Optimizing Message Delivery in Mobile-Opportunistic Networks

Muhammad Arshad Islam* Marcel Waldvogel*
*Distributed Systems Laboratory
University of Konstanz – Germany
firstname.lastname@uni-konstanz.de

***Abstract*—The big challenge of routing in opportunistic mobile networks, overlooked by most researchers, is to not only find any path to the destination, but a path that is stable and powerful enough to actually carry the message. Few attempts addressed this problem, all of them under controlled scenarios, avoiding the complexity of real-world connectivity. As a result of our comparison of selected networks under a wide variety of realistic scenarios, we have not only been able to identify and describe favorable traits of protocols, but also necessary relationship of successful MON protocols with QoS routing in wired networks. We present a novel protocol, Nile[1], that performs both in dense as well as sparse networks. Nile is the first autonomous "controlled flooding" protocol that keeps the link loads in check, to push replicas only on those paths that are both promising and may sustain more load. It is a multi path protocol that deploys replication based on heuristic for disjoint path calculation. Other protocols' performance, when simulated in real-world traces, highly depends on parameter choice. Nile, however, consistently performs among the top protocols without any external tuning and exerts far less overhead than other replication protocols.**

I. Introduction

Although the problem of routing in opportunistic networks is getting a lot of attention, we have not still seen a robust and reliable solution. The reason being that challenges involved in opportunistic network routing are totally different than the traditional wired networks. We not only can design and plan the structure of wired networks but also have real time information about the route changes in the network in the case part of a network fails. On the other hand, opportunistic networks (as the name suggests) cannot be designed or planned. They are implicitly created and evolved due to wireless devices that come in to each others radio range. These wireless devices then behave as data mules as well as routers. They make routing decisions to bring the messages to their respective destinations based on the local knowledge that they have obtained earlier from the network. In our view, the routing problem has two distinct steps. (i)to ensure connectivity to the destination(ii) to ensure that the given paths are reliable enough to deliver the message. The current trends in the field show that most of the protocols are tested on networks that are dense, predictable or aided by communication infrastructure[11], [2]. On the other hand, simulations are performed mostly on artificially generated networks where movement of the nodes is defined by a prescribed velocity in a random direction in a designated area. Our observation on the contrary, is that the opportunistic nature of these networks brings so much unpredictability to path structure that the shortest path existing at one point in time may either not turns out to be the shortest one or in the worse case, this path may even fail to exist in future. To ensure the connectivity to the destination, it is, therefore necessary to propagate multiple replicas of a message along several paths simultaneously. The replicas, on one hand increase the probability of the message delivery to the destination but also create congestion in the network, thereby reducing the delivery probability of other messages being propagated at the same time through common paths. This in turn, raises the reliability question, as given a path does exist to the desired destination but does not posses the capacity to deliver the amount of data in question. The lack of capacity may either be original characteristic of the path or arise from the current traffic trends. This paper presents a routing strategy that creates replicas in such a way that it not only increases the delivery probability but also keeps a check on congestion.

The network community has already faced the reliability challenge with regard to network traffic while ensuring real-time multimedia streaming with the help of QoS. Traditionally, QoS is employed to provide better service to selected network traffic over various technologies. It is here imperative to mention that ensuring QoS in wired networks is relatively easy as traffic capacity and demand can be estimated to an accurate level because the capabilities and specification of routers are known and the traffic generated by the applications can be estimated accurately. We argue that we must couple QoS routing practices similar to those of wired networks, with the routing problem in opportunistic networks because storage and transmission capabilities of wireless devices in opportunistic networks can severely degrade if QoS aspects are fully ignored. In opportunistic networks, not only individual devices have highly variable capacities but also the traffic is less predictable. It is well known that wireless communication highly suffers from traffic congestion and in case of a bottle-neck, path recalculation can be a resource expensive process. As already stated, traffic congestion in opportunistic networks, not only create problems for those messages that are directly involved but it also reduces the delivery probability of those messages that are sharing the same path. It is therefore, even more necessary for opportunistic networks to have as accurate

[1]Motivation comes from the notion of controlling floods from the river Nile by constructing dams and barrages across the river banks.

as possible traffic metrics so that congestion can be avoided. The delay encountered by the messages in opportunistic networks ranges from a few minutes to several days therefore obtaining traffic measures for participating devices is a huge challenge. It is easily understandable that the more accurate paths and traffic measures will lead to better message delivery.

Another issue with the path metric currently in use is the absence of *inter-hop duration*. There are several examples(discussed briefly in next section) where past contact frequency is used to estimate the future contact outcome. We present a novel protocol that we like to call Nile, where we construct paths not only with the help of contact frequency but also augment them with *duration* to reach destination as well as *capacity* i.e. traffic volume that is deliverable through that path.

II. Related Work

Although flooding protocol is argued to be impractical due to the traffic volume it induce, it is mostly used as an upper performance bound. flooding is described as epidemic protocol by Vahdat and Becker[22] where every nodes exchanges its messages with every other node in its range. Though, notorious for overhead, in the case of sparse opportunistic networks, Islam and Waldvogel[8] have shown that epidemic routing is the only solution that may provide reasonable performance[8]. The reason being that contact patterns in such networks are so irregular that any kind of profile based routing has a very little chance of delivering the message to destination. Moreover, no one has contested the fact that given enough resources, flooding style replication can improve the timely delivery of messages. [8] shows, protocols that are solely dependent on contact frequency to classify paths fail to scale with variations with bandwidth whereas, protocols that use replication show a definite improvement when sufficient bandwidth is available.

