Distributed Clustering Algorithm in Dense Group-Based Ad Hoc Networks

R. Massin and C. J. Le Martret
Thales Communications & Security
Waveform Design Department
92622 Gennevilliers, France
Email: {raphael-a.massin, christophe.le_martret}@thalesgroup.com

P. Ciblat
Institut Mines-Telecom/Telecom ParisTech
Communications and Electronics Department
75013 Paris, France
Email: ciblat@telecom-paristech.fr

***Abstract*—For dense ad hoc networks, clustering is an appropriate strategy to efficiently organize the network. Moreover, public safety or military networks are structured through a hierarchical organization via operational groups. This organization has an impact on both the mobility of nodes which move in groups, and the data flow since the traffic is mainly intra-group. In this work we propose a novel distributed clustering algorithm suited to such networks, called Dynamic Clustering with Operational Groups (DCOG). This algorithm is designed in order to achieve the following properties: each cluster includes the highest possible number of members of some operational groups, and each cluster size is the closest possible to a given maximum. We first prove the theoretical convergence of DCOG and then compare by simulation its performance against five other clustering schemes from the literature. Our simulations show that DCOG leads to a lower end-to-end communication delay and offers a better stability to mobility.**

***Keywords*—Ad hoc network, distributed clustering, operational group, dense network, end-to-end delay, stability.**

I. Introduction

Ad hoc networks (formerly known as Packet Radio Networks) are well known to be suitable solutions for public safety or military deployments [1] (and references therein).

In order to implement practical large ad hoc networks, gathering nodes in clusters was proposed in the early 80's for networking purposes, and then in the 90's to sustain good Quality of Service (QoS) [1]. Since the radio resource is finite, implementation of clustering requires the spatial reuse of the resources using sets of (pseudo-)orthogonal codes or frequency channels (not addressed in this paper).

It was later theoretically proved in [2] that ad hoc networks do not scale with density since the throughput from one node to any destination goes to zero when the number of nodes N goes to infinity within a finite area region. The authors suggested that clustering the network in groups, connecting smaller numbers of users, may be of interest to mitigate this problem, yet without giving any specific solution. Similar results in [3] prove that cluster-based hierarchical routing introduces exponential savings in the amount of information to be stored and exchanged in a large scale ad hoc network. More recently in [4] clustering has been proposed in the context of device to device communications to improve energy efficiency, capacity and user fairness.

Public safety and military networks are organized into a hierarchical structure leading to the existence of *operational groups* (e.g., squad, section, etc.). In those networks, nodes exhibit group mobility behavior. Additionally, the traffic is strongly dependent on the network hierarchical organization, being mostly concentrated within operational groups. For these two reasons, the clustering solution may take into account operational group information in order to provide a better end-to-end QoS and to improve network stability. As demonstrated in [5] the trivial solution consisting in building one cluster per group is not acceptable, for instance it does not handle a node outside of the radio coverage of the other members of its group. Thus to cope with networks based on operational groups, new clustering solutions are required. For the sake of readability, in the sequel *group* refers to *operational group*.

Numerous distributed clustering schemes have been proposed in the literature. These algorithmic solutions first select Cluster Head (CH) nodes, and then the other nodes affiliate to them, leading to the different clusters. A weight is commonly associated with each node and the nodes with the highest weights in a neighborhood are selected to be the CH nodes. The weights can be the node identifier, the node degree, the remaining battery power, metrics related to radio measurements etc., or a combination of them [6]–[8]. To get the node weight, other solutions rely on the knowledge of nodes' location and speed, obtained thanks to for instance a GPS device [9].

Yet only a few papers consider group information for building the clusters. The authors in [10] introduce the type-based clustering algorithm (TCA). This clustering scheme associates a stability factor to each node and selects as CH the nodes that have the highest stability factor in a radio neighborhood. The stability factor takes group membership (identified thanks to the IP subnet of each node) into account. A limitation of TCA lies in the fact that two CH nodes cannot be neighbors. A direct consequence in dense networks is the formation of big clusters (with a large number of members). Two distributed clustering algorithms, GDMAC [7] and VOTE [8], are well-known to handle the size of clusters but they do not take into account the group structure. Extensions of these two algorithms were proposed in [11], taking advantage of the stability factor of TCA and thus taking into account group-based networks characteristics. A conclusion of this paper is that properly handling the cluster size and making sure that, when possible, nodes of the same group belong to the same cluster, is instrumental in achieving good performance. Because our previous extensions of GDMAC and VOTE are inherently limited by the algorithms from which they are derived (notably in relation to end-to-end delay and cluster structure stability), we have decided to follow a clean slate approach to design a new clustering solution suited to group-based networks.

In this paper we propose the new Dynamic Clustering with Operational Groups (DCOG) distributed clustering algorithm to be used in group-based ad hoc networks, and prove its convergence. A rigorous numerical evaluation against the five aforementioned clustering solutions is also conducted. These simulations show that DCOG outperforms all these alternative clustering schemes.

The paper is organized as follows. The system model is described in Section II. The DCOG clustering algorithm is described and its convergence proved in Section III. Then Section IV is devoted to simulation results. Finally, Section V concludes the paper.

