# A Location Service Mechanism for Position-Based Multicasting in Wireless Mobile Ad hoc Networks

Yoav Sasson David Cavin André Schiper
{yoav.sasson,david.cavin,andre.schiper}@epfl.ch

École Polytechnique Fédérale de Lausanne (EPFL)
1015 Lausanne, Switzerland

***Abstract*— In this paper we propose a novel location management scheme tailored for multicasting in Mobile Ad-hoc Networks (MANETs). We furthermore propose AMDLM, a location-based multicast algorithm relying on the location management service. Such an approach avoids fragile data structures such as trees or DAGs to manage multicast groups, without reverting to more reliable, yet overhead-prone mesh-based algorithms. AMDLM additionally enables us to derive analytical bounds due to its location-based nature.**

## I. Introduction

*Mobile ad hoc networks* (MANETs) are self-organizing mobile wireless networks that do not rely on a preexisting infrastructure to communicate. Nodes of such networks have limited transmission range, and packets may need to traverse multiple other nodes before reaching their destination. Research in MANETs was initiated 30 years ago by DARPA for packet radio projects [1], but has regained popularity nowadays due to the widespread availability of portable wireless devices such as cell phones, PDAs and WiFi/Bluetooth enabled laptops.

*Multicasting* provides a means for multipoint communication by enabling applications to seemingly communicate with groups of nodes. Traditionally a well suited tool for collaborative applications, multicasting is especially useful in ad hoc networks where tasks may be carried out by groups of nodes. Due to scarce bandwidth, varying network connectivity and frequent topology changes caused by node mobility and transient availability, routing algorithms tailored for wired networks will not operate well if directly transposed to MANETs. All the more so with multicasting, which adds to the difficulties of unicast routing the complexity of maintaining and handling dynamic multicast group membership changes.

In this paper we present a novel MANET location service for multicasting, which is based on an extension of the DLM scheme [2]. Dedicated servers are distributed throughout the network in order to store the geographic location of multicast group members. Coupled with an underlying geographic forwarding layer (e.g. [3], [4]), the solution offers an alternative approach for multicast routing. Since no end-to-end routes are maintained, the benefits of location-based multicasting are reduced overhead, increased robustness to mobility and fault-tolerance. There exist several location-based algorithms for *unicast* routing ([5], [6], [2]). In [7] the authors extend unicast position-based routing for multicasting: the paper generalizes routing and assumes that the position of the destination(s) is known in advance through a location service. Because of node mobility and dispersion of multicast node members, we claim that location services designed for unicast routing are not exploitable as such for multicasting. The contribution of this paper is to devise a novel location management scheme adapted for multicasting in MANETs. We furthermore present *AMDLM* (*Adaptive Multicast Distributed Location Management*), a location-based *multicast* algorithm built on top of the location service[1].

The remainder of the paper is organized as follows. The next section provides an overview of other works that address multicasting in MANETs. In Section III we present the DLM [2] location management scheme for unicast routing, which serves as a basis for our multicast algorithm. Section V presents *AMDLM*, a novel location based multicast algorithm. In Section VI we undertake an analytical study of the algorithm, followed in Section VII by a qualitative comparison between *AMDLM* and the other popular approaches for MANET multicasting. We finally conclude and describe future work in Section VIII.

## II. Related Work

There have been numerous multicast algorithm proposals for MANETs. In this section we present the most representative of each approach.

As with unicast routing, multicast routing comes in *proactive*, *reactive*, or a combination of the two flavors (*hybrid*). Reactive algorithms [9], [10] present reduced maintenance

The work presented in this paper was supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant 5005-67322.

[1]Location-based multicasting may be confused in the MANET community with *geocasting* [8]. Whereas for geocasting nodes join and leave groups by entering and leaving geographic regions, multicasting enables nodes to join and leave groups at any time, regardless of their location. *AMDLM* provides the latter service.

overhead by maintaining state information only when a multicast session is active. The drawback is decreased responsiveness. Proactive algorithms [11], [12], [13] react faster since multicast routing information is readily available, but at the price of introducing high overhead for maintaining multicast group structure even when no multicast session is active. The hybrid approach [14] aims at obtaining a satisfactory balance among the characteristics of both methods by limiting the scope of the proactive procedures to the local neighborhood of nodes and implementing reactive procedures for longer distances.

Various algorithms rely on different data structures to manage multicast group membership. Due to the highly dynamic and everchanging topology of MANETs, solutions that offer multiple routes through more robust data structures perform better. Therefore, mesh-based solutions [10] generally outperform tree-based solutions [9] due to the availability of alternative paths, which in turn tend to perform better than directed acyclic graph (DAG) solutions [13]. In extreme cases of high mobility and frequent multicast group membership changes, basic flooding still remains the best performing multicast algorithm [15]. Performance comparison studies of MANET multicast protocols may be found in [16], [17], [18].

Our (reactive) location-based approach for multicasting, *AMDLM*, differs from previous MANET multicast algorithms by relying on a location service composed of dedicated nodes distributed across the network[2]. Communication between these nodes is provided by an underlying geographic forwarding routing mechanism. Therefore, to the contrary of the tree, DAG or mesh approaches, no multihop data-structure needs to be maintained, reducing the vulnerability to the frequent link-breakages that occur in MANETs.

## III. The Distributed Location Management Scheme (DLM)

DLM [2] is a location management service for MANETs tailored for *unicast* routing, which addresses the shortcomings of GRID [6]. As with GRID, DLM partitions the mobile node deployment region into a hierarchical grid with squares of increasing size, as shown in Figure 1(a). The location service is offered by location servers assigned across the grid, storing node location information. DLM assumes a *uniform* distribution of the location servers. The server density is a parameter that may be adapted to better suit the characteristics of the network. *To the contrary of GRID, location servers in DLM are not directly nodes, but regions in the grid*. Nodes that happen to be located in these region offer the location service. This solution increases DLM's robustness to mobility. The selection mechanism for the predetermined regions is carried out through a hash function, which maps node identifiers to region addresses.

