SYSTAS: Density-based Algorithm for Clusters Discovery in Wireless Networks

Apostolos Kousaridas, Markus Dillinger
European Research Center
Huawei Technologies Duesseldorf GmbH
Munich, Germany
{apostolos.kousaridas, markus.dillinger}@huawei.com

Stefanos Falangitis, Panagis Magdalinos, Nancy Alonistioti
National and Kapodistrian University of Athens
Athens, Greece
{sfalangitis, panagis, nancy}@di.uoa.gr

Abstract—**The Internet of Things will comprised of billions of devices that will be randomly placed, forming a dense and unstructured network environment with overlapping wireless topologies. The management of the available network resources and the control of the increased signaling traffic are crucial for the reliable and efficient provision of novel services. The grouping of involved entities into clusters will be a promising approach for solving locally and autonomously these issues. This paper proposes the SYSTAS algorithm for the distributed discovery and formation of clusters in small-world graphs of fixed wireless nodes by exploiting local topology knowledge and without having any information about the expected number of clusters. The density of the network graph, the interactions among participating nodes and the model of preferential attachment are used by the proposed scheme. The effectiveness of SYSTAS is evaluated using different topologies; experimental evaluation demonstrates that SYSTAS results are comparable or outperform other clustering algorithms.**

Keywords— wireless network; IoT; autonomous systems; clusters; modularity;

I. INTRODUCTION

The Internet of Things (IoT) will merge computing and communication capabilities with physical processes to support healthcare, automotive systems, smart cities, robotics etc. All these novel services require a communication framework that will enable the collaboration of involved objects, satisfying the Quality of Service (QoS) requirements (e.g., reliability, latency), which in many cases are strict. The interactions and the collaboration of hundreds or even thousands of entities create a complex environment, where scarce radio resources should be effectively managed, without increasing the control signaling overhead. Thus, while we are moving towards IoT world, autonomicity is required for the evolution of management and control functionalities of networks that operate in a dynamic and dense environment [1].

An autonomous communication system is capable of monitoring its network-related state and modifying it based on specified policy rules, thresholds or business goals with the minimal involvement of human administrators. A key design issue for the deployment of an autonomous communication system is the selection among a) a centralized, b) a distributed, c) or even a hybrid approach. A centralized solution facilitates optimal decision making due to the more holistic view of the network status, but comprises a single point of failure since scalability, computational, local optimization and local search issues arise. Although a fully distributed solution can cope with these issues it requires continuous coordination of various nodes, both spatially and temporally so as to avoid conflicts or dependencies. In order to address the disadvantages while in parallel capitalize on the key advantages of both paradigms, a hybrid approach that is based on the formation of clusters (groups) could be adopted. Nodes that support Device-to-Device (D2D) communication, IoT objects, and small cells are examples of wireless nodes that could form clusters for various group-based network functionalities e.g., group-based resource allocation, group-based access control [2], [3].

Clustering is a multidisciplinary research field, which has been studied by different research communities, e.g., mobile ad hoc networks (MANETs), Wireless Sensor Networks (WSN), graph theory and data mining. Various algorithms have been proposed for clusters discovery/formation (cf. Section II). The majority of MANET clustering algorithms is distributed, but they establish clusters focusing on specific applications or performance improvements operations (e.g., low-maintenance, mobility-aware clustering, energy-efficiency, load-balancing, etc). Additionally in many cases the formed clusters overlap and the maximum distance from the head node is pre-defined. Graph theory algorithms are effective, but they require global knowledge (i.e. a fully described graph) in order to discover the appropriate clusters. Finally, many of the data mining clustering algorithms are centralized, while in the case of distributed schemes it is necessary to indicate the number of expected clusters, before running the algorithm. In our context, the devices of an IoT environment form small-world graphs, and existing clustering algorithms have not been designed to support management and control functions. Considering the above as well as the structural properties of random graphs, an application-neutral, distributed cluster discovery algorithm is required that will operate by exploiting local information and will be agnostic towards the expected number of clusters.

