VANET in Eyes of Hierarchical Topology

Yair Allouche and Michael Segal
Communication Systems Engineering Department
Ben-Gurion University of the Negev, Beer-Sheva, Israel

In this paper we will show a self-organizing hierarchical topology that can serve as infrastructure for efficient and reliable safety related communication aiming to minimize the interference between network participants. We will show how reliability, fairness, and efficiency can be achieved in our presented D-CUT algorithm. Our solution addresses these challenges in a local, distributed manner by exploiting the vehicle proximity map, needed from a safety point of view, as the building block for constructing the hierarchical topology. The D-CUT algorithm produces a geographically optimized clustering of the network, by grouping dense and consecutive nodes into clusters which are separated by maximally possible gaps. This type of clustering allows strong connections between cluster members and reduces the inter cluster interference. In addition, we present the primitives for interference aware communication system design, based on the awareness of vehicles to their surrounding vehicle proximity map partitioned into geographically optimized clusters. We present theoretically provable bounds demonstrating the ability of the algorithm to deal with dynamic nature of the VANET environment supported by simulation results.

***Keywords*— beacon dissemination, distributed algorithm, optimal clustering assignment, self-organizing topology.**

I. INTRODUCTION

Vehicular ad-hoc network (VANET) is a promising branch of traditional MANET. VANET is designed to provide wireless communication between vehicles and between vehicles and nearby roadside equipment. This communication intends to improve both safety and comfort on the road. To this end, the US FCC has allocated 75 MHz of the spectrum in the 5.9 GHz band for Direct Short Range Communication (DSRC). VANET have a number of difficulties regarding to the traditional MANET. Due to the mobile nature of VANET nodes, configuration in this environment is always changing, especially when considering short range communication, where links may appear and disappear very quickly. Furthermore, this highly dynamic configuration nature results with constantly changing node density. On the other hand VANET has some inherent advantages over the traditional MANET. It is generally assumed that vehicles will be aware to their own geographical position (which can be obtained, for example, by a Global Positioning Satellite). In addition, vehicles in a VANET environment move in an organized fashion within the constraints of two-dimensional traffic flow.

In order to serve as the infrastructure for safety applications, highly reliable, real-time communication is required. This means that packets must be successfully delivered before a certain deadline. Meeting the tight delay restriction based on the unreliable wireless medium, combined with the dynamic VANET environment, becomes a very challenging task. When considering event driven dissemination this tight delay restriction drop to 0.1 second as in the Emergency dissemination Emergency Electronic Brake Lights [1].

A key component of safety applications are the periodic beacon messages, providing nodes with an updated and accurate *vehicle proximity map* of their surroundings. Based on this map, safety applications - usually refer to as Cooperative Awareness applications – can be used for accident prevention by informing drivers about evolving hazardous situations. In addition, an accurate vehicle proximity map can facilitate other essential multi-layer objectives such as optimized geographic oriented forwarding [2] and addressing methodologies. From routing point of view, high awareness can be very beneficial in terms of route discovery, end-to-end delay, and number of retransmissions [3]. Torrent-Moreno et al. [4] propose a transmit power control method, based on the vehicles' location proximity, to control the load of beacon messages. To be used as a reliable infrastructure for safety applications, the surrounding vehicle proximity map should be as broad and accurate as possible. Hence, while considering a fully deployed high-density vehicular scenario combined with the dynamic topology of the vehicular environment (e.g. a free highway), creating a broad, and accurate vehicle proximity map becomes challenging. Such an accurate estimation in a dynamic environment requires a high transmission frequency of beacon messages, in broadcast fashion, from numerous nearby vehicles; which in turn, resulting in a high data load on the channel which can escalating to broadcast storm.

To provide reliability, the medium access issues need to be addressed. Medium access scheme for VANET must corroborate different types of data traffic. In some applications, like information services, communications are based on unicast traffic. Many applications, as warning messages dissemination, or the exchange of information regarding nearby traffic situation, are based on broadcast transmission. This is because such messages' content can be useful for all vehicles around.

The IEEE 802.11p standard, designed for vehicular ad hoc networks (VANET) uses the CSMA-CA as its MAC method, despite the fact that it suffering from three well-known problems: First, when considering broadcast transmission, RTS (Request To Send)/CTS (Clear To Send) mechanism is infeasible. In such case, the CSMA provides no means to solve the *hidden station problem*, which can lead in a heavy traffic load to a high rate of packet collisions. In unicast transmission the RTS/CTS mechanism, which solves the *hidden station problem*, can be applied. However, this mechanism raises a new problem, known as the *blocked station problem*, when viable transmissions are disallowed. Finally, the CSMA/CA can be resulted under high, though realistic, traffic load with unacceptable channel access delays [5], and therefore, unable to support real-time communications.

In order to provide reliable topology links which support real time deadlines, we prefer using a channelization based approach (as TDMA) rather than the current CSMA/CA contention based approach. In this manner we insure every user receives a fair, time bounded access to the medium. A proposed medium access scheme in this direction is the Space Division Multiple Access (SDMA) [6]. In this approach a one-to-one map between the space divisions and the bandwidth divisions is used so within each bandwidth divisions a TDMA scheme is mapped. This scheme provides users with collision-free access to the communication medium, and guarantees delay-bounded communication in real-

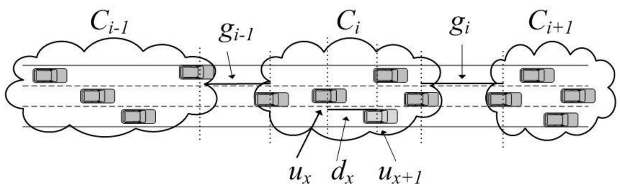

Fig. 1. The model basic notations.

time. However, this mapping is likely to be impractical in a real system [7], mainly because the lacking of flexible adaptation to the scalable and dynamic vehicular environment. In this work, we aim to obtain a spatial based TDMA[1] scheme, but with adaptivity required to fit the scalability and the dynamic of the VANET environment.

One of the main approaches to optimize the communication within the network is to organize it in a hierarchical topology fashion. The benefits of hierarchical topology are well known, and include routing [8], rebroadcasting [9], increasing security [10], and in addition, provide the flexibility required to attain Quality of Service. In this paper, we suggest self-organizing hierarchical topology to serve as the infrastructure for inter-vehicles safety related communication. The key design goal of this topology is to optimize the efficiency and reliability of the topology links in order to meet the highly demanding communication constraints described above. Moreover, for a hierarchical topology to be feasible for VANET, it must handle the challenging dynamic behaviors of vehicular networks. In addition, constructing the topology by designated messages in such dynamic scenarios can lead to significant overhead. The Distributed Construct Underlying Topology (D-CUT) algorithm addresses these challenges in a local, distributed manner by leveraging the two main qualities of VANET environment described above – namely, the vehicles' location awareness, and vehicles' organized movement fashion – as the building block for constructing the hierarchical topology. Broadly speaking, the D-CUT algorithm partitions the vehicle proximity map into road sections where each section contains geographically optimized clusters. Given the new location of the nearby vehicles attained by the beacon dissemination process, the algorithm updates the partitioning according to the most recent topological changes while aiming to maintain geographically optimized clusters.

The rest of this paper is organized as follows. In Section II, we summarize other approaches for building hierarchical topology in VANET. In Section III we present an interference-aware system design and the clustering strategy, and according to both, we give a formal definition of the clustering optimization problem considered in this paper. Then, in Section IV, we describe the D-CUT algorithm and in Section V, we show theoretically provable bounds for the algorithm performance. In Section VI, we show a simulation study which supports our analytical results. Finally, Section VII discusses the ability of the D-CUT algorithm to serve as the infrastructure for safety application, under the very dynamic nature of the VANET environment.

II. Related Work

There are several, well known, clustering mechanisms for mobile ad hoc networks, see for instance [11]-[15]. One approach for clustering formation in VANET is by adopting MANET algorithm according to the characteristics of the vehicular environment. In [16], Fan et al. analyzes the obtained network structure taking direction, mobility features, and leadership duration into consideration. Another approach for cluster formation, presented in [8] and [17], is to distribute the state of nodes (undecided, member, gateway or cluster-head) on the regular transmission of beacons. Each node chooses its appropriate state according to the state of the nodes nearby. Both approaches try to maximize the clustering stability in to avoid the overhead caused by clustering formation designated messages. However, they are not taking advantage of the vehicle proximity map required by safety considerations. In several papers, see [9],[10],[18], it is demonstrated how to dissect the roads into predetermined area cells which define clusters. This method does not take into account the placement of the vehicles on the road. As a result, unbalanced clusters can be produced and dense vehicles can be partitioned into different clusters. An additional drawback of this method is the requirement for preloaded dissection of the area map into cells.

III. System Model and Problem Defenition

In this section we describe the geographic clustering optimization problem. Alongside with optimizing the clustering according to the following suggested communication system design, we seek for a clustering scheme which can handle the environmental conditions of VANET.

