Stable Clustering Through Mobility Prediction for Large-Scale Multihop Intelligent Ad Hoc Networks

S. Sivavakeesar
Center for Communication Systems Research
University of Surrey, Guildford GU2 7XH
United Kingdom
S.Sivavakeesar@eim.surrey.ac.uk

G. Pavlou, A. Liotta
Center for Communication Systems Research
University of Surrey, Guildford GU2 7XH
United Kingdom
{G.Pavlou, A.Liotta}@eim.surrey.ac.uk

Abstract— **In this paper we present a framework for dynamically organizing mobile nodes (MNs) in large-scale mobile ad hoc networks (MANETs), with the eventual aim to support Quality of Service (QoS). Our dynamic, distributed clustering approach is based on intelligent mobility prediction that enables each MN to anticipate the availability of its neighbors. We present a scalable way to predict the mobility, and thus availability, of MNs, achieved with the introduction of geographically-oriented *virtual clusters*. We name the proposed model as the (p, t, d)-clustering model that facilitates the formation of stable clusters. Simulation results demonstrate the performance advantages of our approach.**

Keywords- Ad-hoc networking; Mobility prediction; Hierarchical clustering, QoS.

I. INTRODUCTION

Wireless communication and the lack of centralized administration pose numerous challenges in MANETs [1]. Node mobility results in frequent failure and activation of links, causing a routing algorithm reaction to the changes in topology and hence increasing network control traffic. Ensuring effective routing and QoS support while considering the relevant bandwidth and power constraints remains a great challenge. Given that MANETs may comprise a large number of MNs, a hierarchical structure will scale better [2]. This fact has made researchers focus their attention in partitioning the multihop network into clusters, and electing cluster heads (CH) [5][6][7][8]. This clustering technique brings in a number of benefits as stated in [4]. It is expected that future generation wireless networks will evolve towards non-authority based, self-organized, large-scale MANETs, which will have a significant impact on future communication models and m-business. In this work we envisage large-scale deployment of long-term multihop MANETs, which is similar but complementary to mobile telephony systems. The network model considered in this work is, thus, similar to that assumed in the 'Terminodes' project [3]. We adopt a hierarchical clustering approach, which is fully distributed and dynamic in nature. Our approach differs from other similar approaches in two important aspects: a cluster head is elected based on mobility prediction and we also introduce the concept of *virtual clusters*. Location information may be obtained using the Global Positioning System (GPS), or a self-positioning algorithm as specified in [3]. Our mobility prediction approach is derived from data compression techniques that are both theoretically optimal and good in practice. Although making prediction based on accumulated history demands storage and calculation capacity, these are affordable if we can limit the need for updates as much as possible while maintaining up-to-date topology information. This way we can limit the waste of scarce wireless bandwidth and transmission power. Our work is further motivated by the fact that both memory and processing power continue to get cheaper.

II. PREVIOUS WORK AND OUR MOTIVATION

While many clustering techniques with CH selection have been proposed in the literature, almost none of them consider node mobility as a criterion in the clustering process effectively [5][6][7][8]. As a result, they fail to guarantee a stable cluster formation. In a MANET that uses cluster-based services, network performance metrics such as throughput and delay are tightly coupled with the frequency of cluster reorganization. Therefore, stable cluster formation is essential for better QoS. The most popular clustering algorithms available in the literature are the lowest identifier (Lowest-ID) and maximum-connectivity [4][8]. But these two, along with others, do not provide a quantitative measure of cluster stability. In the former, a highly mobile lowest ID CH will cause severe re-clustering; in addition, if this CH moves into another region it may unnecessarily replace an existing CH, causing transient instability. In the latter, depending on MN movement and traffic characteristics, the criterion values used in the election process can keep on varying, and hence also result in instability. This is also the case in the Lowest Distance Value (LDV) and the Highest In-Cluster Traffic (ICT) approaches [5]. Another scheme referred to as (∝, t)-clustering focuses on mathematical characterization of the probability of link and path availability as a function of a random walk based mobility model [6]. In the latter, it is considered that a link is active between two MNs at time $t_1 + t_0$ ($t_1 > 0$) given that there is an active link between them at time t_0. This scheme leads to ambiguity as to how big t_1 is and also it does not consider events that might have happened in the interval $t_1 + t_0$. A clustering scheme based on a mobility-metric is proposed in [7]. Since this bases the CH selection criteria on received power measurements, its accuracy depends heavily on how well a varying channel condition is modeled and, as such, it is not optimal. Our approach is motivated by the fact that in MANETs link bandwidth and MN transmission power are scarce, and any effective solution should take this into account and try to conserve them [1]. Given we are eventually seeking a QoS solution, MNs should be able to predict the availability

