Dynamic Local Clustering for Hierarchical Ad Hoc Networks

Satu Elisa Schaeffer
Laboratory for Theoretical Computer Science
Helsinki University of Technology
P.O. Box 5400, FI-02015 TKK, Finland
elisa.schaeffer@tkk.fi

Stefano Marinoni
Laboratory for Theoretical Computer Science
Helsinki University of Technology
P.O. Box 5400, FI-02015 TKK, Finland
stefano.marinoni1@studenti.unimi.it

Mikko Särelä
Laboratory for Theoretical Computer Science
Helsinki University of Technology
P.O. Box 5400, FI-02015 TKK, Finland
mikko.sarela@tkk.fi

Pekka Nikander
Nomadiclab
Ericsson Research
Jorvas, Finland

Abstract

Hierarchical, cluster-based routing greatly reduces routing table sizes compared to host-based routing, while reducing path efficiency by at most a constant factor [9]. More importantly, the amount of routing related signalling traffic is reduced [7, 11, 19]. On the other hand, address changes caused by nodes changing their cluster produces address management traffic. In this paper, we present a new local clustering method that produces dense and stable clusters, thereby minimizing address changes and allowing better and more stable network conditions for ad hoc routing.

1 Introduction

When clustering is introduced to an ad hoc routing system, locally computable clustering is a necessity in order to avoid generation of excess control traffic. In the ideal case, each arriving node is able to determine the appropriate cluster simply by consulting its immediate neighbors, who will not need to communicate further to determine the best cluster. Proposals for and analysis of cluster-based routing in dynamic networks include [10, 19].

Within a clustered network, routing can be divided into two subproblems: finding a route of clusters to the destination node and finding a route within each cluster either to the next cluster or to the destination node within the cluster. If two previously disconnected clusters become connected or vice versa, the inter-cluster routing is affected. Desirably inter-cluster connectivity changes are rare and nodes only switch from one cluster to another in order to minimize intra-cluster routing and maintenance costs. Avoiding cluster changes helps stabilize routing by cluster hops in comparison to routing based on individual links.

It is common for many clustering algorithm proposals that nodes are at most two hops away from the members of their corresponding clusters [2, 4, 5, 6, 12]. Methods differ for example in the usage of cluster heads and possible cluster overlaps. Ohta et al. [16] propose a clustering algorithm similar to the one presented in this paper, where the clusters are chosen from neighboring ones, bounding the size of each cluster. Our contribution is in choosing the clusters based on a particular method for local graph clustering that helps achieve dense clusters [18]. Our clustering protocol does not impose explicit constraints on the cluster diameter and hence the intra-cluster hop counts are not limited. The goal is to produce such a clustering where topology changes are concentrated *inside* clusters and changes in inter-cluster connectivity are avoided.

We aim at clusters with high local density and only few links to the rest of the network desirable as they simplify the routing task. Link state algorithms, such as OLSR [3], require dense and relatively small networks in order to be efficient [17] and perform well for intra-cluster routing with dense and stable clusters. Inter-cluster routing, on the other hand, may well use on-demand routing protocols that construct routes based on cluster hops and gain the advantage of more stable routes, as the clustering hides many route-breaking topology changes that occur within single clusters.

2 Cluster fitness

In this paper, we model ad hoc networks as dynamic graphs, consisting of nodes and edges (bidirectional links). The focus is on the clustering protocol. We use a graph-theoretical fitness measure [18] to locally select the cluster of an arriving node. We adopt the following notation to define the fitness measure used: in a graph $G = (V, E)$, a cluster candidate is a set of nodes $C \subseteq V$, and the set of edges of the subgraph induced by C is $E_c = \{(m, n) \in E \mid m, n \in C\}$. The *size* of the cluster is the number of nodes included in the cluster, denoted by $|C|$. The (local) *density* $\delta_\ell(C)$ of a cluster C is $|E_c| / \binom{|C|}{2}$ for clusters with more than one node and zero otherwise. The density of the entire graph $\delta(G)$ is simply $|E| / \binom{|V|}{2}$. Clusters for which $\delta_\ell(C) \gg \delta(G)$ can be considered good. The *relative density* $\delta_r(C)$ [14] is defined in terms of the *internal degree* $\deg_{\text{int}}(C) = |E_c|$ and *external degree*