A. Model

We are given a network N with n ordered nodes $U=\{u_1,u_2,\ldots,u_n\}$ that are moving along a road from left to right (see Fig. 1). Instead of denoting the location of nodes explicitly, we use their relative locations. Let us denote by $D=\{d_0,d_1,\ldots,d_n\}$ the set of inter-distances such that d_i is the inter-distance between u_i and u_{i+1}. The inter-distances d_0,d_n denote the space at the edge of the model and are set to ∞. In some cases, we will need to observe subsets of the sets U and D. Hence, let $U(d_i,d_j)$ be the subset of U framed by the inter-distances d_i,d_j, i.e., $U(d_i,d_j)=\{u_{i+1},u_{i+2},\ldots,u_j\}$. Similarly, let $D(d_i,d_j)$ be the D subset: $\{d_{i+1},d_{i+1},\ldots,d_{j-1}\}$. To indicate that one or both of the endpoints is to be included in the set, we substitute a square bracket for the corresponding parenthesis, e.g $D[d_i,d_j)=\{d_i,d_{i+1},\ldots,d_{j-1}\}$. In addition, let us denote by $S=\{C_1,C_2,\ldots,C_m\}$ the set of clusters such that C_i is a set of consecutive nodes that forms the i'th cluster in set, and m is the number of clusters in the model. Accordingly, let $G=\{g_0,g_1,\ldots,g_m\}$ be the set of inter-cluster gaps, such that g_i represent the inter-distance located between the clusters C_i and C_{i+1}, and g_0,g_m represent the end-points d_0,d_n, respectively. Notice that according to the above notations $C_i=U(g_{i-1},g_i)$.

Remark: The D-CUT algorithm is based on comparing the length of inter-distances and gaps. In order to deal with ties in gap or inter-distance comparisons, the gap/inter-distance having the smaller index wins.

B. Interference aware communication system design

We consider the following, two-level general hierarchy (see Fig. 2). First, the network is split into clusters of adjacent vehicles which cover the entire vehicle population. Each cluster contains a designated vehicle referred to as the *clusterhead* which acts as a relay point of communication for the cluster members. Thus, the first level of the topology consists of links between each clusterhead and its cluster members (i.e., a star topology). On top of these *intra-cluster links*, clusterheads can aggregate and disseminate information from and to its cluster members in a centralized manner. To prevent clusterheads from becoming bottlenecks of their clusters, we limit the number of members within each cluster. The second level of the topology consists of

[1] We remark that our scheme is compatible with any other multiple access schemes such as FDMA, CDMA, etc.

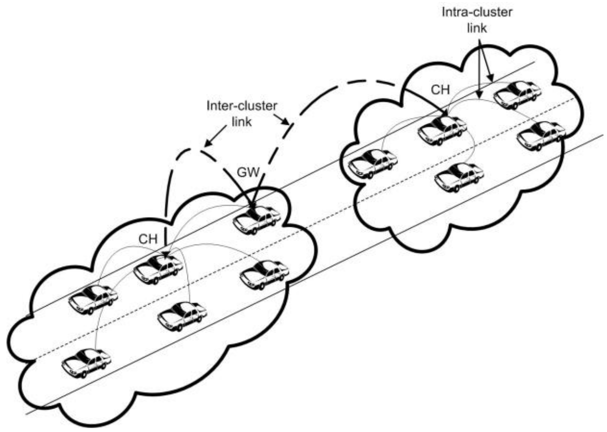

Fig. 2. The hierarchical network topology is created by grouping sets of sequential nodes into clusters. At the intra cluster level, the cluster members of each cluster are linked to a designated clusterhead (CH). At the inter-cluster level, CH's are linked, if needed via gatetways (GW's), to their adjacent clusters.

the *inter-cluster links* which connect between adjacent clusterheads. When one clusterhead is not in transmission range of its adjacent clusterhead, communication takes place through intermediate nodes referred to as *gateways* (if connectivity exists). Subsequently, the clusterheads and gateways generate the backbone infrastructure of the network. This general topology provides, as mentioned above, multi-layer benefits; however, to receive reliability and efficiency we aim to design a communication system which provides an optimal interference aware channel access scheme.

As in [10], we suggest spatial based TDMA approach. In this work we design communication system which intends to deal with VANET dynamicity. For this purpose, we group adjacent channel contesters into not overlapping clusters, and synchronize their channel access according to their current location within the cluster (which can be derived from the vehicle map). A feasible TDMA scheme requires limiting the cluster size. The intra-cluster synchronization prevents interference among cluster members (i.e., *intra-cluster* interference is avoided). Bandwidth efficiency is achieved by bandwidth reuse among clusters. However, this bandwidth reuse causes interference from adjacent clusters as vehicles from adjacent clusters are assigned with the same time slot.

In order to minimize this interference, we demand the clusters to be as dense as possible, and far apart from each other. Furthermore, to optimize the capacity, i.e., to maximize the benefit of the spatial reuse, we aim to increase the clusters size up to their limits.

C. Clustering scheme strategy

To fit the environmental conditions of VANET, we identify the following clustering scheme strategy requirements: self-organization, locality and stability. In order to find the balanced way between stability and adaptation, we seek for road dissection strategy which follows the trends rather than a single vehicle's behavior. Hence, we propose dissecting the road by prioritizing the dissection candidate - the inter-distances - according to their size. By this simple yet meaningful strategy we gain stability by disregarding small scale (intra-cluster) reconfiguration changes.

D. Problem definition

At the cluster level, we look for star topology which allows one hop aggregation/dissemination. This objective requires the existence of at least one clusterhead candidate that is in the transmission range of all cluster members. Our second objective is to limit the cluster size for the two reasons mentioned above: (i) preventing clusterhead to become a bottleneck, and (ii) feasible medium access allocation. Each cluster that fulfills these objectives will be defined as a *valid cluster.*

Definition 1. The Boolean objective function F receives two inter-distances d_i,d_j, which form the subset $U(d_i,d_j)$, and returns true if and only if this subset satisfies the following two conditions:

- Exist a clusterhead candidate $u' \in U(d_i,d_j)$, where $dist(u,u') \leq R_{max}$ for all $u \in U(d_i,d_j)$, when $dist(u,u')$ denotes the Euclidian distance between u and u', and R_{max} denote the maximal transmission range.
- $k \leq k_{max}$, where $k=|U(d_i,d_j)|$.

We note here that the D-CUT algorithm properties are preserved for any objective function which satisfies:

$$If\ (U(d_i,d_j) \subseteq U(d_x,d_y)\ \&\ F(d_x,d_y)=true) \rightarrow F(d_i,d_j)=true$$

(e.g., an objective function which allows some p hops connection between clusterhead and its cluster members).

Based on this definition, we define a *valid solution* for the network N as follows.

Definition 2. Given the network N with the set of nodes $\{u_1,u_2,...,u_n\}$, the *Clustering Assignment* (*CA*) is a function assigning each node in the network to a cluster; for which, the received cluster set S fulfils: (i) every cluster in S satisfies the objective function; (ii) each node belongs to only one cluster; (iii) the union of all clusters in S contains all nodes in the network.

We group consecutive nodes into clusters which are separated by maximally possible gaps. This type of clustering allows strong connection between cluster members and reduces the inter cluster interference. Having *fairness* design goal in mind, we consider a *Max-Min* gap objective as the first objective of the optimization problem. In addition, in order to enhance the advantages of the hierarchical topology, that is, to maximize the spatial reuse and minimize the network diameter, we consider minimizing the number of clusters in the network as the second objective of the optimization problem.

Let $V(N)$ be the set of all possible clustering assignments of the network. Now we are ready to formally define the optimal geographical clustering objectives described above:

- Objective 1: $\min_{i \in [1...m-1]} g_i$ is maximized over all solutions from $V(N)$.
- Objective 2: The number of clusters is minimized over all solutions from $V(N)$. Let us denote by S_{opt} the optimal solution such that $|S_{opt}|=\min_{S \in V(N)} |S|$.

The D-CUT algorithm produces the *Geographically Optimal Clustering Assignment (GOCA)* with the resulted cluster set S' which meets Objective 1 and approximates Objective 2 by a factor of 3 (i.e., $|S'| \leq 3 \cdot |S_{opt}|$).

IV. The D-CUT algorithm

In this section we present the Distributed Construct Underlying Topology (D-CUT) algorithm. The D-CUT algorithm is an iterative algorithm, which strives to discover and maintain a geographically optimal clustering for the current network configuration. At each iteration, the D-CUT algorithm gets a snapshot of the local vehicle proximity map and updates the clustering solution according to the changes in the network configuration. The D-CUT algorithm is stand only on top of a strong connection between adjacent clusters. Based on these connections, each cluster obtains information about last iteration *CA* of its adjacent clusters, and their updated location. Both can be obtained by the beacon dissemination process (which described in the following section).

