Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 159 (2016) 82 – 88

Procedia Engineering

www.elsevier.com/locate/procedia

Humanitarian Technology: Science, Systems and Global Impact 2016, HumTech2016, 7-9 June 2016, Massachusetts, USA

Tracking user-movement in opportunistic networks to support distributed query-response during disaster management

Somprakash Bandyopadhyay[a]*, Apratim Mukherjee[b]

[a] *Social Informatics Research Group, Indian Institute of Management Calcutta, Kolkata 700104, India*
[b] *Department of Computer Science & Engineering, Michigan State University, MI 48824, USA*

Abstract

Effective communication amongst diverse rescue and relief workers is a primary requirement in any disaster management. Since pre-existing communication infrastructure may not be available, the Opportunistic Network framework provides a potential platform for information communication, where individual smart-phones of rescue and relief workers (the *nodes*) spread across an environment form a disjoint, peer-to-peer network. Here, a source node communicates with a destination node following hop-by-hop, store-wait-forward cycle, since an end-to-end route connecting them never exists. Also, due to mobility and disconnectedness, nodes have scarce or no knowledge about the network topology. However, in the context of disaster management, in order to evaluate the situation, rescue and relief workers often need to generate different field-related queries and the response to those queries must come from other workers in the field. Since source node (generating the query) is not aware of the location of destination node (answering the query) and all nodes are mobile, it is difficult to implement a query-response mechanism. This paper proposes and evaluates a distributed query-response mechanism that enables any node to track approximate location of other rescue and relief workers, which is turn helps to handle query-response operations.

Peer-review under responsibility of the Organizing Committee of HumTech2016

Keywords: Opportunistic Networks; Disaster Management; Multi-Agent System; Location Tracking.

1. Introduction

In order to support disaster rescue and recovery efforts after any disaster, effective communication amongst the diverse rescue workers, as well as providing connectivity to survivors is a primary requirement. Police, Fire Departments, Public Health, Civil Defence and other organizations including local volunteers / field workers have to react not only efficiently and individually, but also in a coordinated manner. This results in the need for both intra and inter-organization information-exchange at several stages. However, disasters, whether natural or man-made, have severe impact on communication infrastructure. Services that are relied on for everyday communications (e.g., cell phone / internet connectivity) immediately become non-functional in emergency situations due to the failure of the supporting infrastructure through both system damage and system overuse [1].

In contrast to the vulnerable fixed network infrastructure, it is very likely that battery-powered wireless personal mobile communication devices (PDA, cell-phones) will survive disasters. Additionally, even more mobile devices will be brought to the scene by rescue workers, volunteers, and local authorities. Currently, those devices have powerful processors and high storage capacity with GPS and multi-radio interfaces (Cellular, Wi-Fi, Bluetooth). Such devices are, therefore, promising candidates to contribute in forming ad hoc wireless network structure to support disaster communication.

However, end-to-end connectivity can never be assumed in this kind of scenario and long disconnections and network partitions are the rule. Thus, in this context, Opportunistic Network framework [2,3] provides a potential platform for information communication. In opportunistic networks, the devices (PDA, cell phones) spread across the environment form the

* Corresponding author. Tel.: +91-33-24678300; Fax: +91-33-24678307.
E-mail address: somprakashb@gmail.com

Peer-review under responsibility of the Organizing Committee of HumTech2016
doi:10.1016/j.proeng.2016.08.068

network. In this type of networks, the mobility of devices is an opportunity for communication rather than a challenge. Mobile nodes communicate with each other even if an end-to-end route connecting them never exists. Any possible node can opportunistically be used as the next hop, if it is likely to bring the message closer to the final destination(s).

In Opportunistic Networks, it is usually assumed that nodes have scarce or no knowledge about the physical network topology. However, in our context of application, it is important to track the location of mobile rescue workers and other agencies so that effective intra- and inter-organizational communication can be handled. In general, in this scenario, X may want to send a query to Y and like to get a response from Y, when (i) X and Y are not connected through network link(s), (ii) both X and Y are mobile, and (iii) message from X to Y will not travel instantaneously, but will travel opportunistically (following hop-by-hop, store-wait-forward cycle) which implies there may be significant delay in message communication from X to Y and vice versa.