Given the performance promised by epidemic routing, several approaches have been proposed to reduce its overhead of redundant replication [14], [25], [18]. Small et al. [18] examine a number of different strategies to suppress redundant transmissions and clean up valuable buffer space after a message has been delivered with epidemic routing. One can introduce a timer associated with every epidemic message after which the node gets "cured" and the message is deleted from the network[6]. Zhang et al. [25] describe a system, where a message is forwarded to another node with some probability smaller than one (i.e. data is "gossiped" rather than flooded). Other techniques used to control flooding include limiting the number of copies a node may forward, the time interval node must wait before retransmitting and number of hops a message may cover[6]. Finally, there are some strategies that try to use the inherent advantages associated with erasure coding techniques[23], [13], partitioning the messages to reduce the load on bandwidth and then routing all of them over same or different paths. Erasure coding has the potential to improve the delivery ratio of the messages as not all fragments have to reach the destination but it is hard to tune the code rate for general case, as it heavily depends on the network environment.

It is imperative to mention replication-less delivery methods[20], also called "smart strategies" where the aim is to push the message closer to the destination with every hop. Each node has to decide in the given circumstances that whether the next encountered node can bring the message closer to destination or not. The simplest of them is Direct Transmission [17] where source of the message delivers the message directly to destination without intervention of any other node. This scheme, as expected, is used as maximum delay baseline for routing algorithms. Jain and Fall[9] have proposed several oracles with future insight that although are unrealistic yet can help to understand the nature of underlying networks. [20] argues that given infinite node buffers, a protocol using contact oracle performs as good as flooding. This claim has been though contradicted in work by Islam and Waldcogel[8] arguing that oracle-based algorithms perform worse than flooding as the shortest path provided by the oracle may not be able to carry the message of a considerable size through to the destination. In other words, the minimum-delay or shortest path does not necessarily delivers the message to destination. We therefore need more intelligent, as well as, unpractical oracles that can give full assurance to deliver the message by providing a path that not only has big enough contacts duration as well as avoids the bottlenecks due to peripheral traffic.

In a mobile environment, where every mobile node is autonomous, it is not a simple task for every node to have such a picture of the whole network that it can ensure a message delivery based on that local knowledge. Although, all smart strategies utilize contact history information in their own way to predict the paths to the destination, there is no analysis available that shows prediction accuracy of these methods. There are several aspects involved in history gathering that directly or indirectly affect the routing decision. These aspects include (i) way metrics are gathered or calculated from node encounters (ii) effect of transitivity on the gathered metrics (iii) age of the metrics. Juang et. al [10] and MaxProp [2] propose two practical ways of collecting contact history information that is used to maximize the utility function that in turn gets the next hop. MaxProp [2], moreover also utilizes a fancy local message queuing method to give priority to messages that have better chance of being delivered. [14] has proposed to attach an aging factor with history information where recent history gets priority over the older history.

Although, simulation is a powerful tool to analyze any phenomena and its side effects that are either not in our control or hard to repeat, the reliability of the results gathered are highly dependent on environments and parameter used to simulate. In opportunistic networks, it is customary to create an artificial environment of designated area with assumptions for node speed/velocity, node coverage area [21], [23], [13]. Despite of the fact that these variables can be tuned to create dense or sparse networks, Islam and Waldvogel[8] have shown that artificially generated simulation environments are not a good replacement for real world scenario. Moreover, confidence level in the results must be crosschecked by testing the protocol in several scenarios. This may, in turn, complicate the comparative analysis as it is difficult to find multiple real

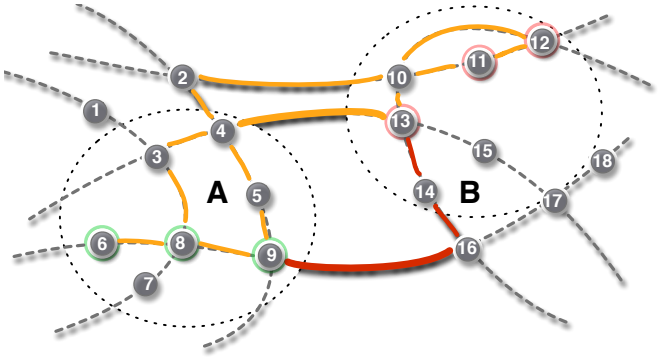

Fig. 1. Congestion-adaptive routing takes advantage of dispersion

world traces with matching life span. On the other hand, some efforts have been made to learn the characteristics of social networks and corresponding mobility model so that artificial models with different parameters can be built [12], [16]while validity of a social network model for communication in mission critical environments is an open question.

III. Nile design background

In our previous attempt[8], we analyzed the performance on 3 (discussed in section IV) different kind of networks given different data transmission rates. Single copy protocols that inherently depend on the information from the network, are not able to perform unless the required information flows throughout the network. This information is shared very rapidly and frequently in the case of strongly connected networks, this helps these protocols to out-perform others. flooding, in the case of strongly connected algorithm suffers very strongly from the bandwidth shortage as every node tries to send every byte it has. Moreover, no protocol other than flooding has the ability to adapt convincingly to different bandwidth scenarios. In the case of a dense network, the improvement of routing protocols (from low to high bandwidth) has been good but in sparse networks, only flooding has been able to deliver 70+% of the messages otherwise performance of remaining strategies struggle to meet the satisfactory level[8].