II. System model

We consider a graph $\mathcal{G}$ defined by its set of nodes $\mathcal{V}$ and its set of edges $\mathcal{E}$. The number of nodes of $\mathcal{G}$ is $N := |\mathcal{V}|$. Two nodes are neighbors if there is an edge between them. Let $\mathcal{P}$ be the set of partitions of graph $\mathcal{G}$. A clustering solution leads to a partition p of $\mathcal{G}$. A partition p contains N_c clusters. The size of cluster $k \in \{1, \ldots, N_c\}$ is n_k^c. In this paper any cluster is subject to the following constraints: 1) it must be connected, 2) its size must be no greater than n_{max}, and 3) its diameter must be no greater than d_{max}. Two clusters $\mathcal{C}_k$ and $\mathcal{C}_\ell$ are neighbors if at least one member of $\mathcal{C}_k$ and one member of $\mathcal{C}_\ell$ are neighbors. The nodes are organized in groups. The set of groups $\mathcal{O}$ is defined as $\{\mathcal{O}_1, \ldots, \mathcal{O}_{N_g}\}$ with N_g the number of groups. Let us note n_ℓ^g the size of group $\mathcal{O}_\ell$ with $\ell \in \{1, \ldots, N_g\}$. If n_ℓ^g is constant then it is noted n^g. Each node belongs to only one group. Each cluster has members from a limited number of groups. Let $\mathcal{I}(\mathcal{C}_k)$ include the index of groups with at least one member in cluster $\mathcal{C}_k$. The number of members of group $\mathcal{O}_\ell$ in cluster $\mathcal{C}_k$ is $n_{\ell,k}^{gc}$.

Similar to [5] the user traffic depends on the group affiliation: the probability that one node communicates with a node of the same group is equal to $\alpha \in [0, 1]$ and thus the probability that one node communicates with a node in another group is equal to $1-\alpha$. Let us define $\pi_{j|i}$ as the probability that a source node i communicates with a destination node j. If $j = i$, then $\pi_{i|i} := 0$. When $j \neq i$ we have:

$$\pi_{j|i} := \begin{cases} \dfrac{\alpha}{n_\ell^g - 1} & \text{if } j \in \mathcal{O}_\ell, \\ \dfrac{1-\alpha}{N - n_\ell^g} & \text{otherwise,} \end{cases}$$

with $i \in \mathcal{O}_\ell$.

In clustered networks, inter-cluster communications can be implemented using a medium access control (MAC) different from the one used for intra-cluster communications. This is justified by the fact that within a cluster a resource allocator (RA) node can optimize the radio resource management (RRM) on behalf of the whole cluster. Conversely, inter-cluster RRM is done in a more distributed way (e.g., among the RA nodes of neighbor clusters) and is thus more difficult to optimize. Therefore, we reasonably assume that the performance (delay, data rate, etc.) of intra-cluster and inter-cluster communications are different. In this context, to measure the performance of a clustering solution resulting in a partition p of $\mathcal{G}$, we use the performance metric defined in [5] and denoted by $J(p)$.

This performance metric corresponds to the average cost (additive metric) of communication between all pairs of nodes in the network. For instance this value reflects the average end-to-end delay from all nodes to all nodes. It is defined as:

$$J(p) := \frac{1}{N} \sum_{(i,j) \in \mathcal{V}^2} \pi_{j|i} \cdot J_{i,j}(p), \qquad (1)$$

where the factor $1/N$ embodies the fact that all nodes i have equal probability to transmit. $J_{i,j}(p)$ is calculated as the cost of communication between source node i and destination node j along a shortest path. This shortest path is calculated taking into account the different costs $\tilde{\gamma}$ for inter-cluster and $\hat{\gamma}$ $(< \tilde{\gamma})$ for intra-cluster communications.

III. Distributed clustering algorithm

A. Clustering with Operational Group Algorithm

DCOG is an iterative algorithm. During an iteration clusters are modified through node cluster swaps. A node cluster swap (simply referred to as a swap) is defined by a set of nodes $\{u_i\}$ currently in a cluster $\mathcal{C}_k$ that leave $\mathcal{C}_k$ to join a neighboring cluster $\mathcal{C}_\ell$ $(k \neq \ell)$, subject to connectivity, cluster size and cluster diameter constraints. The DCOG algorithm associates a cost $c(\mathcal{C}_k) \in [0, 1]$ to each cluster $\mathcal{C}_k$, and calculates a swap gain $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell)$ for each possible swap. The gain $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell)$ corresponds to the difference between: 1) the gain associated with the arrival of nodes $\{u_i\}$ in $\mathcal{C}_\ell$, and 2) the loss associated with the removal of nodes $\{u_i\}$ from $\mathcal{C}_k$. Thus the gain can be expressed as:

$$\begin{aligned} g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell) \quad := \quad & \big[c(\mathcal{C}_\ell) - c(\mathcal{C}_\ell \cup \{u_i\})\big] \\ & -\big[c(\mathcal{C}_k \setminus \{u_i\}) - c(\mathcal{C}_k)\big]. \qquad (2) \end{aligned}$$