This paper addresses a distributed location tacking mechanism to handle query-response process in an effective manner. In other words, the proposed mechanism would help a node know the approximate location-related information of other nodes in the system. The degree of accuracy of this information would depend on the inter-node distance and connectivity pattern of the opportunistic network. Once the approximate location information of destination node is known to the source node, query-response mechanism can be implemented using Geographic Routing [4].

The mechanism proposed in this paper is primarily based on a mobile multi-agent based framework. Mobile agents are an effective paradigm for distributed applications, and are particularly attractive in a dynamic network environment involving partially connected computing elements. Intensive research on the "Insect-like Agent Systems" has been done over the last few years. Of particular interest is a technique for indirect inter-agent communication, called stigmergy, in which agents leave information in the cache (which other agent can use) of the nodes they have visited. Stigmergy serves as a robust mechanism for information sharing [5].

We propose two types of agents: ***satellite agent*** and ***query agent***. Each node n_i has a dedicated ***satellite agent*** S_i. Task of S_i is to help exchanging information between its host node n_i and each of the neighboring nodes of n_i. To do this, the satellite agent S_i periodically hops from n_i to one of its neighbours with all location-related information as perceived by n_i. The neighbouring node has a different perception regarding location-related information of other nodes. S_i and the neighbouring node mutually exchange this information, forms a "consensus view" regarding the location related information of other nodes and Si then comes back to the host node n_i with this "consensus view". This would then change the perception of n_i about location-related information of other nodes. In the next time-slots, S_i visits other neighbouring node of n_i and the process is repeated. This mechanism is designed to mimic the real-life scenario in the context of disaster management. When a relief worker arrives at the field, s/he has no knowledge about the whereabouts of other workers. However, s/he exchange information with his/her neighbors one by one to get some idea (although imperfect) about others. Since no other communication mechanism is available, information assimilation by an individual can only be done using nearest neighbor interaction principle and exchange of information always takes place at a local level. Since nodes are mobile, information stability would never happen; but in the process, nodes will have approximate location information about other nodes.

Second type of agent is ***query agent*** that starts from the source node (query generator) with the query and approximate location information of the destination node, navigates autonomously hop by hop to reach the destination (response generator), collects the response and returns back to the source node. As it approaches closer to destination node, it will have more accurate location information about the destination node and accordingly it modifies its navigation.

2. Related Works

Opportunistic Networks and opportunistic computing has become a major research area in the recent past. Designing routing and forwarding schemes is one of the main challenges in this environment. However, forwarding and routing protocols are merged in this context, because routes are actually built while messages are forwarded [2]. The forwarding scheme has been primarily referred as "store, carry, and forward". Each intermediate node evaluates the suitability of encountered nodes to be a good next hop towards the destination. Another form of routing technique exploits some form of flooding. The heuristic behind this policy is that when there is no knowledge about a possible path towards the destination or of an appropriate next-hop node, a message should be disseminated as widely as possible. The most representative protocol of this type is Epidemic Routing [6] and some optimizations of the same [e.g.,7]

However, flooding-based approach generates multiple copies of the same message. In Forwarding-based approach, though there is only one single custodian for each message, it may suffer long delays and low delivery ratios. Several schemes have been proposed considering mobility pattern / context information into account. The Haggle Project [8] has developed mechanisms for measuring and modelling pair-wise contacts between users and devices by means of two parameters: contact durations and inter-contact times. The statistical properties of these parameters are used to drive the design of forwarding policies. Probabilistic Routing scheme [9] calculates the delivery predictability from a node to a particular destination node based on the observed contact history, and it forwards a message to its neighbouring node if and only if that neighbour node has a higher delivery predictability value. Leguay et al. [10] have taken the mobility pattern into account, i.e., a message is forwarded to a neighbour node if and only if the neighbour node has a mobility pattern more similar to the destination. However, in many application scenarios (such as ours), mobility patterns are largely unpredictable. Also, there are some efforts to use opportunistic networks in the context of disaster management [11,12].