of network resources in order to support QoS. This in turn necessitates that each MN should have up-to-date information on network topology with minimal control traffic overhead. In order to achieve a compromise between these two extremes, accurate prediction of future state is necessary for the network control algorithms to keep pace with rapid and frequent state changes. Since MN movement is the main cause of uncertainty, we propose a scalable mobility prediction scheme based on the accumulated past behavior history of a specific MN. In typical mobile networks and in the type of longer-term MANET that we consider in our work, MNs exhibit some degree of regularity in their mobility pattern and this is what we are trying to use in our proposed solution [9][12].

III. (p, t, d)-CLUSTERING MODEL

Having taken into account the common deficiencies of other approaches, our algorithm selects a MN as CH, if it satisfies the following criteria: 1) it has the highest probability, in comparison to other MNs within the same *virtual cluster*, to stay for longer time within that cluster (see section A below), 2) it has the minimum distance from the respective *virtual cluster* center (VCC). The first requirement tries to ensure that a highly mobile MN is not elected as a CH. The second is to ensure that by being located very closed to a VCC, the CH can have a uniform coverage over a specific *virtual cluster*. This in turn ensures that in subsequent CH changes, the area covered would not be impaired. We name our model the (p, t, d)-clustering model. More accurately it is (p_{xk}, t_{xk}, d_{xk})-model, where 'p_{xk}' is the probability that x^{th} MN within k^{th} *virtual cluster* having a distance 'd_{xk}' from the center of that cluster stays within the cluster for some specific time period 't_{xk}' (residence-time). Any x^{th} MN within k^{th} *virtual cluster*, having $p_{xk} \geq p_{max}$, for $t_{xk} \geq t_{yk} \geq t_c$ (where 'x' ≠'y') and $d_{xk} \leq d_{min}$, can become a head of that *virtual cluster* - equation (2) introduced later is used in this process. Here 'p_{max}', 'd_{min}' and 't_c' are system dependent, and 'p_{xk}'and 't_{xk}' are determined based on the mobility prediction model described in section B. The necessary ingredients of the (p, t, d)-model, which are 1) The concept of *virtual clusters*, 2) Mobility prediction model, 3) Clustering algorithm and protocol, are explained below.

A. The Concept of Virtual Clusters

In order to make our mobility prediction viable, and our clustering mechanism scalable, we introduce the notion of *virtual clusters*. The idea is that a geographical area (or even the whole earth) is divided into equal regions of circular shape in a systematic way that each MN can determine the circle it resides in if location information is available. Each circular region is centered on a *virtual cluster* center (VCC) as depicted in Fig. 1. These VCCs are associated to a particular region in such a way that the resulting *virtual clusters* are nearly overlapping. These circular regions are our *virtual clusters*; a *virtual cluster* becomes an actual cluster if MNs exist in it. Unlike in other clustering scheme discussed in section II, in our approach each *virtual cluster* has a unique identifier based on the geographic location, which can be calculated using a publicly known function [3]. It is necessary that each *virtual cluster* have a unique identifier for our mobility prediction algorithm to work in a scalable way.

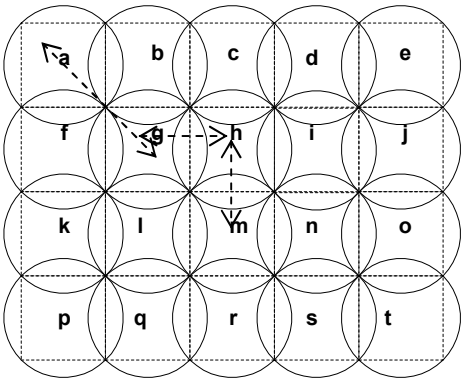

Figure 1. Concept of Virtual Clusters.