In this paper, we propose SYSTAS, a novel algorithm for the distributed discovery of clusters of wireless nodes based on physical network topology features. SYSTAS identifies clusters by exploiting the density of the network graph, the interactions among nodes and the preferential attachment model. SYSTAS efficiently leads to the specification of clusters, which consist of simple members and a head node. The identified clusters are non-overlapping and with no restrictions regarding their diameter. The quality of the formed clusters affects the effectiveness of group-based network processes. The discovery of modular clusters will facilitate the operation of the different functions (management and control) of autonomous systems. For that reason, in this work we evaluate and compare SYSTAS using the modularity score *Q* metric [4], which measures the strength of division of a network into modules. Modularity score is neutral to other QoS indicators of communication networks.

The remainder of the paper is organized as follows: Section II presents relevant research efforts, while the proposed algorithm for clusters discovery is described in Section 3. Section 4 presents the experimental validation procedure and the derived results. Section 5 concludes the paper and sketches future research directions.

II. Background Work

Clustering in MANETs is defined as the virtual partitioning of the dynamic nodes into various groups. Groups of nodes are formed with respect to their closeness. Two nodes are assumed neighboring when both of them lie within their transmission range and set up a bidirectional link between them [5]. From a graph-theoretic perspective, clustering is the task of grouping the vertices of a graph into disjoint groups. The derived partitioning attempts to minimize the sum of edges between clusters, while in parallel maximize their sum in every cluster [6]. In some of the research literature, a cluster in a graph is called a community [7]. Finally, in data mining, clusters are formed among (data) objects typically by exploiting their pairwise similarities [8].

Several clustering algorithms have been proposed for MANETs the last decade, exploiting different criteria (nodes' ID, nodes' degree, mobility patterns, closeness, etc) and having different targets (e.g., load balancing, energy efficiency). According to the Lowest-ID Cluster (LID) algorithm that is one of the first proposed solutions, every node is assigned a unique non-negative identification number (ID), which is the deciding factor for the status of a node. Each node broadcasts its ID to its neighbors and receives theirs. If a node listens to all the neighbors' IDs that are higher than its own ID, it declares itself as the cluster head among its immediate neighbors. This process is repeated till all the nodes are assigned with the role of a head or a member of a cluster [9]. MOBIC is a mobility-metric based version of LID which was proposed by Basu et al. [10]. Authors in [11] have proposed the Highest Connectivity (HC) algorithm, where cluster heads become the nodes that enjoy the highest degree of connectivity. In [12] Chatterjee et al. proposed a weighted clustering algorithm (WCA) where a set of node parameters such as degree of connectivity, mobility, transmission power and available battery power are taken into account for the selection of a cluster head. Additionally, depending on the network scenario, a weighted scheme is also applicable. Basagni et al. presented a distributed clustering algorithm (DCA) and the mobility adaptive clustering algorithm (DMAC) [13]. In DCA, each node is associated with a unique parameter (weight), which is used for the identification of the node's role. In [14], the authors propose an enhancement of LID, the Least Cluster Change (LCC) algorithm. In [15], the Hierarchical control clustering (HCC) is introduced which creates a hierarchical control structure for multi-hop wireless networks. HCC cluster formation involves constructing a spanning tree in time proportional to the diameter of the network by performing a distributed breadth-first search. Additional cluster formation algorithms have been proposed for energy saving purposes especially in Wireless Sensor Networks (WSN) [16]. A more detailed analysis of the different algorithms that have been proposed for MANETs or WSNs is available in [5], [17] and [18].

Cluster discovery has been also studied in the context of graph theory. The work can be broadly divided in two main areas ([6]), namely Graph Partitioning and Community Structure Detection. Graph partitioning (GP) has been pursued particularly in computer science and related fields, with applications in parallel computing and integrated circuit design. Community Structure detection (CS), has been pursued by sociologists, with applications especially to social and biological networks. In GP it is usually assumed that we know or at least have an indication about the number of groups, while in CS, we assume that the network of interest divides naturally into subgroups and the experimenter's job is to find those groups. The number and size of the groups are thus determined by the network itself and not by the experimenter. Repeated Random Walks (RRW) and Markov Cluster Algorithm (MCL) are well known schemes for CS-based clusters discovery [19].