DLM distinguishes between a *full length* address policy and a *partial length* address policy. Under the full length address policy, location servers store the precise position of nodes. When nodes change regions due to mobility, it is necessary to update all location servers. Under the partial length address policy, the accuracy of node location information stored at the location servers increases along with the proximity of the location servers to the nodes. To the contrary of the full length address policy, several queries are necessary to locate a node. Nevertheless, the partial addressing scheme offers overall increased performance, since it reduces the scope and frequency of location server updates due to node mobility. Indeed, only the location servers affected by the distance travelled by the nodes need to be updated. We therefore consider the partial length address policy whenever we refer to DLM in this paper.

Figure 1 illustrates an example of how a node location query is carried out in DLM. Figure 1(a) depicts the location server hierarchical partitioning, which is an abstract overlay above the full grid in which the nodes evolve (Figure 1(b)). A location server is responsible for its entire region, which may not be within single-hop communication reach. Node $B$ desires to find the location of node $A$. Node $B$ will first query location server $L10$, which is $A$'s location server in the same region as $B$. $L10$ will reply to $B$ with information about the quadrant $A$ belongs to. $B$ will now contact a location server for $A$ in that quadrant, e.g. $L4$. $L4$ will similarly reply to $B$ with the smaller subregion containing $A$. The process continues until $B$ eventually contacts a location server that knows $A$'s exact position, $L3$ in our example.

## IV. DLM and Multicast

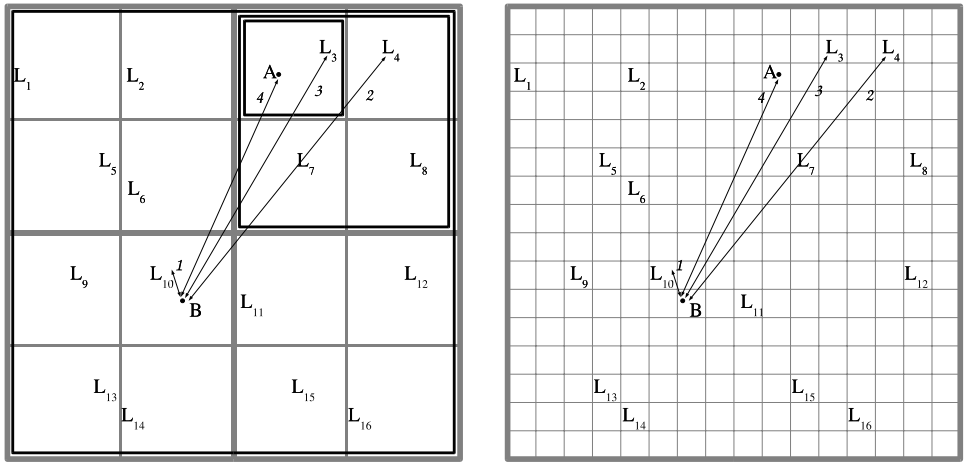
In this section we examine the straightforward modifications required for DLM to offer a multicast service.

### A. MDLM: Extending DLM for Multicast

*DLM* scheme relies on a *hash function* for assigning and locating servers responsible for storing node location information. The assignment is essentially based on the identifier of the node we wish to locate. DLM may be extended to offer a multicast service by replacing the node ID parameter by a multicast group ID. Furthermore, location servers will now store a *set* of links to regions that contain multicast group members. The rest of the algorithm remains similar. In the remainder of the paper, we denote by *MDLM* the straightforward extention of DLM for the multicast operation.

Figure 2 illustrates an example of a node $B$ multicasting to a group $G$ composed of members $G = \{A, A', A''\}$, with the multicast extensions brought to DLM. $B$ must first obtain the location of the group members. Node $B$ will first query location server $L10$, which is group $G$'s location server in the same region as $B$. $L10$ will reply to $B$ with a set of quadrants that contain group members. $B$ will then contact a location server for $G$ in each one of the quadrants returned, i.e. $L4$, $L12$ and $L13$. These location servers will similarly reply to $B$ with the smaller subregions containing members of $G$. The process continues until $B$ eventually contacts the location servers that know the exact position of each group

[2]See the comment concerning [7] in the introduction.




(a) Location Server Partitioning Representation

(b) Full Grid Representation

Fig. 1. Node Location Query under DLM's Partial Address Policy

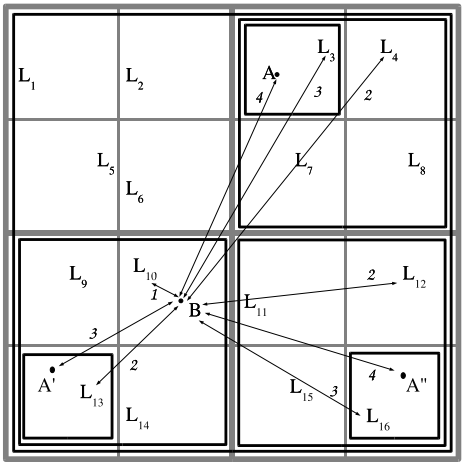



Fig. 2. DLM Multicast Extension

member, $L3$, $L13$ and $L16$ in our example. $B$ may now send a message to $A$, $A'$ and $A''$.

There are also trivial modifications that may be brought to DLM's approach in order to greatly enhance performance. In particular, instead of location queries being sent back and forth between a requesting node and the location servers until location information is gathered, a multicast may be sent and routed through the location servers themselves. Multicast messages will be forwarded along the location servers until they reach their destination. Among the benefits are reduced latency, increased reliability and robustness to mobility, since messages may still reach a moving target.

### B. Discussion

The solution presented in Section IV-A to transform DLM into a multicast service (MDLM) is rather straightforward but has nevertheless drawbacks in terms of performance, overhead and scalability.

In the case of a uniform distribution of *multicast group members*, MDLM may be satisfactory. This is however not the case with a non-uniform geographic distribution of the group members, a situation that may be very frequent. If group members are not uniformly distributed, having a uniformly distributed set of location servers is not optimal in terms of cost to maintain the location information (in regions void of group members) when nodes move, join, or leave the group.

Ideally, the presence or absence of location servers in a region should dynamically adapt to the presence or absence of group members in that region. In the rest of the paper we present a solution that has this property. We also discuss how nodes *join* and *leave* a group and the handling of node mobility.

## V. The Adaptive Multicast Distributed Location Management Algorithm (AMDLM)

### A. Design Choices for Efficient Location Management Multicasting

The most desired property for a location-based MANET multicast algorithm is to minimize the number of required location servers without harming overall performance. This may be achieved through maintaining a higher concentration of location servers around multicast group members, which requires a *dynamic* assignment and adaptation of the location servers. Multicast group members and nodes in their proximity will have privileged access to multicast group membership information. Nodes far from group members can still multicast, yet with a higher average cost.