Each MN is supposed to have a complete picture of the locations of VCCs. This information can be either embedded in a MN at the time of manufacture or a MN may obtain it by accessing a common location service. In our context, each VCC is assumed to be away from each other by a fixed distance (which is not exactly true due to the earth's spherical shape). In order to make our mobility prediction analysis simpler, the MANET is modeled as a connected graph G = (V, E), where the vertex-set V represents the *virtual cluster*, and the edge-set E represents the adjacency between pairs of *virtual cluster*.

B. Mobility Prediction Model

1) User Mobility Pattern: Mobility management is important in wireless networks in order to route packets according to users' locations. This becomes extremely difficult in a MANET, but mobility management is still imperative for effective routing. This subsection presents how this can be achieved in a novel way without too much complexity and waste of bandwidth and transmission power. It is important at this point to distinguish between movement-history and mobility-model [12]. The former is past and deterministic, whereas the latter is probabilistic and extends to future. Since the location-tracking problem is user-oriented by definition, a tacit assumption is that a user's movement is a reflection of the patterns of his life. As such, we try to make use of the personal mobility of each user [9][12]. Users tend to have favorite routes and habitual movement patterns, and those can be learned. Learning aids decision-making when reappearance of those patterns is detected, given that 'history repeats itself' [9][12]. Our model exploits this, and attempts to derive probabilistic prediction of particular user mobility by utilizing his accumulated movement-history. Our mobility prediction scheme is motivated by computational learning theory, which has shown that prediction is synonymous with data compression. The movement history of a MN is represented by a string '$\nu_1,\nu_2, \nu_3\ldots$' of symbols where V = $\{\nu_1, \nu_2, \nu_3\ldots\nu_n\}$ is the set of virtual clusters and ν_i denotes the virtual cluster id (VID), and such strings are generated by using a combination of time-based and movement-based tracking schemes [9][12]. In time-based schemes, tracking

takes place periodically, whereas in the movement-based schemes tracking takes place whenever virtual cluster crossing is detected. The proposed scheme attempts to create a dictionary of virtual cluster ids, which are treated as character symbols, and uses the dictionary to gather statistics based on movement history contexts, or phrases. The proposed mobility prediction algorithm is based on the Ziv-Lempel (LZ78) algorithm for data compression, which is both theoretically optimal and good in practice [9][12]. There have been similar algorithms developed, e.g. LeZi-update, in order to minimize updating cost in regular cellular networks [12]. Our approach differs from similar algorithms in the literature, because in ours each MN is responsible for generating the strings of VIDs and maintaining its respective dictionary in its memory. In addition to making predictions as to future movements of a particular MN, our model is used by each MN to predict its approximate residence-times of the virtual clusters it visits.

2) Mobility Prediction Model: The need for finding a universal variable-to-fixed coding scheme gave rise to the emergence of the LZ78 algorithm, where the coding process is interlaced with the learning process for the source characteristics [9][12]. The learning process is aided with the de-correlating process, which works by efficiently creating and looking up an explicit dictionary. This algorithm parses the input string '$\nu_1, \nu_2, \ldots \nu_n$' ($\nu \in V$) into c(n) distinct substrings $\omega_1, \omega_2, \ldots, \omega_{c(n)}$ such that for all $j \geq 1$, the prefix of substring ω_j is equal to some ω_i, for $1 \leq i < j$ [12]. Because of this prefix property, substrings parsed so far can be efficiently maintained in a multiway tree or trie [9]. Since in our model, each MN is responsible for generating and constructing such tries in real-time, depending on its movement and time, each MN will act as an encoder. Since each MN is expected to find its residence-time in each virtual cluster it visits from its own trie, it will also function as a decoder. Accordingly, since each MN maintains its mobility database at a specific time in terms of a trie, we name such a trie Mobility Trie [9]. Each leaf except the root in the trie preserves the relevant statistics that can be used to predict the probabilities of following events.

```
initialize Mobility_Trie := null
initialize phrase ω := null
initialize Num_of_Event := 0
loop
  wait for next event (symbol) ν
  if (ω.ν in Mobility_Trie)
    Num_of_Event := Num_of_Event + 1
    ω := ω.ν
  else
    create a leaf ν
    encode <index(ω), ν>
    ω := ν
  endif
  calculate the probabilities of
  possible events based on the
  Num_of_Event of leaves
forever
```

Figure 2. Encoder of a Mobile Node.