The DCOG algorithm performs swaps whose gains are strictly positive, and is run independently by all clusters. It is described in Table I. Each iteration of DCOG is split in two steps. The first step (lines 1-22) lets each cluster search for the swap with highest strictly positive gain. During step 2 (lines 23-35), nodes swap clusters as determined in step 1. Because DCOG is distributed, the decisions made during step 1 by some nodes $\{u_i\}$ of a cluster $\mathcal{C}_k$ to join a cluster $\mathcal{C}_\ell$ may no longer make sense, e.g., because step 2 of cluster $\mathcal{C}_\ell$ has been executed before step 2 of cluster $\mathcal{C}_k$, and $\mathcal{C}_\ell$ has been modified. Consequently, in step 2, before modifying clusters it is verified if the decided swaps still make sense. Also, line

30 checks that the destination cluster $\mathcal{C}_\ell$ is still ready to accept new nodes, e.g., if $\mathcal{C}_\ell$ is not currently involved in another swap operation. These two steps are repeated until there is no more possible swap.

```
Step 1
1 Set $\mathscr{M} = \emptyset$, $\mathcal{L} = \emptyset$ and $g = 0$
2 For each set of nodes $\{u_i\}$ members of the same group in $\mathcal{C}_k$:
3   If $\mathcal{C}_k \setminus \{u_i\}$ is not connected, continue to line 2. End If.
4   If $d(\mathcal{C}_k \setminus \{u_i\}) > d_{max}$, continue to line 2. End If.
5   For each cluster $\mathcal{C}_\ell$ neighbor of cluster $\mathcal{C}_k$:
6       If $|\mathcal{C}_\ell \cup \{u_i\}| > n_{max}$, continue to line 5. End If.
7       If $\mathcal{C}_\ell \cup \{u_i\}$ is disconnected, continue to line 5. End If.
8       If $d(\mathcal{C}_\ell \cup \{u_i\}) > d_{max}$, continue to line 5. End If.
9       Calculate $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell)$ thanks to (2).
10      If $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell) > g$:
11          Set $g = g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell)$.
12          Set $\mathcal{L} = \{\mathcal{C}_\ell\}$.
13      Else if $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell) = g$:
14          Set $\mathcal{L} = \mathcal{L} \cup \mathcal{C}_\ell$.
15      End If.
16  End For.
17  If $g = 0$ then nodes $\{u_i\}$ remain in cluster $\mathcal{C}_k$.
18  Else:
19      Choose randomly $\mathcal{C}_\ell \in \mathcal{L}$.
20      Set $\mathscr{M} = \mathscr{M} \cup (\{u_i\}, \mathcal{C}_\ell, g)$.
21  End If.
22 End For.
Step 2
23 For each $(\{u_i\}, \mathcal{C}_\ell, g) \in \mathscr{M}$ considered in decreasing $g$ values:
24  If $\mathcal{C}_k \setminus \{u_i\}$ is not connected, continue to line 23. End If.
25  If $\mathcal{C}_\ell = \emptyset$, continue to line 23. End If.
26  If $|\mathcal{C}_\ell \cup \{u_i\}| > n_{max}$, continue to line 23. End If.
27  If $d(\mathcal{C}_\ell \cup \{u_i\}) > d_{max}$, continue to line 23. End If.
28  Calculate $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell)$ thanks to (2).
29  If $g(\{u_i\}, \mathcal{C}_k, \mathcal{C}_\ell) \leq 0$, continue to line 23. End If.
30  If $\mathcal{C}_\ell$ is available:
31      Set $\mathcal{C}_\ell = \mathcal{C}_\ell \cup \{u_i\}$.
32      Set $\mathcal{C}_k = \mathcal{C}_k \setminus \{u_i\}$.
33      Exit For loop.
34  End If.
35 End For.
```

TABLE I: Dynamic clustering algorithm applied to cluster $\mathcal{C}_k$.

DCOG is initialized with all nodes forming their own singleton cluster (i.e., each node corresponds to a cluster).

In each cluster DCOG can be executed using one of the two following approaches (not detailed in this paper). First a member of the cluster (that can be called the CH) is in charge of running it. In the second approach the cluster members coordinate using a distributed protocol.

Fig. 1 illustrates a swap search by DCOG within a 16-node network organized into four clusters, in which all nodes are in range. The group of each node is identified by its shape. Clusters boundaries are indicated by solid lines. During the execution of DCOG, the source cluster evaluates the swap gain induced by the two nodes from the circle group joining each of the neighboring clusters. Among these three possible swaps, it is the one inducing the maximal strictly positive gain which is selected, i.e., the one where the two nodes join C_2.

B. Properties of DCOG

In this section we prove the convergence of DCOG to a stable partition. We now define the concept of a stable partition:

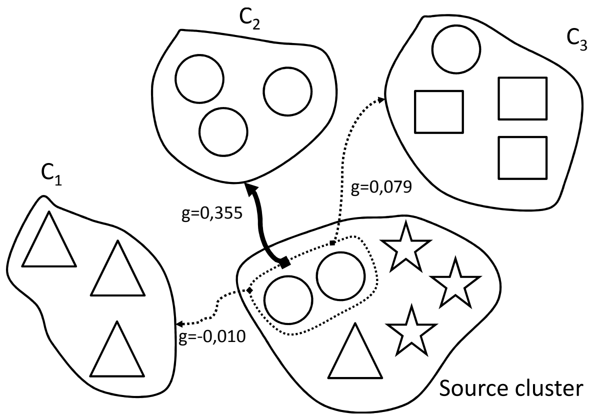

Fig. 1: Illustration of swap search and associated gains during DCOG operation in cluster source for its members belonging to the circle group.

