A Comparative Study of various Community Detection Algorithms in the Mobile Social Network

Ankit Didwania
IT Department, Faculty of Technology
Dharmsinh Desai University
Nadiad, Gujarat, India
Email: ankit.didwania@gmail.com

Zunnun Narmawala
Institute of Technology
Nirma University
Ahmedabad, Gujarat, India
Email: zunnun80@gmail.com

***Abstract*—Mobile social network is a type of delay tolerant network of mobile devices in which there is no end-to-end path available in advance for communication. It works on the principle of a store-carry-forward mechanism. The community is a very useful property of the mobile social network as humans are social animals and they like to live in a community. Such community structure enables efficient communication between devices carried by humans without any infrastructure. We have analyzed various community detection methods and identified those suitable for mobile social network. We have also analyzed various existing distributed community detection algorithms in mobile social networks based on important parameters like complexity and type of community detected. Such analysis will help in discovering strengths and shortcomings of various existing algorithms. As the mobile social network is self-organizing real network working on highly resource constraint mobile devices, it is necessary to enable each mobile device to detect its own community with minimal information, computation and space requirements. This is a very challenging task and very little work is done in it. So there is an immense opportunity available for research in this area.**

Keywords: Community, Delay Tolerant Network, Detection, Distributed, Dynamic, Mobile Social Network, Opportunistic Network, Overlapping

I. INTRODUCTION

Rapid exponential growth of the mobile network and devices have called for deeper and better understanding of underlying structures and explore its various usage opportunities. It motivates us to think about how the underlying structure of the network looks like and how do mobile devices organize themselves in such networks so that mobile devices can themselves detect their local community. By understanding their structural behavior, we can get benefit from the self-organizing networks like using them for the message forwarding mechanism where there is no infrastructure available. It is very useful in military and disaster situations.

Mobile Social Networks (MSN) or Opportunistic Networks utilize the obvious qualities of social animals like humans such as their day by day schedules, similarities, correlated interaction, etc. for performing message exchanges. Such networks are a variant of the mobile ad hoc network and come under Delay Tolerant Networks (DTN) space. MSN is different from Mobile Ad Hoc Networks (MANET) as MANETs require complete path from the source to destination for routing messages but MSN requires only the establishment of opportunistic contacts among mobile nodes.

A. DTN

According to [1], DTN is a heterogeneous network that may lack continuous network connectivity. DTN can have the presence of a disconnected or intermittent network with increased round-trip time, queuing time and latency but has reduced data rate and resource availability. Routing in DTN includes space and time dependency as links between nodes are a function of time whereas other networks have only space dependency. DTN follows store-carry-and-forward routing mechanism as compared to the store-and-forward routing mechanism in other networks. It can be used for information dissemination in an infrastructure-less environment where delay in the delivery of information is tolerable like villages, islands, disaster affected or wild areas, etc. Routine stuff like transport vehicles, postman or even pigeons can be used as data mules in such environment.

B. Simulator for Testing MSN

The implementation of various algorithms developed for MSN can be easily simulated in ONE simulator. The Opportunistic Network Environment (ONE) is a Java based simulator targeted for research in Delay Tolerant Networks (DTN) [2]. It's benefits include simulating different scenarios quickly and easily, generate various statistics in an easy manner, can be run on any Java-supporting platform and simulator can be easily carried in USB as its source code is less. The default configuration of the simulator can be conveniently overridden as per developer requirements. The simulation can be run either in batch mode through command prompt or in GUI mode.

II. COMMUNITY STRUCTURE

The community can be defined as a graph G = (V, E) where V is the set of vertices representing nodes or objects and E is the set of edges representing their relationship. As seen in the graph of Figure 1 from [4], community structure or clustering is a group of nodes, devices or individuals which has a dense connection or more interaction in its community (i.e. dense intra-community edges) and less or no connection or interaction across the community (i.e. less inter-community edges) [3], e.g. a community structure of families, friends, animals, sensors, etc.

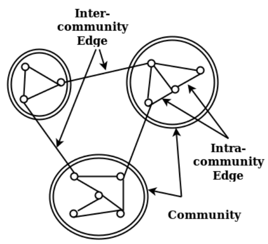


Fig. 1. Basic Community Structure

A. Benefits of Community Structure Analysis

Analysis of community structure helps to disclose relationships between nodes which may be hidden. This information can be used to define the network structure and use it for efficient data forwarding mechanism like identifying a shared node who can reliably transfer data between two or more different communities or an influential node which can quickly transfer data to all the nodes within its community.

B. Categories of Community Structure

There are 3 major categories of community structure as shown in Figure 2:

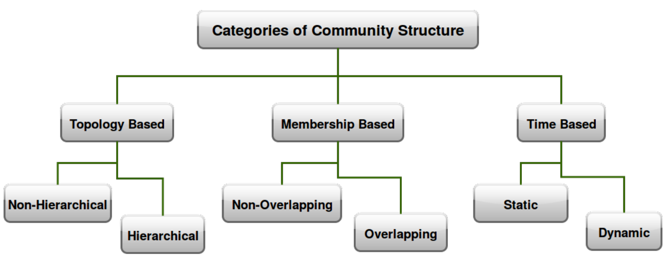


Fig. 2. Categories of Community Structure

a) Topology-Based Community Structure: It is based on exhibiting the multi-level structure of the graph as shown in Figure 3. In its hierarchical sub-category, the graph may exhibit various levels of communities containing smaller sub-communities within a larger community whereas non-hierarchical sub-category has a flat structure. Social networks often exhibit hierarchical structure as many subgroups may exist within a larger group like groups of friends, class-mates, colleagues within a specific university community.

b) Membership Based Community Structure: It is based on sharing of vertices or nodes between various communities as shown in Figure 4. In its overlapping sub-category, the node may be a member of more than one community whereas non-overlapping sub-category has single community member nodes only. Social networks often exhibit overlapping structure like a student could be a member of more than one community related to subjects, friends and activities.

c) Time Based Community Structure: It is based on the structure of a community changing with time as shown in Figure 5. In its dynamic sub-category, the community structure like the number of nodes and their relationships may evolve with time whereas its static sub-category has the same community structure at all time. Social networks often exhibit dynamic structure like a student community structure may change based on the time of the day like lecture time, lunch time, sports time, etc.

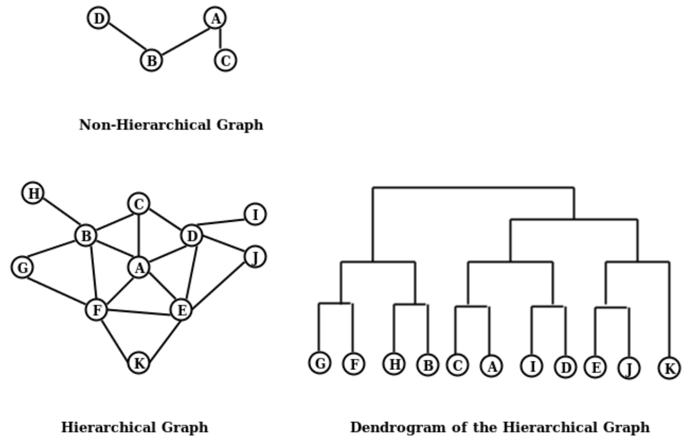


Fig. 3. Topology-Based Community Structure

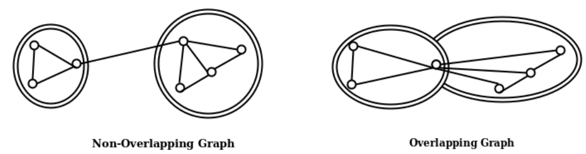


Fig. 4. Membership Based Community Structure

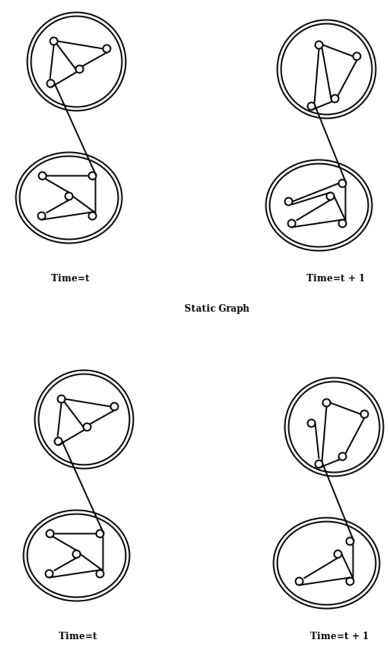


Fig. 5. Time Based Community Structure

A social network community structure can be a combination of any of the above-mentioned categories. There exist one more type of graph called random graph [5] which is a disordered graph as its edge distribution is highly homogeneous among the vertices. Random graphs cannot exhibit social network characteristics as they do not contain any group of nodes. Figure 6 from [4] is a random graph depicting a small world graph with 5 vertices and having the same rewiring probability for each vertex.

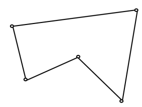


Fig. 6. Random Graph

C. Detection of Community Structure

Community detection is an important and challenging task in MSN because the mobile network is large and dynamic as the network is constantly changing due to new nodes coming in and out. Also, in overlapping community structure, correct community labels have to be given to each node.

Initially, it is required to test that the given graph is a random graph or contains any community structure. A null model [6] can be used to test this by matching the original graph with its randomly generated graph based on the structural features. After this test, community structure detection can be carried out.

There are 2 types of community structure detection methods [3] as shown in Figure 7.

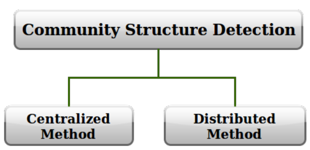


Fig. 7. Community Structure Detection

1) Centralized Community Detection
Such detection method can analyze data only in some centralized location (i.e. it requires centralized processing). It works only in off-line mode and could have high resource availability requirements. It's algorithm requires knowledge of whole network topology and mostly used to detect static communities.

2) Distributed Community Detection
Such detection method can analyze data on every node and does not require any centralized location for processing. It works in on-line mode (i.e. real time) and could have less resource availability requirements. It's algorithm does not require knowledge of whole network topology and can detect static as well as temporal/dynamic communities.

According to [4],the performance of various methods proposed for community detection is mostly evaluated on the basis of complexity like time, space, etc. and is tested using benchmark graphs which are simple and their community structure is also easy to detect. Due to this, it is not possible to judge the usability and effectiveness of the specified method with other existing methods. So, reliable and standard benchmark graphs (or quality functions) should be used whose community structure is complex but known, like models given in [7], [8], [9], [10], etc.

III. Community Detection Methods for Social Network

The social network is considered as a human network having social interactions using mobile phones and the Internet. It forms communities of friends, co-workers, like-minded people, etc. whose analysis is useful for social science. It is generally assumed to be a large scale and a complex network having dense clique (i.e. containing fully connected sub-graphs).

According to [11], a social network differs in two important ways from other types of networks including technological and biological networks. Firstly, in social network degrees of adjacent vertices have various patterns of correlation, mostly positive or assortative mixing whereas non-social networks have negative or non-assortative mixing. Secondly, there is a high level of network transitivity or clustering in social networks which is not seen in non-social networks. So, only a social network can be divided into communities [11].

We have summarized in Table I only those methods from [4], which are suitable for community detection in a social network.

In Hierarchical clustering method [12], a multi-level structure of the graphs containing hierarchy is revealed using dendrograms as shown in Figure 3. A similarity matrix is created by calculating the similarity between each pair of (connected or unconnected) vertices. It aims at detecting communities of vertices with high similarity. This method is divided into two categories. First, agglomerative algorithms which use bottom-up approach and merge highly similar clusters iteratively, but it does not scale well. Second, divisive algorithms which use top-down approach and split less similar clusters iteratively. It removes edges to make disconnected communities. The second method has now become more popular.

Greedy Technique [13] merges vertices iteratively to form a large community such that their overall modularity value is maximum. It is a quick method but has to compromise on accuracy.

Random Walk [14] works on the concept that a random walker will spend a long time in a dense community. Algorithms based on the random walk can be easily extended for weighted graphs also, but they cannot decide the best partition among available partitions.

Statistical Inference methods [15] use hypotheses based model to deduce the properties of a graph. Hypotheses are based on connectivity patterns of vertices. Bayesian inference under generative model and block modeling is used for analysis and modeling of social networks.

Label Propagation [16] repeatedly propagates unique information like vertex labels among its neighborhood until all vertices have the majority label of their neighbors. Community is formed by the vertices having the same labels. Overlap communities can be found by merging partitions. No pre-information or parameter is required but may give different results for different vertices.

Clique Percolation method [17] uses a clique which means a group of highly interactive people. The community of three nodes in Figure 8 is a connected sub-graph with a clique of size 3. It is based on the idea that inter-community edges will not form any clique whereas intra-community edges will form cliques. Nodes shared between different cliques can be used as shared or overlapped nodes as seen in Figure 9. This method can easily identify whether the graph is random or not but does not perform well if only a few cliques are available.

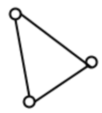


Fig. 8. k-Clique Graph

IV. A Literature Survey of the Community Detection Algorithms in MSN

Community detection in MSN has some unique characteristics as compared to other social networks like the structure of the entire network is not known and no predefined information like the number and size of communities is available. Also, there can be a mixed size of communities and a mixed number of cliques. The community detection should also be scalable.

A centralized community detection algorithm needs to know the structure of the network and also have some predefined information like the number and size of a community. Also, MSN works on highly resource constrained mobile devices with limited computation and space capabilities. So, the mobile devices in such self-organizing networks should quickly sense and locate their nearby groups themselves as opposed to depending on a centralized server. So the community detection algorithms in MSN needs to be distributed in nature as it possesses the characteristics mentioned above.