Fig. 3 presents the D-CUT algorithm run by vehicles which belong to the cluster C_i. The algorithm use as input the last iteration

CA of its vicinity (C_{i-1},$C_{i,}$,C_{i+1}) with updated members' locations. As output, the algorithm produces the new *CA* of C_i members. The algorithm is based on split and join operations between adjacent clusters, and can be logically partitioned into 3 parts according to the different clustering reorganization procedures. (i) The Split-Join procedure (stages 1,2) enables two adjacent clusters to greedily replace the inter-cluster gap trapped between them, by larger gaps. The algorithm tries to replace first (stage 1) cluster's left inter-cluster gap, and then its right inter-cluster gap (stage 2). (ii) By the Split procedure (stage 3), invalid or discontinued cluster is Split. (iii) Finally, the Join procedure (stages 4-5) enables two adjacent clusters to greedily remove the inter-cluster gap located between them by Join operation (Objective 2). To guarantee coordinated join operation, the join conditions check whether the cluster to be joined with is not going to be split at the same iteration.

Definition 3. The *Max-Min Inter-Distance Pair* (*MMIDP*) is a function that finds an inter-distances pair (denoted by $(d^{(l)},d^{(r)})$) from adjacent clusters, such that, the minimal value in the pair is maximized over all possible pairs which form a valid cluster.

More formally, given the inter-cluster gap g_i, let $X = \{(d,d')|d\in D[g_{i-1},g_i),d'\in D(g_i,g_{i+1}],F(d,d') = true\}$. The split candidates pair $(d^{(l)},d^{(r)})$ is the pair that maximizes $\min(d,d')$ over all possible choices of $(d,d')\in X$. When more than one pair satisfied the condition, the pair with the maximal second pair value determines the unique *MMIDP*. In some cases we will refer the output of the function, $(d^{(l)},d^{(r)})$, as *MMIDP*.

Definition 4. We define the following Split-Join Condition (*SJC*): $SJC(d^{(l)},d^{(r)},g_i)= \min(d^{(l)},d^{(r)})>g_i$.

Split-Join Procedure (Stages 1-2): Given the inter-cluster gap g_{i-1} (stage 1), the Split-Join procedure (see Fig. 4a) enables two adjacent clusters to form the optimal cluster in the range $U(g_{i-2},g_i)$ in terms of Objective 1. For this purpose, the procedure begins with finding the *MMIDP*, $(d^{(l)},d^{(r)})$. Then, the *SJC* verifies whether $\min(d^{(l)},d^{(r)})$ is larger than the inter-cluster gap, g_i, trapped between them. When this condition is satisfied, the Split-Join procedure removes g_i by joining $U(d^{(l)},g_{i-1})$, $U(g_{i-1},d^{(r)})$ to form the new cluster $U(d^{(l)},d^{(r)})$. In case $d^{(l)} \neq g_{i-2}$, preceding Split operation is applied on $d^{(l)}$, resulting with the additional cluster: $U(g_{i-2},d^{(l)})$. Symmetrically, when $d^{(r)} \neq g_i$, $U(d^{(r)},g_i)$ is formed. Only the members of new cluster $U(d^{(l)},d^{(r)})$ terminate this iteration of the algorithm at the end of this stage, the rest continue to the successive stages. By combining together the (optional) Split, and Join operations to the same iteration, intermediate clustering reorganizations are avoided. Each cluster applies the procedure first (stage 1) on its left inter-cluster gap and then (stage 2) on its right inter-cluster gap. Nevertheless, as we shall see later, this procedure is performed in a coordinated fashion between the clusters.

Definition 5. We define the following Split Conditions:

- $SC1(C_i)= !F(g_{i-1},g_i)$;
- $SC2(C_i)= d'>g_{i-1},g_i$, where $d'= \max(D(g_{i-1},g_i))$.

Split Procedure (Stage 3): Given a cluster C_i and some inter-distance d', the split procedure is defined to partition the cluster C_i into two clusters: $U(g_{i-1},d')$ and $U(d',g_i)$. In order to maintain stable *CA* which consists of large clusters, the D-CUT tries to modify the current *CA* by Split procedure, only when the current *CA* contains clusters which are: (i) not satisfied by the objective function *F*, or (ii) discontinuous. When cluster ceases to satisfy the objective function *F*, the first split condition (*SC1*) is fulfilled. The second split condition (*SC2)* is satisfied when inner gap becomes larger than its delimiting inter-cluster gaps. In both cases, the split operation is done on the maximal inter-distance that results in creation of 2 valid clusters.

Remark: The D-CUT algorithm produces a valid *CA* at each iteration, due to the *SC1*. An invalid *CA* will be received when the last iteration *CA*, updated by the new node's locations, creates one or more invalid clusters. When some of the clusters do not satisfy the objective function *F*, Split operation, triggered by *SC1*, will occur. As a result, each invalid cluster is replaced by two[2] *valid* clusters. Since this operation is triggered independently among clusters, the split operations are occur simultaneously, and valid *CA* is received.

Definition 6. We define the following 2 Join Conditions:

- $JC1(g_i)=(g_{i-1}>g_i)\ \&\&\ !F(g_i,g_{i+2})\ \&\&\ F(g_{i-1},g_{i+1})$;
- $JC2(g_i)=(g_{i+1}>g_i)\ \&\&\ !F(g_{i-2},g_i)\ \&\&\ F(g_{i-1},g_{i+1})$.

Join Procedure (Stages 4-5): Given the gap g_i, the Join procedure (see Fig. 4b) is defined by removing the gap g_i to create the new cluster $U(g_{i-1},g_{i+1})$. The Join procedure is motivated by Objective 2, i.e., reducing the number of clusters in the model. Two join conditions allow continuously increasing cluster size to its limit as long as this operation is not preventing a more beneficial future Join or Split-Join procedures. For this purpose, the join conditions allow two clusters to join not only when a gap is trapped between two larger gaps as in the *SJC*, but also when it is trapped by a larger gap from one side, and non-joinable clusters from the other side (*JC1* and *JC2*).

V. ANALYTICAL ANALYSIS

In this section we conduct an analytical analysis of the ability of the D-CUT algorithm to self start and maintain the *GOCA* in the dynamic environment of VANET. This section is organized as follows.

A. Lower bound

First, let us show lower bound for approximation ratio for Objective 2, to every *CA* satisfies Objective 1.

THEOREM 1: There is a network *N* such that any valid[3] *CA* that meets Objective 1 has to approximate Objective 2 with factor of 2.

Proof: We consider a network *N* organized in dense, equally spaced, groups of $k_{max}/2$ and $k_{max}/2+1$ nodes, where each group of $k_{max}/2$ nodes is followed by group of $k_{max}/2+1$ nodes. Moreover, the inter-distances that separate the groups are larger than the inter-distances that separate the group member's. Let us denote by S_1 and S_2 the *CA*s that satisfy Objective 1 and Objective 2, correspondingly. Under this configuration, the size of each cluster in S_2 is maximal, i.e., $|C|=k_{max}$ for $\forall C\in S_2$. Accordingly, $|S_2| = n/k_{max}$. On the other end, the *CA* that meets Objective 1 clusters each group into a cluster. Hence, $|S_1| = 2\cdot n/(k_{max}/2+k_{max}/2+1)$. So, the ratio between $|S_1|$ and $|S_2|$ is 2. ■

B. Self-Organization

In this section we will demonstrate that the D-CUT algorithm self-organizes the hierarchical topology. For this, we will show that even though nodes hold only a local portion of the vehicle map, and therefore nodes from different clusters hold different section of the vehicle map, the *CA* produced by the D-CUT algorithm is coordinated among all nodes, as long as there vehicle proximity

---

[2] Here we assume that an invalid cluster, which was valid cluster in the previous iteration, can be split into 2 valid clusters. The algorithm can intuitively be expanded to deal with the case where invalid cluster is required to be split into more than 2 clusters.