An additional desirable property is the independence of the core multicast algorithm from a particular location server placement scheme. This is achievable by isolating key hash function properties from a specific implementation. The benefit will be a more flexible location-based multicast algorithm,

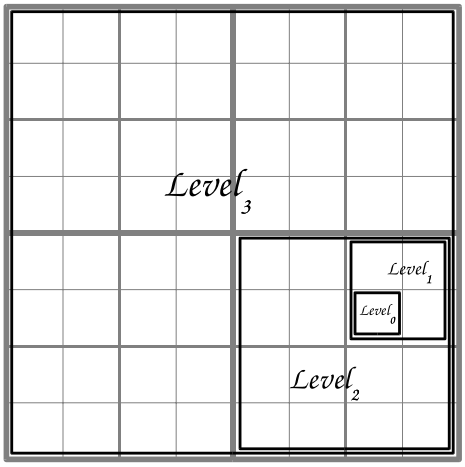
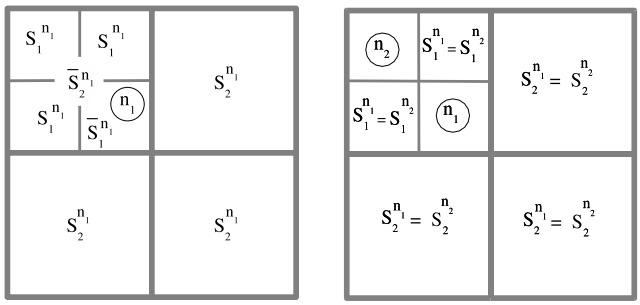



Fig. 3. Grid Partitioning Scheme



(a) Property 1: Partitioning Rule

(b) Property 2: Sharing Rule

Fig. 4. Hash Function Properties

since the hash function responsible for placing the location servers may be chosen to better suit a particular MANET topology.

### B. Model

The model for *AMDLM* is similar to GRID [6] and DLM [2]. Wireless nodes evolve in a geographic area partitioned into a hierarchical grid with squares of increasing size. The smallest region contains one square and is referred to as the $level_0$ region. Four $level_0$ regions form a $level_1$ region and so on, as shown in Figure 3. Regions do not overlap, so a $level_k$ region belongs to exactly one $level_{k+1}$ region (and thus nodes belong to exactly one region of each size). We further assume, as with position-based routing algorithms, that nodes are aware of their location through a positioning system such as GPS. Since the grid is static and predetermined, nodes know of their current region within the grid, as well as its boundaries. Nodes sharing the same $level_0$ region are called *neighbors*. All neighbors have knowledge of each other, even though they may not be within a 1-hop communication range (i.e. through flooding or token passing)[3]. Finally, similarly to DLM, we assume node density and distribution such that statistically over time, at least one node will be present per $level_0$ region (the grid dimensions may be chosen accordingly)[4].

Given two regions $reg_1$ and $reg_2$, we denote by $reg_1 \cup reg_2$ the geographic region that corresponds to the union of $reg_1$ and $reg_2$, by $reg_1 - reg_2$ the geographic region such that $reg_2$ is removed from $reg_1$; moreover $reg_1 \subset reg_2$ is true if $reg_1$ is included in $reg_2$.

### C. Logical Servers and their Placement

Given $k \geq 0$, we define $R_k$ as any $level_k$ region. For a node $n_i$, $R_k^{n_i}$ is the region $R_k$ such that $n_i \in R_k$. *Logical servers correspond to $level_0$ regions*: nodes happening to be in a given $level_0$ region that correspond to a logical server, participate in the service. Note that, as already explained, not every $level_0$ region is necessarily a logical server.

The logical servers are denoted by $S$. More precisely, we denote by $S_k^{n_i}(G)$ the logical server(s) in a region $R_k$ for node $n_i$ and group $G$. We further define $\overline{S}_k^{n_i}$ as being the $level_k$ location server located in $n_i$'s $level_{k-1}$ region. $S_k^{n_i}(G)$ will simply be denoted $S_k^{n_i}$ in the rest of the presentation. We now present the placement rules of our logical servers for a given group $G$.

*Property 1 (Partitioning Rule, see Figure 4(a)):*
$k = 0$: For each node $n_i$ member of $G$, there exists exactly one logical server of level 0 (denoted by $S_0^{n_i}$), which is region $R_0^{n_i}$.
$k > 0$: For each level $k > 0$ and for each node $n_i$, there exist exactly four logical servers of level $k$ (denoted by $S_k^{n_i}$). Moreover, there is exactly one of these four servers $S_k$ in each of the four sub-regions region $R_{k-1}^{n_i}$ of region $R_k^{n_i}$ (denoted by $\overline{S}_k^{n_i}$).

*Property 2 (Sharing Rule, see Figure 4(b)):*
$\forall k > 0$ and for two nodes $n_i$ and $n_j$ member of the same group, if $R_k^{n_i} = R_k^{n_j}$, then $n_i$ and $n_j$ share the same level $k$ server(s), i.e. $S_k^{n_i} = S_k^{n_j}$.

### D. Locating Logical Servers

Nodes determine the geographic position of the *logical servers* by means of a *hash function* that implements the placement rules presented in Section V-C. For a given multicast group $G$ and level $k > 0$ (i.e., for group $G$ and region $R_k$), the hash function returns the four $level_0$ regions corresponding to the four logical servers in $R_k$ for group $G$. Note that these $level_0$ regions are not necessarily logical servers, since the presence of logical servers depends on the presence or not of members of $G$ in region $R_k$. So a node $n_i$ — located in the level 0 region of address $a$ [5] — wanting to multicast a message to group $G$ first needs to find the smallest $k \geq 0$ such that there is a logical server of level $k$ in the $R_k$ region of $n_i$.

[3]The size of $level_0$ regions may also be chosen as to be fully covered by the transmission range of the nodes. In this case, the term *neighbors* still refers to two nodes sharing the same $level_0$ region and not to any two nodes within communication range.

[4]This assumption does not mean uniform distribution of group members!

[5]Each $R_0$ region can be uniquely identified with an address.