We have studied the major algorithms as shown in Table II, which can detect communities of MSN in a distributed manner.

The Simple method [28] detects a community by classifying each local node pairs contact duration based on an apriori threshold value. It gives good performance even in a resource constrained environment, but the quality of the community detected is not so good.

Modularity method [28] uses modularity value for community detection. It is not a suitable method of community detection in MSN due to its high resource requirements and comparatively poor results than K-clique. It uses the modularity-based method specified in Table I.

K-clique method [28] can detect k-clique community which is a union of all k-cliques that can reach each other through a series of adjacent k-cliques who share k-1 nodes as seen in Figure 9. The communities detected are much better, but it requires more space and computation in large networks which is painful in a resource constrained environment like mobile devices. Also, high overlap density decreases its performance. It uses the CPM technique specified in Table I.

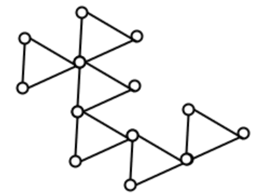


Fig. 9. Adjacent k-Clique Graph

TopGC method (Top Graph Cluster) [29] uses locality sensitive hashing which means that same community shares same hash value. It finds a set of nodes whose neighborhoods are highly overlapping as these nodes should be clustered together. So, it only searches clusters with highest scores. It's resource requirement is less, but can detect overlapping community up to a given percentage only. The algorithm can also detect directed, weighted and disjoint communities.

The SLPA method (Speaker-Listener Label Propagation Algorithm) [30] uses an agent-based algorithm. Each node sends its unique label value and listens to each of its neighbors. Then each node detects its community by choosing the most common label in its memory after repeating the process multiple times. So, it uses more memory as each node stores labels of all its neighbor. A single common label means a single community. A multiple label means an overlapping community. Although it is scalable but needs to process more label in large networks. So, its time complexity also scales linearly with the number of edges. It is good for detecting low overlapping density and nested hierarchy communities. The algorithm can detect directed, weighted and disjoint communities, but lacks precision in communities detected. It uses the label propagation technique specified in Table I.

PEC method (Periodic Encounter Communities) [31] uses a decentralized algorithm to detect periodic communities. Each node can detect and share its community periodicity by mining its encounter history and extract globally maximal PEC . It assumes sufficient opportunities for exchanging information among nodes which are not always possible in real-world. Also, the current algorithm is not able to maintain an updated view of the knowledge base as it does not prune old encountered nodes. The algorithm performance decreases with increase in the size of the community, so it is not good for large networks. Only this algorithm among all the algorithms mentioned in Table II can detect communities which are encountered periodically in the opportunistic network.

The AD-Simple method (Autonomic Detection using Simple) [32] is an extension of the Simple method mentioned in Table II. It is used to detect dynamic communities which the simple method fails to detect thus improving its results. It maintains a consistent and updated view of the neighborhood at each node and so reduces the computation and storage requirements.

DOCNET method (Detecting Overlapping Community in NETwork) [33] is based on local optimization of a fitness function and a fuzzy belonging degree of different nodes. It greedily extends from a seed node until a stopping criterion

TABLE I. SUMMARY OF VARIOUS COMMUNITY DETECTION METHODS FOR SOCIAL NETWORKS

Sr. No.	Methods	Main observations	Advantage	Disadvantage	Algorithms
A	Traditional Methods	They use traditional ways to detect communities			
1	Hierarchical Clustering [12]	It aims at detecting groups of vertices having high similarity. It reveals the multilevel structure of the graph using dendrograms. It has two categories of algorithms: 1. Agglomerative 2. Divisive	Can find large number of partitions	Partitions found are not equally good	Newman method [18], CONGA [19]
B	Modularity-Based Methods	They are popularly used to assess the stability of partitions.			
1	Greedy Technique [13]	It's based on the agglomerative hierarchical method for performing modularity optimization	Fast method	Accuracy is not good	Greedy method of Newman [13], Clauset [20], Wakita et. al. [21]
C	Dynamic Algorithms	They are used to detect and evolve dynamic communities			
1	Random Walk [14]	A random walker spends a long time inside a strong community structure	Can be easily extended for weighted graphs	Cannot decide the most meaningful partition from different partitions.	Zhou et. al. [22], Latapy et. al. [23]
D	Statistical inference methods [15]	It's methods aims at deducing properties of datasets. Important models are Generative Model, Block Modeling and Model Selection	Can analyze and model real graphs like social network	Vertices are to be classified based on their connectivity patterns	Handcock et. al. [24], Koskinen et. al. [25], Rhodes et. al. [26]
E	Alternative Methods	These are non-traditional methods.			
1	Label Propagation [16]	Repeatedly scans all vertices (initially having unique labels) and shares label with the majority of neighbors. Community is obtained from the dominating labels at each vertex.	A simple and fast method even for large systems. No parameter or pre-information needed.	A unique solution is not always obtained	Leung et. al. [27]
F	Overlap communities: Clique Percolation Method [17]	The idea is that internal edges of a community are likely to form cliques due to their high density. Clique Percolation Method (CPM) is a popular method for detecting overlapping communities.	Clear distinction between random graphs and graphs with community structure.	Assumes graph has large number of cliques, so fails with few number of cliques	Palla et. al. [17]

is not met. It builds community in an agglomerative manner as a two steps process: build core and then extend the core. It can find homogeneous communities and community exhibiting antagonistic behaviors which means set of community pairs behaving opposite to each other. According to [33], DOCNET is better than CPM method in detecting overlapping communities. It uses the hierarchical clustering technique specified in Table I.

According to [33] there are 5 broad methods of detecting overlap communities:

1) Graph Theory Partition: This method is used to detect community in the clique graph. It is a costly method as it uses high time and space complexity. It cannot work properly in a sparse network. CPM uses this method.
2) Link Partition: It uses line graph method. It is a costly method as it uses high time complexity.
3) Local Expansion and Optimization Partition: It relies on fitness function and neglects small size communities.
4) Fuzzy Partition: It quantifies the strength of association and determines adhesion to a group based on its degree. It computes communities with spherical shape only and not any arbitrary shapes.
5) Agent-based Partition: It uses labels to identify membership of vertices and propagates it. It provides better results only for small communities.

V. OPEN ISSUES

Algorithms detecting communities in MSN have some open issues in general like detecting community in a noisy network that exhibit a high number of changes over time, ability to cover uncertainty in nodes periodic patterns, ability to detect weighted and directed networks, ability to detect various types and density of communities in real time self-organizing networks, maintaining privacy of users and securing message content, reducing algorithmic overhead and improving the quality of service, ability to handle selfish nodes not willing to be the routing path for others, etc. Also community detected needs to be meaningful, stable, scalable and accurate. Reducing complexity in terms of the computation (time taken to detect and report community), resource requirements (like memory space) and implementation or coding difficulty in large networks is also a major issue. The algorithm must be able to work on encounter traces from a variety of domains from the real world.

VI. CONCLUSION

Many centralized algorithms for detecting community/cluster/groups from graphs exist for centuries, but they cannot be directly applied to MSN because of the unique characteristics of MSN. Although, they can be extended to work in MSN as shown in Table II. Not much work is done in detecting communities of MSN in a distributed and efficient manner and so there is a huge scope available for conducting research in this area.

REFERENCES

[1] Daly, Elizabeth M., and Mads Haahr, *Social network analysis for routing in disconnected delay-tolerant manets*, In Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing, pp. 32-40. ACM, 2007.

TABLE II. SUMMARY OF VARIOUS COMMUNITY DETECTION ALGORITHMS IN MSN

Algorithm	Complexity	Type Detected	Main Observations
Simple [28]	Quick, Less Mem.	Static, Non-hierarchical, Non-overlapping	Uses fewer resources, but does not provide results better than K-clique
Modularity [28]	Slow, More Mem.	Static, Non-hierarchical, Non-overlapping	Uses modified modularity method, but does not provide results better than K-clique even after using more resources
K-Clique [28]	Quick, More Mem.	Static, Non-hierarchical, Overlapping	Uses CPM which assumes groups consist of fully connected subgraphs but fails to terminate in large social network
TopGC [29]	Slow, More Mem.	Static, Non-hierarchical, Overlapping	Top Graph Clusters uses locality sensitive hashing and can detect groups in directed and weighted networks using a probabilistic clustering algorithm
SLPA [30]	Quick, More Mem.	Static, Hierarchical, Overlapping	Speaker-listener label propagation method can detect groups in directed and weighted networks, but lacks accuracy
PEC Detection method [31]	Slow, More Mem.	Static, Hierarchical, Overlapping	Detects the periodic encounter community for each node using decentralized algorithm
AD-Simple [32]	Quick, Less Mem.	Dynamic, Non-hierarchical, Non-overlapping	It adaptively detects dynamic social communities and gives results better than Simple
DOCNET [33]	Slow, More Mem.	Dynamic, Hierarchical, Overlapping	Detects overlapping community in a complex network using agglomerative hierarchical clustering based on fuzzy membership degree

[2] Kernen, Ari, Jrg Ott, and Teemu Krkkinen, *The ONE simulator for DTN protocol evaluation*, In Proceedings of the 2nd international conference on simulation tools and techniques, p. 55. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

[3] N. P. Nguyen, T. N. Dinh, S. Tokala, and M. T. Thai, *Overlapping communities in dynamic networks: their detection and mobile applications*, Proc. 17th Annu. Int. Conf. Mob. Comput. Netw., pp. 85-96, 2011.

[4] S. Fortunato, *Community detection in graphs*, Phys. Rep., vol. 486, no. 3-5, pp. 75-174, Feb. 2010.

[5] Erdos, Paul, and Alfrd Rnyi, *On random graphs*, Publicationes Mathematicae Debrecen 6 : 290-297, 1959.

[6] Newman, Mark EJ, and Michelle Girvan, *Finding and evaluating community structure in networks*, Physical review E 69, no. 2 : 026113, 2004.

[7] Condon, Anne, and Richard M. Karp, *Algorithms for graph partitioning on the planted partition model*, Random Structures and Algorithms 18, no. 2 : 116-140, 2001.

[8] Girvan, Michelle, and Mark EJ Newman, *Community structure in social and biological networks*, Proceedings of the National Academy of Sciences 99, no. 12 : 7821-7826, 2002.

[9] Fan, Ying, Menghui Li, Peng Zhang, Jinshan Wu, and Zengru Di, *Accuracy and precision of methods for community identification in weighted networks*, Physica A: Statistical Mechanics and its Applications 377, no. 1 : 363-372, 2007.

[10] Sawardecker, Erin N., Marta Sales-Pardo, and L. A. Nunes Amaral, *Detection of node group membership in networks with group overlap*, The European Physical Journal B-Condensed Matter and Complex Systems 67, no. 3 : 277-284, 2009.

[11] Newman, Mark EJ, and Juyong Park, *Why social networks are different from other types of networks*, Physical Review E 68, no. 3 : 036122, 2003.

[12] Friedman, Jerome, Trevor Hastie, and Robert Tibshirani, *The elements of statistical learning*, Vol. 1. Springer, Berlin: Springer series in statistics, ISBN 0387952845, 2001.

[13] Newman, Mark EJ, *Fast algorithm for detecting community structure in networks*, Physical review E 69, no. 6 : 066133, 2004.

[14] Hughes, B. D, *Random walks and random environments. Vol. 1. Random walks. Clarendon*, ,1995.

[15] MacKay, David JC, *Information theory, inference, and learning algorithms*, Vol. 7. Cambridge: Cambridge university press, 2003.

[16] Raghavan, Usha Nandini, Rka Albert, and Soundar Kumara, *Near linear time algorithm to detect community structures in large-scale networks*, Physical Review E 76, no. 3 : 036106, 2007.

[17] Palla, Gergely, Imre Dernyi, Ills Farkas, and Tams Vicsek, *Uncovering the overlapping community structure of complex networks in nature and society*, Nature 435, no. 7043 : 814-818, 2005.

[18] Newman, Mark EJ, *Analysis of weighted networks*, Physical Review E 70, no. 5 : 056131, 2004.

[19] Gregory, Steve, *An algorithm to find overlapping community structure in networks*, In Knowledge discovery in databases: PKDD, pp. 91-102. Springer Berlin Heidelberg, 2007.

[20] Clauset, Aaron, Mark EJ Newman, and Cristopher Moore, *Finding community structure in very large networks*, Physical review E 70, no. 6 : 066111, 2004.

[21] Wakita, Ken, and Toshiyuki Tsurumi, *Finding community structure in mega-scale social networks:[extended abstract]*, In Proceedings of the 16th international conference on World Wide Web, pp. 1275-1276. ACM, 2007.

[22] Zhou, Haijun, and Reinhard Lipowsky, *Network brownian motion: A new method to measure vertex-vertex proximity and to identify communities and subcommunities*, In Computational Science-ICCS 2004, pp. 1062-1069. Springer Berlin Heidelberg, 2004.

[23] Latapy, M., and P. Pons, *Lecture Notes Computer Science*, 3733, 284, 2005.

[24] Handcock, Mark S., Adrian E. Raftery, and Jeremy M. Tantrum, *Model based clustering for social networks*, Journal of the Royal Statistical Society: Series A (Statistics in Society) 170, no. 2 : 301-354, 2007.

[25] Koskinen, Johan H., and Tom AB Snijders, *Bayesian inference for dynamic social network data*, Journal of statistical planning and inference 137, no. 12 : 3930-3938, 2007.

[26] Rhodes, C. J., and E. M. J. Keefe, *Social network topology: a Bayesian approach*, Journal of the operational research society 58, no. 12 : 1605-1611, 2007.