[3] For every objective function, when the number of cluster's members is bounded.

```
// Stage 1 - Split-Join procedure on g_{i-1}
(d^(l),d^(r)) = MMIDP(g_{i-1});
 if (SJC(d^(l),d^(r),g_{i-1}))
      if u∈U(d^(l),d^(r)) then C_{i-1}=U(d^(l),d^(r)) and exit;
      else C_i=U(d^(r),g_i);
// Stage 2 - Split-Join procedure on g_i
(d^(l),d^(r))= MMIDP(g_i);
 if(SJC(d^(l),d^(r),g_i))
      if u∈U(d^(l),d^(r)) then C_{i+1}=U(d^(l),d^(r)) and exit;
      else C_i=U(g_{i-1},d^(l));
// Stage 3 - apply Split procedure on C_i
if(SC1(C_i)||SC2(C_i))
     d'=max(D(g_{i-1},g_i)) where F(g_{i-1},d')= F(d',g_i)=true;
     if u∈U(g_{i-1},d') then C_i=U(g_{i-1},d') and exit;
     else C_{i+1}=U(d',g_i) and exit;
// Stage 4 apply Join procedure on g_{i-1}.
if(JC1(g_{i-1})||JC2(g_{i-1})) && !(SJC(g_{i-2})||SC2(C_{i-1})))
     C_i=U(g_{i-2},g_i);
// Stage 5 apply Join procedure on g_{i-1}.
if(JC1(g_i)||JC2(g_i)) && !(SJC(g_{i+1})||SC2(C_{i+1})))
     C_i=U(g_{i-1},g_{i+1});
```

Fig. 3. The D-CUT algorithm.

map overlapping section is the same. More formally, assume the output of D-CUT for some node u_x is C_p, and for u_y is C_q; if $u_y \in C_p$ then $C_p=C_q$. Before proving the above assertion, let us establish the following:

Observation 1: In case $SJC(d^{(l)},d^{(r)},g_i)$ is satisfied: (i) if $d^{(l)} > d^{(r)}$ then $d^{(l)} = \max(D[d^{(l)},d^{(r)}])$ and $d^{(r)} = \max(D[g_i,d^{(r)}])$; (ii) symmetrically, if $d^{(l)} < d^{(r)}$, $d^{(r)} = \max(D[d^{(l)},d^{(r)}])$, and $d^{(l)} = \max(D[d^{(l)},g_i])$.

Observation 2: If $SJC(d_1^{(l)},d_1^{(r)},g_{i-1})= true$ and $SJC(d_2^{(l)},d_2^{(r)},g_i)=true$ then $U(d_2^{(l)},d_1^{(r)}) = \phi$.

Lemma 1: Given that one of the join conditions is satisfied on g_{i-1}, then g_i is not satisfying any of the join conditions at the same iteration.

Proof: Since g_{i-1} is satisfying one of the join conditions we can conclude that $F(g_{i-2},g_i)=true$ and either $g_{i-1} < g_i$ or $F(g_{i-1},g_{i+1})=false$. But for g_i to satisfy join condition, the expression $F(g_{i-1},g_{i+1})=true$ must be fulfilled and either $g_{i-1}>g_i$ or $F(g_{i-2},g_i)=false$. Thus, the lemma holds. ■

THEOREM 2: Let the output of D-CUT for some node u_x be C_p and for u_y be C_q. If $u_y \in C_p$ then $C_p=C_q$.

C. Independent sub-model clustering

In a good clustering algorithm, configuration changes in a certain place of the model would influence the clustering process of only some *local* sub-model around it. The algorithm, as we prove below, partitions the model N into local sub-models, where each sub model is clustered *independently*.

Definition 7. Let $Q(d',t)$ be the set of any inter-distances d that satisfies either $F(d',d)=true$, or $F(d,d')=true$ at iteration t. We define d' as a *local maximum* inter-distance in the timeframe $[t', t'']$, if and only if, $d' > \forall d \in Q(d',t)$ at any iteration t, $t' \le t \le t''$.

Now we shall confirm that as long as the two inter-distances remains *local maximum* in the time interval $[t', t'']$, the sub-model trapped between them is clustered independently.

Definition 8. Let $g_{v(t)}$ be the inter-cluster gap located at the inter-distance d_v in iteration t, i.e., $d_v = g_{v(t)}$. Accordingly, $g_{v(t)-1},g_{v(t)+1}$ are the 2 inter-cluster gaps that frame d_v from left and right, respectively, at the iteration t.

THEOREM 3: Consider d_v,d_u, 2 consecutive local maximum inter-distances in the time frame $[t',t'']$. Then, the D-CUT algorithm is clustering the sub-network $U(d_v,d_u)$ independently with the rest of the model, in the time frame $[t'+1, t'']$.

D. Convergence Process

In this section we would like to show the fast and strict convergence of the D-CUT algorithm, from any given *valid CA* to a *GOCA*. Furthermore, when assuming uniform distribution of the inter-distances' length, we will show logarithmic time convergence. Consequently, when the configuration is stable we maintain a stable *CA*. In dynamic configuration, the algorithm is promptly reacts to the configuration changes.

In order to demonstrate the above, we will take advantage of the correlation between the D-CUT convergence processes, and the Split Binary Tree (*SBT*), a particular tree representation of the inter-distance set D. Below, we analyze the convergence process by the following three stages: firstly, we present the *SBT* and prove that it is a Binary Search Tree with expected height of $O(\log(|D|))$; secondly, we limit the convergence process duration of the D-CUT algorithm by the height of the *SBT*; thirdly, we want to express the *SBT* height as a function of the distance between the initial *CA* and the *GOCA*.

1) The Split Binary Tree (SBT)

In what follows, we refine the notation of D to represent only the subset of the inter-distances which are involved in the convergence process. More formally, let D be subset which contains all the inter-distances that at some iteration, during the convergence process, served as a inter-cluster gap, i.e., $D = G(t_0) \cup G(t_0+1) \cup G(t_0+2) \ldots \cup G(t_0+t_2)$ where t_0,t_2 denote the first and last iterations in the conversance process, respectively.

Definition 9. Given a network N with configuration D, the Split Binary Tree *(SBT)* is a tree representation of the given configuration (see Fig. 5). The root entry of the *SBT* is the associated with the full set $D(d_s,d_f)$. Each subsequent *SBT* entry is associated with subset of D obtained by the following process: We start by setting d_k, the maximum inter-distance of the set $D(d_s,d_f)$, as the root entry. Then, we partition the set $D(d_s,d_f)$ into 2 subsets: $D(d_s,d_k)$, and $D(d_k,d_f)$; where the first subset associated with the root's left child, and the second with the right child. Then, we set the maximum inter-distances d_y and d_z - where $d_y = \max(D(d_s,d_k))$, and $d_z=\max(D(d_k,d_f))$ - as the left and right child of d_k, respectively. We continue with recursive process to the point when each received subset contains single inter-distance which acts as its own maximum. As key entry, we use the index of the maximum inter-distance (e.g. if d_v is the maximum in distance in the entry we set the key entry as v). By $l(d)$ and $r(d)$ we denote the left and right end points of associated range of d. Finally, the function $h(d_v)$ returns height of the subtree rooted at the entry v .

Corollary 1: Given inter-distance set D, where D values are uniformly distributed, $SBT(D)$ produces a Random Binary Search Tree on the indices of inter-distances with expected height of $O(\log|D|)$.

Proof: Consider the $SBT(D)$ produced by the inserting the tree's entries in decreasing order. That is, we set the maximal inter-distance as the root. Then, at each stage we insert into the *SBT* the subsequently maximal value, which has not yet inserted. We end when all inter-distances in D have been inserted. This *SBT* of the values of D is a Binary Search Tree considering the indices of D as entry's keys. When the inter-distances' values are uniformly distributed, this process inserts into the binary tree a random permutation of the keys set. This process produces a Random

Binary Search Tree on the indexes of inter-distances. As shown in [19], the expected height of *Random Binary Search Tree* is $O(\log |D|)$. ■

To show convergence, we need to assume a *stable configuration*. We define a *stable configuration* as a configuration where: (i) d_s,d_f remain local maximums during all the convergence process and (ii) the *SBT* representation of the sub-model $D(d_s,d_f)$ is unchanged.

2) *Bounding the convergence process duration by the height of SBT*

For bounding the convergence process duration by the height of *SBT*, we will show that each inter-distance is classified to its final state in the *GOCA* according to its height in the *SBT*. First, we represent the *CA* at some iteration t by $G(t)$, the set of *inter-cluster gaps* separating the *CA* clusters at t, in the sub-model $D[d_s,d_f]$.

Definition 10. We say that the inter-distance d is *classified* at iteration t' as inner gap if $d\notin G(t)$ for every $t > t'$. We say that the inter-distance d is *classified* at iteration t' as inter-cluster gap if $d\in G(t)$ for every $t > t'$.

Definition 11. Let us define a *refined height* $h'(d_v)$ of the sub tree rooted at the entry v by counting only entries that will be classified as *inner gap*.

We associate any inter-distance with one of 4 types.

Definition 12. Given $d_v\in D(d_s,d_f)$, we associate d_v according to the validity of the clusters trapped between $l(d_v)$, d_v, $r(d_v)$ as follows: (a) $d_v\in A_1$ if and only if $F(l(d_v),r(d_v))=true$; (b) $d_v\in A_2$ if and only if $F(l(d_v),d_v))=F(d_v,r(d_v)) = true$ and $F(l(d_v),r(d_v))=false$; (c) $d_v\in A_3$ if and only if $F(l(d_v),d_v) = true$ and $F(d_v,r(d_v))= false$; (d) $d_v\in A_4$ if and only if $F(l(d_v),d_v) = false$ and $F(d_v,r(d_v))= true$.