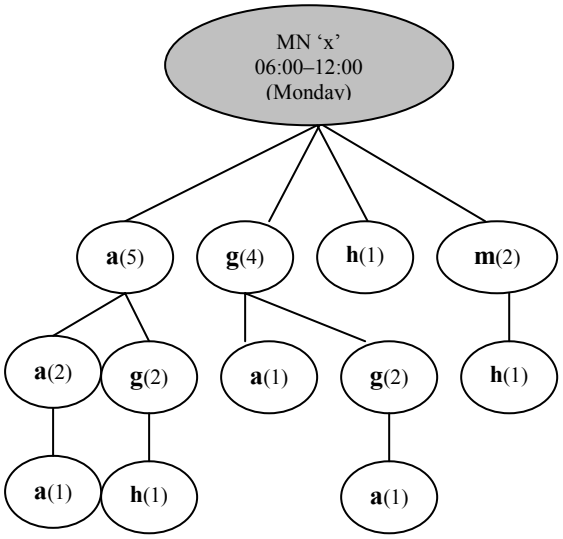

Figure 3. A Mobility Trie of a Specific Mobile Node 'x'.

When a MN is switched on for the first time in a specific day of a week, the encoder (or predictor) of that MN would initialize the root of the trie according to the time and VID, and should be able to calculate the probabilities of all possible future events associated with that MN. In our model, an event occurs due to either the time-based or the movement-based updating. On seeing the actual event, the predictor of each MN walks down the trie and is ready to predict again. When an event is not in the *Mobility Trie*, a prediction fault is generated and the trie is updated by adding a leaf. Fig. 2 outlines the greedy parsing technique of classical LZ78 as used in our context, and the dot represents concatenation [12].

In addition to representing the dictionary, the *Mobility Trie* can store statistics for contexts explored. A path from the root to any leaf ω in the trie represents a context. Fig. 3 shows an example trie formed while parsing the movement-history "aaagaggggaaghhmmhgaaaa…" – obtained from Fig. 1 – as "a, aa, g, ag, gg, gga, agh, h, m, mh, ga, aaa, …". The commas separate the parsed phrases and indicate the points of trie updates. As the process of incremental parsing progresses, larger and larger phrases accumulate in the *Mobility Trie*. Consequently estimates of conditional probabilities for larger contexts start building up. Intuitively, it would gather the predictability or richness of higher and higher order Markov Models [11][12]. In other words, by modeling the sequence of events (symbols) generated during a specific time duration (for e.g. a specific day of a week) as those generated by a stationary r^{th} order Markov source, and predicting next events using the mobility prediction scheme derived from the LZ78 algorithm, we can predict not only which *virtual cluster* a MN will visit but also the approximate residence-time in it .

Let ξ, $N(\omega)$, $L(\omega)$, $\delta^k(\omega)$, $\rho(\omega)$ and Λ denote the last updated phrase, number of occurrences of a phrase ω, its length, k^{th} suffix, prefix, and null phrase respectively. The probability of any phrase φ can be estimated by the recursive formula: $\Pr[\varphi] = \frac{N(\varphi \mid \rho(\delta^k(\xi)))}{\sum_{\omega} N(\omega \mid \rho(\delta^k(\xi)))} +$

$\frac{N(\Lambda \mid \rho(\delta^k(\xi)))}{\sum_{\omega} N(\omega \mid \rho(\delta^k(\xi)))}$ X $\Pr[\delta^1(\rho(\delta^k(\xi)))]$. In this way, the blending probabilities, associated with the occurrences of next possible VIDs on the path segment to be reported by the next update event are calculated if the movement history context is known [12]. This technique is here used by every MN 'x' to calculate the state probability (p_{xk}) for it to stay in *virtual cluster* 'k'.

The time-interval (T_e) at which update events are triggered based on the time-based updating, and the radius (R) of *virtual cluster* are two important parameters, and thus determines the accuracy of prediction, and hence the performance of our clustering algorithm. The shorter the 'T_e', the greater the accuracy of the residence-time, but the higher the tracking overhead. Similarly, the smaller the 'R', the better the prediction capability, but the higher the tracking overhead. Therefore, a compromise decision is necessary, when selecting values for these two parameters.