[27] Leung Ian , Pan Hui, Pietro Lio, and Jon Crowcroft, *Towards real-time community detection in large networks*, Physical Review E 79, no. 6 : 066107, 2009.

[28] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, *Distributed community detection in delay tolerant networks*, Proc. first ACM/IEEE Int. Work. Mobil. Evol. internet Archit. - MobiArch-07, p. 1, 2007.

[29] K. Macropol and A. Singh, *Scalable discovery of best clusters on large graphs*, Proc. VLDB Endow., vol. 3, no. 1-2, pp. 693-702, Sep. 2010.

[30] Xie, Jierui, Boleslaw K. Szymanski, and Xiaoming Liu, *Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process*, In Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, pp. 344-349. IEEE, 2011.

[31] M. J. Williams, R. M. Whitaker, and S. M. Allen, *Decentralised detection of periodic encounter communities in opportunistic networks*, Ad Hoc Networks, vol. 10, no. 8, pp. 1544-1556, Nov. 2012.

[32] E. Borgia, M. Conti, and A. Passarella, *Autonomic detection of dynamic social communities in Opportunistic Networks*, 2011 10th IFIP Annu. Mediterr. Ad Hoc Netw. Work., pp. 142-149, Jun. 2011.

[33] D. Rhouma and L. Ben Romdhane, *An efficient algorithm for community mining with overlap in social networks*, Expert Syst. Appl., vol. 41, no. 9, pp. 4309-4321, Jul. 2014.

QoS Aware Geographic Routing Protocol for Multimedia Transmission in Wireless Sensor Network

Vijay Ukani
Computer Science and Engineering,
Inst of Technology, Nirma University,
Ahmedabad, India
Email: vijay.ukani@nirmauni.ac.in

Dhaval Thacker
Computer Science and Engineering,
Inst of Technology, Nirma University,
Ahmedabad, India
Email: dhaval88.thacker@gmail.com

***Abstract*—Wireless sensor network (WSN) is a network of small, low power, resource constrained devices distributed in a large area. The principle focus of such network is to monitor some well-defined phenomenon in the sensor field. Availability of low power CMOS camera has offered wings to WSN applications. Broadening the range of applications from monitoring to multimedia surveillance. Multimedia transmission brings newer challenges primarily provisioning of the quality of service. Real-time video surveillance is one class of application which expects timely delivery of the packets. QoS aware routing protocols are required to cater to this demand. Geographic routing protocols are most suited WSN due to its scalability. GPSR, SPEED, and MMSPEED are the protocols which explore 1-hop neighborhood information in packet forwarding. Most of these protocols stations their routing decision on the location of sink and ignoring energy reserves of a node, which is crucial in each WSN application. A geographic routing protocol is proposed in this paper, which is energy aware. Also rather than sticking to maintaining 1-hop neighborhood information, it maintains n-hop neighborhood which helps early detection of void.**

***Index Terms*—SPEED, MMSPEED, N-Hop Neighborhood, Geographic routing protocol, QoS aware routing, Wireless Multimedia Sensor Networks**

I. Introduction

Wireless sensor network is a collection of small low-power, resource constrained sensor nodes capable of sensing many environmental parameters like temperature, pressure, humidity etc[1]. These devices are distributed autonomously across the large area to monitor physical or environmental conditions[2]. WSN mainly consists of sensor nodes with small irreplaceable batteries & densely deployed in the monitoring area.

Upon detecting an event of interest, the event information is disseminated in the network so that it reaches the sink or base station node. Using only scalar sensor nodes tends to suffer from false alarm rate as it possible to trifle scalar sensors. The visual capture of the phenomenon can gather the event information from multiple dimensions resulting into confirming the event. Thus reducing false triggers[3]. As visual i.e. video information needs a large number of bits to be captured and transferred, bandwidth consumption becomes a challenge. Bandwidth consumption can be reduced to some extent by using suitable video encoding like distributed video coding[4], [5], [6]. Also in applications like battlefield surveillance, where timely delivery of the information is very critical to the success of the application, routing protocol assumes an important role[7], [8], [9]. So, a key challenge in the design of WSN is to prolong the lifetime of sensor nodes. Packet routing plays an important role in assuring optimum energy usage and timely delivery of the packet.

Availability of low-power CMOS camera has lead to tremendous growth in applications of WSN. A network containing multi-modal sensors capable of retrieving scalar along with audio/video information is called as Wireless multimedia sensor network(WMSN). With the advancements in hardware availability and design, signal processing techniques, advanced coding theory, and optimal protocol design have made WMSNs a reality[10]. WMSNs have applications in video surveillance, smart traffic control, person locator services, and health care. When a multimedia sensor is used, audio and video data is to be transmitted. Most applications will expect delay bound in data communication as these are assumed to be real-time applications.

This paper aims at designing a QoS and energy aware geographic routing protocol which tries to send the packets over a path which can meet the end-to-end deadline of the packet. Most of the geographic routing protocols uses greedy forwarding, which might encounter void/hole in the network where no further progress is possible. In such cases, the packet is to be rerouted in the backward direction, leading to higher probability of missing deadline for such packets. Geographic routing protocols are scalable as they maintain information about 1-hop neighbors only. Availability of the limited information forces the nodes to make a greedy decision which might lead to void later. Protocol proposed in this paper gathers n-hop neighborhood information. It will enable the node to estimate the availability of path up to n-hops. Thus avoiding possibility of a void. The residual energy of the neighbor nodes also aids in choosing the next hop, making it an energy aware protocol. The objective of the paper is to improve the ratio of number of packets meeting deadline in

978-1-4799-9991-0/15/$31.00 ©2015 IEEE

an energy efficient manner.

The rest of the paper is organized as follows. Section II describes an overview of existing protocols to support multimedia packets. Section III provides proposed enhancements in existing protocols. Design details of the protocol is described in Section IV. Section V provides and insight into simulation setup and performance evaluation. Finally, Section VI concludes the paper with summary and directions for future work.

II. Routing Protocols for WMSN

Most of the applications of WMSNs demands timely delivery of packets. Many approaches in literature are proposed which are either multipath based or ensuring end-to-end delay. Multipath routing protocols works by distributing the packet stream in multiple paths[11], [12], [13]. This will reduce chances on congestion on the individual paths. The packets are transmitted over multiple path to ensure that packets do not suffer much delay due to congestion on different path. Other class of routing protocols operate by estimating the remaining distance to travel and slack deadline. The packets are assigned priority based urgency required to meet the deadline.

Several protocols exists like SPEED [14], MMSPEED [12], [15], PASPEED[16], EAQoS[17], RPAR[18], RAP[19] etc. GPSR is a geographic routing protocol designed for wireless sensor network[20]. It uses greedy forwarding for selecting the next hop. The node which makes maximum progress towards the destination is used as next hop. Sometimes it meets void/hole where it may not be possible for a packet to progress forward as no path towards the destination is available from that node. This kind of void/hole should be avoided by the protocol. GPSR uses perimeter routing to avoid the void. It does not provide any timeliness assurance to the packets. So it is not suitable for real-time video transmission.

SPEED and MMSPEED are well known algorithm based on GPSR. Major goals of these routing protocols are providing timeliness and reliability guarantee. SPEED[14] provides timeliness guarantee but lacks on reliability assurance. It maintains predefined network-wide speed and efficiently handles congestion control. There is no priority, delivery guarantee or condition based dynamic speed. It has random packet drop behavior in case of congestion. It uses geographic positions to make the decisions locally. Fig. 1 shows architecture of SPEED protocol and relationship of routing layer between MAC and application layer.

SPEED has six different phases Beacon interchange, Delay estimation, Stateless Non-deterministic Geographic Forwarding (SNGF), Neighbor Feedback Loop (NFL), Back pressure routing and Void avoidance.

Beacon interchange is responsible for interchanging three different types of information like - location information, delay estimation to neighbor nodes and on demand back pressure beacons.

Delay estimation is responsible for collecting delay information of neighboring nodes by broadcasting HELLO packets. Each node maintains delay information of its neighboring nodes and uses it to make routing decisions.

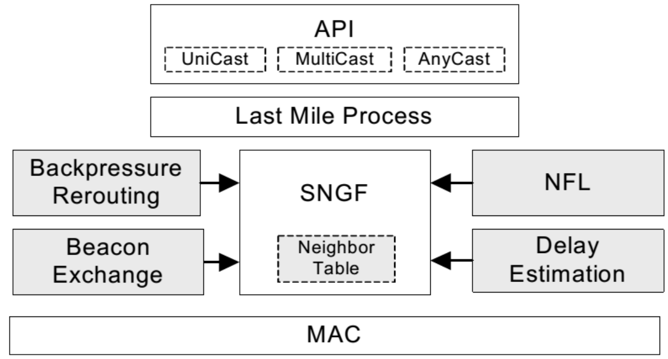


Fig. 1. Modules of SPEED[14]

SNGF will divide all nodes into two different sets - Forwarding Set and remaining set. Nodes in forwarding set are in the direction towards the sink. SNGF calculates distance and speed based on location information and delay information. As per calculated distance and speed, it finds best next hop to forward a packet. If it can not provide predefined speed then it will drop packet randomly and activate NFL mechanism.

NFL module maintains system's performance and it decides whether to drop a packet or not in case of lack of nodes which provides predefined speed. NFL also generate back pressure beacon and send it to neighbor nodes to reduce the traffic and inform neighbor nodes about the severity of congestion.

Backpressure rerouting is used in the event of congestion. If a node is congested, it sends increased delay to an upstream neighbor node from where data packet came. This ensures that the node is less preferred as next hop.

Void avoidance is responsible for sending back pressure beacons with infinite delay in case of no forward path is possible from that node. The scenario is demonstrated in Fig. 2.

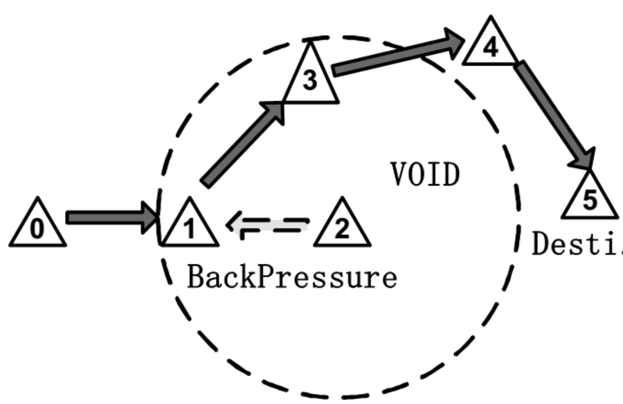


Fig. 2. Void avoidance with backpressure beacon in SPEED[14]

MMSPEED[12], [15] is an extension of SPEED protocol. It provides timeliness guarantee inherently and reliability additionally. It supports various speed and reliability levels.

MMSPEED works dynamically by providing change of speed and path as per service requirements. It offers various speed levels. Different packets will need different speeds to meet the end-to-end deadline. Based on the speed requirement, MMSPEED assigns packets to different speed layers. It also assures reliability to the packet by estimating the reliability of the path through a node based on the current error rate. Since the reliability of a path is estimated based on 1-hop neighbor information only, it may not be accurate enough. The estimation can be improved if n-hop information is available. The absence of energy consideration is still a problem with MMSPEED. PASPEED [16] is a power-aware variant of SPEED. It dynamically controls the transmission power based on distance to the selected next hop. It uses minimum power required to transmit a packet. RPAR[18] maintains a neighbor table which includes neighbor and power level. RPAR forwards the packet using most energy efficient forwarding choice that meets the velocity requirements of the packet.

III. Proposed Enhancement

Based on the literature study, two types of problems were identified. Nodes are static, and destination is also fixed in most applications. Packets are sent to some particular node every time, which causes the reduction of energy in that node and as a result that chances of that node running out of energy is very high. It will affect the connectivity of network.

Random deployment or failure of some nodes creates void in the network. These routing protocols uses geographic location information of immediate neighbor. A node does not know have any information about the further condition of the path. If nodes have information about nodes which are further than 1-hop, it can estimate the presence of hole before taking the routing decision. It can save energy and efforts required for rerouting the packet. Therefore to achieve better results, the proposed protocol uses the residual energy of neighbor nodes as well as multi-hop neighbor information for identifying the hole in a network as early as possible.

A. N-hop neighborhood

Several changes are required to be made in SPEED protocol's neighbor table structure, the way of exchanging of neighbor's information and taking routing decisions. The objective is to have multi-hop neighborhood information in the table. Maintaining single-hop information makes it a reactive routing protocol. If global information about the network is maintained, it becomes a proactive routing. A strike balance between the two, it maintains n-hop information in the table. If n is high, energy consumption and delay in maintaining the table will increase and lifetime of nodes will decrease. Lower values of n were tried in the protocol. The proposed protocol was tested with 2-hop neighborhood information.

B. Routing Decision Parameters

Energy is an important factor in routing decisions because if a forwarding node is selected repeatedly, it might run out of energy and that part of network may be disconnected from sink node. So, forwarding node is selected based on speed offered as well as remaining energy of the node.

IV. Design of Proposed Protocol

The overall architecture of the proposed protocol is similar to SPEED protocol. Modifications were made in several components.

A. Alteration in Node's local structure

SPEED maintains a setSpeed parameter, which is the target speed to be achieved across the network. In proposed protocol, a variable, N-Hop-count, is available to all nodes. It informs the nodes about depth of neighborhood to be maintained in the table.

B. Structure of neighbor table

Originally SPEED maintain the Node Id, Delay information, Time stamp, X and Y coordinates, No. of times queried for finding next best hop and No. of times failed to fulfill desire speed. Last two parameters useful for NFL i.e. Neighbor Feedback Loop.