The analysis of huge amounts of data has led to the need for the development of data mining clustering schemes. These algorithms facilitate the identification of groups of similar objects. Typically these objects are database entries describing key features of the set members; the clustering procedure is executed simultaneously on the entire dataset. Depending on the algorithm, the process has variable time constraints and cluster assignments. The procedure may lead to clusters' formation, hierarchies of clusters, etc. Typical methodologies are the Partitioning clustering (e.g., k-Means) and Hierarchical clustering [8].

III. Proposed Algorithm

A. *Structure and Properties of Dense Wireless Networks*

There are various structural metrics that could be used in order to describe the properties of a graph topology formed by a network of IoT objects e.g., node degree (i.e., number of edges incident to the node), degree distribution, path length, node clustering coefficient (i.e., ratio of the number of edges between node's neighbors to the maximal possible number of such edges), geodesic distance. According to these properties four main categories of structural models are identified: a) Random network: a fixed number of nodes are connected randomly to each other, b) Small-world network: any two nodes can be connected with a path of only a few links, c) Scale-free network: network's properties are determined by hubs, d) Hierarchical network: seamlessly integrates a scale-free topology with an inherent modular structure by generating a network that has a power-law degree distribution. Each structural model has specific characteristics regarding the values of the key structural metrics e.g., degree distribution and clustering coefficient. The wireless networks in an urban dense environment appear the properties of a small-world graph, since the number of nodes, connections between them and graph edges are determined in a random way, but with small average distances between node pairs [20] .

B. SYSTAS

The proposed algorithm assumes zero or low mobility levels for the involved nodes. All nodes have the capability to discover their physical topology and by exploiting this commonly known scheme, form in a distributed way, the appropriate number of clusters. Clusters are non-overlapping and consist of two types of nodes: a) simple member nodes, and b) head nodes. The application of SYSTAS leads to the election of a single head for each cluster and the specification of cluster borders through the allocation of member nodes to elected heads. It should be noted that the number of elected heads equals to the number of clusters, while their role in the algorithm is to facilitate the formation of the clusters.

SYSTAS is based on the topological characteristics of the network area (nodes' degree, clustering coefficient). Heads are elected according to their degree (number of edges incident to the node) and then clusters are formed based on the network density and by a process of "preferential attachment", where nodes prefer to join the more "popular" clusters. Two nodes are considered neighboring if they are within each other's transmission range.

We assume that a network topology is represented by a connected and undirected graph $G = (V, E)$, where V is the set of network nodes and E the set of edges (connectivity links) between network nodes. An edge exists between two nodes, if one is within the coverage area of the other. Let the cardinalities of V and E be denoted by n and e, respectively; i.e., n is the number of nodes, and e the number of connectivity links. The input variables of the proposed algorithm are the following:

- h_{max}: the maximum number of hops that each head node advertises its presence.
- m: the maximum number of member nodes that triggers a cluster to be merged with a neighboring cluster.
- R_{thr}: the threshold of inter-cluster and intra-cluster edges ratio (denoted by R) that initiates a cluster merging process.

The proposed algorithm consists of three phases:

1. Formation of initial clusters with heads and members selection.
2. Expansion of clusters with high clustering coefficient.
3. Merging of clusters with high inter-connectivity factor.

In the first phase, the initial set of clusters is formed based on the degree parameter (Table I). Each node i monitors its proximity, measures its degree D_i -which is the number of its one hop away neighbors- and advertises it to its proximity (one-hop away nodes). After receiving all neighboring nodes' degrees -$D_{i,j}$-, each node i evaluates local and neighboring degrees and selects as head (H_i) the node with the maximum degree. All head nodes that have a zero number of member nodes are merged with the neighboring cluster where the head has the largest degree.

The head-node is not a member node of any other cluster and each member is allocated to only one head/cluster. Every member node stores the degree and the ID of its head, while the head knows the number of member nodes that constitute its cluster and their IDs. With the end of this phase, the initial clusters have been formed, consisting of heads and members.