Nevertheless, a successful information forwarding scheme in opportunistic networks not only needs to consider delay performance, but it must also consider the nature of its application. Effective schemes dealing with different application requirements remain challenging and desirable. As indicated earlier, in our context of application, it is important to track the

location of mobile rescue workers and other agencies so that effective intra- and inter-organizational communication can be handled. As examples, public health department, after reaching the site with a team of medical expert, must be able to locate the other health workers, who are already working in the field. Secondly, in order to distribute relief resources to the designated rescue workers, those workers must be located and the agency carrying the relief resources must be able to send a query to those designated workers in order to assess the requirements. Also, the organizers need to delegate tasks to volunteers working in the field, and therefore those volunteers need to be located. In such scenarios, forwarding scheme should be location-based.

3. System Description

An Opportunistic Network is modelled as a time-dependent disconnected graph G(t) = (N,L, τ) where N is a finite set of nodes, L is a finite set of unidirectional links and τ is a set of time-values indicating life-span associated with the links. Each link $L_i \in L$ is associated with $\tau_i \in \tau$, indicating life-span of L_i at time t. Since the graph represents an opportunistic network, graph G(t) should usually be disconnected. G(t) consists of multiple connected pieces called components C(t). When G(t) is fully connected, C(t) =1; when G(t) is fully disconnected, C(t) = number of nodes N.

Since opportunistic network is usually a disconnected network, it is important to define a parameter W that indicates the degree of disconnectedness over a period of time T. For each pair of nodes, W=0 means there always exists a path between them between <0..T>. W is said to be 100%, when no path exists between any source-destination pair at any point of time between <0..T>. In the first case, the network is always connected (C(t) =1) and ceases to be an opportunistic network. In the other extreme, the graph is fully disconnected with C(t) = number of nodes N.

In order to quantify W, we need to take a set of snap-shots of G(t). $G(t_i)$ is snap-shot of G at $t=t_i$. W_i, the degree of disconnectedness for $G(t_i) = \{C(t_i)-1\}/(N-1)$. When $G(t_i)$ is fully connected, $W_i = 0$; When $G(t_i)$ is fully disconnected, $W_i = 100\%$. W is the average of W_i over the number of snap-shots taken. So, if number of snap-shots taken is α, $W = [\Sigma_{i=1 \text{to } \alpha} W_i] / \alpha$

While designing and testing the robustness of any algorithm designed in the context of opportunistic networks, it is important to consider the parameter W. Specially, while testing and validating algorithms designed for opportunistic network in a simulated network environment, disregarding the parameter W may result in a network condition where nodes are always forming a network or forming a network with a few numbers of disconnected components. Any algorithm designed for opportunistic network should work well for a wide range of W.

We define the ***physical neighbours*** of node n at time t as $N_n(t) \in N$, where $N_n(t)$ is the set of nodes within the transmission range of n at time t. It is assumed that each node knows its position, velocity and direction of movement using Global Positioning System (GPS). It is also assumed that each node periodically broadcast a beacon with its id to its entire physical neighbours at that instant of time.

4. Distributed Location Tracking

As indicated earlier, we propose to use ***satellite agents*** for distributed location tracking. In a seminal paper in Physical Review Letters, Vicsek et al.[13, 14] propose a simple model of n autonomous agents moving in the plane with the same speed but with different headings. Each agent's heading is updated using a local rule based on the average of its own heading plus the headings of its "neighbours." In their paper, Vicsek et al. demonstrated that the nearest neighbour rule can cause all agents to eventually move in the same direction despite the absence of centralized coordination and despite the fact that each agent's set of nearest neighbours change with time as the system evolves. Other studies also indicate that multi-agent systems that interact through nearest-neighbour rules can synchronize their states regardless of the size of communication delays [15].

In our mechanism, each ***satellite agent*** interacts with their respective neighboring nodes only and come back to the respective host node with a localized "consensus view" about location-related information of other nodes. The technique used here for indirect inter-agent communication is ***stigmergy***, in which agents leave information in the cache (which other agent can use) of the nodes they have visited.