Remark: The final case where $F(l(d_v),d_v))=F(d_v,r(d_v)) = false$ is already defined as local maximum, i.e., the two sub-model end points.

The next observation exhibits the relationship between inter-distance type and the type of its *descendants* in the *SBT*.

Observation 3: Consider some inter-distance d_v, if $F(l(d_v),d_v)=true$ then all $d\in D(l(d_v),d_v))$ belong to A_1.

From this observation we can conclude that if $d\in\{A_1\cup A_2\}$ then all d descendants belong to A_1. Furthermore, the left *descendants* of $d\in A_3$ and the right descendants of $d\in A_4$ belong to A_1 as well.

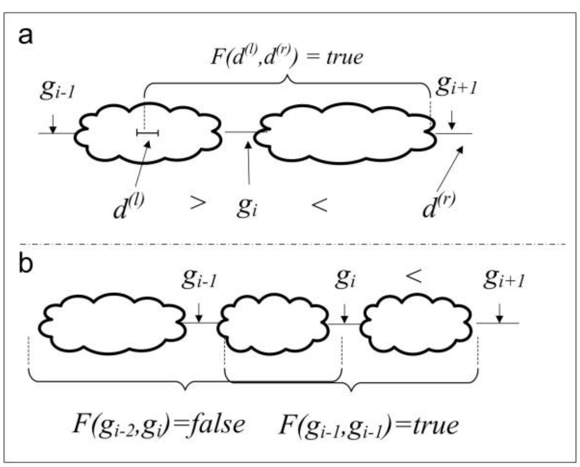

Fig. 4. (a) The Split Join procedure. In this example $SJC(d^{(l)},d^{(r)},g_i)$ is satisfied. As a result, Split operation on $d^{(l)}$ is triggered, which followed, at once, by Join operation over g_i. Thus, the new *CA* of this range is the 2 clusters $U(g_{i-1},d^{(l)})$ and $U(d^{(l)},g_{i+1})$. (b) The Join procedure. Here, $JC2(g_i)$ is fulfilled. Consequently, Join operation over g_i produces the new cluser.

Next, we will show the bottom-up classification process on the *SBT*. This process begins with inter-distances associate with A_1 that placed (if exist) in the bottom of the *SBT*. Lemma 2 assures that every $d\in A_1$ is classified as inner gap at iteration $t = h'(d)$. Lemma 3 shows that the *CA* obtained at the end of this phase satisfies Objective 1. Then, in Lemma 4 we ensure that $d\in A_2$ is classified as inter-cluster gap once its descendants, that are all associated with A_1, are classified. Notice that this condition is fulfilled at iteration $t= h'(d)$. We continue with bottom up process by demonstrating (Lemma 5) that $d\in\{A_3\cup A_4\}$ is classified either as inner gap or as inter-cluster gap at iteration $t = h'(d)$. To conclude, we prove that after the classification of all $d\in D[d_s,d_f]$ the obtained *CA* is in fact the *GOCA* (Lemma 6). Note that the sub-model end points d_s,d_f are classified as inter-cluster gaps at iteration t_0 as we have shown in the proof of Theorem 3.

In order to demonstrate the classification of inter-distance d as inner gap, we will ensure that if $d\in G(t)$ at iteration $t = h'(d)$ then Join operation will be applied on d. In case $d\in A_1$ we will demonstrate that *SJC* is satisfied, and when $d\in\{A_3,A_4\}$ the operation will be triggered by *JC1* or *JC2*. However, to guarantee the classification, we need to prove that this operation will not be overturned by future Split operation.

Remark: As we assume valid *CA* at iteration t_0, and as all the D-CUT operations produce valid clusters, in the following we assume Split operation to be trigger either by *SC2* or by *SJC*.

Observation 4: If $SC2(C_i)$ is satisfied on d' then $d' = \max(D[g_{i-1},g_i])$.

Observation 5: Let $g_{i(t)-1},g_{i(t)}$ be two consecutive inter-cluster gaps at iteration t. For every $t > t_0$, all $d\in D(g_{i(t)-1},g_{i(t)})$ are smaller than $\max(g_{i(t)-1},g_{i(t)})$.

Observation 6: If $D(l(d_v),r(d_v))\cap G(t') = \phi$, then $D(l(d_v),r(d_v))\cap G(t) = \phi$ for every $t > t'$.

Lemma 2: If $d_v\in A_1$, then d_v is classified at iteration $t = h'(d_v)$, as inner gap.

Lemma 3: Let $t_1 = t_0 + \max(h(d)) + 1$, for all $d\in D(d_s,d_f)\cap A_1$. $G(t)$ satisfies Objective 1 for every $t \geq t_1$.

In order to demonstrate the classification of d as inter-cluster gap we will show that d is located between two clusters, such that their union produces an invalid cluster. Considering such d, and assuming that all d descendants from A_1 are classified as inner gaps, the following ensures that this state is irreversible.

Observation 7: Consider $d_v\notin A_1$ ($d_v = g_{v(t)}$). If $F(g_{v(t)-1},g_{v(t)+1})=false$ at some iteration $t \geq h'(d_v)$, then $F(g_{v(t)-1},g_{v(t)+1})=false$ at any iteration $t' \geq t$.

Lemma 4: If $d_v\in A_2$ then d_v is classified as inter-cluster gap at the iteration $t = h'(d_v)$

To show the classification of $d_v\in\{A_3\cup A_4\}$ at the iteration $t = h'(d_v)$, in the following two observations we characterize the inter-cluster gaps $g_{v(t)-1},g_{v(t)+1}$, framing $d_v=g_{v(t)}$ from left and right, respectively, at this iteration.

Observation 8: Let $t' = h'(d_v)$. If $d_v\in A_3$ then $g_{v(t)-1} > g_{v(t)}$ for every $t \geq t'$.

Observation 9: Let $t' = h'(d_v)$. If $d_v\in A_3$ then $g_{v(t)+1}\in\{A_2\cup A_3\}$ for every $t \geq t'$.

In the following we subdivide the set A_3 into two subsets.

Definition 13. Given $d_v\in A_3$ ($d_v = g_{v(t)}$), if $F(g_{v(t)-1},g_{v(t)+1})=true$ at iteration $t = h'(d_v)$ then d_v is associated with the subset A_3*, else d_v is associated with the subset $A_3\backslash A_3$*.

Lemma 5: If $d_v \in A_3$ then d_v is classified at iteration $t=h'(d_v)$ either as inner gap when $d_v \in A_3*$ or as inter-cluster gap when $d_v \in A_3 \backslash A_3*$.

Proof: Following Observation 7, if $d_v \in A_3 \backslash A_3*$ then d_v is classified as inter-cluster gap. Hence, to prove the lemma, we will ensure that if $d_v \in A_3*$, then Join operation, triggered by $JC1(g_{v(t)})$, occurs at iteration t. We will demonstrate it by induction on the inter-distance refine height. By Observation 8 we get that $g_{v(t)-1} > g_{v(t)}$ at iteration t. To show that $JC1(g_{v(t)})$ is satisfied it is suffice to show that $F(g_{v(t)},g_{v(t)+2})= false$.

For the base case, we consider the d_v without descendant from A_3*. Thus, $h'(d_v)$ is the height of the highest d_v descendant belonging to A_1. Notice that $g_{v(t)+1}$ is $g_{v(t)}$ descendant and belongs to $\{A_2,A_3\}$ (Observation 9). According to Lemma 4, if $g_{v(t)+1} \in A_2$ then $F(g_{v(t)},g_{v(t)+2})= false$. If $g_{v(t)+1} \in A_3$, $F(g_{v(t)},g_{v(t)+2})= false$ as d_v has no descendant from A_3* (i.e., $g_{v(t)+1} \in A_3 \backslash A_3*$). Therefore, $JC1(g_{v(t)})$ is satisfied. Following Observation 6, Split operation on d_v will not be applied at any iteration $t' > t$, and thus, d_v is classified as inner gap. Assume that our induction hypothesis holds for all d such that $h'(d) \le t\text{-}1$. Here as well, the case where $g_{v(t)+1} \in A_2$ yield from Lemma 4. If $g_{v(t)+1} \in A_3$, $F(g_{v(t)},g_{v(t)+2})= false$ follows directly the inductive hypothesis. As in base case, the assertion is concluded by Observation 6. ■

For reasons of symmetry the above lemma holds for $d_v \in A_4$.

Lemma 6: Let $t_2 = t_0 + \max(h'(d)) + 1$, for all $d \in D(d_s,d_f)$. $G(t)$ satisfies Objective 2 with an approximation ratio of at most 3 for every $t \ge t_2$.