TABLE I. PHASE I - FORMATION OF INITIAL CLUSTERS AND HEADS IDENTIFICATION

	Each node: InitialClustersFormation()
1:	Discover neighboring nodes;
2:	Calculate Degree (D_i);
3:	Advertise D_i one hop away;
4:	Collect $D_{i,j}$ from all neighboring nodes;
5:	Selected Head (H_i): Node with the maximum degree ($D_{i,j}, D_i$)
6:	**If** node i is a head without members **then**
7:	Join the cluster of neighboring cluster where head k has the maximum degree D_k;
8:	**End**

The goal of the second phase is to expand the clusters that have high clustering coefficient. Thus, areas of the graph with high modularity will be identified. For this step, both head nodes and member nodes participate, undertaking different tasks, presented in Table II and Table III, respectively. Firstly, each head node H_k calculates the existing number of its member nodes, denoted by M_k, and advertises it h hops away (Table II, steps 1-3). The initial value of h is one (h=1). The head node then waits for a period T_k for the completion of the forwarding phase of advertisements messages and the re-allocation of member nodes to clusters. The re-allocation phase is discussed below and presented in Table III. After T_k, every head node re-calculates the number of its member nodes and updates M_k (Table II, steps 5).

Clusters that exhibit low cardinality ($M_k < m$) are merged with the neighboring cluster that has the largest dominance factor, denoted DF (Table II, steps 6-8). DF quantifies the influence (in terms of edges) of cluster g on cluster k. Hence, for its calculation it is necessary to measure the total number of inter-cluster edges of cluster k (head and member nodes), denoted by I_k^e, as well as the number of inter-cluster edges between cluster k and neighboring cluster g, denoted $I_{k,g}^e$. DF is calculated as follows:

TABLE II. PHASE II - EXPANSION OF CLUSTERS (HEAD SIDE)

	Head node: ClustersExpansion_HeadSide()
1:	$h = 1$;
2:	Calculate the number of cluster members (M_k);
3:	Advertise M_k h hops away;
4:	Wait for a time interval T_k to complete the ClustersExpansion_MemberSide();
5:	re-Calculate M_k;
6:	**If** $M_k < m$ **then**
7:	Merge cluster of Head k with the neighboring cluster g that the maximum DF_k^g;
8:	**End**
9:	**If** $h < h_{max}$ **then**
10:	h ++
11:	GOTO step 2;
12:	**End**

$$DF_k^g = \frac{I_{k,g}^e}{I_k^e} \tag{1}$$

The head of cluster k calculates the DF of each neighboring cluster and opts to merge with the cluster that has the largest DF value. After merging, the advertisement hops are increased by 1 (h++) provided that $h < h_{max}$ and the process is repeated (Table II, steps 9-12).

Table III depicts the process for the second phase from the perspective of a member node. Each member that receives an advertisement message forwards it to a neighboring node if the following conditions are met:

- $h - 1 \neq 0$ and
- It belongs to the same cluster with the head that initially created the advertisement message.

The head node does not forward the advertisement messages of other heads. The goal of these conditions is to avoid the creation of disconnected cluster areas.

Table III presents the functions of each member in the context of the second phase of SYSTAS. Each member node collects all advertisement messages that are in maximum h hops away. Using these messages, each member calculates the influence factor of every head. The influence IF_i^k of a head H_k on a member node i, is calculated as follows:

$$IF_i^k = \begin{cases} M_k, & if\ TTL = 1 \\ S_{i,k}, & if\ TTL > 1 \end{cases} \tag{2}$$

where M_k is the number of members of the head node k. $S_{i,k}$ denotes the number of advertisement messages that member node i has received from H_k via the different paths that connect node k and node i. Each member node selects the head (i.e., cluster to join) with the largest IF value. Hence, member nodes, following the preferential attachment model, join the closest cluster with the larger influence on it.