4.1. The Node

In order to facilitate agent-based distributed location tracking, each node is assumed to have the following structure:

- Node Id (n_i)
- Current Location (x_i, y_i)
- Neighbourhood List : <for all k : list of n_k>
- Location Table: Information about all nodes (n_p)
  → (for all p : n_p, x_p, y_p , $timer_p \uparrow$)

Each node is assumed to know its id and current location (through GPS). Each node broadcast a periodic beacon to its neighbours to inform its id. This would help a node to form Neighbourhood List. Initially, the Location Table of a node would contain only the location information of itself only. Location Table of node n_i would be augmented by : (i) visits of satellite agents from other neighbouring nodes containing the neighbourhoods' perception (ii) returning of satellite agent of node n_i from a neighbouring node containing the "consensus view" of n_i and that neighbouring node. It is no be noted that the size of location table is a N x N where each cell is storing 20-25 bytes of data. Since N (number of relief and rescue workers working in a disaster zone) in our context is not large (e.g. 200 max), the size of location table is around 10 MB.

It is to be noted that the entire scheme is based on multi-agent interaction via Location Table of nodes. Since navigation of satellite agents is asynchronous and there is an obvious time gap between the updation of information by one satellite agent in one node and carrying this information to another node by another satellite agent, there is a notion of timer↑ with each entry, depicting the ageing of information. The information is aging as agents percolates from one node to another and the nodes will therefore have new information about close neighbours and old information about remote nodes. The symbol ↑ indicates that each timer is counting up locally, unless overwritten by more recent information about that entry.

This is illustrated in figure 1. For the sake of simplicity, we are assuming unidirectional communication and we only show percolation of location information of node n1 to node n4. In figure 1, current location of node N_1 is x_1, y_1. Since this information is always recent, no aging factor is associated with it and timer value is 0. Let us assume, this information is taken to node N_2 by N_1's satellite agent S1 at t_1=0. The timer associated with x_1, y_1 now starts getting incremented at node N_2, indicating that the location information of N_1 at N_2 is aging. At t_2=10, say, N_2's satellite agent S_2 carries this information to N_3. The timer associated with x_1, y_1 at N_3 now starts getting incremented at node N_3 with a starting value of 10. Let us assume further that after 25 time-unit, i.e. at t_3=25, N_3's satellite agent S_3 carries this information to N_4. The timer associated with x_1, y_1 at N_4 now starts getting incremented at node N_4 with a starting value of 35. This implies that the perception of node N_4 about the location of N_1 is 35 time-unit old and it is aging, unless overwritten by some more recent location information of N_1.

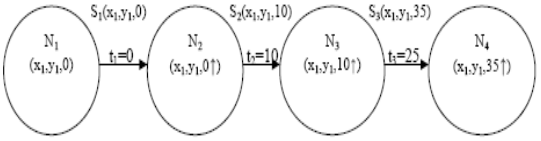

Figure 1. Percolation of Location Information

4.2. The Satellite Agent

As explained earlier, satellite agent is a dedicated agent for any node. Its task is to carry information from its host node to neighbouring nodes in a time-sequenced fashion (i.e. one neighbour at a time) and return to its host node with information of that neighbouring node. For example, if node N_2's satellite agent has not visited N_1 in the recent past, the satellite agent of Node N_1 visits Node N_2, updates N_2's table to form a consensus view between N_1's and N_2's perception, then comes back to N_1 with this information. After some time, it again collects the latest information from N_1, and moves to another neighbouring node, say N_3, provided N_3's satellite agent has not visited N_1 in the recent past. This process occurs for every node since each node has a dedicated satellite agent.

If a satellite agent loses its host node due to mobility of host node, it kills itself. The host node, on the other hand, generates a new satellite agent if the satellite agent doesn't come back after a certain time. The structure of a satellite agent is given below:

- Host Node Id (n_i)
- Current Location of Host Node (x_i, y_i)
- Location of Target Neighbor to be visited now
- Location Table of Host Node: Information about all nodes (n_p)
  - → (for all p : n_p, x_p, y_p , $timer_p$ ↑)

5. Query-Response Processing: Query Agent

As indicated earlier, a query agent starts from the source node (query generator) with the query and approximate location information of the destination node, navigates autonomously hop by hop to reach the destination (response generator), collects the response and returns back to the source node. As it approaches closer to destination node, it will have more accurate location information about the destination node and accordingly it modifies its navigation. The objective of the navigation procedure used by a query agent is to minimize the distance between the agent's current location (current location of the node where the agent is residing) and the location of the destination. This criterion would enable an agent to select a physical neighbour which is closer to the location of the destination node and migrate there. If there is no physical neighbour available at that instant of time satisfying the above-mentioned criterion, the agent waits for a pre-specified amount of time and tries again. Because of high degree of node mobility, the topology will change, and, it is assumed that the agent will eventually succeed to migrate.