In proposed protocol, neighbor table maintains existing information. Over and above this, it is required to maintain new fields like, previous-Hop which maintains previous node's information e.g. If node 3 is two hop neighbor from node 1 via node 2 then it keeps an entry of node 3 in node 1's neighbor table with previous-Hop as 2. In case of immediate neighbor node, this value will be treated as -1. One more variable, hop-Count, will keep the information about the node's distance in terms of number of hops from self node.

C. Way of exchanging neighbor's information

SPEED keeps immediate neighbor information. So, when a HELLO beacon packet is broadcasted, all 1-hop neighbors will receive the beacon and they acknowledge the HELLO beacon. Upon receiving the acknowledgment, the source node can estimate the delay to that node from round trip time. Here it is assumed that all the network nodes are synchronized in terms of the clock as well as start up time.

In proposed protocol, at every HELLO beacon interval, node broadcast HELLO beacon with a counter indicating hop count the packet should travel. This counter value is dictated by the requirement of N-Hop neighborhood. It will be decremented each time until counter gets 0. On the way, each node acknowledges the sender and broadcast it to the neighbor. In case of acknowledgment, hop count will be set to 1 and further incremented by each hop upon forwarding by intermediate node.

D. Neighbor table updation

SPEED has simple rule to update and refresh the neighbor table like, whenever it receives the acknowledgment, it first checks neighbor table, whether an entry for particular node is exist or not? If one exist, it overwrites otherwise creates a new entry with current time. The neighbor entries are invalidated after refresh-interval.

In proposed protocol, whenever a node receives an acknowledgment of HELLO packet, it first check the existence of entry and act accordingly. It also stores nodeId from where the acknowledgment is received in previous hop field. In case there are multiple entries for the same neighbor with one of the previous hops as some nodeId and other entry with -1, then later entry will be prioritized. The neighbor entries are invalidated like classical SPEED.

E. Routing decision

SPEED protocol divides nodes in the neighbor table into two categories: forwarding set and another set. Nodes in forwarding set are in the direction of the sink and are only considered as probable forwarding node. If this set is empty the packet is dropped. In forwarding set, search for the node with best speed and use it as a forwarding node. It is biased toward speedy nodes i.e. whenever node which provides the best speed is alive, it always provides the best speed. If the network is assumed to have stationary nodes, it will lead to failure of such node due to energy depletion.

In proposed protocol, the neighbor table has more number of records compared to SPEED. It has records of all neighbors up to n-hop. So, searching for a node with best speed will not be straight forward. It searches for nodes which provide iterative best speed among the Hop-count's highest value to lowest (i.e. 0 in case of no 2-hop nodes found). The idea is to search for best speed node which provides the longest connectivity towards sink from the current node. Once a node is identified, find the immediate neighbor through which that best node is reachable using recursion and then forward the packet to such node, it ensures long connectivity in the network. It will also incorporate energy reserves of the selected node. So, to select a node with significant remaining energy, it multiplies speed with remaining energy to find the best node.

V. Simulation and Result Analysis

The performance of protocol was evaluated in OMNeT++ based Castalia simulator. $M3WSN$ framework[21] which is extension to Castalia was used to generate video data. Following table summarizes simulation scenario.

TABLE I
Simulation Parameters

Parameter	Value
No. of Nodes	18
Sink node	Node Id : 14
Simulation Time	1500 Seconds
Area	110 x 110
Set Speed	4800 m/s
N-Hop-Value	2
MAC Protocol	Tunable MAC
Radio parameters	CC2420
Initial Energy	100 Joules

Topology used to simulate the protocol is shown in Fig. 3. The placement of the nodes are as shown in the figure. It is designed to create to void in the network. A hole is present near to node 7 and 8 in topology.

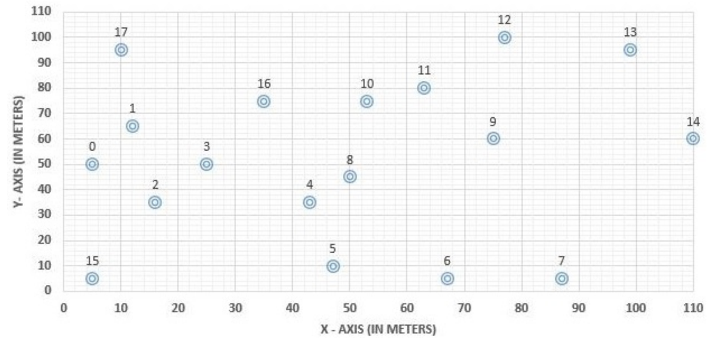


Fig. 3. Topology with 18 nodes

A. Results Analysis

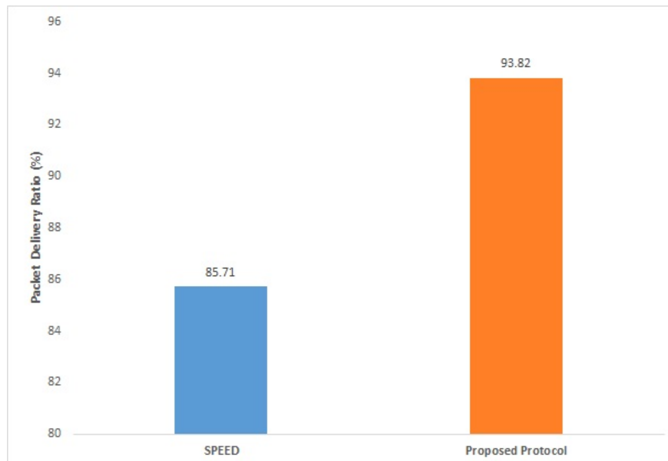


Fig. 4. Packet Delivery Ratio

1) Packet Delivery Ratio: Average packet delivery ratio of all nodes is shown in graph in Fig. 4. It is ratio of total number of packets transmitted from all node to received at sink. It is clearly observed that proposed protocol has better packet delivery ratio. The reason for the improvement is that it maintains n-hop neighborhood information in the neighbor table. Due to this a node before hand can estimate quality of the path in terms of congestion or presence of void. It reduces the chances of packet drop and backpressure rerouting.

2) Deadline Meeting Ratio: The primary goal of the this class of routing protocol is to make sure maximum packets meets the end-to-end deadline. Graph in Fig. 5 shows percentage of packets meeting deadline. Each packet is set with deadline within which it is supposed to reach destination. If packet reaches within given deadline, it is considered a meet. As 2-hop neighborhood is explored in the simulation, protocol identifies the void in network early as compared to SPEED algorithm. In case a void is encountered, the packet is to be rerouted backward for less number of hops. Ultimately time is saved in unnecessary forwards to non optimal nodes. First shows packet delivery ratio for number of packets reached within deadline against total number of packets sent. Second shows same ratio against number of packets delivered.

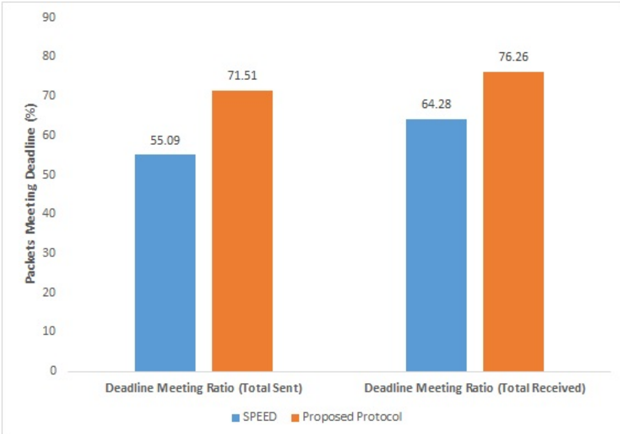


Fig. 5. Ratio of Packets Meeting Deadline

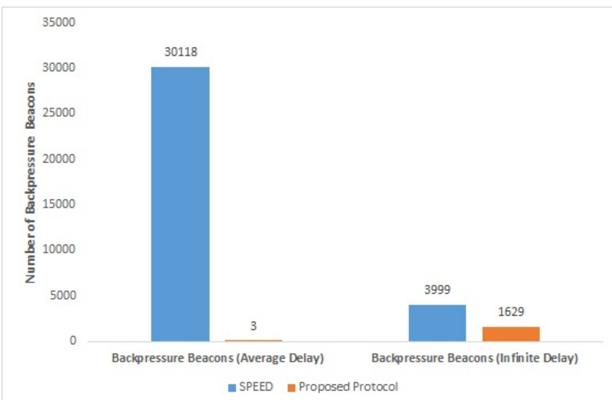


Fig. 6. Backpressure beacons

3) Backpressure Beacons: Back pressure beacons are generated when a node is facing congestion or it can not make any progress forward. Depending on congestion, there are two types of beacons - One sets infinite delay in case of hole in network and Second will set average delay in case of congestion i.e. none of the nodes are providing required speed. It can be observed that number of average delay beacons are very less in proposed protocol compared to SPEED. Considerably less number of infinite back pressure beacons are generated in proposed protocol compares to SPEED. We can conclude that proposed protocol has less possibility of congestion and void and thus less number of beacons are generated.

4) Energy Consumption: Graph in Fig. 7 shows average energy consumption in network by all nodes. There are less number of back pressure beacons generated in proposed protocol compared to SPEED. We can observe slight improvement in energy consumption in proposed protocol compared to traditional SPEED.

5) Network Lifetime: Life time of the network is measured as seconds till first node dies. Fig. 8 shows comparison of both protocols in terms of the lifetime. As the protocol is energy aware, it makes forwarding choice based on residual energy.

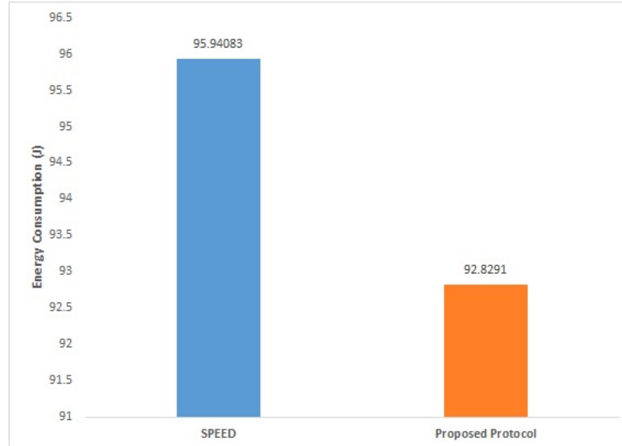


Fig. 7. Energy Consumption (Joules)

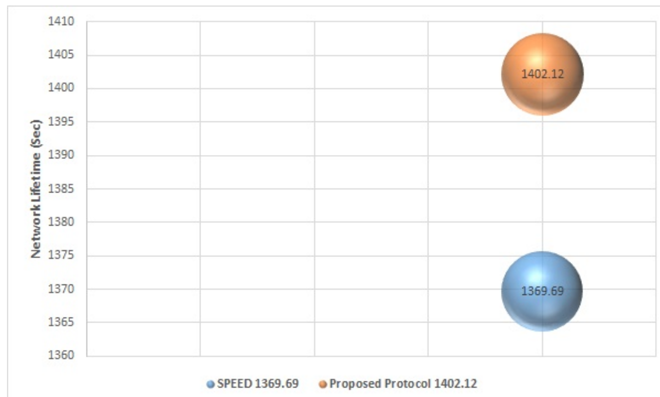


Fig. 8. Network Lifetime

This will avoid selecting the same node repeatedly. As a result load is balanced in the network and network lifetime should improve.

VI. Conclusion

Most deadline driven QoS aware routing protocols are greedy in nature while forwarding a packet. If multi-hop information is made available to the protocols, it can perform better by estimating the congestion, presence of void, reliability etc over the path. This will enable the routing engine in making a better decision than in case of greedy forwarding. Also most protocols while considering quality of service, ignores the other important parameter, i.e. energy efficiency. The protocol proposed in this paper looked at both, the QoS and energy efficiency and results clearly shows improvements compared to traditional QoS aware, deadline driven routing protocol, SPEED. The proposed protocol outperforms SPEED when the network has voids present. The protocol was tested on sufficient number of topologies with voids in the network. We could get improvements in packet delivery ratio, deadline meet ratio, energy consumption and thus network lifetime.

The work can be extended to set value of N in N-hop neighborhood dynamically. It can change based on some network parameters like network size, node density, neighbor density and velocity offered by nodes etc.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, pp. 102–114, Aug. 2002.

[2] M. Bokare and M. A. Ralegaonkar, "Wireless sensor network: A promising approach for distributed sensing tasks," *Excel Journal of Engineering Technology and Management Science*, vol. 1, pp. 1–9, 2012.

[3] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer Networks*, vol. 51, pp. 921–960, 2007.

[4] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," in *PROC. IEEE, SPECIAL ISSUE ON ADVANCES IN VIDEO CODING AND DELIVERY*, pp. 71–83, Stanford University, IEEE, January 2005.

[5] A. Vitali, "Multiple description coding: A new technology for video streaming over internet," tech. rep., ST Microelectronics, 2007.

[6] S. Misra, M. Reisslein, and G. Xue, "A survey of multimedia streaming in wireless sensor networks," *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, vol. VOL. 10, NO. 4, pp. Pages 18–39, 2008.

[7] T. Bokareva, W. Hu, S. Kanhere, and B. Ristic, "Wireless sensor networks for battlefield surveillance," *Proceedings of the Land Welfare Conference*, 2006.