Proof: In order to compare between the values of Objective 2 in optimal *CA* and the *CA* produced by the D-CUT algorithm we will bound the number of inter-cluster gaps in each sub-model range separately. As the sub-model $D[d_s,d_f]$ shares the endpoint d_s with its left sub-model and d_f with its right sub-model, we count only the left endpoints in each sub-model. To be exact, we charge $|G(t_2)|\text{-}1$ inter-cluster gaps for the sub-model $D[d_s,d_f]$. Each inter-cluster gap $g_{i(t2)} \in G(t_2)$, excluding the sub-model end points $\{g_{s(t_2)},g_{f(t_2)}\}$, satisfies $F(g_{i(t_2)-1},g_{i(t_2)+1}) = false$. Accordingly, $G(t_2)$ can be segmented into $\left\lfloor \frac{|G(t_2)-1|}{2} \right\rfloor$ pairs of consecutive clusters, where the union of each clusters pair produces an invalid cluster. More formally, $F(g_{s(t)+2\cdot i},g_{s(t)+2\cdot i+2}) = false$ for $i= 0,1,2,\ldots, \left\lfloor \frac{|G(t_2)-1|}{2} \right\rfloor - 1$.

This implies that every valid *CA* has at least $\left\lfloor \frac{|G(t_2)-1|}{2} \right\rfloor$ inter-cluster gaps in the range $D[d_s,d_f]$ since every valid *CA* contains at least one inter-cluster gap in the range of each consecutive pair of clusters. Let G_{opt} be the set of the inter-cluster gaps in optimal *CA* in the range $D[d_s,d_f]$. We obtain $|G(t_2)-1| \le \left(2+\frac{1}{|G_{opt}|}\right)|G_{opt}|$. As $F(d_s,d_f) = false$, $|G_{opt}| \ge 1$. In worst case we get an approximation ratio of 3. ■

Corollary 2: The D-CUT algorithm converges to the *GOCA* after no more than t_2 iterations.

3) D-CUT strict convergence

After we have limited the convergence process duration by the *SBT* refined height, we want to express the *SBT* refined height as a function of the distance between the initial *CA* and the *GOCA*, i.e., $|(G(t_0) \cup G(t_2)) \backslash (G(t_0) \cap G(t_2))|$. We consider only the following distance subset to express the refined height of *SBT:*

Definition 14. Let $\Delta = G(t_0) \backslash G(t_2)$, be the set of inter-cluster gaps in the range $D(d_s,d_f)$ which belong to the initial *CA* $G(t_0)$, but not belong to the *GOCA*, $G(t_2)$.

Definition 15. Let $\Psi = D(d_s,d_f) \backslash (G(t_0)) \cup G(t_2))$, be the set of temporary inter-cluster gaps in the range $D(d_s,d_f)$, that appears (by Split operation), and then removed (by Join operation), during the course of the convergence process.

Notice that the union of the sets Δ and Ψ gives the set of all inter-distances that classified as inner gaps. As the refined height is a function of the inter-distances that classified as inner gaps, and Δ is a lower bound of the distance between the initial *CA* and the *GOCA*, our goal is to express the ratio between the size of sets Δ and $\Psi \cup \Delta$.

Below, we will demonstrate that $|\Psi \cup \Delta| \le 3.5 \cdot |\Delta|$ by showing that $|\Psi| \le 2.5 \cdot |\Delta|$. In order to give this bound, we will relate all Split operations that occur during the convergence process to an explicit subset of Δ. In particular, we define the subset Δ_v to be the set of inter-cluster gaps that are located in the range $D(l(d_v),r(d_v))$ at iteration t_0, i.e., $\Delta_v = D(l(d_v),r(d_v)) \cap G(t_0)$, where $d_v \in A_1$ and both $l(d_v),r(d_v) \notin A_1$. Since every $d \in D(l(d_v),r(d_v))$ belongs to A_1, we can conclude that $\Delta_v \subseteq \Delta$. The right neighbor of Δ_v is denoted by $\Delta_u = D(r(d_v),r(d_u)) \cap G(t_0)$.

Firstly, we relate each Split operation that takes place in the range $D[l(d_v),r(d_v)]$ to the subset Δ_v. According to Observation 6, if $\Delta_v = \phi$, then Split operation will not take place in the range $D(l(d_v),r(d_v))$. Moreover, from Observation 9 follows that if $\Delta_v = \phi$ and Split operation, triggered by *SC2*, takes place on $r(d_v)$, then Δ_u, is not empty. Accordingly, any Split operation triggered by *SC2* can be related to one of Δ subsets. However, Split operation triggered by $SJC(d^{(l)},d^{(r)},g_i)$, $g_i \in D[l(d_v),r(d_v)]$, can be spread outside the range $D[l(d_v),r(d_v)]$. In Observation 10 we demonstrate that in such case, e.g., $d^{(l)} \notin D[l(d_v),r(d_v)]$, $d^{(l)} \notin A_1$. Thus, $d^{(l)}$ will not trigger additional Split operation. Hence, by relating any Split operation, triggered by $SJC(d^{(l)},d^{(r)},g_i)$, where $g_i \in D[l(d_v),r(d_v)]$, to the subset Δ_v we ensure that any Split operation triggered by *SJC* will be related to one of Δ subsets.

Definition 16. We say that Split operation on d is *resulted* by Δ_v, $\Delta_v \ne \phi$, if one of the following is satisfied: (i) $d \in D[l(d_v),r(d_v)]$, (ii) $d \notin D[l(d_v),r(d_v)]$ and the operation is triggered by $SJC(d^{(l)},d^{(r)},g_i)$, where $g_i \in D[l(d_v),r(d_v)]$.

Observation 9: Let Δ_v,Δ_u be two adjacent Δ subsets. Let $d' = r(d_v) = l(d_u)$. If *SC2* is satisfied on d' at iteration t_0, then either $\Delta_v \ne \phi$, or $\Delta_u \ne \phi$.

Observation 10: Consider the case when $SJC(d^{(l)},d^{(r)},g_i)$ is satisfied. If $g_i \in D(l(d_v),r(d_v))$ and $d^{(l)} \in A_1$ then $d^{(l)} \in D(l(d_v),r(d_v))$.

According to the above, we can establish the inequality $|\Psi| \le 2.5 \cdot |\Delta|$ by showing that the number of Split operations *resulted* by the set Δ_v is bounded by $2.5\,|\Delta_v|$.

We first consider the base case where $|\Delta_v| = 1$. In this case no more than 2 Split operations (on $l(d_v),r(d_v)$) will be resulted by the Join operation on $g_{i(t_0)}$. This is because $SJC(l(d_v),r(d_v),g_{i(t_0)}) = true$ at iteration t_0. After the Join operation on $g_{i(t_0)}$, the range $D(l(d_v),r(d_v))$ (which does not contain any inter-cluster gap) will not be split, as demonstrated in Observation 6.

Next, we show that if $|\Delta_v| \ge 2$, no more than $2 \cdot |\Delta_v| + 1$ Split operations will be resulted by the set Δ_v. As we seek for an upper bound, we are allowed to assume that if $\Delta_v \ne \phi$ then $SJC(l(d_v),r(d_v),d_v)$ will be satisfied. Therefore, we presume that

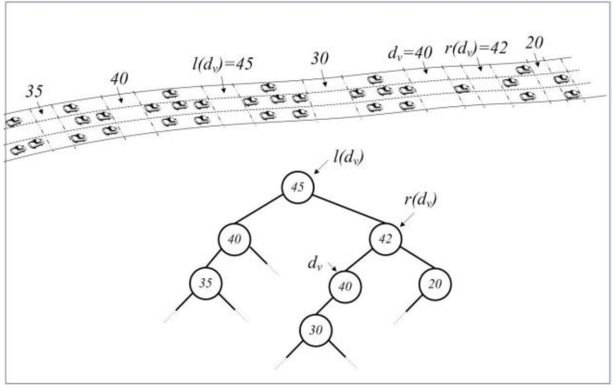

Fig. 5. The *SBT* of the set $D'=\{35,40,45,30,40,42,20\}$.

Split on $l(d_v),r(d_v)$ will be resulted by $\Delta_v \neq \phi$. According to the above, the total number of Split operations resulted by the set Δ_v is limited to the sum of: (i) the number of Split operations on the both ends of the sub-model: $l(d_v),r(d_v)$,(ii) the number of Split operations (either by fulfilling *SJC* or *SC2*) taking place in the range $D(l(d_v),r(d_v))$, and (iii) the number of Split operations taking place (by fulfilling $SJC(d^{(l)},d^{(r)},g_i)$) outside the range $D[l(d_v),r(d_v)]$, where $g_i \in D(l(d_v),r(d_v))$.

In the following two observations we will characterize the split candidates in the range $D(l(d_v),r(d_v))$ according to the initial *CA* at this range. Let $D_{v,j}(t) = D[l(d_v),r(d_v)] \cap D[g_{j(t)-1},g_{j(t)}]$.

Observation 12: If $SJC(d^{(l)},d^{(r)},g_{i(t)})$ is satisfied, and both $d^{(l)},g_{i(t)} \in D_{v,i}(t)$ at iteration t, then $d^{(l)} = \max(D_{v,i}(t))$.