Definition 1: A partition is stable if no swap induces a strictly positive gain.

Since the network is initialized by singleton clusters, DCOG must not be blocked in this state and must allow the merge of two clusters. Thus the following property must be fulfilled:

Property 1: The cluster cost c satisfies $g(\mathcal{C}_k, \mathcal{C}_k, \mathcal{C}_\ell) > 0$ for any clusters $\mathcal{C}_k$ and $\mathcal{C}_\ell$.

Let us now define the network cost function ψ as the sum of the clusters costs for a given partition p:

Definition 2: The network cost function ψ is defined as:

$$\psi : \mathcal{P} \mapsto \mathbb{R}$$
$$p \mapsto \psi(p) := \sum_{k=1}^{N_c} c(\mathcal{C}_k).$$

The key property of the function ψ is reported hereafter:

Property 2: If between time t_n and t_{n+1}, nodes $\{u_i\}$ have left cluster $\mathcal{C}_k$ to join cluster $\mathcal{C}_\ell$, the following equality holds:

$$\psi(p(t_{n+1})) = \psi(p(t_n)) - g(\{u_i\}, \mathcal{C}_k(t_n), \mathcal{C}_\ell(t_n)).$$

where $p(t)$ is the partition at time t.

Proof At time t_n we have:

$$\psi(p(t_n)) = c(\mathcal{C}_k(t_n)) + c(\mathcal{C}_\ell(t_n)) + \sum_{m \neq k, m \neq \ell} c(\mathcal{C}_m(t_n)).$$

At time t_{n+1} we have:

$$\begin{aligned}
\psi(p(t_{n+1})) &= c(\mathcal{C}_k(t_{n+1})) + c(\mathcal{C}_\ell(t_{n+1})) \\
&\qquad + \sum_{m \neq k, m \neq \ell} c(\mathcal{C}_m(t_{n+1})) \ , \\
&= c(\mathcal{C}_k(t_n) \setminus \{u_i\}) + c(\mathcal{C}_\ell(t_n) \cup \{u_i\}) \\
&\qquad + \sum_{m \neq k, m \neq \ell} c(\mathcal{C}_m(t_n)), \\
&= c(\mathcal{C}_k(t_n) \setminus \{u_i\}) + c(\mathcal{C}_\ell(t_n) \cup \{u_i\}) \\
&\qquad + \psi(p(t_n)) - c(\mathcal{C}_k(t_n)) - c(\mathcal{C}_\ell(t_n)), \\
&= \psi(p(t_n)) - g(\{u_i\}, \mathcal{C}_k(t_n), \mathcal{C}_\ell(t_n)).
\end{aligned}$$

∎

Since DCOG allows nodes to change clusters only if the associated gains are strictly positive, a direct consequence of property 2 is the following property:

Property 3: If a swap is performed between time t_n and t_{n+1}, the following inequality holds: $\psi(p(t_{n+1})) < \psi(p(t_n))$. A direct application of property 3 is the following theorem:

Theorem 1: For a fixed topology, the DCOG algorithm defined in table I converges to a stable partition in a finite number of iterations.

Proof Any swap selected in step 1 and executed in step 2 implies a decreasing of the network cost function ψ thanks to property 3. Consequently DCOG cannot choose a partition that has already been selected, which prevents loops. Furthermore since there are a finite number of partitions, the algorithm converges to a stable partition after a finite number of iterations. ∎

C. Cluster cost function

As introduced in section I, the cost function associated with a cluster must fulfill the following goals:

1) lead to the construction of clusters with the highest possible number of members from the same groups,
2) favor clusters whose size are maximal, i.e., equal to n_{max}.

As an example we propose here a cost function achieving these two goals:

$$c(\mathcal{C}_k) := 1 - \left[f_{n_{max}}(n_k^c) \cdot \epsilon + \frac{1-\epsilon}{n_{max}} \sum_{\ell \in \mathcal{I}(\mathcal{C}_k)} n_{\ell,k}^{gc} \cdot f_{n_\ell^g}(n_{\ell,k}^{gc}) \right], \tag{3}$$

with $\epsilon \in [0,1]$ selected to favor either goal 1 or goal 2, and function $f_n(m)$, $n \in \mathbb{N}$, defined from $[0,n]$ to $[0,1]$ verifying:

$$\begin{cases} f_n(0) = 0, \\ f_n(n) = 1. \end{cases}$$

To fulfill the goals 1 and 2, the function f_n must be strictly increasing. Moreover we choose f_n such that:

$$f_n(m+1) - f_n(m) < f_n(m+2) - f_n(m+1). \tag{4}$$

The rationale behind expression (4) (not demonstrated here) can be explained as follows. The term $f_{n_{max}}(n_k^c)$ in (3) ensures that from a cost perspective it is better to add nodes to a bigger cluster, thus achieving goal 1. Due to the $f_{n_\ell^g}(n_{\ell,k}^{gc})$ terms, it is better from a cost perspective to add members of a group to a cluster already including the highest number of members of this group. Also, thanks to the sum over $\mathcal{I}(\mathcal{C}_k)$, collecting together members of the same groups in the same cluster leads to a reduced cluster cost, allowing to reach goal 2. The term $n_{\ell,k}^{gc}$ allocates more weight to larger groups and $1/n_{max}$ is used as a normalizing factor. In this paper we have selected the function $f_n(m)$ defined as:

$$f_n(m) := \frac{1}{n(n+1)} \cdot m(m+1).$$

Note that the cost function defined by (3) satisfies Assumption 1. The proof is omitted due to space limitation.