[8] E. Felemban, "Advanced border intrusion detection and surveillance using wireless sensor network technology," *Int'l J. of Communications, Network and System Sciences*, vol. 6, no. 5, pp. 251–259, 2013.

[9] M. Kumhar and V. Ukani, "Survey on qos aware routing protocols for wireless multimedia sensor networks," *International Journal of Computer Science & Communication*, vol. 6, pp. 121–128, March 2015.

[10] I. Akyildiz, T. Melodia, and K. Chowdhury, "Wireless multimedia sensor networks: Applications and testbeds," *Proceedings of the IEEE*, vol. 96, no. 10, 2008.

[11] Y. Chen and N. Nasser, "Energy-balancing multipath routing protocol for wireless sensor networks," in *Proceedings of the 3rd international conference on Quality of service in heterogeneous wired/wireless networks*, QShine '06, (New York, NY, USA), ACM, 2006.

[12] E. Felemban, C.-G. Lee, E. Ekici, R. Boder, and S. Vural, "Probabilistic qos guarantee in reliability and timeliness domains in wireless sensor networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 4, pp. 2646 – 2657, march 2005.

[13] B.-Y. Li and P.-J. Chuang, "Geographic energy-aware non-interfering multipath routing for multimedia transmission in wireless sensor networks," *Inf. Sci.*, vol. 249, pp. 24–37, 2013.

[14] T. He, J. A. Stankovic, C. Lu, and T. F. Abdelzaher, "Speed: A stateless protocol for real-time communication in sensor networks," in *International Conference on Distributed Computing Systems (ICDCS 2003)*, 2003.

[15] E. Felemban, C.-G. Lee, and E. Ekici, "Mmspeed: Multipath multi-speed protocol for qos guarantee of reliability and timeliness in wireless sensor networks.," *IEEE Trans. Mob. Comput.*, vol. 5, no. 6, pp. 738–754, 2006.

[16] V. Ukani, A. Kothari, and T. Zaveri, "An energy efficient routing protocol for wireless multimedia sensor network," in *International Conference on Devices, Circuits and Communication*, September 2014.

[17] K. Akkaya and M. Younis, "An energy-aware qos routing protocol for wireless sensor networks," in *Proc. of the IEEE Workshop on Mobile and Wireless Networks (MWN 2003*, pp. 710–715, 2003.

[18] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. A. Stankovic, and T. F. Abdelzaher, "Real-time power-aware routing in sensor networks.," in *IWQoS*, pp. 83–92, IEEE, 2006.

[19] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He, "Rap: A real-time communication architecture for large-scale wireless sensor networks.," in *IEEE Real Time Technology and Applications Symposium*, pp. 55–66, IEEE Computer Society, 2002.

[20] B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proceedings of the sixth annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'00)*, (Boston, Massachusetts), pp. 243–254, August 2000.

[21] D. Rosário, Z. Zhao, C. Silva, E. Cerqueira, and T. Braun, "An omnet++ framework to evaluate video transmission in mobile wireless multimedia sensor networks," in *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, SimuTools '13, (ICST, Brussels, Belgium, Belgium), pp. 277–284, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013.

Adaptive Routing with Congestion Control in Delay Tolerant Network

Pravesh Patel[1], Ph.D. Scholar,
Ganpat University,
Gujarat, India
pravesh.patel@ganpatuniversity.ac.in

Hemal Shah[2], Professor
Ganpat University,
Gujarat, India
hemal.shah@ganpatuniversity.ac.in

Binjal Patel[3], PG Student
Ganpat University,
Gujarat, India
bhadi.madhiya@gmail.com

Abstract— Delay-Tolerant Networking is a communication networking paradigm that enables communication in environments where there may be no end-to-end paths. Routing in Delay Tolerant Network is different from other traditional networks. It used store carry and forward mechanism by overlaying new protocol known as bundle layer. Bundles are known as messages and it suppresses the delay. The bundle layer stores and forwards entire bundles (or bundle fragments) between nodes. In this paper, we propose a routing protocol named as Adaptive routing protocol with congestion control mechanism (AJRMP) that includes two mechanisms: Message Migration and Message Replication. Each mechanism has two steps: Message selection and Node selection. In message replication, we select a smallest TTL message to replicate. The TTL Threshold is used to control the replication speed. We propose a metric called 2-hop Activity Level to measure the relay nodes transmission capacity, which is used in node selection. This protocol includes a novel message migration policy that is used to overcome the limited buffer capacity and bandwidth of DTN nodes. Then AJRMP protocol is compared with the existing protocols and Joint Replication Migration Based Routing Protocol known as JRMBRP for delivery ratio, overhead, average buffer time and the number of messages dropped. This has been carried using Opportunistic Network Environment simulator under real and synthetic datasets.

Index Terms — Delay Tolerant Networks (DTNs), Message Replication, Message Migration, Buffer Management, Routing, AJRMP, JRMBRP.

I. INTRODUCTION

Delay Tolerant Network[1][2] is kind of opportunistic network where most of the time there does not exist end to end path from source to destination. Here, the performance of the network may vary significantly, depending on how the mobile nodes move, how dense the nodes populations are, and how far apart the sender and the receiver are. Delivery latency may vary from a few minutes to hours to days, and a significant fraction of the messages may not be delivered at all[3]. DTN use store-carry-forward mechanism. In this mechanism, node takes the custody of data until opportunity arrives. In DTN, the two or more nodes can exchange message when they move in transmission range of each other. This event is called "contact" in DTN. DTN is characterized by intermittent connectivity, long or variable delay, asymmetric data rate, and high error rates[4]. In Delay Tolerant Network, each node has a resource to store the package. Now node will check if a link between node is established or not. If no, then the node will store the packet & forward the same when the link is established[5]. The researchers investigated various routing protocols: Flooding-based approaches[4][5], encounter history based approaches [6][7], and social behavior-based approaches[8].

In this paper, we propose an adaptive routing with congestion control mechanism based scheme that includes message replication: replicate message to a relay node and message migration: migrate message to an alternative node with enough buffer space when the current buffer space is almost full. We use 2-hop activity level[9], to control message replication during a contact. The notion of the activity level is based on the observation that an active node has a higher chance of contacting more nodes later to improve its performance. We use 2-hop neighborhood information (or 2-hop information) to predict the node's activity level, which combines local information and encounter node's information. Our message replication scheme is an encounter history based multiple copy models that consider the 2-hop information so that destination can be reached quickly with a high delivery rate.

There are two steps to controlling the message replication: message selection (selecting the highest priority message to replicate) and node selection (selecting the best relay node to carry, store and forward the message). We use message TTL to prioritize the message in the buffer. The TTL threshold (Th) is used to control the message replication speed. In other words, if the TTL of the message is smaller than the threshold, a message will be replicated directly. Otherwise, we will use the node activity level to select the good relay node to replicate. However, the threshold is low the replication speed is slow. Hence, it has a higher latency. When the Threshold is high, it performs the significant replications, which may lead to buffer congestion when the buffer space is limited.

Although a lot of efforts has been invested in the design of buffer management policies for DTNs [9, 10], most of them deal with choosing the appropriate message to discard when the available buffer space is below threshold(Tb). I this paper, we propose a novel congestion control scheme, called message migration that support early migration to an alternative node when the available buffer space is below Tb. The message migration policy

978-1-4799-9991-0/15/$31.00 ©2015 IEEE

has two parts: message selection (selecting the lowest priority message to migrate) and node selection (selecting an alternative node to store the message by considering the available buffer space in the alternative node). By using the message migration scheme, the message can be migrated from overloaded node to alternative node early which can increase the delivery rate with a cost of increasing the number of forwarding.

II.RELATED WORK

Routing in DTNs has attracted the attention of the research community recently. The most simple replication based approach is flooding, also known as epidemic routing [12]. To control the copies of the message in epidemic routing, Spray and Wait protocol [13] and Spray and Focus have proposed. In the former one, the source node spray the message to the neighbour and waits for one of the neighbours meets the destination. The latter one goes further to allow multi-hop even when there is one copy, which is based on a quality metric from the node's encounter history. In this paper, our approach is also a form of replication based routing, which consider 2-hop information to improve the accuracy. Our scheme combines message selection and node selection to improve the performance.

Several solutions have been proposed to handle storage congestion control problem in DTNs. Most of them are based on message dropping policies [9, 10, 13]. In [14], Seligman et al. investigated storage routing to avoid storage congestion in DTNs. The basic idea is that once a node becomes congested, it forwards some fraction of its stored messages to alternative custodians in order to decongest itself; therefore, the node will be able to serve the incoming traffic. Our proposed message migration scheme has a similar spirit to that of [14], but we do not rely on global information. Our message migration policy is made using the local information only.

III.ADAPTIVE ROUTING WITH CONGESTION CONTROL MECHANISM

Our approach is an adaptive congestion control mechanism that is based on 2-hop information. We consider the limited buffer space scenario in DTNs. Our protocol is based on prioritizing the schedule of the message being transmitted to other nodes when there are multiple messages in the buffer based on message TTL. Four performance metrics are used: *(1)* Delivery Rate: the rates of message that reach the proposed destination within a given time limit; *(2)* number of forwardings: the average number of forwardings for each message in the entire routing process. This can be considered as the overhead for the routing process; *(3)* latency: the average duration between the generation time and arrival time of a packet; *(4)* number of lost messages: the number of messages that have been dropped.

We assume that there are N nodes in the whole network. M messages will be generated. Each of the messages has a message Id. Each of the nodes has a buffer in which it can store up to F Messages in transit: either message belonging to other nodes or generated by it. Each node has a priority ranking table for the messages stored in its buffer based on the message TTL (Time to live).

A. Message Replication

1) Replication Message Selection: in message selection stage, we use TTLThreshold (Th) as the priority metric to select the smallest TTL message m as the candidate replication message in a contact. If there are two message with the same TTL, select message randomly to replicate.

To reduce the message delivery time, we propose a hop-count threshold (Th) to control the message replication speed. If the TTL of the selected message is smaller than Th, we will replicate this message to relay node without node selection. Otherwise, if the TTL is larger than Th, we will go to the replication node selection stage to decide whether to replicate the message to the encountered node.

2) Replication Node Selection: the node selection is based on the 2-hop activity level. Next, we present the destination of the 2-hop activity level. The 2-hop activity level[9] of a node is the combination of its own encounter history and its neighbor's encounter histories before this contact. Node a's activity level (A_a) can be calculated as following Equ *(i)*

$$A_a = cE_a + (1-c)\sum_{k=1}^{i} w_k E_k, \quad \text{------- } (i)$$

Where, E_a is node a's total number of encounters with another node, W_k is neighbor *k's* contribution to node *a's* activity level, which is the ratio of *k* that appeared in node *a's* encounter list, *c* is a constant value to present the weight of the neighbor's contributions. For example, node *a* has encounter two other nodes *x* and *y*, 2 and 3 times respectively. By the definition E_a=5, and we assume that E_x=8 and E_y=6. If we set *c* to 0.8 and by using equation, we get the activity level of a: A_a = 0.8*5+0.2*(2/5 * 8 + 3/5 * 6) = 4.68.

When node *a* and node *b* are in each other's communication range, if the TTL of the selected message *m* (TTL) is smaller than *Th*, *a* replicate *m* to *b*. Otherwise, they will compare their activity levels. If node *b's* activity level A_b is larger than node *a's* activity level A_a, a replicate message m to b. Otherwise, a replicate message *m* to node *b* with probability A_b/A_a. The message replication process is shown in Algorithm *1* when node *a* and node *b* are in contact.

Calculation of Dynamic TTL Threshold (Th):

Here, we can consider the average of TTL value as a Threshold (*Th*). For example, node *a* has four messages and whose TTL are 225, 275, 256, 154 respectively. Then Threshold = (225+275+256+154)/4 = 227.5. Threshold value will be varying from node to node.

Algorithm 1 Message Replication (at Node a)

1. Node a and b first exchange the metadata

2. Node a select smallest *TTL* message (*m*) to replicate

3. **If** $TTL < Th$ **then**

Node *a* replicate *m* to node *b*.

4. **else if** $A_b > A_a$ **then**

node a replicate *m* to node *b*.

5. **else** node a replicate m to b with probability *Ab/Aa.*

B. Message Selection

Although there are many works that have been designed message dropping policies, we believe that using message dropping policies will reduce the delivery rate in the following situation: if the buffer size is relatively small, the buffer of an alternative node will be overloaded quickly. Then, the message in the buffer may all have a relatively small TTL. Hence, dropping any message in the buffer will decrease the delivery rate.

Therefore, we propose a message migration scheme to control the buffer congestion [9]. When the available buffer space is below a threshold ($S_a < Th$), the node will migrate to an alternative node that has enough buffer space early on, even though it may not be a high priority node. In the message migration process, the sending node will forward smallest *TTL* message to the alternative node and will not retain a copy in its buffer.

1) Migration Message selection: we select smallest TTL message to migrate. This is because, before discard it may reach the destination.

2) Migration node selection: in order to make sure that the message will not be discarded quickly after the migration, the selected migration message will be migrated to an alternative node with enough buffer space ($S_b - 1 \geq T_b$.)

When the available buffer space of node a (S_a) is below the buffer space threshold Tb in the message dropping schemes, m will be discarded to release the buffer space for new messages. However, it will reduce the delivery probability to the destination and will increase the delivery time because of the message loss. By using the message migration scheme, node a migrate message m to node b with enough buffer space (S_b-1 $\geq$ T_b) before the buffer becomes full. B will later replicate or migrate m to other encountered nodes. At the same time, a has enough buffer space to accept the new message. The message migration process is shown in Algorithm 2, when nodes a and b are in contact. When either node *a* or *b* is overloaded, we do message migration first.