After completing the advertisement phase, the number of formed clusters and consequently the number of heads have been reduced. The third phase of SYSTAS entails the discovery of clusters that show high inter-connectivity and can be merged (Table IV). The inter-connectivity measurement R_k is calculated by every head node k as the inter-cluster and intra-cluster edges ratio for its cluster:

$$R_k = \frac{I_k^e}{I_k^a} \tag{3}$$

where I_k^a denotes the total number of intra-cluster edges of cluster k. If $R_k > R_{thr}$ then cluster k, regardless of its constituent number of members, triggers its merging with the

TABLE III. PHASE III - EXPANSION OF CLUSTERS (MEMBER SIDE)

Member node: ClustersExpansion_MemberSide()
Do
Collect advertisement messages about M_k;
While (T_k has not expired)
For each Head (H_k)
Calculate IF_i^k;
End
Join Cluster with maximum IF_i^k;

TABLE IV. PHASE III - MERGING OF CLUSTERS

Head nodes: ClustersMerging()
Calculate I_k^a and I_k^e;
Calculate R_k;
If $R_k > R_{thr}$ **then**
Calculate DF_k^g;
Merge cluster k with the neighboring cluster g that has the maximum DF;
End

neighboring cluster that has the maximum DF value, using (2). The rationale of this merging phase, is to avoid forming clusters that are not modular and member nodes are not clearly independent in term of interactions from neighboring clusters.

Merging will lead to the reduction of clusters (i.e., elected heads) and consequently to the increase of the allocated member nodes per cluster. After the end of the merging phase, the clusters have been formed and the heads of each cluster have been elected. Finally, each head node is aware of the member nodes that constitute its cluster, while each member node is aware of the head that is assigned to and its distance in terms of hops.

IV. PERFORMANCE EVALUATION

In this section we evaluate the performance of SYSTAS using various network topologies and different values for the three input parameters (h_{max}, m, R_{thr}). Useful conclusions are derived for the values of the parameters based on the density of the graphs, which have been generated using the NS-3 network simulator [21]. We compare SYSTAS with two other cluster discovery algorithms: a) Hierarchical Agglomerative Clustering (HAC) [8], and b) Markov Cluster Algorithm (MCL) [19]. The quality of the discovered clusters is assessed using the modularity score Q metric introduced in [4].

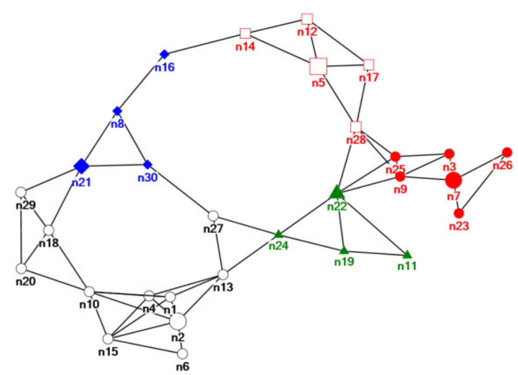
$$Q = Tr(e) - \|e^2\| \tag{4}$$

where e is a symmetric matrix whose element $e_{i,j}$ is the fraction of edges in the network that connects vertices in communities i and j, and $Tr(e)$ is the trace of matrix e (i.e., the sum of elements from its main diagonal). Q gives an indication for the quality of the formed clusters over all possible divisions of the graph, measuring the density of links inside communities as compared to links between communities. Its range is [0,1], while the higher is Q the better is the quality of the discovered clusters. In the experiments we consider graphs, where nodes are connected randomly to each other following the structure of a small-world network topology. An edge between two nodes denotes that they are within each other's transmission range. In all experiments we set $m=1$.

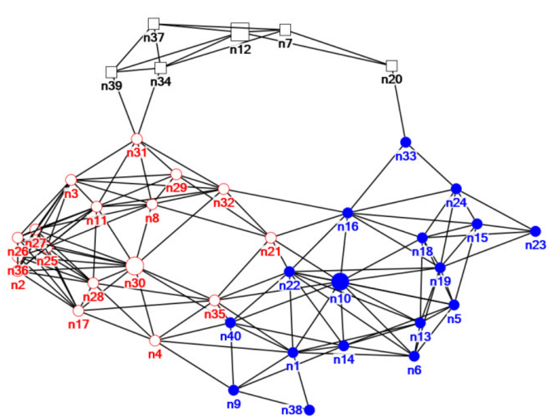
Fig. 1-a depicts the formed clusters after the application of SYSTAS on a small and sparse graph of 30 nodes (density = 0.128 and average geodesic distance=3.56). We note that the density of a graph is calculated using the formula in [22]. Fig. 1-b presents the discovered clusters for a network graph of 40 nodes, which density is 0.198 and its average geodesic distance = 2.7. Three clusters have been formed, which heads are n30, n12, and n10.