D. Adaptation to node mobility

Because of node mobility, some clusters may no longer satisfy the connectivity or diameter constraint. In this case, the procedure detailed in Table II has to be added in order to split those clusters in sets gathering members of the same groups. In Table II, lines 4-7 find connected components and lines 9-15 find subcomponents satisfying the diameter constraint. After execution of this procedure, all clusters in the network satisfy the connectivity and diameter constraints.

```
1 If $\mathcal{C}_k$ is not connected or $d(\mathcal{C}_k) > d_{max}$:
2   Let $\mathscr{M} = \emptyset$.
3   For each set of nodes $\{u_i\}$ members of the same group in $\mathcal{C}_k$:
4       Let $\mathscr{C} = \{\{u_i\}\}$.
5       If $\{u_i\}$ is not connected:
6           Split $\{u_i\}$ into its $m$ connected components $\{\{u_i^1\},\dots,\{u_i^m\}\}$.
7           Set $\mathscr{C} = \{\{u_i^1\},\dots,\{u_i^m\}\}$.
8       End If.
9       For each set of nodes $\{u_i^j\}$ in $\mathscr{C}$:
10          If $d(\{u_i^j\}) > d_{max}$:
11              Split $\{u_i^j\}$ into its $m'$ subcomponents $\{\{v_i^1\},\dots,\{v_i^{m'}\}\}$ each one satisfying the diameter constraint.
12              Set $\mathscr{M} = \mathscr{M} \cup \{\{v_i^1\},\dots,\{v_i^{m'}\}\}$.
13          Else set $\mathscr{M} = \mathscr{M} \cup \{u_i^j\}$.
14          End If.
15      End For.
16  For each set of nodes $\{v_i\}$ in $\mathscr{M}$:
17      Create a new cluster with $\{v_i\}$.
18  End For.
19 End If.
```

TABLE II: Adaptation to node mobility for cluster $\mathcal{C}_k$.

Refering to line 11, a way to enforce diameter constraint when $d_{max} = 2$ is to split collection of nodes $\{u_i^j\}$ thanks to the following heuristic: 1) gather the node with highest degree and all its neighbors, and 2) build connected components with the remaining nodes. Note that the $\{v_i\} \in \mathscr{M}$ available after line 15 have a diameter lower or equal than 2 only if the algorithm in Table II is invoked often enough w.r.t. the node mobility.

IV. Simulation results

A. Reference clustering algorithms

The performance of DCOG is compared to five distributed clustering solutions. GDMAC [7] and VOTE [10] are respectively referred to as GDMAC-std and VOTE-std. We selected these protocols because they allow adapting the number of clusters to the network node density. We have introduced extensions of GDMAC and VOTE in [11] in order to take into account the group structure of the network. They are referred to as GDMAC-new1, GDMAC-new2, and VOTE-new. Note that the DCOG approach proposed in this paper does not need any CH node and is completely different from the ones proposed in [11].

B. Simulation setup

The simulated network has N nodes deployed in a square area whose side is 1500 distance units long. The range of a node is $r = 250$ distance units. The operational group size is set to the same value $n^g = 10$ for all groups. The group mobility model is defined as in [12]. In this model a virtual center moving with random speed and direction is associated to each group. Nodes from the same group move randomly such as never being farther than a specific distance from the virtual center of the group. In our simulation, this distance has been set to the node range r. If the virtual center of a group is closer than r from the deployment area boundary, then a new location is drawn for this virtual center to make sure all nodes of the group are always in this area. Simulations have been performed in static and mobile conditions.

As for the traffic parameters, we set $\alpha = 0.9$, meaning that 90% of the traffic is exchanged between members of the same groups. The inter-cluster RRM is considered to be twice as efficient as the inter-cluster RRM: we fix $\hat{\gamma} = 1$ and $\tilde{\gamma} = 2$.

The DCOG parameter n_{max} has been set to 20. In order to always favor the construction of clusters with the highest number of members from the same groups (summation in (3)), the value of ϵ has been set to $1/(1 + n_{max})^4$ (justification is omitted due to lack of place). Concerning the other clustering algorithms parameters, we use the values defined in [11].

To be independent from the performance of the radio access scheme, we used a custom simulator which splits time in rounds. For DCOG, each round is split in two parts during which the steps 1 and 2 of Table I are run: first step 1 is applied sequentially to all clusters and then step 2 is applied sequentially to all clusters. Concerning the other clustering algorithms, during a round all nodes run the algorithm independently: CH nodes maintenance is performed, and non-CH nodes select their CH. At the end of the round, all nodes are assumed to have perfectly learned the information about their neighbors that is required to run the selected clustering algorithm in the next round.