For example, If the current location of an agent is (15,25), locations of three of its physical neighbours are (10,15), (30, 35) and (20, 30), and location of the destination is (65, 75), the agent would migrate to the node whose location is (30,35). However, if the current location has two physical neighbours with locations (10,15) and (5,10), the agent would wait and retry after some time.

6. Performance Evaluation

6.1. Evaluation Criteria

***Average Perception Deviation of a node*:** We have developed a metric Average Perception Deviation of a node i, denoted by P_i (t), to quantify the average deviation of actual node position of each node with the node-position of corresponding node perceived by node i at any instant of time t. Let us assume that (x_k, y_k) is the actual co-ordinates of node k at time t. Let (x_k^i, y_k^i) be the coordinate of node k as perceived by node i.

$$P_i(t) = \Sigma_{k=1 \text{to } n} [(x_k - x_k^i)^2 + (y_k - y_k^i)^2]^{1/2} / n$$

Average Perception Deviation of a network N with n number of node, denoted by $P_N(t)$, is defined as

$$P_N(t) = \Sigma_{i=1 \text{to } n} P_i(t) / n$$

Wait-before-Migrate (WbM): In order to control agent-traffic in the network, a satellite agent, after finishing its first visit to a neighbouring node of its host node, is not allowed to migrate immediately to another neighbouring node. A satellite agent will be forced to wait in its host node for a pre-specified period of time, termed as *wait-before-migrate* (WbM), before migrating to another neighbouring node. By controlling WbM, the network congestion due to satellite agent traffic can be controlled. For example, if WbM = 200 msec, and an agent takes approximately 4 msec. to physically migrate from one node to another, the agent would occupy the medium for 4 ms out of 200 msec.

Degree of Disconnectedness: In section 3, we have introduced a parameter W that indicates the degree of disconnectedness over a period of time T. In our simulation, by controlling the transmission range, we have controlled W and evaluated the performance.

6.2. Simulation Setup

The proposed scheme is evaluated on a simulated environment under a variety of conditions to estimate average perception deviation against time and delivery performance of a query agent. In the simulation, the environment is assumed to be a closed area of 1500 x 1500 square meters in which mobile nodes are distributed randomly. We present simulations for networks with 40 mobile hosts, operating at a transmission range from 150 to 250 meters. In order to study the time-related parameters, every simulated action is associated with a simulated clock. The speed of movement of individual node is selected from 2 m/sec (walking) and 10 m/sec (vehicle). Each node starts from a starting location, selects a random direction and moves with a uniform, predetermined velocity along that direction. In order to simulate disaster management environment, we define a set of 10 shelter-points within the area. Once a node reaches a shelter-point, it waits there for a pre-specified amount of time, selects randomly another direction and moves towards that.

6.3. Results and Discussions

Figure 2 shows the average perception deviation $P_N(t)$ w.r.t time at different Wait-before-Migrate and mobility is randomized between of 2 m/ sec and 10 m/ sec. With 40 nodes in the system, transmission range is adjusted to 200 meters to get an average degree of disconnectedness W= 28%, indicating that the network is having 10 to 12 disconnected components (Fig. 3). Considering the maximum possible perception deviation to be 2100 meters in a 1500 m x 1500 m area, the perception deviation is around 4 to 8 %, when WbM=120 to 200 msec. However, when WbM is low (=40 msec), perception deviation is erratic because of congestion due to agent traffic.

Next, we have considered $P_N(t)$ vs time at different W (Degree of Disconnectedness) with mobility randomized between of 2 m/ sec and 10 m/sec (Fig 4). For the sake of simplicity, we have mentioned approximate value of W in the graph. With increase in W, $P_N(t)$ will increase as expected. But even with W=50% (number of disconnected components is around 20 in a 40-node system), average perception deviation is around 10%.