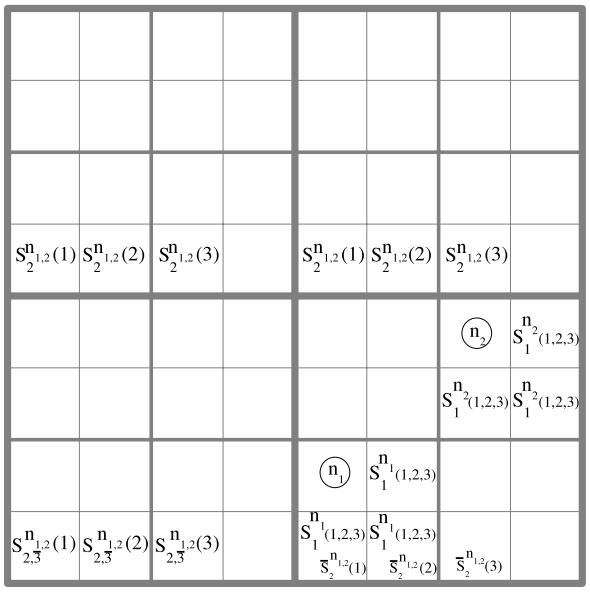



Fig. 5. Logical servers determined by the hash function. In the figure, $S_{2,\overline{3}}^{n_{1,2}}(1)$ denotes the $level_2$ $S$ logical servers and $level_3$ $\overline{S}$ logical servers for nodes $n_1$ and $n_2$ belonging to the multicast group 1. The $level_0$ and $\overline{S}_1^{n_i}$ servers are not shown.

This is achieved using the hash function, which returns the potential logical server for $G$ in region $R_k$ of $n_i$. Examples of how the hash function is used for updating logical servers and multicasting are given in Sections V-E and V-F.

The idea of how a hash function verifying the properties of Section V-C can be constructed is easy to understand based on the following example. The formal specification does not contribute to the intuition and can be found in [19], along with corresponding proofs.

*Hash function example:* The assignment of the $level_0$ regions designated to act as $level_l$ logical servers is determined fairly, by means of a round-robin policy based on the multicast group member identifier. Say we would like to obtain the ($level_0$) addresses of the four $level_2$ logical servers for a multicast group $G$. By the *Partitioning* rule, there will be one such server in each one of the four $level_2$ regions. Each $level_2$ region contains $2^{2\cdot 2} = 16$ $level_0$ regions (cf. Figure 3). In each $level_2$ region, the $level_0$ region whose address verifies $G \bmod 16$ is selected to serve as a $level_2$ logical server, ensuring a fair distribution of the service assignment across the groups. The *Sharing* rule further ensures that, for the same group, any two nodes belonging to the same $level_0$ or $level_1$ regions will share the same $level_2$ logical servers. Figure 5 shows the logical servers returned by the hash function for two nodes, $n_1$ and $n_2$, sharing the same $level_2$ region and belonging to multicast groups 1, 2 and 3.

### E. Joining and Leaving Groups

We now describe how nodes join and leave a group. The corresponding pseudo-code can be found in the appendix. Let node $n_1$ want to join a new group $G$ (Figure 6(a)). $n_1$ first issues a `join(G)` request to its $S_0^{n_1}$ (located in the same $R_0$ region than $n_1$, not represented in the figure), which will add an entry for $n_1$ as belonging to $G$. Then $S_0^{n_1}$ will in turn forward the $join$ to the three neighboring $S_1^{n_1}$ servers, activating them as $level_1$ *logical servers* for group $G$. Activating means that each $S_1^{n_1}$ stores a reference to $n_1$'s $level_0$ region. Through the hash function, the $S_1^{n_1}$ servers in turn forward the `join(G)` request to three $S_2^{n_1}$ servers. Requests to $S_{k+1}^{n_1}$ servers are forwarded by the geographically closest $S_k^{n_1}$ servers. As before, each $S_2^{n_1}$ stores a reference to the corresponding $S_1^{n_1}$ from which it received the `join(G)` request. This procedure is repeated on each level until the maximum level is reached.

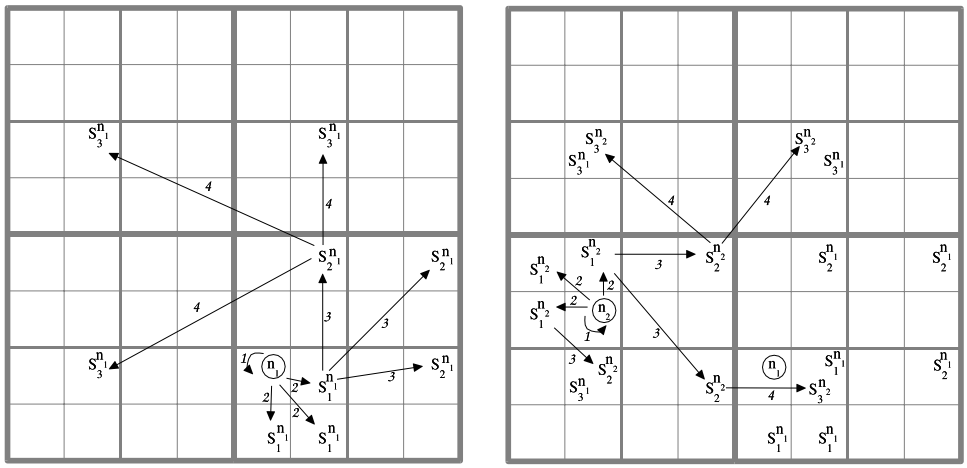
The leave operation is the reverse of the join operation (see Algorithms 1 and 2 of the appendix).

### F. Multicasting

We illustrate the principle of AMDLM on two examples, see Figures 7(a) and 7(b). In Figure 7(a), the source is member of the group to which it desires to multicast. Figure 7(b) illustrates a case where the source does not belong to the group. For sake of clarity, only the logical servers participating in the multicast examples are represented in the figures.