$$\deg_{\text{ext}}(C) = |\{(m, n) \in E \mid m \in C, n \in V \setminus C\}| \quad (1)$$

of a cluster candidate C as the fraction of the internal degree of the total number of edges incident on the cluster. It is commonly acknowledged that a good graph cluster should have many edges connecting the included nodes to each other, and as few as possible connecting the cluster to the rest of the graph, and hence, high relative density [8, 14]. We want each node to be connected to each member of their cluster by at least one path *within* the cluster, preferably directly linking to many cluster members, and linking to only few nodes outside its cluster. The first criterion is fulfilled if only connected subgraphs are considered as cluster candidates. We choose to optimize the product of the relative and local densities to achieve clusters that fulfill the other two criteria:

$$f(C) = \frac{2 \deg_{\text{int}}(C)^2}{|C|(|C| - 1)(\deg_{\text{int}}(C) + \deg_{\text{ext}}(C))}. \quad (2)$$

With respect to this measure, a good cluster is both dense and "introvert", and the combination avoids counterintuitive clusterings produced by optimizing either one of the two density measures alone.

3 Clustering protocol

The clustering algorithm initiates, e.g. after the node first wakes up, by probing the neighborhood with a CLUSTER REQUEST message to which all neighboring nodes respond with a CLUSTER REPLY. The response message consists of the node identifier, cluster identifier and three integers: the number of nodes in the cluster $|C|$, the internal degree $\deg_{\text{int}}(C)$ of the cluster, and the external degree $\deg_{\text{ext}}(C)$ of the cluster. If no responses arrive, the node creates a singleton cluster and becomes its cluster head. This allows the clustering to initialize in a distributed fashion.

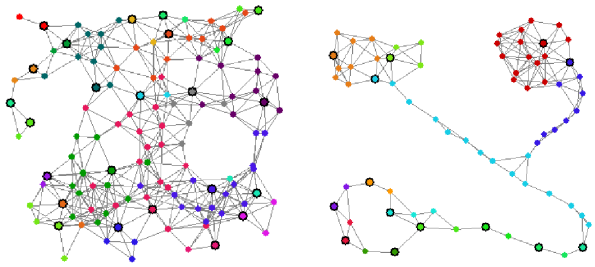


Figure 1. Stationary nodes with fixed range have been added one by one, with existing nodes updating their clusters (indicated with colors) after the newcomer selects a cluster. Cluster heads have a black border. On the right, a more anomalous network structure.

If replies do arrive within a beacon frame, the node chooses among the neighboring clusters by optimizing the change in cluster fitness, choosing the cluster for which its join would cause the highest increase (or smallest decrease). The node declares its selection by broadcasting a CLUSTER JOIN message containing the cluster identifier of the chosen cluster. Upon the creation of a singleton cluster, the node sends a CLUSTER JOIN message containing the cluster identifier it chose.

The CLUSTER REQUEST and CLUSTER REPLY messages are then used periodically to maintain up-to-date neighborhood information and to make decisions of leaving and joining clusters. Generally, a node only performs a cluster switch (through a join operation) when it is *quality-increasing*: a node i executing the cluster-selection protocol switches from its current cluster C_i to another cluster C if the sum of the cluster fitnesses C_i and C grows as i switches from C_i to C.

In addition, we impose upper and lower bounds on the cluster sizes so that nodes primarily choose clusters that are within the bounds. If there are no such neighboring clusters, a node prefers clusters below the lower bound, and in their absence, then will create a new cluster. No node may join a cluster whose size is at or above the upper bound.