For reasons of symmetry, the above observation holds for $d^{(r)}$ as well. In Observation 12 we demonstrated that a split candidate d must satisfy $d = \max(D_{v,i}(t))$. In the following we extend this split candidate prerequisite to $d = \max(D_{v,i}(t_0))$.

Observation 13: If $d \neq \max(D_{v,j}(t_0))$ then $d \neq \max(D_{v,j}(t))$ for every $t \geq t_0$.

After stating the above, we are ready to limit the number of Split operations resulted by the set Δ_v.

Lemma 8: The maximal number of Split operations on $d \in D(l(d_v),r(d_v))$, resulted by the set Δ_v, is $|\Delta_v| - 1$.

Lemma 9: The maximal number of Split operations triggered by $SJC(d^{(l)},d^{(r)},g_{i(t)})$, where $g_{i(t)} \in D(l(d_v),r(d_v))$ and $d^{(l)}$ or $d^{(r)} \notin D[l(d_v),r(d_v)]$, is $|\Delta_v|$.

Proof: First we would like to show that if $SJC(d^{(l)},d^{(r)},g_{i(t)})$ is satisfied, where $g_{i(t)} \in D(l(d_v),r(d_v))$ then either $d^{(l)} \in D(l(d_v),r(d_v))$, or $d^{(r)} \in D(l(d_v),r(d_v))$ holds. Assume the opposite, that is, $D(l(d_v),r(d_v)) \subset D(d^{(l)},d^{(r)})$. Since: (i) by definition $F(d^{(l)},d^{(r)}) = true$, and (ii) following Observation 1, $\min(d^{(l)},d^{(r)}) > \min(l(d_v),r(d_v))$, then $\min(l(d_v),r(d_v)) \in A_1$ which contradicts Δ_v definition, when $l(d_v),r(d_v) \notin A_1$.

Thus, the only scenario where the Split operation, resulted by Δ_v, will take place on $d^{(l)} \notin D[l(d_v),r(d_v)]$ is when both $d^{(r)},g_{i(t)} \in D(l(d_v),r(d_v))$. In case when $d^{(r)} \in D(l(d_v),r(d_v))$ and $d^{(l)} \notin D[l(d_v),r(d_v)]$, we denote $d^{(r)}$ by $d^{(r)*}$. In the symmetric case when $d^{(l)} \in D(l(d_v),r(d_v))$ and $d^{(r)} \notin D[l(d_v),r(d_v)]$, we denote $d^{(l)}$ by $d^{(l)*}$. As demonstrated in Lemma 8, there are only $|\Delta_v| - 1$ inter-distances in the range $D(l(d_v),r(d_v))$ that can play the role of $d^{(r)*}$ (or $d^{(l)*}$) since $d^{(r)*} = \max(D[g_{j(t_0)-1},g_{j(t_0)}])$ for $g_{j(t_0)-1},g_{j(t_0)} \in \Delta_v$. To conclude, notice that any of those $|\Delta_v| - 1$ inter-distances can play the role of $d^{(r)*}$ (or $d^{(l)*}$) only once. This happens because if $SJC(d^{(l)},d^{(r)*},g_{i(t)})$ is satisfied, then $g_{i(t)}$ is the leftmost inter-cluster gap in the range $D(l(d_v),r(d_v))$. After this operation, $d^{(r)*}$ become the leftmost inter-cluster gap in this range, and therefore, will not play the role as $d^{(r)*}$ again. Following the same reason, only the last inter-distance removed from the $|\Delta_v| - 1$ Split candidates can play the role of both $d^{(r)*}$ and $d^{(l)*}$. This is because once inter-distance play the role of $d^{(r)*}$ it can play the role of $d^{(l)*}$ only after the rest of the Split candidates have been classified as inner gaps. ■

Corollary 3: $|\Delta \cup \Psi| \leq 3.5 \cdot |\Delta|$.

THEOREM 4: From any given starting point, the D-CUT algorithm converges to *GOCA* under the assumption of stable configuration status. The convergence process requires $O(|\Delta|)$ worst case time; and $O(\log|\Delta|)$ expected time, under the assumption of random permutation of the size of the inter-distances in the set $D(d_s,d_f)$.

VI. Simulations

In order to evaluate the performance of the D-CUT algorithm under realistic road conditions we have performed the following simulations.

A. Simulation Setup

The D-CUT algorithm highly depends on the inter-distances between cars. Thus, for faithful evaluation of the algorithm, a realistic mobility model for individual vehicles is required. Hence, we base our simulation on a microscopic model presented in [20] designed for multi-lane traffic flow dynamics. Each car experiences a force resulting from a combination of the desire of the driver to attain a certain velocity, aerdynamic drag, and change of the force due to car–car interactions. The model includes multi-lane simulation capabilities. We simulate 200 vehicles on a 3 lane straight road with a single entrance and exit on a 20 Km road section. The velocity of the vehicles was randomly generated according to normal distribution function with a mean of 120 Km/h and a deviation of 15. In addition, R_{max} is set to 500 meters, and k_{max} to 20. Unless stated otherwise, beacon transmission cycle time is set to 0.4 sec.

B. Tracking the Optimal Solution

As proved above, under stable configuration status, the D-CUT algorithm produces the *GOCA* that meets Objective 1 and approximates Objective 2 by factor of 3. Fig. 6 exhibits the ability of the algorithm to satisfy the objectives under real traffic condition. Fig. 6(a) compares the minimal gap of the *CA* produced by the D-CUT algorithm with the minimal gap of optimal *CA* which satisfies Objective 1. In this figure beacon transmission cycle time is set to 0.3 sec, and the first 50 iterations are plotted. This comparison shows that the D-CUT algorithm provides a fast convergence towards the optimal solution, and displays high correlation with it after initial convergence. Fig. 6(b) shows the percentage of iterations where the *CA* produced by the D-CUT algorithm satisfies Objective 1 for different beacon transmission cycle times. At a 0.1 sec cycle time the D-CUT achieved very high correlation (97%). As the cycle time increases we can see the correlation decreases. Fortunately, even at high cycle times of 1 sec a 64% correlation is still achieved.

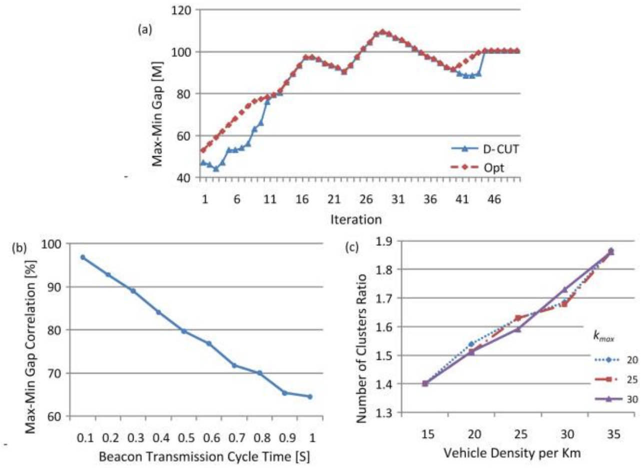

Fig. 6. Objectives evaluation: (a) comparison of the minimal gap produced by the D-CUT *CA* with the optimal *CA* (Objective 1). (b) percentage of iterations where the D-CUT *CA* satisfies Objective 1. (c) comparison of the number of clusters between the D-CUT CA and Objective 2 optimal CA.

Fig. 6(c) presents a comparison between the number of clusters produced by optimal *CA* according to Objective 2 and the *CA* produced by the D-CUT algorithm for various vehicle densities. As we can see from the dynamic simulation, the ratio achieved is even better than the ratio of 3 proved before for static configuration. In addition, we learn that this ratio increases with the density of vehicles. The reason for that is since the density increases, valid clusters become mainly dependent on k_{max} criterion. Since the optimal solution takes into account Objective 2 and is based solely on the number of clusters it produces without taking into account Objective 1, the value $|S_{opt}|$ decreases towards n/k_{max}. On the other hand, since the D-CUT algorithm includes Objective 1 in its consideration, density increase has moderate influence on the number of clusters.

VII. DISCUSSION

In this section we will discuss the ability of the D-CUT algorithm to provide a hierarchal topology suite for safety communication constraints. In order to minimize adjacent clusters mutual interference, we need to coordinate between concurrent channel accesses take place in adjacent clusters. In addition, a power assignment scheme will be designed to fairly equalize the mutual interference. As explained below, the clusters will be colored; this property allows the coordination among adjacent clusters. The advantage of these schemes is in design according to the snapshot of the local vehicle proximity map, partitioned according to current optimized clusters.