In Figure 7(a), node $n_2$ desires to multicast a message to group $G$, containing two members $n_1$ and $n_2$. To do so, it first delivers the multicast message to itself (since $n_2$ is its own $S_0^{n_2}$ server), as well as to $\overline{S}_2^{n_2}$ server. Through Property 2 of Section V-C, $\overline{S}_2^{n_2}$ knows of any group member within region $R_2^{n_2}$, saving the need for sending the request directly to the $S_2^{n_2}$ servers. Since in our case there are no members within $R_2^{n_2}$, the multicast request is forwarded to $\overline{S}_3^{n_2}$ (saving again the need for sending the request directly to the $S_3^{n_2}$ servers). Since $\overline{S}_3^{n_2}$ has an entry for a group member somewhere in a neighboring $R_2$ region (because $\overline{S}_3^{n_2}$ is also $S_3^{n_1}$), it will forward the request to the $S_2$ server it references, i.e. $S_2^{n_1}$. Upon receiving the request, $S_2^{n_1}$ will do the same. The procedure is repeated until the multicast message arrives at destination ($n_1$).

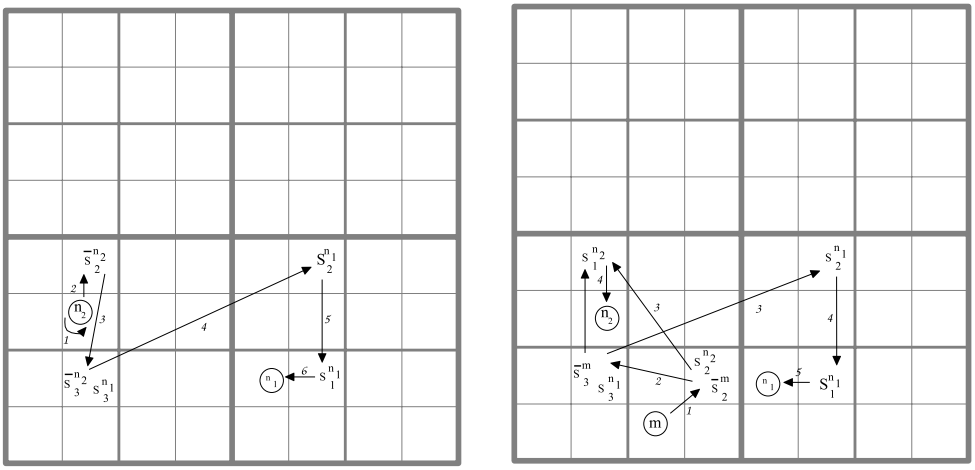
To the contrary of other MANET multicast algorithms mentioned in Section II, multicasting by nodes not belonging to the intended destination group is not more complex. In Figure 7(b), node $m$ desires to multicast a message to a group $G$, composed of nodes $\{n_1, n_2\}$. Had there been a member of $G$ in either $m$'s $level_0$ or $level_1$ regions, $m$ would have served as one of its logical servers and would have directly sent it the message, as a preliminary step. In all cases, $m$ forwards the request to the next upstream logical server that may possess group membership information — $\overline{S}_2^m$. In our example, $\overline{S}_2^m$ happens to be one of $n_2$'s $S_2$ servers. While $\overline{S}_2^m$ will now forward the multicast message to $n_2$, $\overline{S}_2^m$ has still to forward the request to $\overline{S}_3^m$. The reason being that if there are nodes to be reached in the neighboring $R_3$ regions, $\overline{S}_3^m$ will surely reference one of their $S_2$ servers. Indeed, in our case $\overline{S}_3^m$ is also $S_3^{n_1}$. The message will be forwarded to $n_1$ as described in the previous example.



(a) Node $n_1$ joining group $G$

(b) Node $n_2$ joining group $G$

Fig. 6. Join Operation



(a) Node $n_2$ multicasting message to group $G = \{n_1, n_2\}$

(b) Node $m$ multicasting message to group $G = \{n_1, n_2\}$

Fig. 7. Multicast Operation

### G. Mobility

We consider mobility of multicast group members. Mobility is handled as a *join* operation followed by a *leave* operation, with a $level$ parameter (see Algorithm 4 of the Appendix). The $level$ represents the highest region level the node has crossed during its journey. Straightforward for the node to compute, it allows to bound the propagation of the *leave* request. Consider a node $n_i$ changing $level_2$ regions. No action is taken when a $n_i$ starts moving [6]. Upon reaching its destination (i.e., node speed passes back below a given threshold), $n_i$ rejoins $G$ through a `join` request, with $level = 2$ passed as a parameter. In a second stage, $n_i$ must notify its original $level_0$ region that it has left by sending to it a bounded $leave$ request. Finally, the original $level_0$ region propagates the $leave$ message to remove $n_i$'s stale entry from the appropriate logical servers. Bounded `join` and `leave` are required since no assumption may be made on their order of reception at the logical servers. Indeed, if the `leave` is received before the `join`, logical servers will confuse mobility with the normal case of a node desiring to leave a group. The $level$ parameter solves this problem from preventing the `leave` (and afterward the `join`) from being wrongly propagated higher up in the logical server hierarchy. In our example, the requests are bounded to two levels. All

[6]While $n_i$ is moving and before any logical server has been updated, messages may be forwarded to it by following its *trail* — nodes $n_i$ encounters during its move store a forwarding pointer to it, à la [6].

higher-level logical servers need not be updated.

## VI. Quantitative Analysis: AMDLM vs. MDLM

In this section we study the performance of *AMDLM* compared to MDLM, the straightforward multicast extension of DLM presented in Section IV-A. The nature of *AMDLM* enables us to conduct an analysis not possible with other multicast algorithms such as MAODV [9] and ODMRP [10].

### A. Location Servers

*1) Total Number of Location Servers:* The number of required location servers has a direct impact on overhead. Given at least one group member, MDLM assigns $4^m$ uniformly distributed location servers, where $m$ is the desired density.

The total number of location servers for *AMDLM* dynamically grows and shrinks with respect to the number and distribution of multicast group members. The upper bound at any given time for the total number of location servers is calculated as follows. It is equal to the sum across levels, of the number of $level_k$ regions containing at least one group member and *not* belonging to the same $level_{k+1}$ region, multiplied by four: $4\Sigma_{k=1}^{k_{max}} \mid \{R_k \mid \exists n \in R_k, n \in G\} \mid$, where $k_{max}$ is the highest level. This result is an upper bound since we do not subtract from the result overlapping location servers across different levels, implied by Property 2 of Section V-C.