Motivated by these observations, we see Nile as a sound compromise between flooding and intelligent techniques, borrowing the strong points of both of the methodologies. As already stated, single copy protocols have to be unrealistically intelligent to give acceptable outcome in sparse networks. In our opinion, a protocol may replicate aggressively in a sparse but resource-rich network and at the same time can restrict the replication in a dense network where replication will not benefit. Moreover, there are many pivotal and unexplored aspects of opportunistic networks that replication-based algorithms can exploit to optimize the performance. In this work, we utilize the lessons learned from our previous protocol comparison work[8]for identifying and reducing overheads of flooding, resulting in a protocol that performs in all possible varieties of network conditions.

It is not possible for a node to have a global picture of the opportunistic mobile network(as several oracles have), rather a node may only have have the local neighborhood information such as the current node density, local congestion, replicas present for a particular message and so on. As discussed earlier, it is imperative to replicate messages to obtain acceptable results in sparse networks but it is also important to control congestion in dense environments thus this raises the question of replication factor. In our opinion, it is desirable to keep replication approach very flexible and dynamic i.e. the extent of replication should not be fixed by the source at the time of birth of message. The nodes that act as middle hops between source and destination can replicate the message depending on the local traffic conditions. The node that is at the centre of a cluster must refrain from generating extra replica as compared to a node which has very few direct contact; i.e. at the boundary of the network. Congestion can also be controlled by prioritizing the messages depending on the TTL, proximity to the destination or message sizes. In our opinion, messages having low TTL, smaller size, or are closer to destination deserve the priority over their counterparts.

Another issue involving clustered opportunistic networks is that traffic between two cluster is always routed through the set of minimum delay links between the clusters. As long as volume of the messages is less than the threshold that such a inter-cluster link can transfer, this mechanism works fine. As depicted in Fig 1, when this limit is crossed than all the messages being routed through this link can get choked due to lack of bandwidth and local limited storage. Here, source nodes depicted in green(6,8,9) are trying to find a path to destination nodes depicted in red(13,11,12). Assuming the link (9,16) being the most frequently occurring link between cluster A and B, every node in cluster A may select it for routing all the traffic destined for cluster B, creating congestion at this link(shown by thickness of the edge(9,16)). Earlier approaches do not not utilize the secondary path, because they use metric that have no notion of current traffic volume to rank the paths. Message switching is one of the simple solution where large messages can be fragmented and routed through distributed paths towards the destination. The appropriate fragmentation factor can be chosen at any middle hop depending on the congestion it is facing at current point in time where performance of such a solution is highly dependent on the throughput of the nodes. Nile, on the other hand, deploys a different strategy where paths are ranked according to not only delay that they may incur but also current traffic volume being transferred through them. Initially, all the paths will be prioritized according to time-delay but as the time goes on, paths with high traffic volume loose their priority, giving an opportunity to other paths that have not necessarily the minimum delay, to come in service. Such a mechanism plays very important role in the case of opportunistic mobile networks, where not only bandwidth is very scarce but also mobility. Pictorially in Fig 1, when edge(9,16) has been carrying data to its limits, node 9 can preferably lowers the ranking of this link and routes the messages through the next available path containing links (4,13),(2,10). This way the load on the network is balanced thus increasing the messages

Message count	100
Message size	1.6E3...1.6E7 B
Size distribution	Power law
Replication	$r = 4$
Erasure coding	$k = 4$
Bandwidth (low)	100 kiB/s
Bandwidth (med)	1000 kiB/s
Bandwidth (high)	10,000 kiB/s

TABLE I
SIMULATION PARAMETERS

delivery probability.

As far as mobility model is concerned, our belief has gone even stronger that routing protocols must be tested on real life traces. In our opinion, protocol simulations are not supposed to be bounded with any particular mobility model, may it be dense, sparse, social, or infrastructure network. Instead, the routing strategy is supposed to be adopt according to given conditions given reasonable time. Intuitively, more mobile the network is, quickly an algorithm has the possibility to adapt as there will be quick information flow throughout the network. A similar notion that mobility increases the capacity of the network, mobility increases the delivery probability of the messages has been shown presented in [5]. On the other hand mobility can cause negative impact on the network stability by introducing intermittent connectivity [19].

IV. SIMULATION AND TRACE SETUP

a) Trace: We have considered three different kinds of data sets, all of which have been obtained from CRAWDAD. The motivation behind choosing these three traces have been to have a broad spectrum between dense and sparse networks. Two of the data sets have been synthesized from reality mining project [4] from MIT spans on 16 months i.e. February 2004 to August 2005 whereas, the third data consist of the SNMP logs for one month from a IBM campus[1]. As the duration span of MIT reality mining is longer than IBM trace, we have filtered the MIT data to match the time span of IBM traces.