A node stays in the same cluster until it either announces a join to another cluster or the cluster splits. The periodic cluster request and subsequent cluster join messages are the basic cluster maintenance mechanisms. We utilize the cluster-head status of a node in coping with cluster splits and having the cluster heads periodically broadcast keep-alive messages that are flooded only within the respective clusters. The lack of a keep-alive message indicates to a node that it has disconnected from its cluster head and it

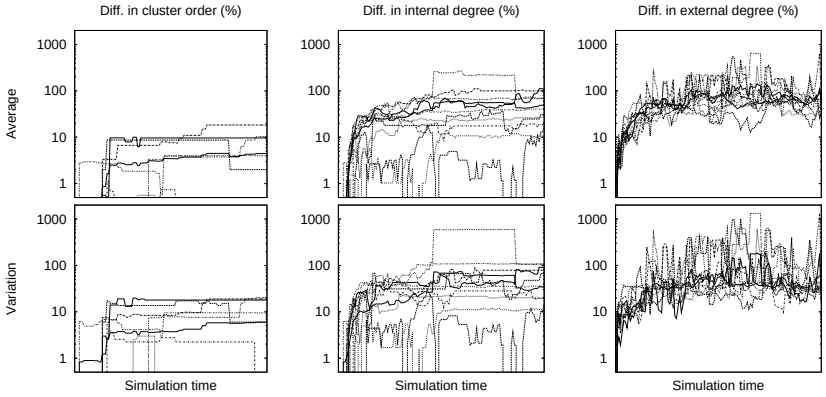


Figure 2. Differences in $|C|$ (left), $\deg_{\text{int}}(C)$ (middle), and $\deg_{\text{ext}}(C)$ (right) over a set of $N = 10$ runs of $D = 250$ seconds. Each line relates to a single run. The upper plots show the average difference $Avg_p = 100 * 1/30 \sum_{i=1}^{30} \frac{abs(E_p(i) - V_p[C(i)])}{V_p[C(i)]}$ over time, for $p \in \{$size,in,out$\}$, and the lower plots the corresponding deviation.

must reinitiate the cluster selection protocol.

4 Experiments

We sketched a small-scale simulator to visualize clusterings [20] (examples shown in Figure 1). We also built an `ns-2` implementation [13] of the algorithm for larger scale experiments. Our experiments with simulation tools are promising: the clusters achieve a proper sense of locality in space and their structure corresponds well to the intuitive global clusterings of the network.

In the `ns-2` simulations, we used networks of 30 nodes in a one square-kilometer area. The minimum cluster order was set to five and the maximum to eight nodes; the simulator was very slow for larger networks. Each node probed its neighborhood, with a range of 250 meters, on five-second intervals and the cluster heads broadcasted a status message for intra-cluster flooding on five-second intervals.

4.1 Effects of outdated information

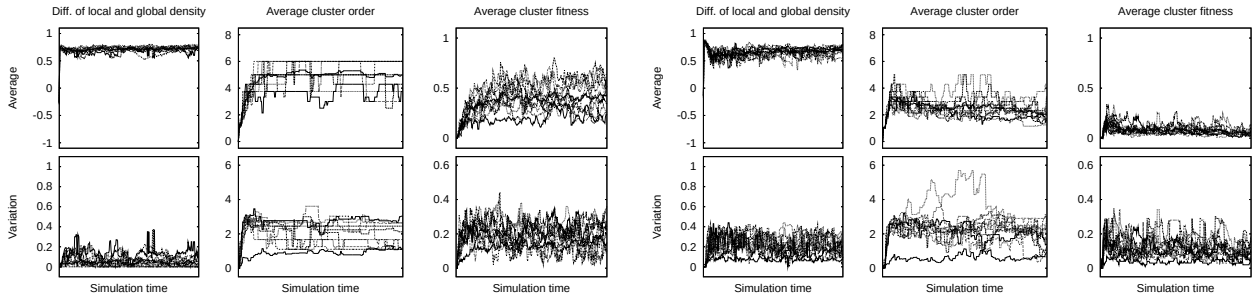
Observing the behavior of the clustering method on the simulators, it also seems feasible to approximate the fitness function using estimates of $|C|$, $\deg_{\text{int}}(C)$ and $\deg_{\text{ext}}(C)$. Such "lazy updates" would allow for a more relaxed control traffic within the cluster, as not all nodes need to be immediately aware of newcomers, departing nodes, or changes in edges. The effects of outdated information can be deduced from the fitness function (Equation 2); the magnitude of the difference between the actual value, and the assumption made at a single node depends on the rate of change in the clustering, as well as the frequency with which updated information is propagated in the network.