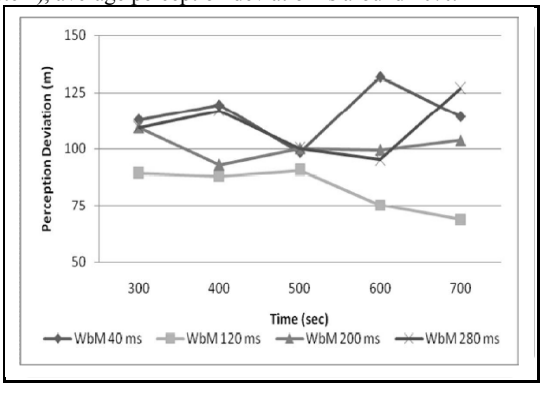

Fig. 2. Average Perception Deviation with Time

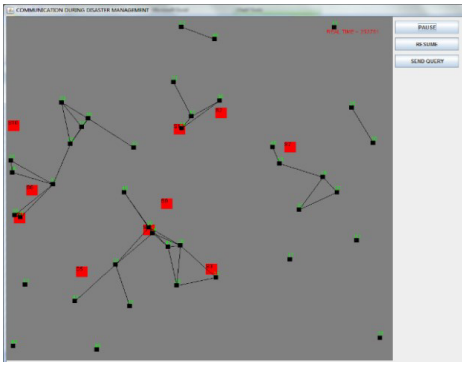

Fig. 3. A Snap-shot: squares are the shelters and dots are the nodes

In order to study *Messenger* navigation pattern for message communication, we have selected an arbitrary source-destination pair and studied the navigation pattern of a *messenger* agent, initiated by a source. Here, N=40, W=25%, WbM=200 msec. The initial distance between source and the destination was 900 meter (approx), when a messenger was initiated. There was no direct connectivity between source and destination. We have studied the variation in distance between the messenger and the destination node during its navigation from source to destination node. Figure 5 shows that the agent, after starting from source node 29, has reached the destination hop-by-hop. At each hop, the distance between itself and the destination node is progressively reducing. The different waiting time at intermediate nodes are also shown. The total number of hops taken is 8 and the total time taken to deliver the message is 55.8 sec.

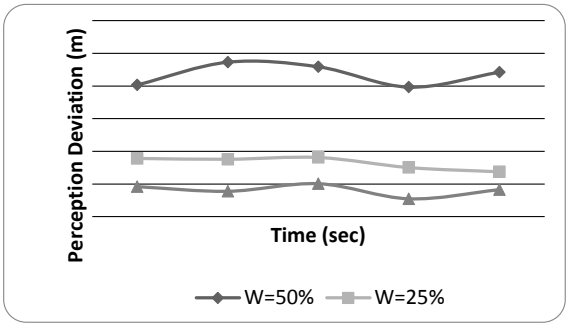

Fig. 4. Perception Deviation with Time at different W

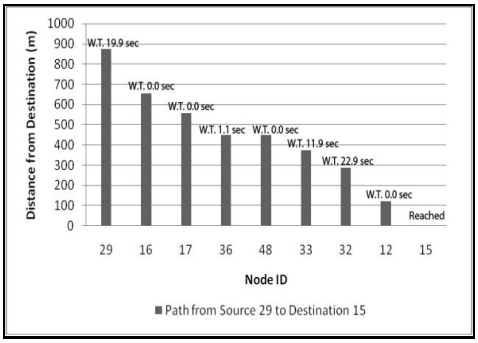

Fig. 5. Navigation Pattern of an arbitrary *Messenger* Agent

Next, we have increased the number of nodes to 60. The transmission range is adjusted to around 100 meters to get a degree of disconnectedness W = 25% (approx). We have selected 30 source-destination pairs randomly and each source is then allowed to generate a messenger. In other words, 30 messages were initiated and we have studied the Average Message Delivery Time, Maximum Message Delivery Time, Average Hop-Count for Message Delivery and Maximum Hop-Count for Message Delivery, as shown in Table 1.