*2) Join and Leave Operations:* Let $n_i$ be the node desiring to join or leave a group $G$ and $n_j$ any member of $G$. When $n_i$ joins or leaves $G$, MDLM requires all $4^m$ location servers to be updated. *AMDLM* requires $4k$ location servers to be updated, where $k$ is the smallest region containing $n_i$ and another member of the group: $k =$ smallest $l$ such that $R_l^{n_i} = R_l^{n_j}$. Upon the first *join* (and last *leave*) all $4k_{max}$ servers are updated.

*3) Mobility:* Let $n_i$ be the mobile node and $k$ be the largest $level_k$ region traversed by $n_i$ upon moving. Both MDLM and *AMDLM* limit the number of location servers to be notified of $n_i$'s new position. *AMDLM* requires at most $2 * 4k$ location servers to be updated. Indeed, *AMDLM* handles mobility as combined *join* and *leave* operations ($n_i$ rejoins the group upon arriving to destination and sends a query to its previous $level_0$ region to issue a *leave* query on its behalf).

MDLM requires $4k$ location servers to be updated with $n_i$'s new position, where $k$ is determined by the density $m$ of location servers. For *AMDLM* $k$ is determined by the total number of $level_0$ regions in the grid. MDLM does not specify how information is removed from the location servers responsible for maintaining $n_i$'s position before having moved.

### B. Cost of Join and Leave Operations

We assume that there is sufficient connectivity in the network in order for any two adjacent $level_0$ regions to communicate. An upper bound on the total distance traveled by the *join* and *leave* operations may now be derived for *AMDLM* based on the number of levels[7]. For sake of simplicity, we consider the Manhattan distance.

[7]Without the assumption about density, we would have been bound by theoretic results obtained for the underlying geographic forwarding layer.

Since a $level_k$ region contains $4^k$ $level_0$ regions, the upper bound $\delta$ on the distance traveled within a $level_k$ region is $\delta(k) = 2\sqrt{4^k} - 2 = 2^{k+1} - 2$ $level_0$ regions (from one corner to the diagonally opposite one). To obtain the upper bound on the distance, we multiply $\delta$ by the number of queries required for each relevant operation, per level, for both algorithms (similarly to Section VI-A.1). We therefore obtain a cost of $4\Sigma_{l=1}^{k}\delta(l)$ for the join/leave operations and $2(4\Sigma_{l=1}^{k}\delta(l))+\delta(l)$ in case of mobility.

### C. Discussion

The recurring dilemma for a location management service in MANETs is the dissemination and maintenance cost of location information versus the accessibility and quality of the information upon retrieval. *AMDLM* addresses this dilemma in the context of multicast by concentrating the effort of maintaining location servers nearby multicast group members, while offering minimal location service in regions with little or no group members.

MDLM's uniform location server distribution, while a reasonable assumption for unicast routing, does not offer the flexibility needed for an efficient access to multicast group information without the overwhelming overhead of maintaining the same quality of service across the entire area of the grid.

## VII. Qualitative Comparison Between the Different Approaches for MANET Multicasting

In this section we conduct a qualitative analysis of the performance of *AMDLM*, MAODV and ODMRP under different factors based on the core design of each algorithm.

The main design choices for the underlying group maintenance structures in MANET multicast algorithms are *tree* and *mesh*. We compare *AMDLM* against the main representant of each approach, MAODV [9] and ODMRP [10]. All these protocols are on-demand.

### A. No Mobility

We first examine the behavior of the various protocols for static networks — nodes do not move.

*1) Managing concentrated group members:* ODMRP does not maintain multicast group membership when no node is interested in multicasting, generating no network activity whatsoever. Both MAODV and AMDLM are expected to exhibit similar behavior in terms of multicast group management. Nevertheless, AMDLM generates additional overhead, since it forwards group member location information (to a bounded number of designated location servers).

*2) Managing scattered group members:* In absence of multicast senders, network activity for ODMRP is non-existent. MAODV actively maintains a multihop tree connecting multicast group member nodes. For few nodes scattered over a large geographic area, the number of non-member nodes required to maintain the multicast tree greatly outnumbers the number of member nodes. In case of numerous group members, MAODV generates important overhead, since it constantly maintains a

multicast tree connecting all members and periodically generates *group hello* packets. AMDLM assigns a bounded number of location servers responsible for managing multicast group membership (See Section VI). A judicious grid decomposition adapted to the number of nodes is required to achieve optimal overhead.

*3) Multicasting:* Since ODMRP does not actively manage group membership, mesh construction is required whenever nodes wish to multicast. Sender nodes create a mesh reaching multicast group members through periodic flooding. ODMRP does not scale well with respect to numerous senders or receivers (multicast group members) [18]. The characteristic behind its robustness (mesh structure offering alternative paths) is that of its high overhead. Indeed, ODMRP generates nearly as much overhead as pure flooding by disseminating data across the forwarding group (*scoped flooding*). With group members scattered over large regions, ODMRP will ultimately lead to network performance degradation similar to the broadcast storm problem discussed in [20].

In MAODV and AMDLM, multicast group members wishing to multicast have immediate access to routing information. For non-member nodes, MAODV requires flooding in order to reach a node on the multicast tree to obtain routing information. AMDLM induces the least overhead among the protocols, since flooding is not required at any stage. Routing information may at any time be retrieved from the logical servers referenced by the hash function.

### B. Mobile Nodes

We now introduce mobility to the analysis. The main factor on performance is the frequent link breakages caused by mobility (node crashes may therefore be considered as a particular form of mobility).

*1) MAODV:* Group members are reachable through a bidirectional multicast tree, providing a single path composed of critical links to reach multicast group members. Node mobility increases the occurrence of link breakages, damaging MAODV's tree structure and leading to reduced multicast efficiency. Furthermore, the tree-based approach may even generate great overhead by incessantly reacting to broken links and initiating repairs. The more scattered the group members, the greater the chance for link breakages to occur. Even a single link breakage may obstruct the path to group members and force MAODV to repair the tree. The physical tree structure therefore proves highly fragile.

*2) ODMRP:* It has been shown in [16], [18] that an increase in mobility has little impact on the efficiency of the algorithm. These results may be explained by two key characteristics of ODMRP. Firstly, ODMRP periodically recreates routes to destination nodes while sender nodes have packets to multicast, maintaining route freshness (although at the expense of increased overhead). Secondly, the underlying mesh structure is more robust than the tree approach by offering alternative paths to reach the multicast group members.