The sparse network is obtained from bluetooth logs of MIT traces where each node scans every five minutes for active Bluetooth neighbors and stored the duration of contact times. For the sake of comparison with other traces and simplicity, we limit ourselves to one month of connectivity trace, where any visible Bluetooth device was considered a candidate connection. Reduction of the trace time span has been done on the basis of connectivity times i.e, one month where nodes have maximum connectivity in terms of time duration. The highest connectivity period i.e. November 2004 showed 1858 bluetooth nodes suggesting a huge number of undesignated nodes as compared to the designated[2] 81 nodes that were designated to gather the data. It is here noteworthy that a few undesignated devices had more connectivity and interaction with the network than the designated nodes.

In the case of IBM Access Point trace, SNMP is used to poll access points (AP) every 5 minutes, from July 20, 2002 through August 17, 2002. A total of 1366 devices have been polled over 172 different access points during approximately 4 weeks. We have extracted the traces of 928 device after discovering existence of 3 clusters in this network and then choosing the biggest cluster with respect to node count. To turn these samples into continuous data, we assume that the snapshot data remains constant for the next 5 minutes. In the rare cases where this would cause an overlap with another snapshot from another access point, we assume that the transition happens halfway between the two snapshots. We assume that two nodes that are connected to one access point during overlapping time period are connected to each other. Thus, key features of such a network are low mobility and medium transmission range.

The third trace, MIT Cell Tower, is used according to the similar principal as that of IBM traces. The only difference being, instead of access points, cell towers are used to gather the contact times of the nodes with each other, thus the resulting network can be characterized as a very dense network due to high range of cell tower. Due to several lapses in data gathering, mentioned by the creators of the data, only 89 of 100 devices are included, which visit 32768 different cell towers. Similarly to Bluetooth traces, November 2004 turns out to be the maximum activity month with 81 devices and 12592 distinct cell towers.

It is imperative to mention that the assumption that two devices connected to one base-station(access Point or cell tower), introduces inaccuracies[3]. On one hand, it is overly optimistic, since two devices attached to the same access point may still be out of range of each other. On the other hand, the data might omit connection opportunities, since two nodes may pass each other at a place where there is no base-station, and this contact would not be logged. Another issue with these data sets is that the devices are not necessarily co-located with their owner at all times (i.e. they do not always characterize human mobility). Despite these inaccuracies, such traces are a valuable source of data, since they span many months and include thousands of devices.

b) Simulator: The motivation behind the simulator is to help us find the delays incurred by messages and overhead suffered by networks during execution of different routing algorithms. The output is analyzed on the basis of both number of messages as well as amount of data delivered. As already mentioned, three different traces have been used that significantly differ in the number of nodes involved, number, frequency, and distinctness of meetings that were taking place among the participants. For the purpose of this simulation, nodes connected to the same access point or the same cell tower are considered to be close enough physically to directly exchange messages with each other. IBM traces come out to be a sparsely connected network and MIT Cell Tower, a dense network, as the range of access points is smaller than that of cell towers. We have created 100 messages for the simulation with different sizes. The smallest size is 1600Bytes where as the the largest message size is 1.6E7 Bytes. We have followed power law to assign the sizes in this range; i.e. many small messages and a few huge messages. The peripheral simulation parameters are summarized in Table I.

It is important to mention that the primary aim of the protocol is to find a reliable path to destination and high

[2]Nodes running the scanning software are referred to as designated

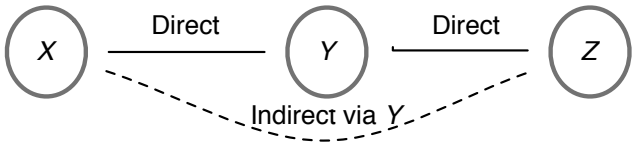

Fig. 2. Tuple sharing among peers

variation of connection time already poses a challenging role. Moreover, there has been no data available for the connection variation therefore, we assume that the links have constant bandwidth when connected. Thus, link parameters can–and have to–be assumed secondary. A detailed discussion of the simulation configuration can be found in[7].

V. NILE

Nile is basically a proactive routing protocol where each network node maintains a delay augmented routing table that in turn is used to control the replica propagation of the messages. Nile constructs these routing tables throughout the network using direct encounters to disseminate the contact information (similar idea is presented by Marina and Das[15]). Nile can be seen as a modified distance vector protocol, where at every node encounter, each node advertises a 3 tuple (details given below) list of nodes that it has logged during the previous encounters.. This information is accumulated in a routing table of each node and this table is referred to, whenever a routing decision is to be made. This implies, mobility of nodes plays a very important role in the performance of Nile, as mobility ensures a precise and accurate local information for each node is shared, that closely represents the true picture of the network.

A. Algorithm Description

For each encounter between two directly connected nodes X and Y as shown in Fig 2, The routing table is populated as follows.

a. Node ID of Y if it is a first contact between X and Y.

b. Update or insert average delivery time ϵ of message transmission from X to Y, ϵ_{XY}, includes the duration messages stay in the local queues as well as inter-hop transmission times.

c. Link Congestion η_{XY}. It is the ratio between the average number of bytes that are actually delivered b_{XY} and potentially could be delivered b^v_{XY} i.e.

$$\eta_{XY} = b_{XY}/b^v_{XY}$$

where $\eta_{XY} > 1$ indicates that the path is congested.