Table 1. Message Delivery Performance (N=60)

No of Message Initiated	No of Message Delivered	Message Delivery Time (Avg)	Message Delivery Time (Max)	Hop Count (Avg)	Hop Count (Max)
30	30	45.5 Sec	121 Sec.	12	20

As indicated earlier, Opportunistic Networks are inherently delay-tolerant; so, a delay of 45.5 sec (average) and 2 minutes (maximum) with a median of 43 sec. and standard deviation = 29 sec (approx) is highly acceptable, when degree of disconnectedness is 25% (i.e. 15 clusters on an average in a 60-node network). A more interesting observation is the number of hops required to deliver a message. Average hop-count is 12 and maximum hop count is 20 with median 11 and standard deviation = 4 (approx). This indicates that the congestion generated during a message delivery is insignificant.

7. Conclusion

In this study, we have designed a mobile, multi-agent-based framework to make the nodes in the network aware of the approximate geographical distribution of other nodes. Our preliminary results indicate that this framework is quite promising in infiltrating GPS oriented topology information into the nodes of the system, without consuming much network capacity and this would improve distributed query-response handing to a great extent.

References

[1] H. Luo, R. Kravets, T. Abdelzaher, The-Day-After Networks: A First-Response Edge-Network Architecture for Disaster Relief, NSF NeTS FIND Initiative, 2006-2010. http://www.nets-find.net/Funded/DayAfterNet.php

[2] Marco Conti, Mohan Kumar: Opportunities in Opportunistic Computing. IEEE Computer 43(1): 42-50 (2010)

[3] Suman Bhattacharjee, Siuli Roy, Somprakash Bandyapadhyay "Exploring an Energy-efficient DTN Framework Supporting Disaster Management Services in Post Disaster Relief Operation", Wireless Networks Journal (Springer), April 2015, Volume 21, Issue 3,

[4] Somprakash Bandyopadhyay and Krishna Paul, "Evaluating The performance of mobile agent based message communication among mobile hosts in large ad hoc wireless network", Proc. of the Second ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems (In conjunction with IEEE/ACM MobiCom'99), Seattle, Washington, USA, August 15-19, 1999.

[5] Romit RoyChoudhury, Krishna Paul, and Somprakash Bandyopadhyay, "MARP: A Multi-Agent Routing Protocol for Mobile Wireless Ad Hoc Networks" Autonomous Agents and Multi-Agent Systems (Kluwer), 8 (1): 47-68, January 2004.

[6] Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical Report CS-2000-06, Duke University, 2000.

[7] K. A. Harras, K. C. Almeroth, and E. M. Belding-Royer. Delay tolerant mobile networks (dtmns): Controlled flooding in sparse mobile networks. In IFIP Networking, 2005.

[8] Augustin Chaintreau,Pan Hui, Jon Crowcroft,Christophe Diot, Impact of Human Mobility on the Design of Opportunistic Forwarding Algorithms, IEEE INFOCOM 2006, April 23-29, Barcelona, Spain

[9] Lindgren and A. Doria. Probabilistic routing protocol for intermittently connected networks. Technical report, draft-lindgren-dtnrgprophet- 01.txt, IETF Internet draft, July 2005.

[10] J. Leguay, T. Friedman, and V. Conan. Dtn routing in a mobility pattern space. In ACM SIGCOMM Workshop on Delay Tolerant Networks, 2005.

[11] Apratim Mukherjee, Souvik Basu, Siuli Roy, Somprakash Bandyopadhyay, "Developing a Coherent Global View for Post Disaster Situation Awareness using Opportunistic Network", in Proceedings of the 7th International Conference on COMmunication Systems & NETworkS (COMSNETS) January, 2015

[12] Somprakash Bandyopadhyay, Debanjan DasDeb, "Post-disaster resource management using peer-to-peer opportunistic networks" Presented in The International Conference on Advances in Computing and Information Technology – (ACIT 2014) Bangkok, Thailand, 04-05 January, 2014.

[13] T. Vicsek, A. Czirok, E. Ben Jacob, I. Cohen, and O. Schochet. Novel type of phase transitions in a system of self-driven particles. Physical Review Letters, 75:1226– 1229, 1995.

[14] Czirok, A. L. Barabasi, and T. Vicsek. Collective motion of self-propelled particles:kinetic phase transition in one dimension. Physical Review Letters, 82:209–212, 1999.

[15] Herbert G. Tanner and Dimitrios K. Christodoulakis, "The stability of synchronization in local-interaction networks is robust with respect to time delays," 44th IEEE Conference on Decision and Control, 2005.