*3) AMDLM:* A location-based scheme offers relative independence from node mobility. Indeed, the logical tree is formed by nodes in geographic regions (logical servers) designated by the hash function. These geographic regions are fixed. Nevertheless, the time required for the relevant logical servers to be updated through geographic forwarding increases the latency of the multicast algorithm, i.e. the overall time required to locate and reach multicast group members. Furthermore, packet loss may occur if the location information in the logical servers is stale.

### C. Additional Factors

AMDLM uses the logical servers not only for storing group membership but also for forwarding packets to group members. The decision to use logical servers for routing reduces the overall latency. The greater number of packets transiting through the geographic regions responsible for managing group membership may however cause possible bottlenecks under high network traffic.

MAODV and ODMRP are independent from geographic concerns and may seamlessly function under reasonable arbitrary node distribution. AMDLM on the other hand expects to find one or more nodes in the regions designated by the hash function to manage multicast group membership. AMDLM therefore benefits from a hash function tailored to the node topology, by favoring densely populated areas for the placement of the logical servers.

### D. Summary

By adopting a significantly different approach for multicast group management, *AMDLM* can offer greater robustness and reliability than the tree-based approach of MAODV without the associated overhead induced by the mesh-based approach of ODMRP.

## VIII. Summary and Future Work

We have presented a location service specifically tailored for multicasting in MANETs. We have additionally specified *AMDLM*, a multicast algorithm built on top of the location service. The advantages of such an approach over multicast algorithms relying on a tree, DAG or mesh approach is the absence of a multihop data-structure connecting the source and members of the multicast groups. Such data-structures are indeed less robust to frequent link breakages that occur in the highly dynamic environment of mobile ad hoc networks.

Designing an efficient location service in the context of multicast within MANETs is a difficult task. To the contrary of unicast routing, the location service must efficiently maintain information regarding a *set* of scattered nodes associated with each group. Maintaining accessible multicast group membership across the network may quickly lead to high communication overhead, reducing performance and wasting limited resources. *AMDLM* addresses the dilemma of the location service overhead versus multicast group information accessibility by dynamically adjusting the density of the location servers.

For future work, we intend to compare *AMDLM* through simulation to MAODV [9] and ODMRP [10] in order to fully comprehend the performance of each approach in various

situations and validate the qualitative analysis presented in Section VII.

## REFERENCES


[1] J. Jubin and J. D. Tornow, "The DARPA packet radio network protocol," vol. 75, no. 1, pp. 21–32, Jan. 1987.

[2] Y. Xue, B. Li, and K. Nahrstedt, "A scalable location management scheme in mobile ad-hoc networks," in *In in Proceedings of the 26th IEEE Annual Conference on Local Computer Networks (LCN 2001)*, Tampa, Florida, Nov 2001, pp. 102–111.

[3] B. Karp and H. T. Kung, "Greedy Perimeter Stateless Routing for Wireless Networks," in *Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, Boston, MA, Aug. 2000, pp. 243–254.

[4] I. Stojmenovic, "Position based routing in ad hoc networks," *IEEE Commmunications Magazine*, vol. 40, no. 7, pp. 128–134, Jul 2002.

[5] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A distance routing effect algorithm for mobility (DREAM)," in *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM-98)*. New York: ACM Press, Oct. 1998, pp. 76–84.

[6] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, Aug. 2000.

[7] M. Mauve, H. Füssler, J. Widmer, and T. Lang, "Position-based multicast routing for mobile ad-hoc networks," University of Mannheim, Tech. Rep. TR-03-004, 2003.

[8] Y.B-Ko and N. Vaidya, "Geocasting in mobile ad hoc networks: Location-based multicast algorithms," in *Proceedings of the IEEE WMCSA'99*, New Orleans, LA, Feb 1999, pp. 101–110.

[9] E. M. Royer and C. E. Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," in *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking*. ACM Press, 1999, pp. 207–218.

[10] S.-J. Lee, C. Chiang, and M. Gerla, "On-demand multicast routing protocol," in *Proceedings of the IEEE WCNC'99*, New Orleans, USA, Sep 1999, pp. 1298–1304.

[11] E. L. Madruga and J. J. Garcia-Luna-Aceves, "Scalable Multicasting: The Core Assisted Mesh Protocol," *ACM/Baltzer Mobile Networks and Applications, Special Issue on Management of Mobility in Distributed Systems*, vol. 6, no. 1, 2001.

[12] M. Liu, R. R. Talpade, and A. McAuley, "AMRoute: Adhoc Multicast Routing Protocol," The Institute for Systems Research, University of Maryland, Tech. Rep. 99, 1999.

[13] C. W. Wu and Y. C. Tay, "AMRIS: A Multicast Protocol for Ad hoc Wireless Networks," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, Atlantic City, NJ, Nov. 1999, pp. 25–29.

[14] V. Devarapalli and D. Sidhu, "MZR: A multicast protocol for mobile ad hoc networks," in *IEEE International Conference on Communications (ICC)*, Helsinki, Finland, June 2001.

[15] C. Ho, K. Obraczka, G. Tsudik, and K. Viswanath, "Flooding for reliable multicast in multi-hop ad hoc networks," in *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Seattle, WA, Aug. 1999, pp. 64–71.

[16] S.-J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia, "A performance comparison study of ad hoc wireless multicast protocols," in *Proceedings of IEEE Infocom 2000*. Tel-Aviv, Israel: IEEE, Mar 2000, pp. 565–574.

[17] K. Obraczka, G. Tsudik, and K. Viswanath, "Pushing the limits of multicast in ad hoc networks," in *Proceedings of the IEEE 21st International Conference on Distributed Computing Systems (ICDCS'01)*, Mesa, Arizona, Apr 2001.

[18] T. Kunz and E. Cheng, "On-demand multicasting in ad-hoc networks:comparing AODV and ODMRP," in *22nd International Conference on Distributed Computing Systems (ICDCS '02)*. Washington - Brussels - Tokyo: IEEE, July 2002, pp. 453–454.

[19] Y. Sasson, D. Cavin, and A. Schiper, "A location service mechanism for position-based multicasting in wireless mobile ad hoc networks," EPFL, Tech. Rep. 200329, May 2003.

[20] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Aug. 1999, pp. 151–162.