We traced a set of `ns-2` runs and computed at each time step the true values of the above measures and compared those to the "belief" of each node, calculating the distance in percentage of the real value. Formally, in every instant of time, every node i belongs to a precise cluster $C(i)$. This cluster has its order $V_{\text{size}}[C(i)]$, its internal degree $V_{\text{in}}[C(i)]$ and its external degree $V_{\text{out}}[C(i)]$; they are the actual values. Likewise, at every instant of time, each node i, holds its estimated values respectively as $E_{\text{size}}(i)$, $E_{\text{in}}(i)$, $E_{\text{out}}(i)$.

Figure 2 shows that the estimate for cluster size does not diverge over time and the internal degree often "restores" the correct value, but the estimates for the external degree remain far from the true value. However, we seem to achieve a practical clustering even with the problems in determining the external degree. With additional control overhead, the accuracy could be improved.

One reason for the problematic estimation of the external degree is that in cluster splits, there is a risk that the original cluster will not notice the departure of some nodes. In situations where splits are frequent and the departing nodes will often become completely detached from the old cluster, not even remaining in the neighborhood, the cluster heads should send out time-stamped beacon messages containing the cluster member list that are propagated by broadcast within the respective clusters, and the member nodes respond (through a broadcast tree formed by the order in which the nodes received the beacon message from each other) by stating which of those members are currently their neighbors and how many other neighbors they have.

Such a mechanism allows for the entire cluster to main-



Group-mobility model

Random way-point model

Figure 3. The difference of the average local density and the global density, cluster order, and cluster fitness averaged over the set of clusters for each time step in $\{0, 2, 4, \ldots, 600\}$ (average over the 10 runs drawn thick).

tain a more up-to-date view on the cluster topology. The cluster head should not send out a new beacon before it receives the replies to the previous ones; the waiting time should be reset upon the arrival of a reply and the computation of the current values should only be done after a timeout occurs with no further reply arrivals. If however the cluster head receives a reply *after* the timeout, it should increase the waiting time for the next beacon round. A mechanism for reducing the time if all replies arrive quickly could also be included. Note that by adding a hop counter to the beacon messages, incremented by each forwarding node, nodes can include the value of the counter upon their first reception of the message to their replies and thus inform the cluster head of their "effective" distance from the cluster head; this information could also be used to adjust the waiting time at the cluster head.

As described above, cluster formation is based on an exchange of simple messages that contain the cluster identifier and three integers: the size of the existing cluster, the internal degree of the cluster, and the external degree of the cluster. If a link state routing protocol is used within the cluster, the nodes can use link state information to produce the current values, and do not need to exchange any extra messages for intra-cluster information. Using these figures, together with information about the new or deleted edges, each node under consideration is able to estimate the cluster quality for each cluster candidate.

For moderately sized clusters (at most 256 nodes) and 64-bit cluster identifiers, all of the required information can be fit into 16 bytes. This could be included in existing link-layer frames, IP layer address resolution, neighbor-discovery messages, routing messages, or in Wireless LAN beacon frames.