Calculation of Dynamic Threshold (Tb):

Here, up to the warm up time we consider the threshold value as a half of the free buffer size of the current node. After warm up time, we take threshold as a mean value of free buffer space of all connected nodes. For example, assume warm time is 1000 and current simulation time is 300, and free buffer space of node a is 5MB than the Tb = 2.5 MB. If the current simulation time is 1250 and the available free space of node a is 5 MB and node b's free buffer space is 3 MB then the Tb = (5+3)/2 = 4 MB.

Algorithm 2 Message Migration (at Node a)

1. Node a and b first exchange the metadata
2. **If** $S_a < T_b$ and S_b -1 $\geq T_b$ **then**

   Node a select smallest *TTL* message (*m*) to migrate to node *b*.

IV. Implementation and Simulation

We implement our routing protocol in Opportunistic Network Environment simulator and evaluate performance with synthetic and real trace, Objective for simulation: to measure delivery ratio(delivery probability) overhead ratio,number of message drops and average buffer time of our proposed protocols and compare existing routing protocol. Delivery Ratio is define as ratio of total numbers of created message by the number of successfully delivered messages and Overhead Ratio is define by Equ *(ii)*

$$\text{Overhead Ratio} = \frac{\text{Relayed Messages} - \text{Delivered Messages}}{\text{Delivered Messages}} \quad \ldots..(ii)$$

Dataset Information:

The dataset being downloaded from Community Resource for Archiving Wireless Data at Dartmout[15] (CRAWDAD) is a wireless network data resource for the research community site. Synthetic data is generated on ONE simulator and make compatible with existing real dataset.

Simulation Settings of all dataset:

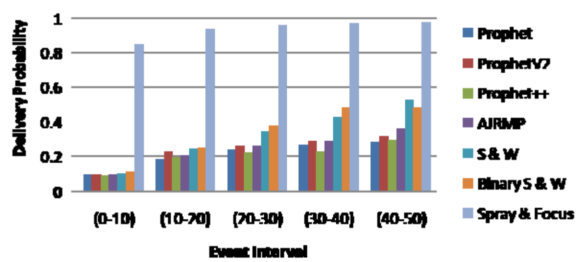
Parameter	***Value***
Group.msgTtl	300
Group.movementModel	StationaryMovement (*for external movement*)
Events.nrof	2
Events1.class	MessageEventGenerator
Events1.interval	0-10, 10-20, 20-30,30-40,40-50
Events2.class	ExternalEventsQueue
Events2.filePath	Path of real and synthetic dataset file
Group.bufferSize	5Mb

In the simulation, T_h and T_b are calculated dynamically and the buffer size of the node is 5MB and the value of c initialised as 0.8 in this paper. Message size is from 500kb to 1 MB. For the best way to evaluate performance regarding congestion control we varying buffer size and compare AJRMP protocol and JRMBRP.

A. Performance evolution with Synthetic trace:

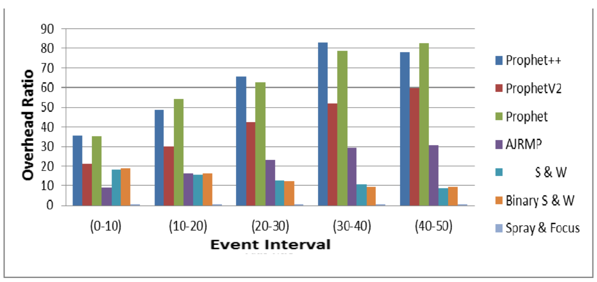
In this phase, we compare the delivery probability and overhead ratio of our protocol with other routing protocols. We compare it with variants of spray and wait protocol and prophet protocol. It was observed that AJRMP perform better than Prophet Versions, because it uses most active node to forward the message.

(i) Comparison of Delivery probability in Synthetic Trace:



AJRMP protocol has higher delivery ratio than Prophet versions and lower than the Spray and Wait version because in spray and wait it spread limited number of copies in the network.

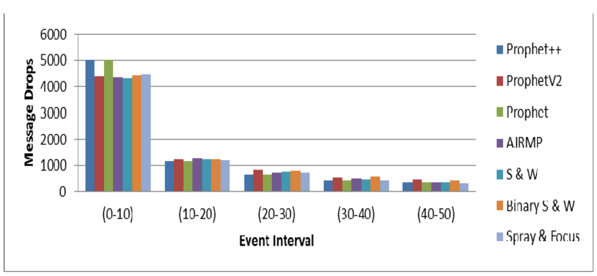
(ii) Comparison of Overhead ratio in Synthetic Trace:



AJRMP protocol replicate and migrate more number of messages, due to sort contact time overhead ratio is higher than spray and wait versions.

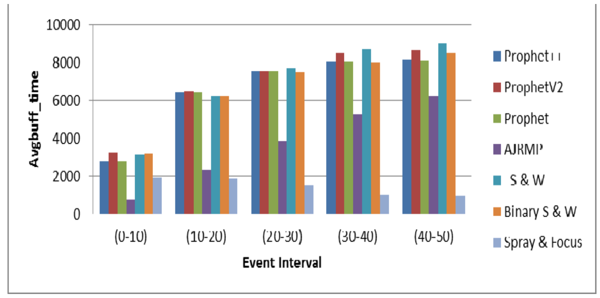
(iii) Comparison of Message drops in synthetic trace:

Compare to the Prophet versions in AJRMP less number of message are dropped because it performs migration when the buffer space is below some threshold value.



(iv) Comparison of Average buffer time in synthetic trace:

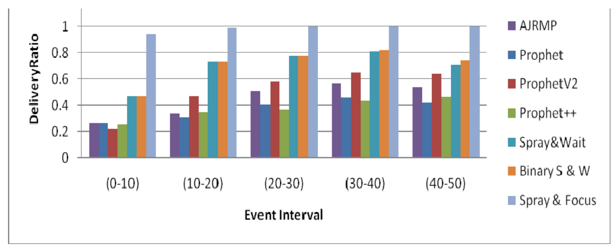
Average buffer time of AJRMP is small compare to other because it performs early replication and migration to the alternative node.



B. Performance evolution with Real traces:

In this phase, we evaluate our protocol in three different real traces. In Dieselnet[15] the simulation time is 5207863 seconds and 8 nodes are there. In rollernet simulation time is 3096 seconds and numbers of nodes are 63. In Infocom the numbers of nodes are 63 and the simulation time is 342915 second. In this phase, we compare our protocol with spray and wait variants and variants of prophet protocol. Reading are taken by varying buffer size and it perform better as a buffer size increased the delivery probability is increased because of enough buffer space less number of messages are dropped from the buffer.

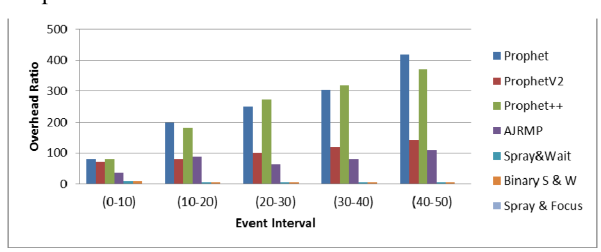
(i) Comparison of Delivery ratio in Rollernet Trace:



Prophet forward the message to other node based on the probability value, AJRMP replicate the message based on Threshold value, also it select the most active node to replicate message and forward smallest TTL message to replicate so before message will be discarded probability to reach the destination is higher than prophet.

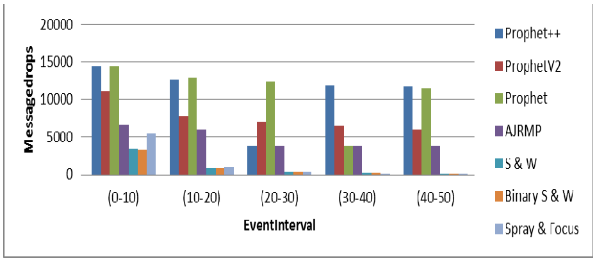
(ii) Comparison of Overhead ratio in Rollernet Trace:

AJRMP has a low overhead ratio because it forward message based to most active node so higher chance to meet the destination, so overhead ratio is lower than Prophet

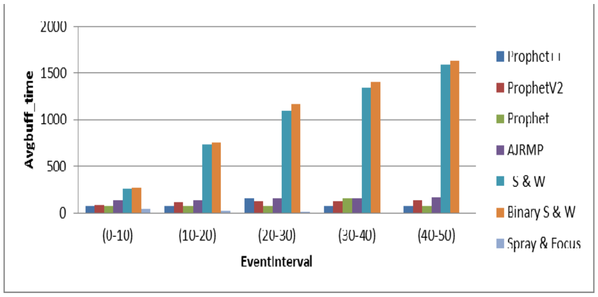


(iii) Comparison of Message drops in Rollernet trace:

AJRMP replicate and migrate more number of messages, more messages are dropped in rollernet compare to other routing protocols.

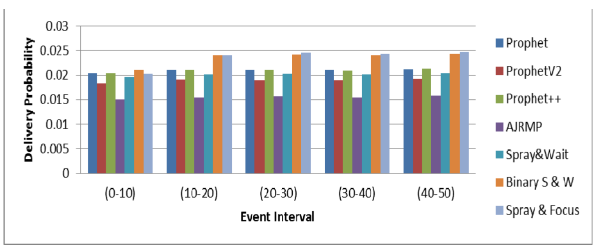


(iv) Comparison of Average buffer time in Rollernet trace:



AJRMP protocol performs early replication and migration and it forward sort remaining life time message so Average buffer time is less.

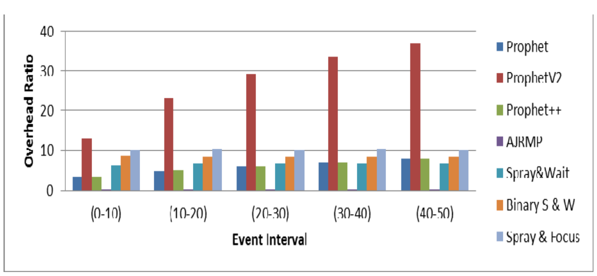
(v) Comparison of Delivery probability in Dieselnet Trace:



Prophet forward the message to other node based on the probability value, AJRMP replicate the message based on Threshold value but in DieselNet number of messages are less.

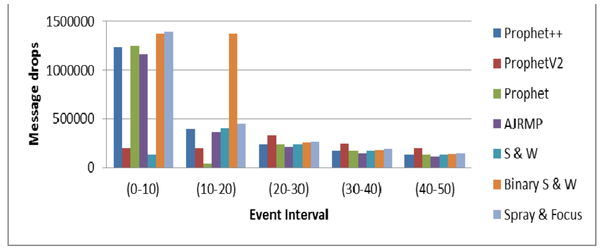
(vi) Comparison of Overhead ratio in Dieselnet Trace:

The overhead ratio is lower in both the cases because AJRMP migrate message to the alternative node and less number of message drops are there.

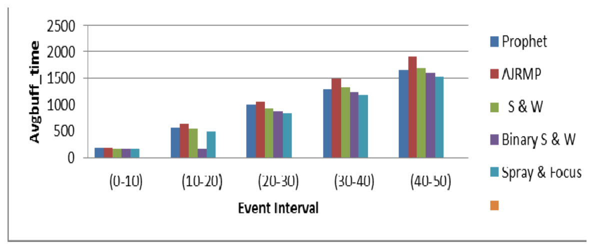


(vii) Comparison of Message drops in Dieselnet trace:

In small Message Event interval, more number of messages is dropped because more number of messages is generated. Instead of dropping AJRMP migrate message to an alternative node so compared to Prophet Versions less number of messages are dropped in AJRMP.

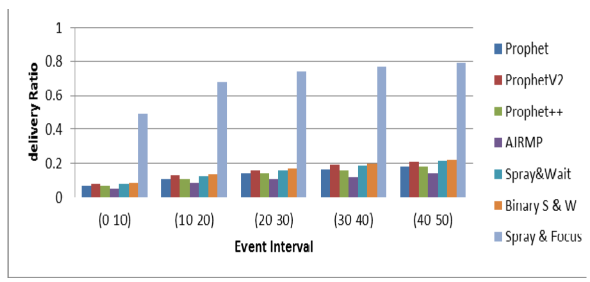


(viii) Comparison of Average buffer time in Dieselnet trace



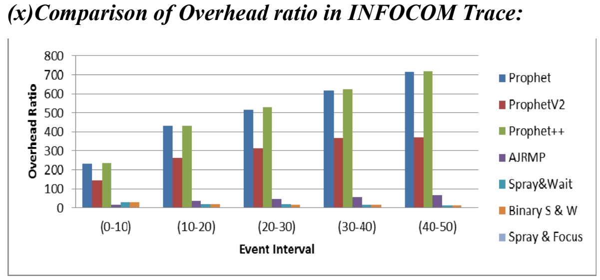
AJRMP only forward a message to the most active node and it takes the time to calculate threshold and activity level.

(ix)Comparison of Delivery probability in INFOCOM Trace:



Delivery Ratio of AJRMP is observed low compare to Prophet Versions and Spray and Wait versions AJRMP calculate and update Threshold value per time interval and perform node selection and message selection each time so in this protocol it found quite low.