A. Efficient and reliable beacon dissemination process

The dissemination process demonstrates dual relation with the D-CUT algorithm. On one hand, the D-CUT algorithm is based on top of this process as a prerequisite that each vehicle will hold the updated *CA* of its own and its adjacent clusters. The dissemination process effectively utilizes the efficient and reliable hierarchical topology discovered by the D-CUT algorithm. Mainly, by replacing the traditional multipoint to multipoint broadcast transmission of the beacon messages with aggregation-dissemination base process. As mentioned above, this approach profit from many advantages, including highly reducing the security related overhead [10].

The beacon dissemination process is initiated where each vehicle holds the Clustered and Colored Vehicle Proximity Map (CCVPM*)* of its vicinity. At the end of the following three phases the vehicles hold the updated CCVMP.

Phase 1- Beacon Aggregation by Clusterheads: At this step clusterheads aggregate the beacon messages from theirs cluster members simultaneously. For this purpose, clusterhead must satisfy one hop connection to any node in its cluster set. The objective function *F* assures the existence of such node. The simultaneous aggregation provides high bandwidth efficiency to the beacon dissemination process. The protocol synchronizes concurrent transmissions taking place in adjacent clusters according to a fair interference minimization criterion. Broadly speaking, we coordinate the channel access between adjacent clusters by taking advantage of the strong links between nodes located next to clusterheads, to deal with the weak links of the nodes located far from the clusterheads.

Phase 2 - Clusterheads communication for D-CUT Algorithm execution: During the D-CUT run, adjacent clusterheads need to communicate, in order to obtain updated CCVMP of their vicinity. Adjacent clusterheads exchange theirs aggregated cluster information. In case of clustering reorganization, supplementary information may be exchanged to ensure that each clusterhead holds the full updated CCVMP. Since the information can be obtained from the adjacent cluster, only unicast communications between adjacent clusterheads are required. We note that the cluster coloring protocol can be embedded within this information exchange. After reducing the amount of channel contenders in Phase 1, the clusterheads communication will be on top of a sparse topology backbone.

Phase 3- CCVMP *dissemination*: In this phase clusterhead disseminates the updated CCVMP to its all cluster members. Again, due to the objective function this can be done in one broadcast transmission. To avoid inter-cluster interference in this curtail transmission, we use the cluster coloring to guarantee that 2 adjacent cluster members will remain silent during this transmission.

4) Handling Transsmission Failures

When the beacon transmission is not received by the clusterhead, current nodes location can be approximated based upon previous cycle data. This approximation will be disseminated in *Phase 3* (along with failure indication), so the rest of nodes will coordinate accordingly. In *Phase 2*, the D-CUT coordination requirement is ensured by the acknowledgment mechanism enabled by the unicast transmission fashion. In the case of unsuccessful cluster information exchange between adjacent clusterheads, the reorganization operations between the two clusters will be coordinately ignored. When a node does not receive the disseminated CCVMP in *Phase 3*, it should skip the following beacon transmission in order to avoid any uncoordinated transmission that can lead to packet collision. Once node receives the updated cluster information in this phase, it can take part in the following beacon aggregation.

5) Scalability

A well-known scheme to control the channel load in dense network configuration is to limit the maximal transmission range. Here, we achieve congestion control by limiting the cluster size. When considering high-density vehicular scenario with high anticipated load on the channel, clusters' size will come up to its limit. In such case, the simultaneous aggregation is most effective, and spatial reuse is optimized. On the other hand, when considering sparse configuration, the load on the channel is low. The main topology design goal is to provide connectivity. Even though the simultaneous aggregation is less effective, connectivity is achieved as transmission range is not limited for providing congestion control.

B. Emergency messages dissemination process

When considering dissemination of emergency message, reliability and delay are of the essence. To this end, Resta et al. [17] proposes a greedy strategy based on the use of the vehicle proximity map combined with a contention based approach. However, this strategy does not take into account a number of vehicles simultaneously detecting a hazard issue and initiating emergency messages. Hierarchical topology can perform well in such challenging scenarios by facilitating the emergency message channel access mechanism, and by allowing clusterheads to discard redundant messages. In addition, clusterheads can supply QoS (quality of service) by applying prioritized re-broadcast scheme.

C. Cluster Coloring

In order to synchronize between adjacent clusters, we want clusters to be colored. Cole and Vishkin [21] demonstrated that ring topology network can be colored by constant number of colors in O(log*n) rounds[4], such that the adjacent nodes are colored by different colors. This scheme can be used for the initial inter-cluster coloring. The split-join technique used by the D-CUT algorithm allows maintaining inter-cluster coloring for any value bigger or equal than 3. To coordinate the coloring procedure, we use the current coloring on top of the synchronized clock. In this way we prevent adjacent clusters coloring at the same step.

D. Clusterhead failure

Hierarchical topology has inherent sensitivity to clusterhead failure. To increase stability we can utilize the objective function to demand number of clusterhead candidates to satisfy one hop connection. Thus, clusterhead failure sensitivity can be reduced by applying contention base approach among the clusterhead candidates.

REFERENCES

[1] M. Shulman and R. Deering, "Vehicle safety communications in the United States", *Conference on Experimental Safety Vehicles*, 2007.

[2] D. Kumar, A. Kherani, and E. Altman, "Route lifetime based optimal hop selection in VANETs on highway: an analytical viewpoint", *Netw. Techn., Services, and Protocols*, pp. 799-814, 2006.

[3] R. Baldessari et al., "Car-2-car communication consortium-manifesto", *DLR Electronic Library [http://elib.dlr. de/perl/oai2](Germany)*, 2007.

[4] M. Torrent-Moreno, P. Santi, and H. Hartenstein, "Fair sharing of bandwidth in VANETs", *VANET*, pp. 49-58, 2005.

[5] K. Bilstrup, E. Uhlemann, E. G. Ström, and U. Bilstrup, "On the ability of the 802.11 p MAC method and STDMA to support real-time vehicle-to-vehicle communication", *EURASIP J. on Wireless Comm. and Netw.* , vol. 5, 2009.

[6] S. Bana and P. Varaiya, "Space Division Multiple Access (SDMA) for robust ad hoc vehicle communication networks", *Intelligent Transp. Syst.*, pp. 962-967, 2001.

[7] X. Chen and H. H. Refai, "SDMA: On the suitability for VANET", *Information and Communication Technologies: From Theory to Applications*, pp. 1-5, 2008.

[8] R. A. Santos, R. M. Edwards, and N. L. Seed, "Supporting inter-vehicular and vehicle-roadside communications over a cluster-based wireless ad-hoc routing algorithm", *Symp. on ICT.*, pp. 1-6, 2004.

[9] M. Durresi, A. Durresi, and L. Barolli, "Emergency broadcast protocol for inter-vehicle communications", *Parallel and Distributed Systems*, pp. 402-406, 2005.

[10] M. Raya, A. Aziz, and J. P. Hubaux, "Efficient secure aggregation in VANETs", *ACM VANET*, pp. 67-75, 2006.

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

[12] K. Xu, X. Hong, and M. Gerla, "An ad hoc network with mobile backbones", *IEEE ICC*, pp. 3138-3143, 2002.

[13] R. Palit, E. Hossain, and P. Thulasiraman, "MAPLE: a framework for mobility aware pro active low energy clustering in ad hoc mobile wireless networks", *Wireless Comm. and Mob. Computing*, vol. 6, no. 6, pp. 773-789, 2006.

[14] R. Ghosh and S. Basagni, "Mitigating the impact of node mobility on ad hoc clustering", *Wir. Comm. and Mob. Comp.*, 8(3), pp. 295-308, 2008.

[15] H. Cheng, J. Cao, X. Wang, S. K. Das, and S. Yang, "Stability aware multi metric clustering in mobile ad hoc networks with group mobility", *Wir. Comm. and Mob. Comp.*, 9, no. 6, pp. 759-771, 2009.

[16] P. Fan, "Improving broadcasting performance by clustering with stability for inter-vehicle communication", *Vehicular Technology Conference*, pp. 2491-2495, 2007.

[17] Y. Gunter, B. Wiegel, and H. P. Großmann, "Cluster-based medium access scheme for VANETs", *Intelligent Transportation Systems Conference*, pp. 343-348, 2007.

[18] L. Wischhof, A. Ebner, and H. Rohling, "Information dissemination in self-organizing intervehicle networks", *Intell. Transp. Syst., IEEE Trans on*, 6(1), pp. 90-101, 2005.

[19] B. Reed, "The height of a random binary search tree", *Journal of the ACM*, vol. 50, no. 3, pp. 306-332, 2003.

[20] N. O. Hodas and A. Jagota, "Microscopic modeling of multi-lane highway traffic flow", *Am. J.l of Physics*, 71, pp. 12-17, 2003.

[21] R. Cole and U. Vishkin, "Deterministic coin tossing with applications to optimal parallel list ranking", *Information and Control*, vol. 70, no. 1, pp. 32-53, 1986.

[4] The function log*() is defined recursively as follows $\log^* 0 = \log^* 1 = \log^* 2 = 0$ and $\log^* n = 1 + \log^* \lceil \log n \rceil$ for n > 2.