4.2 Cluster quality

We studied the quality of the clusterings produced by a series of `ns-2` simulations, studying cluster density, fitness, and stability as the main indicators. We ran $N = 10$ simulations with 30 nodes. The mobility models utilized were reference-point group mobility (GM) model with nodes moving in small groups, random direction (RD) model, random walk model (RW), and random way-point (RWP) model [1, 15].

In all our scenarios the nodes move with speed uniformly distributed in $[0,\ 15]$ m/s after an initial period of $[0,\ 5]$ seconds. In GM, each individual node moves as in RWP, but within a restricted area of 200m^2 surrounding the group imaginary reference point, while reference points also move as in RWP, but within the whole simulation area. For RW, nodes change direction on one-second intervals.

We report averages, and variations of some measured indicators; formally, we denote the average of a set of k values $\{y_1, y_2, \ldots, y_k\}$ as $\mathrm{Avg}\,[y_i]_1^k = \frac{1}{k}\sum_1^k y_i$, and the variation as:

$$\varrho\,[y_i]_1^k = \sqrt{\frac{\sum_{i=1}^{k}\left(\mathrm{Avg}\,[y_i]_1^k - y_i\right)^2}{k}} \qquad (3)$$

where, k denotes the cluster count at a certain time step, and $y_i = m(C_i)$ is an instantaneous measure (concerning the cluster C_i) for a certain metric m. With regard to our clustering algorithm, we considered particularly significant to monitor the overall conditions (average over all clusters) in terms of density, order and fitness. Hence, the metrics measured over a period of $D = 600$ seconds were the following, with a measurement taken for each time step $t \in \{0, 2, 4, \ldots, 600\}$: the local density of the clusters versus the density of the graph $(\delta_\ell\,(C_i) - \delta\,(G))$, the cluster order$|C_i|$, and the cluster fitness $f\,(C_i)$. For the density difference, the range is $[-1\,,1]$ and a positive value indicates

Table 1. Measures of graph (Equation 4) and cluster stability (Equation 5) for the mobility models (MM), averaged over $N = 10$ experiments of duration $D = 600$ seconds.

MM	$\tilde{\mathcal{B}}/D$	$\tilde{\mathcal{B}}_{\text{int}}/D$	$\tilde{\mathcal{E}}/D$	$\tilde{\mathcal{E}}_{\text{int}}/D$	$\tilde{\mathcal{T}}/D$	$\tilde{\mathcal{T}}_{\text{int}}/D$	$\tilde{\mathcal{Q}}$	$\tilde{\mathcal{F}}$	$\tilde{\mathcal{S}}$	$\tilde{\mathcal{T}} \cdot \tilde{\mathcal{S}}$
GM	1.53	0.25	1.55	0.21	3.08	0.46	0.03	0.01	0.04	77
RD	1.04	0.43	1.06	0.20	2.10	0.63	0.10	0.08	0.18	227
RW	1.13	0.47	1.15	0.42	2.28	0.90	0.04	0.02	0.07	89
RWP	1.50	0.59	1.51	0.26	3.01	0.86	0.09	0.07	0.16	289

that dense subgraphs have been selected as clusters; if the value is close to one, almost all links present in the graph are internal to some cluster. For the cluster order the range is [0,8], its value over time it is a first indicator of cluster stability as stable clusters must have few fluctuations. The fitness varies in $[0, 1]$ with values close to one indicating optimal clusters.

The results are shown in Figure 3; results for RD, and RW mobility models were similar and omitted. All mobility models produced clusters with much higher local density than the density of the entire graph. Unexpectedly, the group mobility model produced large clusters with very high density, whereas group mobility scenario had consistently much better fitness than in other mobility models. Both random way-point and random direction acted similarly, producing small, but dense clusters.