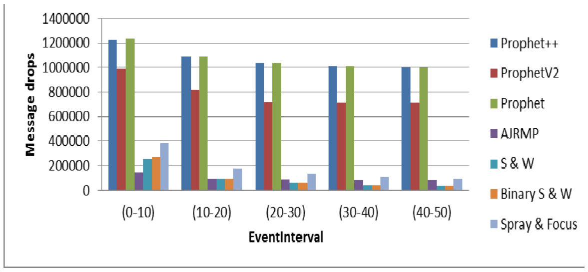
(x)Comparison of Overhead ratio in INFOCOM Trace:



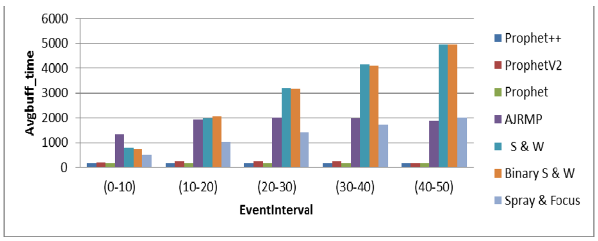
Prophet flooding more messages in network compare to AJRMP so it impacts on the overhead ratio.

(xi) Comparison of Message drops in INFOCOM Trace:

In Prophet because of FIFO drop policy more messages are dropped whereas, AJRMP perform migration so less number of messages are dropped.



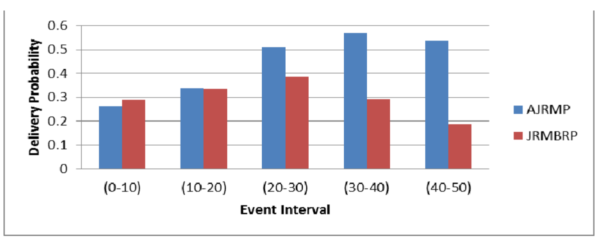
(xii) Comparison of Average buffer time in INFOCOM Trace:



In Spray and Wait, it waits for a destination so messages stay for long time in the buffer. In AJRMP, it migrates and replicate message to the alternative node.

C. Performance evolution of AJRMP and JRMBRP with different Event Internal

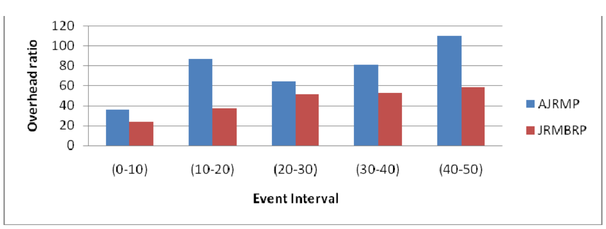
(i) Delivery probability in Rollernet Trace:



AJRMP deliver more messages because it replicates and migrate messages based on dynamic threshold value where as in JRMBRP threshold is static.

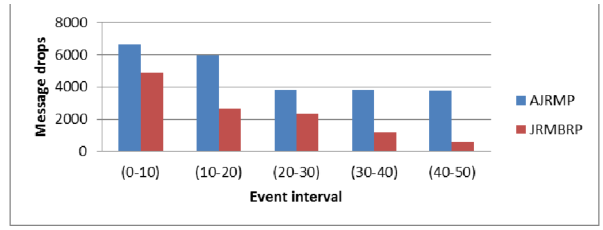
(ii) Overhead Ratio in Rollernet Trace:

The overhead ratio of AJRMP is also higher than existing protocol because it takes much more time to calculate dynamic threshold.

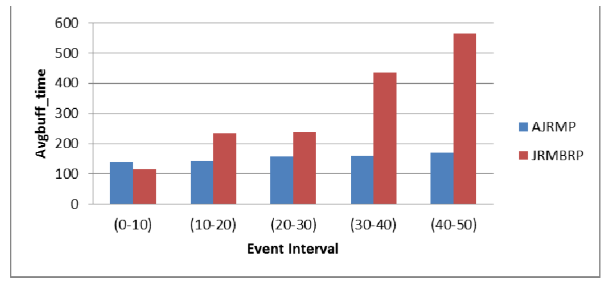


(iii) Message Drops in Rollernet Trace:

In AJRMP protocol threshold value is higher so another node has not enough free buffer space greater than the threshold so at last it drop the message.



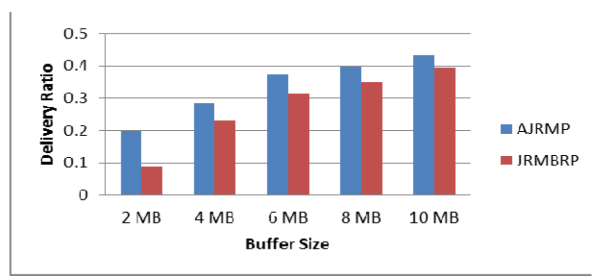
(iv) Average Buffer Time in Rollernet Trace:



AJRMP replicate those messages whose have TTL is less than TTL Threshold so messages are not stay a long time in the buffer.

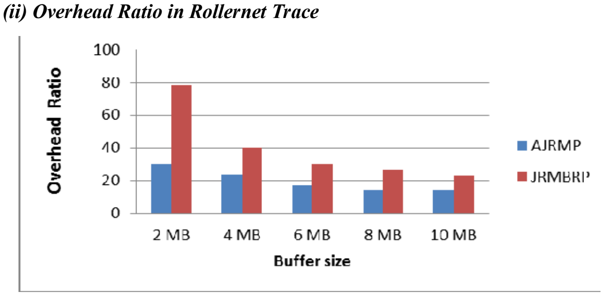
D. Performance evolution of AJRMP and JRMBRP with different buffer size

(i) Delivery probability in Rollernet Trace:



As the buffer size increased the delivery probability is increased in both the protocol because they have enough space to store the messages and migrate the messages so the delivery ratio is higher.

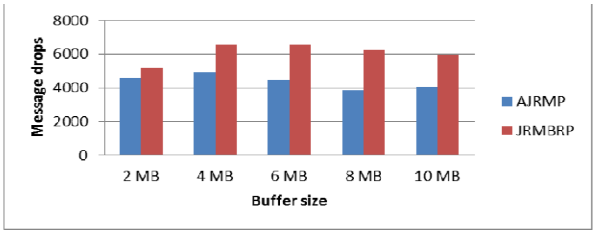
(ii) Overhead Ratio in Rollernet Trace



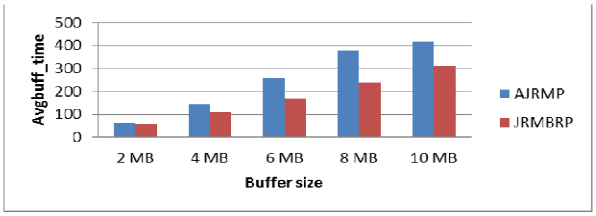
As the buffer size increased the delivery probability is increased so the overhead ratio is decreased in both the protocol because more number of messages are delivered.

(iii) Message Drops in Rollernet Trace:

AJRMP drop less number of messages compare to JRMBRP because it calculate a dynamic threshold and as the buffer size increased the threshold value is increased so less number of messages are dropped.



(iv) Average Buffer Time in Rollernet Trace:



As the buffer size increased the messages are waiting for longer period of time because of enough buffer space no need to migrate message so the Average buffer time increased.

Overall Observations

The AJRMP protocol is also compared with static threshold protocol Joint Replication Migration based Protocol named as JRMBRP. AJRMP is also comparing with other existing protocols and evaluate the delivery probability, overhead ratio, message drops and Avgbuff_time with varying message event interval. By varying the buffer size, the performance is increased with respect to buffer size and the delivery probability is higher.

V. CONCLUSION

AJRMP calculate dynamic threshold using TTL value of available free buffer space of nodes and it's used for replication and migration. AJRMP prioritize messages based on TTL and it will forward the smallest TTL message so a chance to reach the destination is higher before it will discard. Also, select the most active node based on encounter history and replicate messages of node. When the buffer space of any node becomes full it migrate message to an alternative node based on available buffer space, so less number of messages are dropped so it performs well compared to Prophet router because in Prophet messages are forwarded based on probability value. The delivery ratio of AJRMP is higher than Prophet and lower than SprayAndWait because it spread a limited number of message copies in the network. The overhead ratio is also lower than Prophet Protocol. As the buffer size increased the performance is better in AJRMP protocol compared to the JRMBRP because of enough buffer space so there is a less chance for a message drop.

REFERENCES

[1] G. Karlsson, V. Lenders, and M. May, "Delay-Tolerant Broadcasting," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pp. 369–381, Mar. 2007.

[2] F. Warthman, "Delay Tolerant Networks (DTN)," *prius.ist.osaka-u.ac.jp*, no. March, 2003.

[3] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 145–158, Oct. 2004.

[4] C. B. Desai, V. N. Pandya, and P. M. Dolia, "Comparative Analysis of Different Routing Protocols in Delay Tolerant Networks," vol. 4, no. 03, pp. 249–253, 2013.

[5] A. Mehto, "Comparing Delay Tolerant Network Routing Protocols for Optimizing L-Copies in Spray and Wait Routing for Minimum Delay," vol. 2013, no. Cac2s, pp. 239–244, 2013.

[6] C. S. R. T. Spyropoulos, K. Psounis, "Efficient routing in intermittently connected mobile networks: The single-copy case.," *IEEE/ACM Trans. Netw.*, vol. Vol. 16, 2008.

[7] C. S. R. T. Spyropoulos, K. Psounis, "Efficient routing in intermittently connected mobile networks:: The multiple-copy case..," *IEEE/ACM Trans. on Networking.*, vol. Vol. 16, 2008.

[8] I. chung ByJian shen, sangaman moh, "Routing Protocols in Delay Tolerant Networks," *23rd Int. Tech. Conf. Circuits/Systems*.

[9] Y. Wang, J. Wu, Z. Jiang, and F. Li, "A joint replication-migration-based routing in delay tolerant networks," in *Communications (ICC), 2012 IEEE International Conference on*, 2012, pp. 73–77.

[10] D. T. X. Zhang, G. Neglia, J. Kurose, "Performance Modeling of epidemic routing," *Comput. netw*, vol. 51, pp. 28.

[11] C. B. and T. S. A. Krifa, "Optimal Buffer management policies for delay tolerant networks," *IEEE SECON*, vol. pp. 260–26, 2008.

[12] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected Ad Hoc Networks." 2000.

[13] J. Makhlouta, H. Harkous, F. Hutayt, and H. Artail, "Adaptive Fuzzy Spray and Wait: Efficient routing for Opportunistic Networks," in *2011 International Conference on Selected Topics in Mobile and Wireless Networking (iCOST)*, 2011, pp. 64–69.

[14] K. F. and P. mundur M. Seligman, "Alternative custodiansfor congestion control in delay tolerant networks," *ACM CHANTS*, vol. pp. 229–23, 2006.

[15] "CRAWDAD: A community resource for archiving Wirelessdata at dartmouth." .

2015 5th Nirma University International Conference on Engineering (NUiCONE)

TCP with Sender Assisted Delayed Acknowledgement – A Novel ACK Thinning Scheme

Hardik K. Molia
Department of Computer Engineering
Government Engineering College, Rajkot
Gujarat, India
hardik.molia@gmail.com

Abstract— **TCP – Transmission Control Protocol acts as a logical vehicle for process to process communication. TCP ensures reliability in the form of flow control, error control and congestion control via ACK-Acknowledgement based transmissions. Wireless networks have limited spatial reuse scope which limits amount of data that can be in transmission within a specific geographical area at a time. In such scenarios collision is one of the most critical link layer issues which affects overall performance. Inter flow collision occurs among more than one independent TCP connections while Intra flow collision occurs inside a single TCP connection between data flow and ACK flow. ACK Thinning is a process of reducing the rate of ACKs to spare more communication space - bandwidth for the Data flow inside a single TCP connection. This paper introduces an enhanced ACK Thinning scheme SADA - Sender Assisted Delayed Acknowledgement for MANETs. The scheme has been implemented in NS 2.35.**

Keywords— TCP, MANET, ACK Thinning, Sender Assistance, Delayed Acknowledgement

I. INTRODUCTION

TCP – Transmission Control Protocol provides process to process and reliable communication. Packets may get lost during transmission. Standard TCP considers any loss as a cause of network congestion only. On detection of a loss, TCP slows down the transmission rate. This basic interpretation of any loss as a cause of network congestion suits well with the wired networks. In wireless networks, especially in MANETs – Mobile Adhoc Networks, several other issues like route failures, channel losses etc. cause packets lost. TCP's default interpretation of any loss as a cause of congestion degrades performance when the actual cause is something else. Ideally, on congestion loss, transmission rate should be reduced. On route failure loss, further transmission should be suspended until a new route is found. On channel loss, the lost or corrupted packet should be retransmitted immediately without reducing the transmission rate. Unfortunately TCP needs to add some predictability for loss differentiation and to act accordingly [1][2].

Various TCP variants have been proposed which are broadly classified into: Cross – Layer Approaches and Layered Approaches. TCP is a Transport Layer protocol. In Cross Layer approaches, TCP receives feedback – decision making information from one of the lower layers and acts accordingly. Like Network layer informs TCP about route failures or Link Layer informs TCP about contention issues. In Layered approaches, TCP itself predicates the cause of loss without any kind of feedback from the lower layers. In Layered approaches, from the Transport Layer itself, TCP tries to detect and resolve issues related with one of the lower layers[1][2].

TCP provides reliability in the form of send-verify concept. Every data packet sent by sender must be acknowledged by the receiver within a specific time. Data flows from sender to the receiver while ACK flows from receiver to the sender. If Data is not acknowledged after waiting for some time, sender considers it as lost and retransmits. ACK is never retransmitted. Duplicate ACKs are not retransmissions. Wireless networks have limited bandwidth. Shared wireless medium and broadcasting nature often restrict number of simultaneous transmissions. If Link Layer doesn't coordinate transmissions properly then spurious packet loss occurs. Inter flow collision occurs across more than one independent TCP connections while Intra flow collision occurs inside a single TCP connection between data flow and ACK flow. ACK Thinning is a process of reducing the rate at which ACKs are generated to spare more bandwidth for the Data flow inside a single TCP connection. Piggybacking is an optimization technique in which acknowledgement information is sent inside the TCP header carrying Data so separate ACK is not required to send. This technique is suitable for two way communications (peer to peer based) but not for one way communications (client server based). Here some of the ACK Thinning schemes are explained which are suitable for one way communications. A novel Fuzzy Delayed Acknowledgement scheme is proposed which is implemented in NS 2.35 and tested over scenario based MANETs [3].

