Routing for Opportunistic Networks Based on Probabilistic Erasure Coding

Fani Tsapeli and Vassilis Tsaoussidis

Space Internetworking Center, (SPICE),
ECE Department, Democritus University of Thrace, Xanthi, Greece
{ttsapeli,vtsaousi}@ee.duth.gr

Abstract. We investigate the problem of routing in opportunistic networks and propose a novel routing algorithm, which combines probabilistic routing with erasure coding. Erasure coding generates large amounts of code blocks with fixed overhead; we apply a sophisticated but realistic method to allocate the generated code blocks to nodes that relies on a probabilistic metric for evaluating node potential. Our goal is to enhance further the robustness of the erasure-coding based forwarding in worst-case delays but also in small-delay scenarios. We detail our algorithm and evaluate its performance against other well-known routing algorithms. We exploit scenarios with adequate resource storage as well as scenarios with limited storage capacity. In both cases, our algorithm yields promising results.

Keywords: routing; delay tolerant networks; erasure coding.

1 Introduction

Opportunistic networks exemplify characteristics of delay tolerant networking [1] (DTNs) in environments where contacts appear opportunistically. Common examples of such networks are mobile sensor networks [2], underwater sensor networks [3], pocket switched networks [4], or transportation networks [5], where delay tolerance constitutes an inherent property of limited bandwidth, energy or sparse connectivity. In such networks, traditional routing techniques for ad hoc networks, which assume that an end-to-end path between a source and a destination node always exists, cannot be applied. Routing in opportunistic networks is a particularly challenging problem since it does not simply rely on connectivity to establish optimal routes but rather attempts to exploit windows of connectivity opportunities. Therefore, routing in opportunistic networks becomes a scheduling problem with inherently probabilistic properties.

Routing in opportunistic networks is based on the store-carry-and-forward technique according to which messages are transferred using both transmissions and node physical movement. Thus, node mobility is exploited to address the problem of limited connectivity. A general technique to enhance reliability and reduce delivery delay is the use of a replication scheme according to which identical message copies are spread to the network in parallel. Spreading sufficiently large number of replicas in the network increases the probability that at least one of them will reach its final destination. However, transmission of multiple copies wastes network resources. Thus, there is a trade-off between delivery latency and traffic overhead.

Erasure coding has been proposed as an alternative way to generate redundancy, instead of applying simple replication. According to this technique, each message is encoded into such a number of code blocks that the reconstruction of the original message requires only a specific amount of the generated code blocks. The code blocks are spread throughout a large number of relay nodes, which may carry individually less data than the whole message copy. Utilization of a large number of relays for the transmission of messages renders erasure coding based forwarding techniques more robust to the occasional failures of some relays. However, since it takes longer for the source node to complete the distribution of the code blocks, erasure coding based techniques are associated with prolonged delivery latency.

In this work, we propose a novel erasure coding based forwarding algorithm for heterogeneous networks. We apply an effective policy to allocate the generated code blocks based on the node ability to transfer the blocks further to their final destination. We examine our scheme in conjunction with other well-known replication based forwarding schemes, in scenarios with sparse connectivity and limited buffer resources. We evaluate the performance of the proposed algorithm in terms of delivery latency and delivery ratio.

The rest of the paper is organized as follows. In section 2, we briefly discuss the existing routing approaches for opportunistic networks and emphasize on erasure-coding based methods. In section 3, we present our proposed routing policy. In section 4 we present our evaluation based on selected simulation results and in section 5 we summarize our conclusions and future directions.

2 Related Work

Several approaches have been proposed to deal with the problem of routing in opportunistic delay-tolerant networks. Most of them are based on spreading identical message copies in the network. Epidemic routing [6], a flooding-based method, constitutes a typical example of this category of algorithms. Although epidemic routing achieves the best performance in terms of delivery delay, it imposes significant overhead which renders it impractical in reality. *Controlled Flooding* schemes have been proposed towards this direction, namely to alleviate epidemic routing from the burden of resource exhaustion.