C. Clustering Algorithm and Protocol

In this clustering process the *Mobility Trie* each MN constructs plays an important role. As explained in section B.ii, each leaf (or set of leaves) of the trie enables every MN 'x' to determine its residence-time (t_{xk}) in *virtual cluster* 'k' and p_{xk}, if the movement history context is known. In this scheme, the decision as to the next CH to be selected depends on whether the current CH is available or not. If there is a primary CH or deputy CH available, they will make the decision on behalf of all MNs after getting the members' Ω values (see equation (2) below). On the other hand, if any of them does not exist, because of abrupt failure or error in prediction, then MNs within the *virtual cluster* will elect one as their CH in a distributed manner; the cluster formation time in the latter case is higher. The MN that has the highest Ω can become the CH. In forming clusters, the CH has to make sure it can cover the whole area of the *virtual cluster*. Therefore, the CH makes a k-hop cluster where value 'k' is not necessarily uniform within the cluster in terms of distance between any border MN and the CH. Some of the terms used in this paper, such as '*HELLO*' message, adjacent cluster, and gateway MNs, are similar to those specified in [7], [8].

Each *HELLO* message, periodically broadcast by the CH – say every CH_HELLO_INTERVAL – carries the ID of the *virtual cluster* (VID) it covers, the VCC, the cluster's radius and the neighbor-table, the latter being the set of cluster members [8]. Whenever a new MN receives this message from a CH, it can send a *JOIN* message immediately, if it is within the *virtual cluster*. The new MN includes in the *JOIN* message its approximate residence-time (t_{xk}) within its current *virtual cluster* 'k' (by calculating it from its own *Mobility Trie*), its location information, and its *Mobility Trie* corresponding to the next 'T_{mt}' minutes. This system parameter 'T_{mt}' should take an appropriate optimal value. Whenever a CH receives a *JOIN* message, it checks first if the MN is within its *virtual cluster*. If it is, the CH includes it in the cluster, and appends its information to the neighbor-table. The exception to this case is when the MN's expected residence-time within the cluster is minimal. If, on the other hand, the MN is not within the *virtual cluster*, it will simply not be included. In either case, the MN has to wait for at least the next two successive CH_HELLO_INTERVAL periods to check whether it has been included. If not, it has to re-transmit the *JOIN* message. From the periodic neighbor-table that a CH broadcasts, each member of a cluster can build its own neighbor-table. A MN can be a member of up to four maximum adjacent *virtual clusters*. This specific MN would then behave as gateway or forwarder between those clusters [5][8]. Having become a member, each MN within a particular *virtual cluster* is supposed to disseminate a *HELLO* message to its respective CH periodically – say every MN_HELLO_INTERVAL, where MN_HELLO_INTERVAL > CH_HELLO_INTERVAL. In the *HELLO* message, each member specifies if it is acting as a gateway or as an ordinary node. These control messages are relayed by intermediate MNs only within the *virtual cluster*. On the other hand, periodic *HELLO* messages by CHs are unicast by gateways between CHs of adjacent *virtual clusters* to an extent that can be limited for scalability. This is to enable CHs to get the topology information of adjacent clusters. Given that each CH knows the predicted residence-time of each MN within its cluster, it deletes the entry associated with a particular MN exactly 't_{to}' (system parameter) seconds after its residence-time expires. This effect will be reflected in every *HELLO* message a CH broadcasts periodically. Also after having become a member of a cluster, each MN can dynamically increase its MN_HELLO_INTERVAL until the new CH election process is triggered, given that it knows the predicted residence-time of the CH. This is economical with respect to both bandwidth and transmission power.

The unique aspect of our protocol is that, before a particular CH becomes unavailable, it has to trigger the "*CH changeover event*". This happens exactly 't_{ce}' seconds before the time at which the present CH has been predicted to leave the serving *virtual cluster*. Whenever a CH broadcasts "*CH Changeover Event*" message within its *virtual cluster*, each member MN should perform the clustering criterion calculation process.

When this event occurs,

- Each MN has to calculate its distance from the center (VCC) of a particular *virtual cluster* (such information is broadcast periodically by each CH). Assuming an MN with an identifier 'x', whose location co-ordinates at time 't' are ($x_{xk}(t)$, $y_{xk}(t)$), in the k^{th} *virtual cluster*, whose center's Cartesian co-ordinates are (x_{ck}, y_{ck}), its distance at time 't' can be calculated by :

$$d_{xk}(t) = \sqrt{(x_{xk}(t) - x_{ck})^2 + (y_{xk}(t) - y_{ck})^2} \qquad (1)$$

- Each MN has to predict how long it will remain in the present *virtual cluster*, which it can obtain from its *Mobility Trie*. Let the resident-time of a MN with an identifier 'x' within k^{th} *virtual cluster* be 't_{xk}', and the state probability for the x^{th} MN to stay in k^{th} *virtual cluster* be 'p_{xk}'. In case, a MN does not have a trie, its '$t_{xk}(t)$' and 'p_{xk}' will both take zero values.