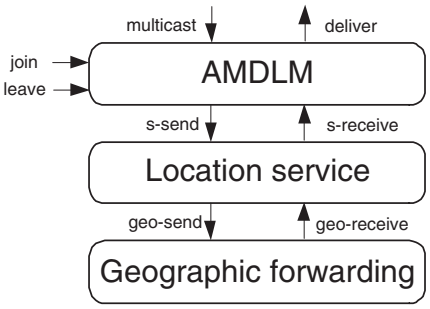




Fig. 8. AMDLM Architecture


[21] M. Mauve, J. Widmer, and H. Hartenstein, "A survey on position-based routing in mobile ad-hoc networks," *IEEE Network Magazine*, vol. 15, no. 6, pp. 30–39, Nov 2001.


## APPENDIX

### *AMDLM Algorithm Pseudo-Code*

Figure 8 represents the layers that together provide the AMDLM multicast service. At the lowest level, the service relies on a geographic forwarding layer ([21], [4]), providing a geographic-based point-to-point unicast by means of a `geo-send` and corresponding `geo-receive` procedures. Above the geographic forwarding layer resides the *logical servers* abstraction. Its interface provides the `s-send` and `s-receive` communication primitives.

Algorithms 1, 2 and 3 describe the *join*, *leave* and *multicast* operations. Algorithm 4 shows how the mobility is handled when a node moves from one $R_0$ region to another. Finally, Algorithm 5 describes the interaction between the logical server and underlying geographic forwarding layers. Every logical server maintains two data structures: (1) $members$ for storing the multicast group membership and (2) $links$, for storing references to active logical servers. Both $members$ and $links$ are initially empty.

**Algorithm 1** : AMDLM - Join

1: {Upon `join(G)` by a node $n_i \in S_0$ :}
2:
3: `s-send(join,`$k_{max}$`,G,`$S_0^{n_i}$`)` to $S_0^{n_i}$;
4:
5: {Upon `s-receive(join,level,G,`$S_l^{n_j}$`)` by a node $n_i \in S_k$}
6:
7: **if** k==0 **and** $(G, n_j) \notin members_i$ **then**
8: $members_i = members_i \cup \{(G, n_j)\}$;
9: **if** $\mid members_i \mid == 1$ **then**
10: `s-send(join,level,G,`$S_0^{n_i}$`)` to $S_1^{n_i}$;
11: **end if**
12: **else if** $(G, S_{n_j}^l) \notin links_i$ **then**
13: $links_i = links_i \cup \{(G, S_l^{nj})\}$;
14: **if** $k < level \leq k_{max}$ **and** $\mid links_i \mid == 1$ **then**
15: `s-send(join,level,G,`$S_k^{n_i}$`)` to $S_{k+1}^{n_i}$;
16: **end if**
17: **end if**

**Algorithm 2** : AMDLM - Leave

{Upon `leave(G)` by a node $n_i \in S_0$ :}

`s-send(leave,`$k_{max}$`,G,`$S_0^{n_i}$`)` to $S_0^{n_i}$;

{Upon `s-receive(leave,level,G,`$S_l^{n_j}$`)` by a node $n_i \in S_k$}

**if** k == 0 **and** $(G, n_j) \in members_i$ **then**
  $members_i = members_i \setminus \{(G, n_j)\}$;
  **if** $members_i == \emptyset$ **then**
    `s-send(leave,level,G,`$S_0^{n_i}$`)` to $S_1^{n_i}$;
  **end if**
**else if** $(G, S_l^{n_j}) \in links_i$ **then**
  $links_i = links_i \setminus \{(G, S_l^{n_j})\}$;
  **if** $links_i == \emptyset$ **and** $k < level \leq k_{max}$ **then**
    `s-send(leave,level,G,`$S_k^{n_i}$`)` to $S_{k+1}^{n_i}$;
  **end if**
**end if**

**Algorithm 3** : AMDLM - Multicast

{Upon `multicast(m,G)` by a node $n_i \in S_0$ :}

**for all** $(G, n_j) \in members_i$ **do**
  **deliver** m to $n_j$;
**end for**
**for all** $(G, S_k^{n_j}) \in links_i$ **do**
  `s-send(m,G,`$S_0^{n_i}$`)` to $S_k^{n_j}$;
**end for**
**if** $k_{max} \geq 2$ **then**
  `s-send(m,G,`$S_0^{n_i}$`)` to server $\overline{S}_2^{n_i}$;
**end if**

{Upon `s-receive(m,G,`$S_l^{n_j}$`)` by a node $n_i \in S_k$}

**if** k == 0 **then**
  **for all** $(G, n_h) \in members_i$ **do**
    **deliver** m to $n_h$;
  **end for**
**else if** $links_i \cap \{(G, S_{k-1}^{n_h})\} \neq \emptyset$ **then**
  `s-send(m,G,`$S_k^{n_i}$`) to server` $S_{k-1}^{n_h}$;
**end if**
**if** $l \leq k < k_{max}$ **then**
  `s-send(m,G,`$S_k^{n_i}$`)` to $\overline{S}_{k+1}^{n_i}$;
**end if**

**Algorithm 4** : Mobility management

{Upon node $n_i$ has moved from $S_0^{old}$ to $S_0^{n_i}$:}

{$n_i$ must update $members_i$ and $links_i$ sets according to its new position}
**if** $members_i \cap \{(G, n_i)\} \neq \emptyset$ **then**
  level = minimum k such that $R_k^{R_0} = R_k^{R_0'}$;
  `s-send(join,level,G,`$S_0^{n_i}$`)` to $S_0^{n_i}$;
  `s-send(leave,level,G,`$S_0^{n_i}$`)` to $S_0^{old}$;
**end if**

**Algorithm 5** : Logical server algorithm

{Upon `s-send(content,G,`$S_k^{n_i}$`)` to a server $S_l^{n_j}$ by a node $n_i \in S_k$ :}

Obtain $S_l$ through the hash function;
Obtain $S_k$ through the hash function;
**for all** $R_0 \in S_l$ **do**
  **if** $dist(R_0^{n_i}, R_0) < dist(R_0', R_0), \forall\, R_0' \neq R_0^{n_i} \in S_k$ **then**
    `geo-send(content)` to $R_0$;
  **end if**
**end for**

{Upon `geo-receive(content)` by a node $n_i \in S_k$ from a server $S_j$ :}

`s-receive(content,`$S_j$`)`;