In the case of a more dense topology consisting of 80 nodes (density =0.186, average geodesic distance = 2.52) two clusters have been identified (Fig. 2-a). A larger graph of 100 nodes, which density is 0.116 and the average geodesic distance is 3.56 is depicted in Fig. 2-b. Using SYSTAS four clusters have been discovered: n56 with 33 members, n37 with 37 members, n7 with 24 members and an isolated cluster where the head node (n96) consists of only two members.

Table V, Table VI, Table VII and Table VIII present the number of formed clusters, the elected head, the number of members and the modularity score Q, setting different values to h_{max} *and* R_{thr} parameters. As expected the increase of advertisement hops (h_{max}) leads to the formation of less clusters for the same R_{thr}, while the resulting modularity score (Q) is increased. However, higher h_{max} and lower m means that more messages are exchanged among involved nodes and the cost of the algorithm is increased. For that purpose it is necessary to make a good estimation for the initial input parameters of graph. The conducted experiments show that in a uniformly dense graph (e.g., Table VII) the achieved modularity (Q) is increased setting a high R_{thr} value (R_{thr} = 1.5, R_{thr} = ∞). In a sparse graph, a larger number of advertisement hops (e.g., h_{max} = 3) and a smaller R_{thr} value (R_{thr} = 1) facilitate the effective discovery of clusters (Table V, Table VIII). It should be noted that the R_{thr} value does not affect significantly the performance of SYSTAS for the case that the partitions of a graph are clearly identifiable.

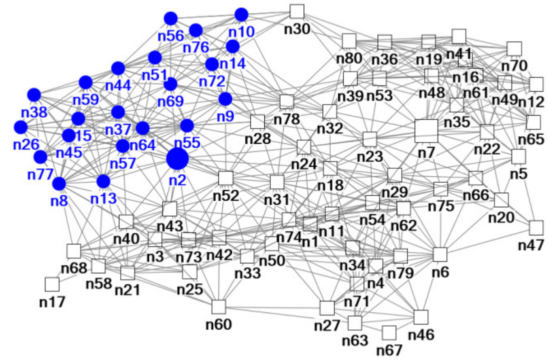


(a)

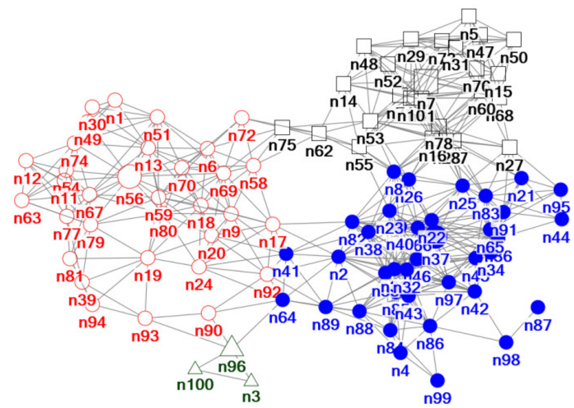


(b)

Fig. 1. Clusters visualization (a) Topology of 30 nodes (h_{max} =2, R_{thr} =1), (b) Topology of 40 nodes (h_{max} =3, R_{thr} =1)



(a)



(b)

Fig. 2. Clusters visualization (a) Topology of 80 nodes (h_{max} =3, R_{thr} = 1), (b) Topology of 100 nodes (h_{max} =3, R_{thr} = 1)

TABLE V. DISCOVERED CLUSTERS FOR 30 NODES TOPOLOGY

h_{max}	R_{thr}	Clusters	Cluster Heads	Q
1	∞	5	n2(10), n7(4) , n5(4), n21 (3), n22(4)	0.523
1	1	4	n2(10), n7(9), n5(5), n21(3)	0.528
2	∞	5	n2(10), n7(4) , n5(4), n21(3), n22(4)	**0.540**
2	1	5	n2(10), n7(5) , n5(4), n21(3), n22(3)	**0.540**
3	∞	5	n2(10), n7(5) , n5(4), n21(3),n 22(3)	0.511
3	1	4	n2(11), n7(9), n5(3), n21(3)	0.521