All simulations have been run over 100 random networks.

C. Performance in static networks

Simulations have been performed for various number of nodes N. Because the deployment area never changes, the density increases with N. For instance when N goes from 100 to 1000, the average degree (density) grows linearly from a moderate value 20 to a high value 150. Whatever the value of N, the number of simulation rounds is always chosen large enough in order to ensure the convergence to a stable partition. To evaluate the general performance of each clustering algorithm we use the following metrics:

- Number of rounds required to reach a stable partition.
- Cluster size, i.e., number of members per cluster. The target cluster size is n_{max}.
- Group cluster diversity (GCD), i.e., average per group number of clusters including at least one member of the group. This metric should have a low value, meaning that the members of a group tend to be in the same cluster.
- Application level performance measured using J as defined by (1).

1) Convergence: Table III provides the average number of rounds required to get a stable partition, for each clustering solution. DCOG convergence duration is low and increases slightly with the number of nodes in the network. The GDMAC variants are quicker to converge than DCOG, and their convergence time does not depend on the network size. The VOTE protocols need more and more time as the network size increases.

Nodes	DCOG	GDMAC -new1	GDMAC -new2	VOTE -std	VOTE -new
100	5.1	3.2	4.3	6.9	6.4
500	6.8	3.1	4.3	47.7	26.5
1000	8.3	3.0	4.0	135	82.7

TABLE III: Number of rounds required to get a stable partition.

2) Cluster size: Thanks to the appropriate choice of design parameters for GDMAC and VOTE (see [11]), GDMAC and VOTE based algorithms yield clusters whose maximum size is almost the same, i.e., 20. In contrast, as illustrated by Fig. 2 their average cluster sizes are very different. The GDMAC based algorithms lead to an average cluster size much smaller than n_{max}, meaning that these protocols build clusters with highly different sizes. Concerning DCOG, the cluster size is close to 11 for 100 nodes, and increases regularly with the number of nodes in the network, up to around 17 for 1000 nodes.

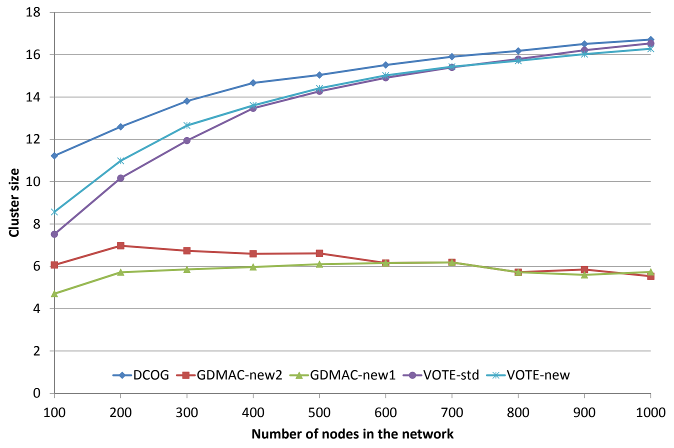

Fig. 2: Average cluster size vs. N in static networks.

3) Group cluster diversity: A low value of this metric indicates that group members are gathered in few clusters, which means that a majority of low cost intra-cluster links and a minority of high cost inter-cluster are used during intra-group communications. As highlighted by Fig. 3, DCOG achieves a GCD which is always about 1 independently of the number of nodes. This means that DCOG succeeds in collecting nearly all members of the same groups in the same cluster. The second best solution is GDMAC-new2 whose GCD is in the interval $[2.5, 4.6]$, which is much worse than the result obtained thanks to DCOG.

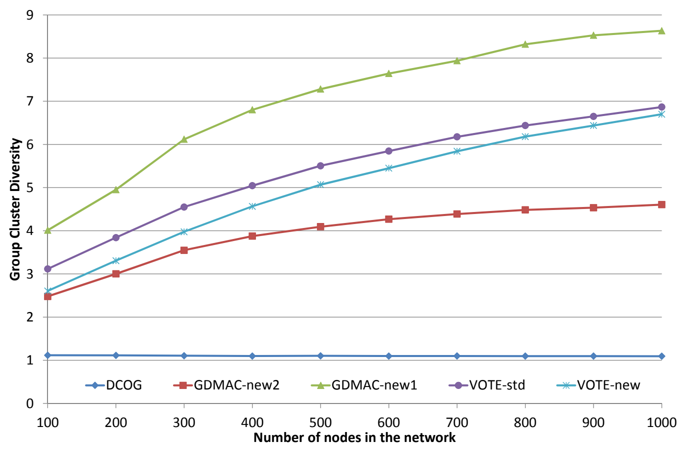

Fig. 3: Group cluster diversity vs. N in static networks.

4) Application level performance: As discussed in section II the J values are associated to the average communication cost between all pairs of nodes and can be associated to average end-to-end delay. Yet these values are dimensionless and to convert them in seconds, some hypothesis must be taken about the MAC. In the example of a TDMA based MAC in which the intra-cluster delay to send data is equal to one 10ms MAC frame, multiplying the J values by 0.01 provides the average end-to-end number of seconds required to transmit a unit of a sender's data to any destination.