For an indirect neighbor Node X inserts the node Z accessible via Y as depicted in Fig 2, augmented information in its routing table as,

a. Node ID Z if X has no earlier record of Z via X.

b. Expected delivery time from X to Z,

$$\epsilon_{XZ} = \epsilon_{XY} + \epsilon_{YZ}$$

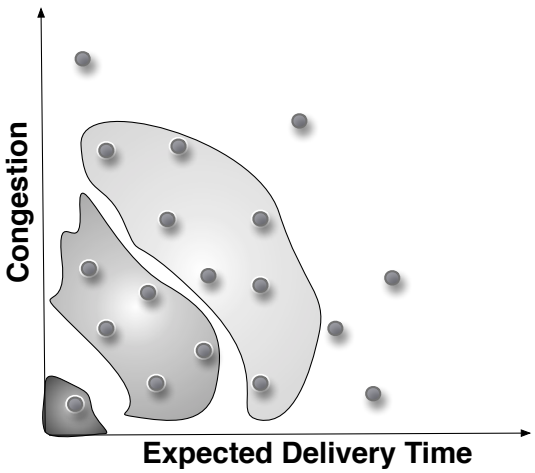

Fig. 3. Route selection methodology

c. Congestion Indicator on the path X, Y, Z

$$\eta_{XZ} = \begin{cases} max(\eta_{YZ}, \eta_{XY}) & \text{if } \eta_{YZ} < 1,\ \eta_{XY} < 1 \\ \eta_{YZ} \times \eta_{XZ} & otherwise \end{cases}$$

If there are multiple paths from Y to Z than the path with the minimum value of ϵ_{YZ} with its corresponding attributes is stored. This ensures that a faster path (irrespective of number of hops) gets the preference over a slower path that may have fewer hops. Any node X carrying a message M with destination D_M looks into the set of all direct peers denoted by P that have advertised the destination D_M through their direct or indirect contacts.

B. Congestion Handling

The critical measure to keep congestion in check is to control the number of replicas a node creates. One simple measure to restrict this number is to push replica on disjoint paths. The idea is to propagate replicas on those non-overlapping paths that are not congested at the moment, specially in dense clusters. To check whether two paths are disjoint, Nile employs *cosine measure similarity*. Literature shows that cosine measure similarity has been used to estimate the similarity between two documents[24] and we have adapted the problem at hand to use this measure in the following way. For each next hop that has a path to destination, Nile forms vectors consisting of ϵ_{XZ} as elements where Z may be any node on the path. Formally, a vector for message M at node X is calculated as,

$$PVec_{D_M} = \{\epsilon_{XZ} | \epsilon_{XZ} \leq \epsilon_{XD_M}\}$$

where D_M is the destination. A next hop vector will be removed from the candidate list If the *CosVectSimilarity* is higher than $1 - \eta_{XD}$, where η_{XD} is the congestion indicator on the path. This implies, a congested path will only be placed in the candidate list, if its highly disjoint than all the earlier paths selected. Conversely, a non disjoint path may only be selected only if it is not experiencing congestion.

Input: Set P of all directly connected neighbors at any node X, Destination D_M of Message M, Ratio r_M between Primary and Secondary path list for M
Output: A set $P_M \subset P$ i.e. set of reliable next hops for M
foreach *Node* $N \in P$ **do**
  if D_m *is accessible via* p **then**
    Add N to P_M;
    $PVec \leftarrow \epsilon_{ND_m} - \epsilon_{NX}$ where $\epsilon_{NX} < \epsilon_{ND_m}$
  end
end
Sort ascending P_M w.r.t. η_{ND};
for $i = 0$ *to* $Size(P_M)$ **do**
  for $j = i + 1$ *to Size*(P_M) **do**
    if *CosVect* $(PVec[i], PVec[j]) > 1 - \eta_{XD}$ **then**
      remove j from P_M;
    end
  end
end
Divide P_M by r_M

Algorithm 1: Path selection for given message M

It is not easy to avoid the question of complexity of computing cosine vector similarity. Nile can be seen as a modified Distance Vector protocol, where a node has to choose only among suitable direct neighbors. If X has n direct contacts that have accessibility to the destination, X will have n such vectors. For. e.g. a node may have 10 paths to destination and only 3 direct neighbors, then it has not to process 10 but only 3 vectors. So the complexity of cosine vector similarity does not depends on the number of paths but on the number of direct contacts through which the destination is accessible. Graphically shown in Fig 3, Nile prefers the paths that are experiencing neither long delays nor bandwidth shortages. The first chosen candidate is the one that has the minimum delay among the available candidate list. The next candidate will be chosen when it has either a relatively disjoint path to destination or its path has relatively low congestion thus keeping a reasonable mix of paths. The route selection algorithm is presented at Alg 1

As discussed previously, Nile may select a congested path provided its disjoint(er) than others paths. To avoid the unnecessary adverse affect of disjoint estimation, Nile further tries to tune the replication factor by maintaining *primary* and *secondary* candidate node lists for any message M. The ratio between the size of these two lists is represent by r_M in Algorithm 1. A node will try to produce a replica for only those paths that are available in primary list. If a node X encounters a secondary list node carrying the message M, X will reduce the size of its primary list. This way, Nile uses an abstract local node density criteria to reduce the replication factor. It can be imagined as an overlay network of all the nodes that are on the route to a particular destination and Nile keeps on pruning this overlay network provided, nodes are able to see the disagreements between their *primary* and *secondary* lists, related to a message M. Whenever a message is able to be replicated in the undesired region of neighborhood of a particular device, that device reduces its capability of replicating that message by a factor of f_M . In our simulation we have used we have used the initial ratio of 1.8 (denoted by f_M) between primary and secondary paths and rate of change f_M is set to 1.5 for the results presented in this paper.