TABLE VI. DISCOVERED CLUSTERS FOR 40 NODES TOPOLOGY

h_{max}	R_{thr}	Clusters	Cluster Heads	Q
1	∞	4	n10(10), n30(16), n12(5), n19(5)	0.372
1	1	3	n10(16), n30(16), n12(5)	0.448
2	∞	5	n10(6), n30(17), n12(5),n19(5),n1(2)	0.404
2	1	3	n10(15), n30(17), n12(5)	0.449
3	∞	3	n10(16), n30(16), n12(5)	**0.468**
3	1	3	n10(16), n30(16), n12(5)	**0.468**

TABLE VII. DISCOVERED CLUSTERS FOR 80 NODES TOPOLOGY

h_{max}	R_{thr}	Clusters	Cluster Heads	Q
1	∞	9	n52(15), n2(6), n50(9), n7(27), n9(4), n43(2), n64(2), n78(2), n55(4)	0.289
1	1	2	n52(48), n(30)	0.342
2	∞	5	n52(12),n2(12), n50(4), n7(38),n9(9)	**0.413**

h_{max}	R_{thr}	Clusters	Cluster Heads	Q
2	1.5	4	n52(12), n2(12) n7(43), n9(9)	**0.416**
2	1	2	n2(22), n7(56)	0.315
3	∞	3	n52(12), n2(21), n7(44)	**0.419**
3	1	2	n2(21), n7(57)	0.307

TABLE VIII. DISCOVERED CLUSTERS FOR 100 NODES TOPOLOGY

h_{max}	R_{thr}	Clusters	Cluster Heads	Q
1	∞	12	n13(2), n2(5), n96(2), n37(26), n7(20), n56(12), n25(3), n69(6), n67(2), n19(5), n83(2), n53(2)	0.448
1	1	3	n56(31), n37(43), n7(23)	0.559
2	∞	8	n2(2), n96(2), n37(30), n7(24), n56(25), n69(2), n19(4), n83(3)	0.445
2	1	4	n56(33), n37(37), n7(24), n96(2)	**0.572**
3	∞	4	n56(32), n37(37), n7(25), n96(2)	0.570
3	1	4	n56(32), n37(37), n7(25), n96(2)	0.570

As mentioned in the beginning of this section, for the evaluation of the SYSTAS we have used as baseline algorithms HAC from the family of hierarchical clustering algorithms and MCL from graph theory. In HAC each object forms a single object cluster. The algorithm continues by merging two clusters with the highest similarity at each step (e.g., choose two clusters with the smallest distance between them) [8]. We selected HAC over DBSCAN (a pure density based clustering algorithm [8]) due to nature of our approach. Although taking into account density, SYSTAS essentially operates similarly to HAC by progressively joining clusters according to a similarity metric until no further merging can take place. MCL partitions a graph by simulating multiple random walks inside the graph [19]. The main idea resides in the fact that by randomly visiting nodes, the number of times strongly connected nodes are visited is much higher than nodes with weak paths between them. In both schemes global knowledge is required (vector distance calculation and matrix multiplication). On the other hand, SYSTAS follows a distributed approach, which is based on local knowledge. Table IX shows that SYSTAS outperforms HAC in all cases, while in four out of eight graphs the same or higher modularity (Q) has been achieved, compared to MCL. We should point out that SYSTAS exploits only local view, whereas MCL needs global view of the graph.

TABLE IX. COMPARISON OF SYSTAS WITH HAC AND MCL

Nodes	Density	HAC	MCL	SYSTAS
20	0.236	0.266	0.386	**0.429**
30	0.128	0.322	0.534	**0.540**
40	0.198	0.398	**0.520**	0.468
50	0.128	0.292	**0.607**	0.572
80	0.186	0.433	**0.543**	0.419
100	0.116	0.562	0.558	**0.572**
128	0.142	0.537	**0.602**	0.595
135	0.084	0.435	**0.611**	**0.611**

V. CONCLUSIONS

In this paper, we have proposed SYSTAS a distributed algorithm catering for the discovery and formation of clusters in a network graph of fixed wireless nodes. SYSTAS is based on the density of the network graph, the interactions among nodes and the preferential attachment model. The local graph view is used, without having any indication about the appropriate or expected number of clusters. Through numerous experiments using various network topologies we showcased the validity, viability and merits of our work. Experimentation results show that the quality of SYSTAS clusters is comparable to competitive centralized solutions from graph theory. Our future work includes the extension of SYSTAS towards maintenance of formed clusters, while the communicational cost of SYSTAS and other solutions will be investigated.