II. ACK THINNING SCHEMES

A. TCP DA: -

DA stands for Delayed Acknowledgement. DA represents default cumulative acknowledgement concept of standard TCP. In this scheme, the delay parameter is fixed to 2 packets.

TCP receiver starts an ADT - Acknowledgement Delay Timer (100- 500 ms) on reception of an expected in-order packet. All the previously received packets must have been acknowledged prior to starting Acknowledgement Delay Timer. Instead of sending an ACK immediately, TCP waits for arrival of an in-order packet or until ADT times out. On arrival of an expected in-order packet, if ADT is already running then TCP sends a cumulative ACK immediately otherwise it starts ADT Timer and delays ACK until another expected in-order packet arrives. At any particular moment, TCP Receiver will not have more than two unacknowledged in-order received packets.[4]

B. TCP ADA: -

ADA stands for Adaptive Delayed Acknowledgement This scheme reduces number of ACKs to only one per congestion window. API – Average Packet Interval is average of inter arrival time durations of subsequent packets. API value is used to calculate AET - ACK Expiration Time. ADA postpones an ACK until either MDT Timer – Maximum Defer Time Timer times out with the condition is that packets are continuously arriving within (β * API) time. TCP immediately sends a cumulative ACK on expiration of either MDT Timer or AET timer. The most recently received packet's arrival time is recorded as LPA – Last Packet Arrival Time. ADA scheme is explained in algorithm below [5].

```
LPA = 0
API  = 0
α = 0.8
β = 1.2
MDT = 500 ms

Packet_Received( )
    If (LPA = 0) Then
        LPA = Now
    Else
        API =  α * API + [(1 − α) * (Now − LPA)]
        LPA = Now
    End If

    If (MDT Timer is not Scheduled) Then
        Reschedule MDT Timer to MDT Time.
    End If

    Reschedule AET Timer to β * API.
End

AET_Timer_Timesout( )
    Send a cumulative ACK immediately.
    Stop MDT Timer.
End

MDT_Timer_Timesout( )
    Send a cumulative ACK immediately.
    Stop AET Timer.
End
```

C. TCP DDA: -

DDA stands for Dynamic Delayed ACK. Any scheme of Delayed ACK improves performance if there are enough number of flight packets (packets in transmission) to allow cumulative concept to work. If congestion window is of 2 packets and TCP receiver delays ACK for up to 3 packets then ACK will be delayed till the expiration on ACK timeout because sender will send 3rd packet only once previous 2 are acknowledged. This issue is tried to resolve with DDA scheme. A delay coefficient d defines number of in-order received packets for which ACK should be delayed. Initially d is set to 2 to implement standard cumulative scheme. DDA increments value of d linearly as more and more packets are received in-order based on three threshold levels l1, l2 and l3. Algorithm is shown below [6].

```
d  =  1
l1 = 2, l2 = 5, l3 = 9

N  =  Initial_Sequence_Number(0)
If (N < l1) Then
        d  =  1
        Else If  (l1 ≤ N < l2) Then
                d  =  2
                Else If (l2 ≤ N < l3) Then
                        d  =  3
                        Else If (l3 ≤ N) Then
                                d  =  4
End If
```

D. TCP DAA:-

DAA stands for Dynamic Adaptive ACK. If receiver delays an ACK, sender should also delay retransmissions. Otherwise there will be unnecessary retransmissions. DAA focuses on avoiding such spurious retransmissions by adjusting delay as per the channel condition. DAA is a sender and receiver side algorithm. Sender side, number of duplicate ACKs to trigger fast retransmission is changed from 3 to 2. At the same time, RTO is increased five times [7].

Receiver has a dwin – Dynamic Delaying Window whose size is from 2 to 4 packets with default value of 2 packets. Pending_acks is number of in-order received unacknowledged packets so far. When Pending_acks becomes equal to current dwin, receiver sends a cumulative ACK immediately. With every arrival of an expected in-order packet, dwnd is incremented by 1 up to its maximum value of 4. ACK Timeout value is determined by statistical analysis of arrival times among the packets [7]. The algorithm is described below.

```
Pending_acks = 0,
dwin = 2

If (an expected in-order packet is received) Then

    If (Pending_acks > 0) Then
        Record arrival time to calculate ACK Timeout
    End If

    Pending_acks = Pending_acks + 1

    If (Pending_acks == dwin) Then
        Send a cumulative ACK immediately.
        Pending_acks = 0
    End If

    If (dwin < 4) Then
        dwin = dwin + 1
    End If

Else
    Send a cumulative ACK immediately.
    Pending_acks = 0
    dwin = 2
End If
```

III. SENDER ASSISTED DELAYED ACKNOWLEDGEMENT

This paper introduces a novel ACK Thinning scheme SADA – Sender Assisted Delayed Acknowledgement. Researchers have proposed many variants of TCP protocol targeting various types of networks and various issues of communication. Every TCP variant's behavior depends upon pattern of incoming ACKs. SADA scheme can be incorporated with any of the ACK clocking TCP variant. SADA scheme has sender and receiver side algorithms. This paper implements SADA scheme with TCP NewReno.

A. Sender Side Algorithm: -

TCP Sender assists TCP receiver to delay ACKs by sending delay parameter information. Delay Parameter is the number of individual ACKs (one individual ACK corresponding to one in order received packet) to delay before the cumulative ACK is required to be sent. Sender sends Delay Parameter information with the TCP header. TCP header has four reserved bits which can be used to define delay information from 0 to 15 packets. Sender also needs to increase value of RTO – Retransmission Time Out Timer value as delay parameter increases. On decrease of delay parameter, sender decreases RTO Timer Value.

DP - Delay Parameter is in the terms of the number of in-order received packets which are not yet acknowledged. These packets will be cumulatively acknowledged. DP value ranges from 2 to the maximum sender window size or 15 whichever is smaller. Initial Size of DP is set to the default delayed acknowledgement scheme size which is of 2 packets. TCP recalculates RTT – Round Trip Time with every successful reception of an expected ACK. This procedure is called on every fresh RTT calculation. This procedure calculates Percentage Change from old RTT value to new RTT value. Four Reserved Bits of TCP Header:- R_3 R_2 R_1 R_0 are used to represent Delay Parameter in binary. Threshold values are set $\alpha = -35$ and $\beta = 35$. The algorithm is shown below.

```
On_RTT_Calculation( )
Begin

   Percentage_Change = [(RTT_New / RTT_Old) * 100] – 100

   If DP < Sender_Window_Size Then

      If Percentage_Change ≤ α Then
         DP++
         RTO_val = RTO_val * (RTT_New / RTT_Old)
      End If

      If Percentage_Change ≥ β Then
         DP--
         RTO_val = RTO_val / (RTT_New / RTT_Old)
      End If

   End If

   R3 R2 R1 R0 = toBinary (DP)

   Set R3 R2 R1 R0 Reserved Bits of TCP Header for next
packet going to be send.

End
```

B. Receiver Side Algorithm: -

TCP receiver receives a packet; it analyzes the TCP header and sets DP from Bits R_3 R_2 R_1 R_0. TCP receiver also changes AET – ACK Expiration Timer value with reference of the change in DP Value.

```
On_Packet_Arrival( )
  Begin

    DP = toDecimal(R3 R2 R1 R0)
    AET = AET * DP

End
```

C. RTT and RTO Calculation: -

RTT – Round Trip Time is the total time the network takes to transmit some data from a source plus time it takes to deliver acknowledgement of it generated by the destination. Ping utility measures the RTT between a sent ping packet and getting ICMP packet back. TCP finds the base RTT during the connection establishment which is the Time interval between a SYN message sent by a Source – Active Open and a SYN+ACK message received from the corresponding destination – Passive Open. RTO- Retransmission Time Out is the time for which source has to wait before it retransmits a packet. RTO is calculated from RTT. As network gets changed, a same RTT cannot be used for a long time. Many mathematical algorithms have been proposed to measure subsequent RTT and RTO efficiently. Karn algorithm's simplified scheme is shown here. This scheme calculates SRTT – Smoothed Round Trip Time. SRTT calculation is based on deriving new RTT from all the past RTTs of a specific connection [3]. (i+1)th SRTT can be calculated from (i)th SRTT and current (i+1)th RTT as,

$$SRTT(i+1) = \alpha * SRTT(i) + (1-\alpha) * RTT(i+1)$$

where α is smoothing factor between 0.8 and 0.9

$$RTO(i+1) = Min [UB , Max [LB , (\beta * SRTT(i+1))]]$$

Where UB is the upper bound of the timeout, LB is the lower bound of the timeout and β is the delay variance factor.

IV. Simulation

Simulation is performed in NS 2.35 with TCP NewReno variant. Scenario based MANETs are used to test SADA scheme. Scenario differs by number of nodes, mobility speed, mobility redefine interval.

TABLE I
MANET Scenarios

	Scenario_1	Scenario_2
Max. Sender Window	15 Packets	15 Packets
Number of Nodes	30	10
Simulation Time	60 Seconds	50 Seconds
Mobility Interval	25 Seconds	35 Seconds

Instantaneous Throughput results are shown in figure 1.

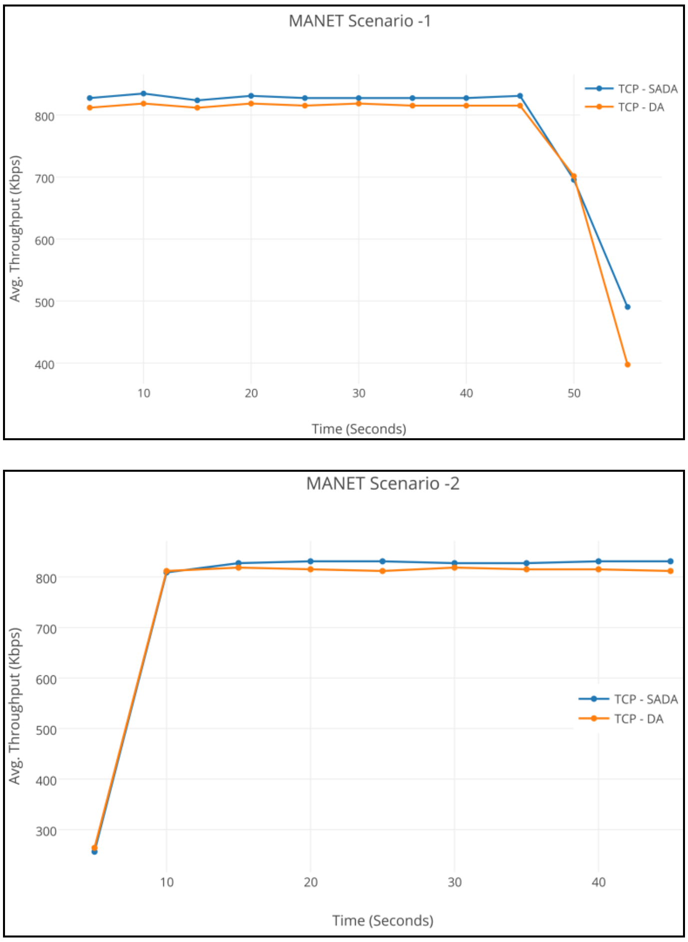


Figure 1. TCP SADA and TCP NewReno Instantaneous Throughput

V. Conclusion

TCP's performance is one of the most important factors in MANETs. Several schemes have been proposed to enhance loss differentiation capability of TCP. SADA is a novel approach to analyze network stability by finding percentage change between subsequent RTTs and to act accordingly. SADA defines dynamic cumulative acknowledgement concept which is useful when MANET is vulnerable to multiple kind of losses. As SADA scheme is based on processing subsequent RTT values, it can be incorporated with any of the reactive TCP variants. SADA is a light weight variant as it requires very less computational resources.

References

[1] Subir Kumar Sarkar, T G Basavaraju, C Puttamadappa , "Ad Hoc Mobile Wireless Networks - Principles, Protocols, and Applications" - Auerbach Publications Taylor & Francis Group

[2] Priyanka Goyal, Vinti Parmar and Rahul Rishi, "MANET: Vulnerabilities, Challenges, Attacks, Application", IJCEM International Journal of Computational Engineering & Management, Vol. 11, January 2011.

[3] G Holland, N Vaidya, “Analysis of TCP performance over mobile ad hoc networks”, in Proceedings of the 5th Annual ACM/IEEE international Conference on Mobile Computing and Networking (Seattle, Washington, United States, August 15-19, 1999) pp. 219–230.

[4] Behrouz Forouzan, "TCP/IP Protocol Suite", 4/e, McGraw-Hill

[5] AK Singh, K Kankipati, "TCP-ADA: TCP with adaptive delayed acknowledgement for mobile ad hoc networks", in Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE. 3, 1685–1690 (2004)

[6] E Altman, T Jiménez, "Novel delayed ACK techniques for improving TCP performance in multihop wireless networks", in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2775, 237–250 (2003)

[7] de Oliveira, T Braun, "A dynamic adaptive acknowledgment strategy for TCP over multihop wireless networks", in INFOCOM, 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE. 3, 1863–1874 (2005)