VI. Results Discussion

Every history algorithm compared in this study has been provided first 10 trace days to get mature history as prescribed by the algorithm. Nile, on the other hand not only requires contact information from network but also the congestion indicator that we have devised. For the purpose of providing a mature history of traffic as well as avoiding any unbiased advantage to Nile, we have gathered the metric related to traffic volume by simulating directed flooding for the period of first 10 days of the trace times. Directed flooding is a predecessor and very crude version of Nile where attempt is made to create a replica for every node that has access to the destination. We have chosen directed flooding so that all the possible path with their corresponding delivery capabilities are exposed.

Fig 4 compares Nile with several different mechanisms for access point traces(details of these mechanism can be found in[8]). In first low bandwidth case in Figure 4(a), we observe that Nile has performed almost equal to flooding as far as number of messages are concerned. If we analyze the sizes of the messages, Fig 4(b) Nile is far better than flooding and is among the best that have performed size wise; i.e. erasure coding based methods [23], [13]. The reason why erasure coding techniques have performed better as far size of the message is concerned is due to the fragmentation of the messages. As far as number of delivered messages in high bandwidth is concerned Fig 4(c), flooding, Directed flooding, oracle and Nile are in the top category. In Fig 4 (c) we observe that Nile has appropriately adapted to the available bandwidth whereas none of the other competitors other then flooding have been able to do so. The best performing method is directed flooding that is the predecessor of Nile where as the rest of the protocols have not been able to take advantage of extra available bandwidth.

In Fig 5(a) in the case of dense and resource deficient MIT cell tower, Nile performs as good as the best competitor i.e. Simple erasure coding [23]. Erasure coding techniques have performed significantly well because the contact durations of MIT network are smaller than that of IBM though the network is quite dense. Majority of the nodes in MIT had a direct contact with each other but the duration of the contact was small hindering the delivery of huge messages. This can be seen in high bandwidth scenario Fig 5(b) where each of the protocol has taken advantage of the dense network and delivered majority of the messages.

The most sparse network in our analysis i.e. MITBT Fig 6 has proven the necessity of replication more than any other network. As there have been paths that do not occur frequently, the history is not very reliable and flooding remains the uncontested winner in this case.

The discussion cannot be complete without having a word about the load exerted on the network. Fig 7 shows the traffic

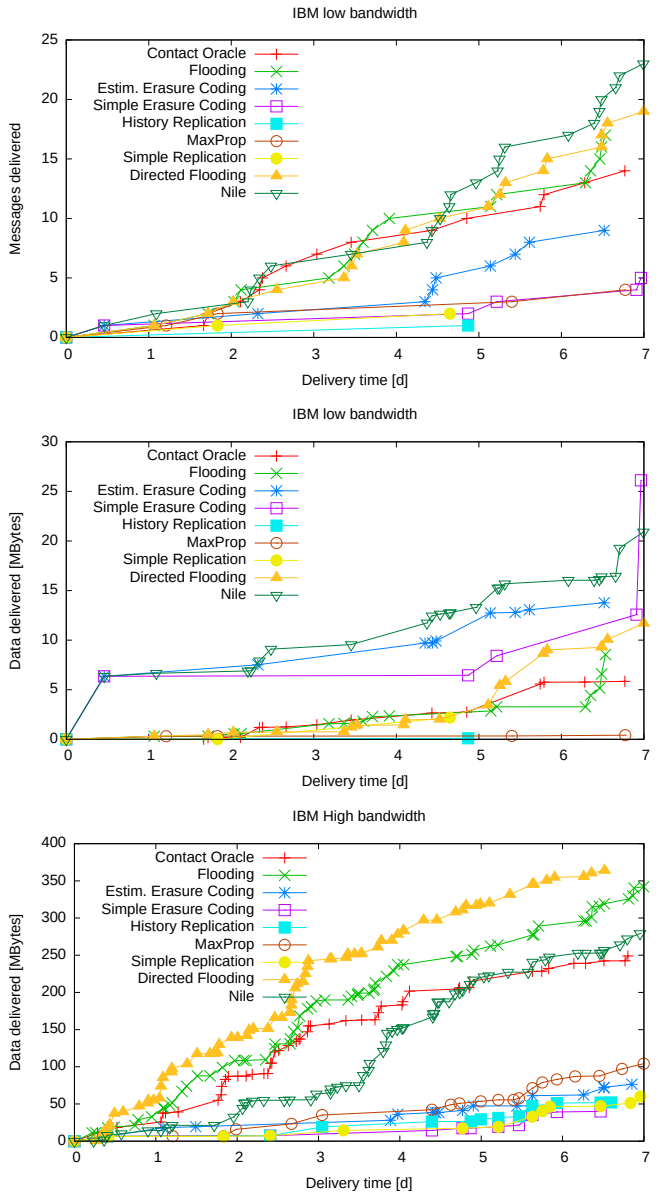

Fig. 4. (a),(b),(c)- Performance comparison in access point trace among different bandwidth scenarios