Based on the above, each MN 'x' is required to calculate, the clustering criterion factor Ω_x which is given by:

$$\Omega_x = \begin{cases} p_{xk}\left[\dfrac{t_{xk} - t_{th}}{d_{xk}(t)}\right] & \forall \quad d_{xk}(t) \neq 0 \\ p_{xk}\left[\dfrac{t_{xk} - t_{th}}{d_{\min}}\right] & otherwise \end{cases} \qquad (2)$$

The formula, given by equation (2), tries to ensure that the resulting clusters are more stable, and have uniform coverage by respective CHs. Ω_x is proportional to expected residence-time and the probability to stay in a *virtual cluster*, and inversely proportional to distance from respective VCC. 't_{th}' is the threshold value (system dependent) for the residence-time, and d_{min} ($\neq 0$) is the minimum value that $d_{xk}(t)$ can take. At the end of calculation, each MN will disseminate a **Ω**-message to the CH. Based on the information received from its members, the CH will select the MN that has the highest **Ω** value as the new primary CH. It will also select two assistant (deputy) CHs for reliability purposes – again based on **Ω**-values. The present CH will then broadcast this information using a *SUCCESSOR* Message. As soon as the new CH receives this, it will assume its status as the new primary CH and so will the two assistants.

If the first assistant CH sees that it has not received a *HELLO* message from the primary CH during the last two consecutive CH_HELLO_INTERVAL periods, it will take over as the primary CH informing its deputy as its first assistant CH. If, however, the second assistant has not received any *HELLO* message either from the primary or first assistant during the last four consecutive CH_HELLO_INTERVAL periods, it will assume duty as the primary CH. In case an ordinary MN notices no *HELLO* message from any CH during six consecutive CH_HELLO_INTERVAL periods, the first MN to notice this will assume duty as the temporary CH, and it will immediately trigger the CH changeover event by broadcasting "*CH Changeover Event*" message. Accordingly, each MN will become aware of other MNs' **Ω**-values. Each MN then compares its own value with that of each MN of the same *virtual cluster*, and one that has the highest value for '**Ω'** will be elected as the new primary CH. Deputy CH election will follow and this information will be broadcast through a *SUCCESSOR* message. The new CH will then start broadcasting *HELLO* message as usual. If however, an ordinary MN has not received any of the above control messages for more than eight consecutive CH_HELLO_INTERVAL periods, then it will elect itself as the CH. In this algorithm, if more than two MNs have the same value for '**Ω',** the one with the lowest ID will be selected as the new CH. Unlike in any other clustering algorithm, our algorithm has another unique feature in that whenever a CH leaves the *virtual cluster* it has served, it will loose its CH status. In this way this algorithm ensures that no other visiting MN can challenge an existing CH within a particular *virtual cluster*, causing instability. Also, since in our algorithm each MN is supposed to be aware of its neighbors' residence-time within a specific *virtual cluster*, the frequency of *HELLO* messages can be made lower than those of other known clustering schemes. In this way, our algorithm minimizes the waste of channel bandwidth and transmission power.

With the *Mobility Trie* information that a CH receives from its new member MN, the CH becomes aware of future mobility patterns of its cluster members at least for the next 'T_{mt}' minutes. In case any member MN continues to stay within the same *virtual cluster* beyond 'T_{mt}', then such MN is expected to unicast its new *Mobility Trie* corresponding to the next 'T_{mt}' to the CH 't_{mt}' seconds before the original 'T_{mt}' expires. The same procedure is expected instantly from every member MN of a cluster, whenever CH changeover occurs.