4.3 Cluster stability

We also studied the *stability* of the graph and cluster topologies (results are shown in Table 1), recording the total amount of link breakages $\mathcal{B}_i$ and new link establishments $\mathcal{E}_i$ and the average topology change rate $\mathcal{T}/D$ by considering the graph variations occurred per second in experiment i, $i \in \{1, 2, \ldots, N\}$:

$$\tilde{\mathcal{B}} = \text{Avg}\left[\mathcal{B}_i\right]_1^N, \tilde{\mathcal{E}} = \text{Avg}\left[\mathcal{E}_i\right]_1^N, \tilde{\mathcal{T}} = \tilde{\mathcal{B}} + \tilde{\mathcal{E}}. \quad (4)$$

We additionally recorded the number of topology changes that were *internal* to clusters, denoting these by $\mathcal{B}_{\text{int}}$, $\mathcal{E}_{\text{int}}$, and $\mathcal{T}_{\text{int}}$, respectively [1]. Cluster stability was measured by the number of cluster changes, distinguishing between two categories: $\mathcal{Q}_i$ is the number of quality-increasing cluster switches and $\mathcal{F}_i$ is the number of switches due to a cluster split;

$$\tilde{\mathcal{Q}} = \text{Avg}\left[\frac{\mathcal{Q}_i}{\mathcal{B}_i + \mathcal{E}_i}\right]_1^N, \tilde{\mathcal{F}} = \text{Avg}\left[\frac{\mathcal{F}_i}{\mathcal{B}_i + \mathcal{E}_i}\right]_1^N \quad (5)$$

We denote $\tilde{\mathcal{S}} = \tilde{\mathcal{Q}} + \tilde{\mathcal{F}}$; note that as $\tilde{\mathcal{T}}$ is the average number of topology changes, $\tilde{\mathcal{S}} \cdot \tilde{\mathcal{T}}$ is the average amount of cluster changes in a single simulation run.

[1] Inter-cluster topology changes could be deduced as a difference between $\mathcal{T}$ and $\mathcal{T}_{\text{int}}$

The results in Table 1 show that group mobility model and random walk show have the most stable clustering structure of the four, although the reasons differ. Random walk creates mainly local movements, which means that the overall topology of the graph will tend to stay the same with small variations. It has as low rate of topology changes as random direction, but causes much less changes in the clustering structure. This is due to the local movements of nodes in random walk vs. global movements of nodes in random direction. Group mobility model creates global movements, but with certain groups of nodes staying close to each other. This causes a high rate of changes in the topology, but low rate of changes in the clustering. The random-direction model also produces global movements.

We experienced very few clusters splits and changes in general. Overall, the rate of changes in clustering is small. Group mobility and random walk cause changes in clustering in 4% and 6% of the cases where topology changes and random direction and random way-point models in 16% and 18% respectively.

5 Conclusions

We introduced a new local measure for clustering quality, and outlined a simple protocol for local cluster management. The simulations show that the clustering algorithm is capable of creating a clustering structure, which hides most of the topology changes within the network and thus making inter-cluster routing task easier. The algorithm is capable of capturing the structure that may exist in the movements of nodes. This is especially marked by the group mobility model having the highest rate of topology change, while having least changes in clustering both per topology change and per unit of time. The algorithm also managed to form clusters with high local density, allowing us to partition the network into smaller subnetworks which are easily managed by proactive routing algorithms such as OLSR [3] that are designed especially for dense networks with small diameter.

As future work we plan to study how the proposed clustering algorithm could be used to further optimize routing and address management. On top of a base-layer cluster-

ing, we could form a hierarchy of clusterings with a very similar cluster-formation protocol, relying on routing the higher-level cluster requests to the cluster heads. Such a layering would however introduce additional duties to the cluster heads, but is an interesting area for further work.

Acknowledgments

The first author was supported by the Academy of Finland (under grants 202205 and 206235), the Nokia Foundation, and the Rotary Foundation.

References

[1] T. Camp, J. Boleng, and V. A. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication and Mobile Computing*, 2(5):483–502, Sept. 2002.

