y. Li, "Complex networks approach for EEG signal sleep stages classification," *Expert Syst. Appl.*, vol. 63, pp. 241–248, Nov. 2016.
[15] R. Ferrero and F. Gandino, "Analysis of random geometric graph for wireless network configuration," in *Proc. 10th Int. Conf. Mobile Comput. Ubiquitous Netw. (ICMU)*, Oct. 2017, pp. 1–6.
[16] E. Forgey, "Cluster analysis of multivariate data: Efficiency vs. Interpretability of classification," *Biometrics*, vol. 21, no. 3, pp. 768–780, 1965.

[17] D. Gabriska and M. Olvecky, "Analysis and risk reduction in operation of hazardous programmable electronic systems," in *Proc. 16th Int. Symp. Intell. Syst. Inform. (SISY)*, 2018, pp. 123–126.
[18] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theor. Comput. Sci.*, vol. 38, pp. 293–306, 1985.
[19] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 7, pp. 1029–1046, Sep. 2009.
[20] D. S. Hochbaum, *Approximation Algorithms for NP-Hard Problems*. Boston, MA, USA: PWS Publishing Company, 1997, pp. 346–398.
[21] A. S. M. S. Hosen, G. H. Cho, and I. H. Ra, "An eccentricity based data routing protocol with uniform node distribution in 3D WSN," *Sensors*, vol. 17, no. 9, p. 2137, 2017.
[22] M. Hostovecký and B. Babušiak, "Brain activity: Beta wave analysis of 2D and 3D serious games using EEG," *J. Appl. Math., Statist. Inform.*, vol. 13, pp. 39–53, Jan. 2017.
[23] N. Jabeur, A. U.-H. Yasar, E. Shakshuki, and H. Haddad, "Toward a bio-inspired adaptive spatial clustering approach for IoT applications," *Future Gener. Comput. Syst.*, to be published. doi: 10.1016/j.future.2017.05.013.
[24] H. Kenniche and V. Ravelomananana, "Random geometric graphs as model of wireless sensor networks," in *Proc. 2nd Int. Conf. Comput. Automat. Eng. (ICCAE)*, vol. 4, 2010, pp. 103–107.
[25] H. Kuang, Y. Qiu, R. Li, and X. Liu, "A hierarchical K-means algorithm for controller placement in SDN-based WAN architecture," in *Proc. 10th Int. Conf. Measuring Technol. Mechatron. Autom. (ICMTMA)*, 2018, pp. 263–267.
[26] S. Kumar and Z. Raza, "A K-means clustering based message forwarding model for Internet of Things (IoT)," in *Proc. 8th Int. Conf. Cloud Comput., Data Sci. Eng.*, 2018, pp. 604–609.
[27] S. Kundu and S. Majumder, "A linear time algorithm for optimal k-hop dominating set of a tree," *Inf. Process. Lett.*, vol. 116, pp. 197–202, Feb. 2016.
[28] L. Lü, D. Chen, X.-L. Ren, Q.-M. Zhang, Y.-C. Zhang, and T. Zhou, "Vital nodes identification in complex networks," *Phys. Rep.*, vol. 650, pp. 1–63, Sep. 2016.
[29] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, Berkeley, CA, USA: Univ. California Press, 1967, pp. 281–297.
[30] M. Newman, *Networks: An Introduction*, 2nd ed. London, U.K.: Oxford Univ. Press, 2018.
[31] A. Nodehi, "Determining cluster-heads in mobile ad-hoc networks using multi-objective evolutionary based algorithm," *J. Adv. Comput. Res.*, vol. 9, no. 3, pp. 133–151, 2018.
[32] G. Oliva, R. Setola, and C. N. Hadjicostis. (2013). "Distributed k-means algorithm." [Online]. Available: https://arxiv.org/abs/1312.4176
[33] D. Pethaperumal, Y. Peng, and H. Qin, "Budget-Hub: A low cost IoT hub selection and neighbor assignment scheme," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, Feb. 2018, pp. 700–705.
[34] D. Pompili, T. Melodia, and I. F. Akyildiz, "Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks," in *Proc. 12th Annu. Int. Conf. Mobile Comput. Netw.*, 2006, pp. 298–309.
[35] D. R. Prasad, P. V. Naganjaneyulu, and K. S. Prasad, "Energy efficient clustering in multi-hop wireless sensor networks using differential evolutionary MOPSO," *Brazilian Arch. Biol. Technol.*, vol. 59, Jan. 2017, Art. no. e16161011.
[36] A. Rauniyar, P. Engelstad, and J. Moen, "A new distributed localization algorithm using social learning based particle swarm optimization for Internet of Things," in *Proc. IEEE 87th Veh. Technol. Conf. (VTC Spring)*, Jun. 2018, pp. 1–7.
[37] B. P. Santos, L. F. M. Vieira, and M. A. M. Vieira, "CGR: Centrality-based green routing for low-power and lossy networks," *Comput. Netw.*, vol. 129, pp. 117–128, Dec. 2017.
[38] Y. Shi, X. Xu, C. Lu, and S. Chen, "Distributed and weighted clustering based on d-hop dominating set for vehicular networks," *KSII Trans. Internet Inf. Syst.*, vol. 10, pp. 1661–1678, Apr. 2016.

MAREK ŠIMON received the Diploma degree in automation engineering from the Slovak University of Technology, Slovakia, in 1996, and the Ph.D. degree in didactics of technical professional subjects from the Faculty of Education, Constantine the Philosopher University, Nitra, Slovakia, in 2013. He is currently a Senior Lecturer and the Deputy Head of the Department of Computer Science, University of Ss. Cyril and Methodius in Trnava, Slovakia. His research interests include network security and the IoT. He is an Editorial Committee Member of the *International Journal of Information Processing and Management*.

DIRGOVÁ LUPTÁKOVÁ IVETA received the Diploma degree in mathematics from the Faculty of Mathematics, Physics, and Informatics, Comenius University in Bratislava, Slovakia, in 2001, and the Ph.D. degree in statistics from the Faculty of Economic Informatics, University of Economics in Bratislava, Bratislava, Slovakia, in 2005. She is currently a Vice-Dean for strategy, accreditation, and promotion with the Faculty of Natural Sciences, University of Ss. Cyril and Methodius in Trnava, Slovakia. She is also a Managing Editor of the *Journal of Applied Mathematics, Statistics and Informatics*. Her research interests include mathematical informatics, statistics, graph theory, theoretical computer sciences, and optimization metaheuristics.

LADISLAV HURAJ received the Ph.D. degree in applied informatics from the Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Slovakia, in 2006. He is currently an Associate Professor of computer science with the University of Ss. Cyril and Methodius in Trnava, Slovakia. His research interests primarily include information and network security, the IoT, and computer education. He is a member of the Editorial Board of the *Journal of Applied Mathematics, Statistics and Informatics*. He served as a Committee Member in many international conferences.

JIŘÍ POSPÍCHAL received the Diploma degree in physical chemistry from the University of Jan Evangelista Purkyně in Brno, Czech Republic, in 1984, and the Ph.D. degree in chemistry from the Faculty of Chemical and Food Technologies, Slovak University of Technology in Bratislava, Bratislava, Slovakia, in 1990. He transferred his interest from quantum chemistry and computer assisted organic synthesis into computer science. He is currently a Professor of applied informatics with the Faculty of Natural Sciences, University of Ss. Cyril and Methodius in Trnava, Slovakia. His research interests include evolutionary algorithms, optimization metaheuristics, artificial intelligence, neural networks, and network science.