IV. EVALUATION THROUGH SIMULATION

The simulation work attempts to compare the performance of our clustering algorithm with the Lowest-ID, maximum-connectivity (Max-Connect), LDV clustering algorithms, in terms of stability of clusters being formed. The cluster instability is measured by determining the number of times each MN either attempts to become a CH or gives up its role as a CH. The y-axis of Fig. 4 shows the frequency of CH changes by each MN, and hence measures the (in)stability associated with each clustering algorithm. (The less frequency of CH changes, the more stable the cluster is). As it can be seen from Fig. 4, the (p, t, d)-clustering algorithm leads to more stable cluster formation. Fig. 5 depicts the average service time duration of each CH in each of the clustering algorithms. The longer the service time of each CH, the better its support for cluster stability is. As it can be seen from Fig. 5, in the (p, t, d)-model each CH has longer average service time than that of any other algorithm. Fig. 6 helps us to find the optimum value for the radius (R) of a *virtual cluster* in our (p, t, d)-clustering algorithm. We try to measure this in terms of the control message overhead, stability of the clusters formed, and the cluster size. This simulation is performed for 100 MNs, and the control overhead is measured in kilobytes. The cluster instability is measured by calculating the total number of CH changes by all MNs, and cluster size is measured in terms of total number of members in each cluster formed. Assuming that nodes have a transmission range of 71m, from Fig. 6 it becomes apparent that the *virtual cluster* radius should be within the range of 200–250m. The optimum value is such that it can minimize the overhead involved, lead to stable cluster formation, and result in a bigger cluster size. We performed our simulations using the GloMoSim simulation package in which we implemented and compared the Lowest-ID, Max-Connect, LDV and our algorithm [10].

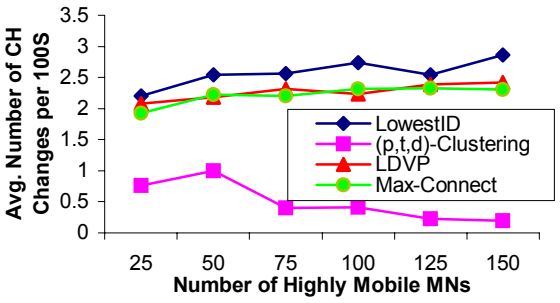

Figure 4. Clustering Instability as a function of Number of Nodes

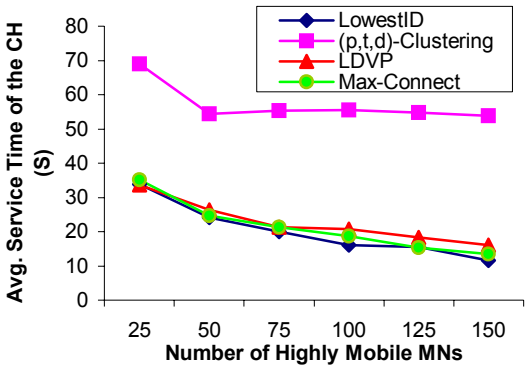

Figure 5. Average Service Time Duration of each CH as a function of Number of Nodes

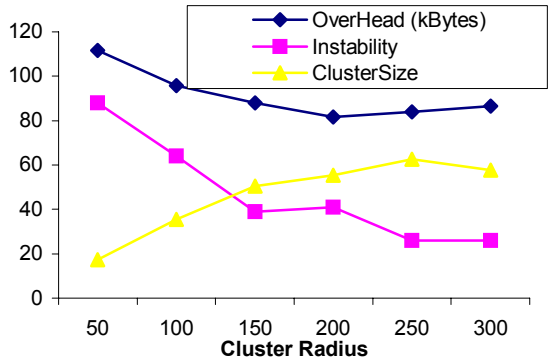

Figure. 6 Contol Overhead, Cluster Instability, and Average Cluster Size as a function of Virtual Cluster Radius.

In simulating the (p, t, d)-model, some MNs take a regular mobility pattern, whilst others always take a random waypoint mobility pattern. The former mobility scenario is realized through the creation of a movement pattern file. In order to simplify the simulation, each leaf of the *Mobility Trie* associated with each MN has two branches only. This again simplifies the calculation of 'p_{xk}' of equation (2). The distance between any two VCCs is 200m, and the radius of a *virtual cluster*, R, is 142m. Lowest-ID, LDV, and maximum-connectivity clustering algorithms form 2-hop clusters. Since it was necessary to ensure that clusters formed by all the schemes approximately cover equal area, the transmission range of each MN is set to 71m. A terrain-area of 600m X 600m with nine *virtual clusters* was considered in our simulations.