The values of J are plotted in Fig. 4. Average values are plotted for all clustering solutions, including the one consisting in 1 singleton cluster per node. This last clustering solution is the one with the highest number of inter-cluster links, and is thus the worst from the point of view of J. A boxplot is drawn to detail the statistical results associated with DCOG. From bottom to top the different bars show the lowest value, 5th centile, first, median and third quartiles, 95th centile, and highest value. The average value is identified by a diamond.

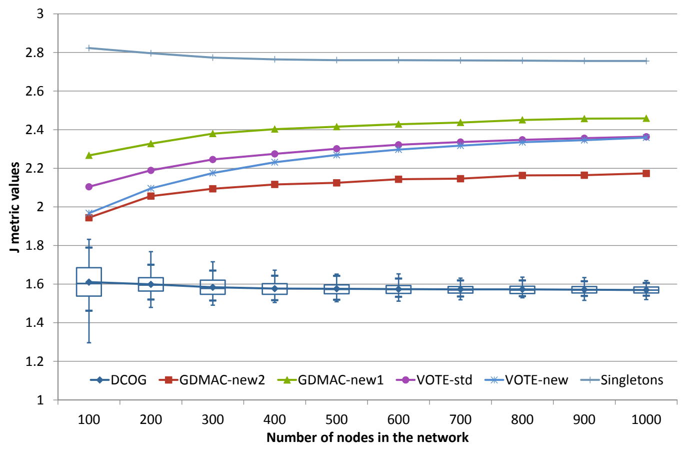

Fig. 4: J values (end-to-end delay) vs. N in static networks.

DCOG achieves the lowest (and consequently the best) J values. This is a consequence of the low GCD achieved by DCOG: J decreases when the amount of inter-cluster traffic decreases, which is the case when the GCD also decreases. Fig. 4 shows that when the number of nodes in the network increases, the performance of the five reference clustering solutions degrades slightly when DCOG performance first increases then remains constant.

Table IV provides J values averaged on all network sizes. DCOG brings a 25% delay decrease when compared to the second best clustering solution, i.e., GDMAC-new2.

DCOG	GDMAC-new1	GDMAC-new2	VOTE-std	VOTE-new
1.58	2.40	2.11	2.28	2.24

TABLE IV: J average values on all network sizes.

5) Additional insights on DCOG behavior: Different group size n^g have been simulated, keeping constant the maximal cluster size $n_{max} = 20$ and leading to different n^g/n_{max} ratios. To ensure that all groups are complete, the number of nodes in the network has been slightly adjusted, leading to the following $(N, n^g, n^g/n_{max})$ 3-tuples:

N	n^g	n^g/n_{max}
105	15	1.3
100	10	2
104	8	2.5
102	6	3.3

TABLE V: N, n_g values for various n^g/n_{max} ratios.

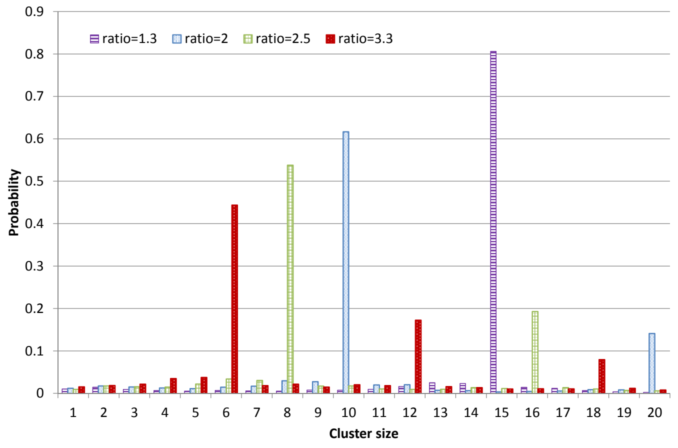

Fig. 5: Cluster size distribution vs. n_{max}/n^g ratio.

The distribution of cluster size for all ratios is plotted in Fig. 5. When the ratio is 1.3, 80.6% clusters built by DCOG are as large as a group, and 15.5% are smaller. When the ratio is equal to 2 or 2.5 then DCOG mainly builds cluster whose size is either equal to the group size or equal to twice the group size. This behavior is justified by objective 1 of the cost function, which aims to the construction of clusters with the highest possible number of members from the same groups. For the same reason, when the ratio is equal to 3.3, DCOG mainly builds cluster whose size is once, twice or thrice the group size. When the ratio is greater than two, the probability to build clusters including a whole group is higher than building clusters including two (or three) full groups. To understand this, recall that the nodes are deployed randomly in groups. There is no guarantee that the members of different groups can be gathered in a cluster satisfying the connectivity, size and diameter constraints. The GCD metric values (not shown here due to lack of place) confirm that the clusters built by DCOG usually include full groups.