[2] C.-C. Chiang, H.-K. Wu, W. Liu, and M. Gerla. Routing in clustered multihop, mobile wireless networks with fading channel. In *Proceedings Of IEEE SICON'97*, pages 197–211, Apr. 1997.

[3] T. H. Clausen and P. Jacquet. Optimized link state routing protocol (OLSR). Technical Report RFC 3626, Internet Engineering Task Force, Reston, VA, USA, 2003.

[4] A. Ephremides, J. E. Wieselthier, and D. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proceedings of the IEEE ?*, 75:56–73, Jan. 1987.

[5] M. Gerla and J. T.-C. Tsai. Multicluster, mobile multimedia radio network. *ACM-Baltzer Journal of Wireless Networks*, 1:255–265, 1995.

[6] 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*, 19(7):1201–1210, July 2001.

[7] F. Kamoun and L. Kleinrock. Stochastic performance evaluation of hierarchical routing for large networks. *Computer Networks*, 3:337–353, Nov. 1979.

[8] J. M. Kleinberg and S. Lawrence. The structure of the web. *Science*, 294(5548):1849–1850, Nov. 2001.

[9] L. Kleinrock and F. Kamoun. Hierarchical routing for large networks: Performance evaluation and optimization. *Computer Networks*, 1(3):155–174, 1977.

[10] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan. A cluster-based approach for routing in dynamic networks. ACM SIGCOMM *Computer Communication Review*, 27(2):49–64, Apr. 1997.

[11] G. S. Lauer. Hierarchical routing design for SURAN. In *Proceedings of the* IEEE *International Conference on Communications (*ICC*)*, pages 93–102, Los Alamitos, CA, USA, 1986. IEEE Computer Society Press.

[12] C. Lin and M. Gerla. Adaptive clustering for mobile wireless networks. *IEEE Jour. Selected Areas in Communications*, 15(7):1265–1275, Sep 1997.

[13] S. McCanne, S. Floyd, K. Fall, and K. Varadhan. The network simulator `ns-2`. The VINT project, `http://www.isi.edu/nsnam/ns/`.

[14] M. Mihail, C. Gkantsidis, A. Saberi, and E. Zegura. On the semantics of Internet topologies. Technical Report GIT-CC-02-07, College of Computing, Georgia Institute of Technology, Atlanta, GA, USA, 2002.

[15] J. Nuevo. Mobility generator program for NS-2, 2002. `http://externe.inrs-emt.uquebec.ca/users/nuevo/NSmobgenerator.htm`.

[16] T. Ohta, S. Inoue, and Y. Kakuda. An adaptive multihop clustering scheme for highly mobile ad hoc networks. In *The Sixth International Symposium on Autonomous Decentralized Systems (ISADS'03)*, pages 293–300, Apr 2003.

[17] C. A. Santivanez, R. Ramanathan, and I. Stavrakakis. Making link-state routing scale for ad hoc networks. In *Proceedings of the Second* ACM *international symposium on Mobile ad hoc networking & computing*, pages 22–32, Long Beach, CA, USA, 2001. ACM Press.

[18] S. E. Schaeffer. Stochastic local clustering for massive graphs. In T. B. Ho, D. Cheung, and H. Liu, editors, *Proceedings of the Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining (*PAKDD*-05)*, volume 3518 of *Lecture Notes in Computer Science*, pages 354–360, Berlin/Heidelberg, Germany, 2005. Springer-Verlag GmbH.

[19] J. Sucec and I. Marsic. Clustering overhead for hierarchical routing in mobile ad hoc networks. In *Proceedings of the Twenty-first Annual Joint Conference of the* IEEE *Computer and Communications Societies*, volume 3, pages 1698–1706, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[20] S. E. Virtanen and P. Nikander. Local clustering for hierarchical ad hoc networks. In *Proceedings of WiOpt'04: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 404–405, Los Alamitos, CA, USA, 2004. IEEE Computer Society.