REFERENCES

[1] Autonomic Communication, A. Vasilakos, M. Parashar, S. Karnouskos, and W. Pedrysz (editors), Springer Verlag, 2009

[2] G. Farhadi, A. Ito, "Group-Based Signaling and Access Control for Cellular Machine-to-Machine Communication," in Proc. IEEE VTC (Fall), pp. 1-6, 2013

[3] A. Hatoum, R. Langar, N. Aitsaadi, et al., "Cluster-based Resource Management in OFDMA Femtocell Networks with QoS Guarantees", IEEE Transactions on Vehicular Technology, vol. PP, no.99, 2013.

[4] M. E. J. Newman, "Modularity and community structure in networks", in Proc. Natl. Acad. Sci. USA 103, pp. 8577–8582, 2006.

[5] J.Y. Yu, P.H.J Chong, "A survey of clustering schemes for mobile ad hoc networks", IEEE Trans. Commun. Surveys & Tutorials, vol. 7, no. 1, pp. 32-48, 2005.

[6] E. Schaeffer, "Graph clustering", J. Computer Science Review, vol. 1, no. 1, pp. 27-64, 2007.

[7] M.E.J. Newman, M. Girvan, "Finding and evaluating community structure in networks", J. Phys. Rev. E 69, 026113, 2004.

[8] Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques. MA, USA:Morgan Kaufmann Publishers, 2012.

[9] C. Lin, and M. Gerla, "Adaptive clustering for mobile wireless networks", IEEE Journal on selected arrears in communications, vol. 15, pp. 1265-1275, September 1997.

[10] P. Basu, N. Khan, T.D.C. Little, "A mobility based metric for clustering in mobile ad hoc networks", in Proc. IEEE ICDCS, pp.413-418, 2001.

[11] A.K. Parekh, "Selecting routers in ad hoc wireless network", Proc. SBT/IEEE ITS, 1994.

[12] M. Chatterjee, S. K. Das, and D. Turgut, "An On-Demand Weighted Clustering Algorithm (WCA) for Ad hoc Networks," in Proc. IEEE Globecom, pp. 1697-701, 2000.

[13] S. Basagni, "Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks", in Proc. IEEE VTC, pp. 889-893, 1999.

[14] C.-C. Chiang, H.-K. Wu , W. Liu, et al., "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel", in Proc. IEEE SICON, 1997.

[15] S. Banerjee, S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks", IEEE INFOCOM, pp.1028-1037, 2001.

[16] S. Yi, J. Heo, Y. Cho, and J. Hong, "PEACH: power-efficient and adaptive clustering hierarchy protocol for wireless sensor networks", J. Computer Communications, vol. 30, no. 14-15, pp. 2842-2852, 2007.

[17] J. Congfeng, Y. Daomin, Z. Yinghui, "Towards Clustering Algorithms in Wireless Sensor Networks-A Survey," in Proc. IEEE WCNC, pp.1-6, 2009.

[18] S. AlMheiri, H. AlQamzi, "MANETs and VANETs clustering algorithms: A survey", In Proc. IEEE GCC Conference and Exhibition (GCCCE), pp. 1-6, 2015.

[19] S. Fortunato, "Community Detection in Graphs", Physics Reports, 2010

[20] A. Helmy, "Small worlds in wireless networks," IEEE Communications Letters, vol.7, no.10, pp.490-492, 2003.

[21] NS-3 network simulator, https://www.nsnam.org

[22] T. Schank and D. Wagner, "Approximating clustering coefficient and transitivity", J. Graph Algorithms and Appl., vol 9, no 2, pp. 265-275, Aug. 2005.