D. Performance in mobile networks

To assess performance in mobility, simulations have been run for $N = 100$ nodes and varying maximum node speed in $\{1, 3, 5, 7.5, 10, 12.5, 15, 20\}$ distance units per simulation round. To assess the stability of the cluster structure built by the different clustering solutions, we define the ratio of the number of simulation rounds when no cluster was modified over the simulation time (set to 1000 rounds). High values of this ratio indicate stable clustering algorithms, and values close to 0 indicate unstable networks. The plot of this metric in Fig. 6 shows that VOTE-std, VOTE-new and GDMAC-std are highly unstable, having less than 50% stability as soon as the node maximum speed attains 3 distance units per simulation round. Thanks to their small cluster size and their use of group information GDMAC-new1 and GDMAC-new2 succeed in keeping a stability greater than 50% up to a 5 distance units per simulation round. DCOG achieves by far the best stability, ensuring more than 60% stability even when the node maximum speed is as high as 20 distance units per simulation round. A first rationale for this good performance is the quick convergence property of DCOG, as shown in section IV-C1. This is also achieved thanks to a GCD whose value is always very close to 1, indicating that all members of a group are usually in the same cluster. Combined with the fact that nodes follow a group mobility pattern, a DCOG cluster is very stable.

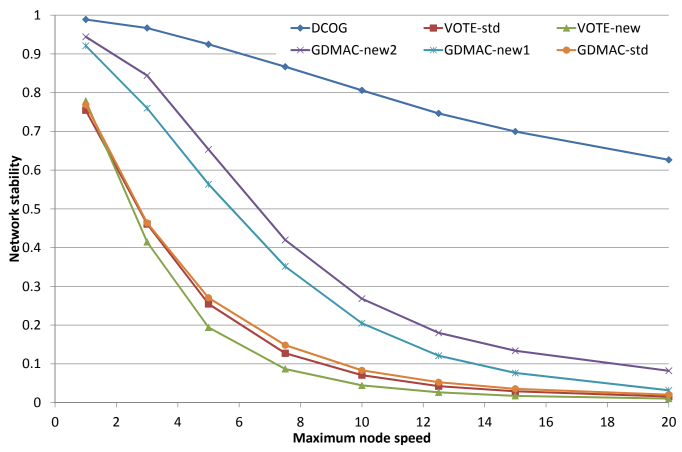

Fig. 6: Network stability vs. node speed in mobile networks.

V. Conclusion

In this paper we have proposed the novel distributed clustering algorithm DCOG suited to group-based ad hoc networks such as public safety or military networks. We have proved the theoretical convergence of DCOG. Our simulations in static and mobile conditions have shown that thanks to DCOG, it is now possible to operate large scale dense networks based on groups and ensure good performance in relation to end-to-end delay and network stability. The comparison with existing solutions has shown that our solution outperforms the ones from the literature.

Another interesting feature of DCOG is its small number of parameters compared to other solutions, such as GDMAC, which makes it easier to use.

Finally, in order to further improve the user QoS, notably from a throughput perspective, we are currently working on algorithms similar to DCOG that take into account the link qualities.

References

[1] C. Lin and M. Gerla, “Adaptive clustering for mobile wireless networks,” *Selected Areas in Communications, IEEE Journal on*, vol. 15, no. 7, pp. 1265–1275, Sep 1997.

[2] P. Gupta and P. Kumar, “The capacity of wireless networks,” *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, Mar 2000.

[3] J. Sucec and I. Marsic, “Hierarchical routing overhead in mobile ad hoc networks,” *IEEE Transactions on Mobile Computing*, vol. 3, no. 1, pp. 46–56, Mar. 2004.

[4] A. Asadi and V. Mancuso, “Dronee: Dual-radio opportunistic networking for energy efficiency,” *Computer Commnunications*, vol. 50, pp. 41–52, Sep. 2014.

[5] R. Massin, C. J. Le Martret, and P. Ciblat, “A network cost function for clustered ad hoc networks: application to group based systems,” in *Proc. IEEE PIMRC*, Washington D.C. (USA), Sep. 2014.

[6] M. Gerla and J. Tsai, “Multicluster, mobile, multimedia radio network,” *Journal of Wireless Networks*, vol. 1, no. 3, pp. 255–265, Jul. 1995.

[7] R. Ghosh and S.Basagni, “Mitigating the impact of node mobility on ad hoc clustering,” *Wireless Communications and Mobile Computing*, vol. 8, no. 3, pp. 295–308, Mar. 2008.

[8] F. Li, S. Zhang, X. Wang, X. Xue, and H. Shen, “Vote-based clustering algorithm in mobile ad hoc networks,” in *Proc. ICON*, Busan, South Korea, Feb. 2004.

[9] Y. Zhang and J. Ng, “A distributed group mobility adaptive clustering algorithm for mobile ad hoc networks,” in *Proc. IEEE ICC*, Beijing, China, May 2008, pp. 3161–3165.

[10] H. Wu, Z. Zhong, and L. Hanzo, “A cluster-head selection and update algorithm for ad hoc networks,” in *Proc. IEEE GLOBECOM*, Dec. 2010, pp. 1–5.

[11] R. Massin, C. J. Le Martret, and P. Ciblat, “Distributed clustering algorithms in group-based ad hoc networks,” in *Proc. EURASIP EUSIPCO (http://perso.telecom-paristech.fr/ ∼ciblat/docs_recherche/congres/ eusipco2015.pdf)*, 2015.

[12] X. Hong, M. Gerla, G. Pei, and C. Chiang, “A group mobility model for ad hoc wireless network,” in *ACM Inter. Workshop on Modeling and Simulation of Wireless and Mobile Systems (MSWiM)*, Aug. 1999, pp. 53–60.