V. CONCLUSIONS & FUTURE WORK

In this paper we presented a new clustering approach that makes use of intelligent mobility prediction and location information in new long-term MANETs. To facilitate this, we introduced the '*virtual cluster'* concept. This way of associating dynamic clusters to geographic locations results in the following benefits: 1) This approach makes the task of mobility management easy, 2) We could predict a specific MN's future positions and route packets accordingly in order to avoid QoS deterioration (this will be part of our future work), 3) The identity of a cluster will not change in subsequent CH changeovers, 4) CH changeover is not frequent, and cluster set up time is minimal. We have demonstrated that this clustering scheme results in more stable clusters than those of other well-known schemes. This stability improvement, however, depends on the accuracy of our mobility prediction. In case no MN is able construct a *Mobility Trie* dynamically, our clustering scheme will behave exactly like the Lowest-ID clustering algorithm. Our future work is going to be on QoS routing, where the construction of longevity routes with sufficient resources is necessary. Work on QoS routing and resource reservation mechanisms will be built on this clustering scheme. Since our clustering mechanism enables each MN to know the availability patterns of its neighbors, our route construction process will avoid costly flooding. Each MN's knowledge of its neighbors' availability will enable it to construct and maintain routes in a proactive way. In addition, the knowledge of neighbors' *Mobility Tries* and the intelligent forwarding decisions by cluster heads will play important roles in improving the routing performance. We also plan to extend our clustering technique to shorter-term MANETs by trying to capture context information. Our findings will be helpful for the design of new intelligent location-aware MANETs. We plan to report such findings in future papers.

REFERENCES

[1] S.Chakrabarthi, and A.Mishra, "QoS Issues in Ad Hoc Wireless Networks", IEEE Communications Magazine, Feb. 2001, pp 142 –148.

[2] S.H.Lee and D-H.Cho, "A new adaptive routing scheme based on the traffic characteristics in mobile ad hoc networks", IEEE Vehicular Technology Conference, vol. 6, 2000, pp. 2911 – 2915 .

[3] L.Blazevic, L.Buttyan, S.Capkun, S.Giordono, J-P.Hubaux and J-Y.Boudec, "Self-Organisation in Mobile Ad Hoc Networks: The Approach to Terminodes", IEEE Communications Magazine, June 2001, pp 166 –173.

[4] T-C.Hou, and T-J.Tsai, "An Access-Based Clustering Protocol for Multihop Wireless Ad Hoc Networks", IEEE Journal on Selected Areas in Communications, vol. 19, July 2001, pp 1201 –1210.

[5] J.Habetha, A.Hettich, J.Peetz, and Y.Du, "Central Controller Handover Procedure for ETSI_BRAN HiperLAN2 Ad Hoc Networks and Clustering with Quality of Service Guarantees", Mobile and Ad Hoc Networking and Computing (MobiHOC), 2000, pp. 131 – 132.

[6] B.McDonald, and F.Znati, "A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks", IEEE Journal on Selected Areas in Communications, vol. 17, August 1999, pp 1466 –1487.

[7] P.Basu, N.Khan, and D.C.Little, "Mobility Based Metric for Clustering in Mobile Ad Hoc Networks", Workshop on Distributed Computing Systems, 2001, pp. 413 –418.

[8] M.Jiang, J.Li, and Y.C.Tay, "Cluster Based Routing Protocol (CBRP) Function Specifications", IETF Draft, August 1999.

[9] F.Yu, and V.C.M.Leung, "Mobility-Based Predictive Call Admission Control and Bandwidth Reservation in Wireless Cellular Networks", IEEE INFOCOM, 2001, pp. 518 – 526.

[10] X.Zengu, R.Bagrodia, and M.Gerla, "GloMoSim: A Library for Parallel Simulations of Large-scale Wireless Networks", Proceedings of the 12th Workshop on Parallel and Distributed Simulations, May 1998.

[11] D.Gross, and C.M.Harris, "Fundamentals of Queuing Theory", 2nd ed., John Wiley & Sons Ltd, ISBN: 0-471-89067-7, 1985, pp.404 – 455.

[12] A.Bhattacharya, and S.K.Das, "LeZi-Update : An Information-Theoretic Approach to Track Mobile Users in PCS Networks", 5th Annual ACM Int'l Conference on Mobile Computing and Networking (MobiCom'99), August 1999 , pp. 1 – 12.