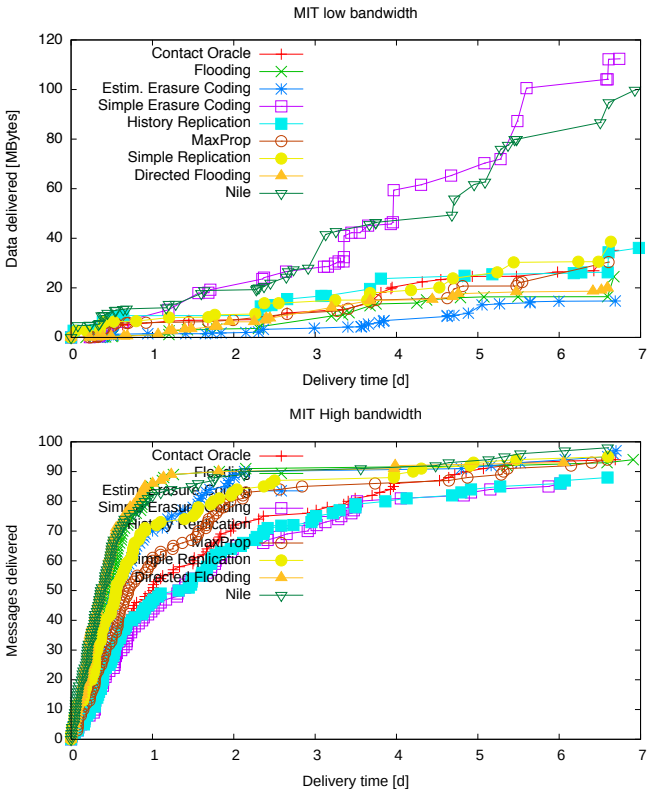

Fig. 5. (a),(b)- Performance comparison in cell tower trace between low and high bandwidth scenarios

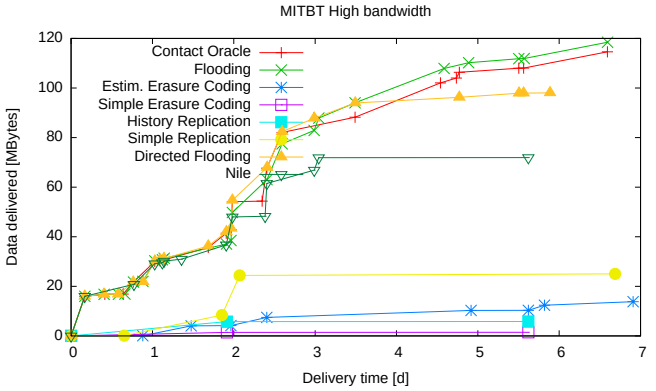

Fig. 6. Performance Comparison of bluetooth trace

that has been created by the simulated protocols. In access point case, Nile has induced approx. 1/3 volume as induced by flooding and directed flooding and in the case of cell tower, the traffic volume created by Nile is far less than the erasure coding based protocols. In this case erasure coding protocols have created the maximum load on the network because, their erasure coding factor has been pre-determined at the source so the irrespective how efficient have been the node to propagate the messages, the traffic volume has been static. On the other hand flooding could not create that much traffic volume because it has failed to create replicas due to lack of resources. We have to keep in mind that Nile in both cases has performed very close to the best performer without exerting the network to the equivalent traffic volume.

VII. Conclusion

In this work, we have investigated the factors that help flooding to perform better in sparse networks and those that degrade it performance in congested networks. We have then, combined the positives of the two worlds to get a solution that not only scales nicely to the network size as well as adapts to available network resources. The reason why Nile adapts very well to different bandwidth and network density scenarios, is the flexible control of replica creation. Furthermore, we see that the other algorithms fluctuate heavily between the different scenarios, indicating that they may perform well in once scenario, but as soon as underlying network changes, their performance is strongly affected. The relatively simplicity

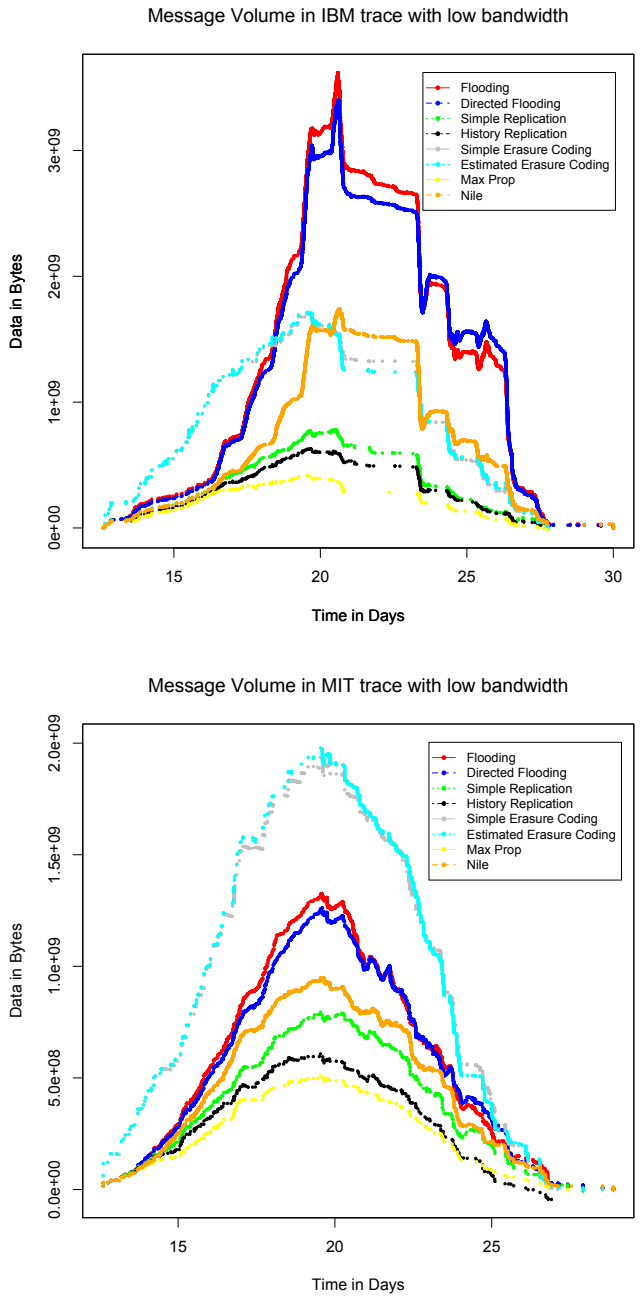

Fig. 7. Traffic volume induced by different protocols

of protocol and overall performance of Nile, combined with the ability to be further fine-tuned to specific environments, in our opinion makes it the prime choice for any mobile opportunistic network.

REFERENCES

[1] M. Balazinska and P. Castro, "CRAWDAD data set ibm/watson (v. 2003-02-19)," Downloaded from http://crawdad.cs.dartmouth.edu/ibm/watson, Feb. 2003.

[2] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "Maxprop: Routing for vehicle-based disruption-tolerantnetworking," in *Proceedings of IEEE Infocom*, Barcelona, Spain, Apr. 2006.

[3] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on the design of opportunistic forwarding algorithms," in *In Proc. IEEE Infocom*, 2006.

[4] N. Eagle and A. S. Pentland, "CRAWDAD data set mit/reality (v. 2005-07-01)," Downloaded from http://crawdad.cs.dartmouth.edu/mit/reality, Jul. 2005.

[5] M. Grossglauser and D. N. C. Tse, "Mobility increases the capacity of ad hoc wireless networks," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 477–486, August 2002. [Online]. Available: http://dx.doi.org/10.1109/TNET.2002.801403

[6] K. A. Harras, K. C. Almeroth, and E. M. Belding-Royer, "Delay tolerant mobile networks (dtmns): Controlled flooding in sparse mobile networks," *Lecture Notes in Computer Science*, vol. 3462, pp. 1180–1192, May 2005.

[7] A. Islam, "Nuntifix-modeling of delay tolerant networks," University of Konstanz, Tech. Rep., 2008.

[8] A. Islam and M. Waldvogel, "Reality-check for dtn routing algorithms," in *ICDCSW '08: Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems Workshops*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 204–209.

[9] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proceedings of SIGCOMM 2004*. ACM Press, 2004, pp. 145–158.

[10] P. Juang, H. Oki, Y. Wang, M. Martonosi, L.-S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet," in *ASPLOS-X*, 2002.

[11] A. Keranen and J. Ott, "Increasing reality for dtn protocol simulations," *Helsinki University of Technology, Tech. Rep., July*, 2007.

[12] J. Kim, V. Sridhara, and S. Bohacek, "Realistic mobility simulation of urban mesh networks," *Ad Hoc Netw.*, vol. 7, no. 2, pp. 411–430, 2009.

[13] Y. Liao, K. Tan, Z. Zhang, and L. Gao, "Estimation based erasure-coding routing in delay tolerant networks," in *Proceedings of IWCMC*, Jun. 2006.

[14] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, July 2003.

[15] M. K. Marina and S. R. Das, "On-demand multipath distance vector routing in ad hoc networks," in *in Proceedings of IEEE International Conference on Network Protocols (ICNP)*, 2001, pp. 14–23.

[16] M. Musolesi and C. Mascolo, "A community based mobility model for ad hoc network research," in *REALMAN '06: Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*. New York, NY, USA: ACM, 2006, pp. 31–38.

[17] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a three-tier architecture for sparse sensor networks," in *Intel Research Technical Report*, 2003.

[18] T. Small and Z. J. Haas, "Resource and performance tradeoffs in delay-tolerant wireless networks," in *ACM workshop on Delay Tolerant Networking*, 2005.

[19] J. Su, A. Chin, A. Popivanova, A. Goel, and E. de Lara, "User mobility for opportunistic ad-hoc networking," in *Mobile Computing Systems and Applications, 2004. WMCSA 2004.*, 2004, pp. 41–50.

[20] K. P. Thrasyvoulos Spyropoulos and C. Raghavendra, "Efficient routing in intermittently connected mobile networks: The multiple-copy case," in *ACM/IEEE journal of Transactions on Networking,*, 2008.

[21] ——, "Efficient routing in intermittently connected mobile networks: The single-copy case," in *ACM/IEEE journal of Transactions on Networking,*, 2008.

[22] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Duke University, Tech. Rep. CS-200006, April 2000.

[23] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-coding based routing for opportunistic networks," *ACM workshop on Delay Tolerant Networking*, 2005.

[24] R. Wilkinson and P. Hingston, "Using the cosine measure in a neural network for document retrieval," in *SIGIR '91: Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 1991, pp. 202–210.

[25] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance modeling of epidemic routing," in *IFIP Networking*, 2006.