Controlled Flooding schemes cannot guarantee a fixed overhead and, in the worst case, they may act exactly like in epidemic routing. In [11], the *Spray and Wait* forwarding scheme has been proposed to ensure fixed overhead while achieving acceptable delay latency. According to this scheme, a specific number of copies is forwarded to the r first relays that the source node encounters. Each of these relay nodes is only allowed to perform direct transmission to the final destination.

Authors in [12], propose a forwarding algorithm based on erasure coding. According to this algorithm, a message of size M bytes is encoded using an erasure coding algorithm (e.g. Reed-Solomon coding) with replication factor *r*, into $N = \frac{Mxr}{b}$ blocks where b is the size of each block in bytes and *r* is the replication factor. These code blocks are equally split among the first $k \cdot r$ relays. A message can be decoded iff at least *N/r* code blocks reach their final destination. Thus, only k out of $k \cdot r$ relays should deliver their blocks to the final destination in order to have a successful message delivery. This method has the same overhead as in *Spay and Wait* with replication factor r, while k more relays are utilized. It has been proved that the erasure-coding based forwarding scheme achieves better worst-case delay performance than *Spray and Wait.* However, it incurs prolonged delivery latency in typical cases. In an attempt to improve the performance of the erasure-coding technique in small delay performance cases, authors in [13] present a hybrid scheme which incorporates aggressive erasure coding forwarding. This technique has improved performance both in worst delay performance cases and in very small delay performance cases. However, this gain comes at the cost of double overhead.

Both the former erasure-coding based routing schemes assume a simple scenario in which all nodes are equally qualified to act as relays for a message. Considering nodes with different characteristics, a more sophisticated method to allocate code blocks could be investigated. Such an approach is presented in [14], where the performance of different allocation methods, in terms of optimizing the delivery probability in the presence of path failures, is examined.

In [15], erasure-coding based forwarding is deployed along with an estimation-based scheme. According to the authors, the generated code blocks are split to the relays in proportion to their delivery predictability. However, to the best of our knowledge, this method has only been evaluated over reliable networks, where nodes never fail or drop messages. We claim here that a simple proportional allocation of code blocks cannot constitute a viable solution but instead, it may frequently result in a typical simple erasure coding forwarding scheme. However, the estimation concept presented in [15] can be further exploited.

Therefore, in our present work we depart from [15] and investigate an enhanced method for the allocation of code blocks. Our goal is to reduce the delivery delay of simple erasure coding for the small delay performance cases while retaining its improved performance to worst-case delays. Thus, we allow relays that are considered to be *good* – according to our justified criteria – to carry as many blocks as it is required in order to successfully decode a message; however we also exploit nodes with worse delivery predictability as relays for less number of code blocks. We determine both an upper and a minimum threshold to the number of blocks that a node can carry in order to confine the number of relays that will be utilized.

3 Probabilistic Erasure Coding Routing

As it was stated earlier, our goal is to improve the performance of erasure-coding based routing in terms of delivery latency in both small delays and worst-case delays. Thus, we apply an effective and quick method for dispatching the code blocks and we define a proper metric to distinguish between relays with high probability to deliver a message to its destination (*good* relays) and relays with lower probability (*bad* relays). However, although our proposal prompts to a binary selection process, the probabilistic nature of our scheme along with the diversity of our dispatching methodology, cancels the main drawbacks of binary – and hence possibly faulty – strategy. In this section, we describe comprehensively our proposed routing scheme.

3.1 Proportional Allocation Method

An effective method of allocating code blocks should utilize each contact opportunity and let *good* relays carry more messages while exploiting also *bad* relays by dispatching them a small portion of code blocks. The technique proposed in [15] satisfies this statement and incorporates high reliability since it accounts for the possibility of fault prediction and consequently false characterization of the binary relay state (i.e., *good* vs. *bad*). However, our analysis shows that dispatching a number of blocks to a node in direct proportion to its delivery predictability is not an effective technique.

Let consider the allocation method proposed in [15]. When two nodes meet and one of them has more code blocks than a threshold G it re-dispatches them proportionally to the estimation metrics. The estimation metric is defined as the number of contacts $N_{i,j}$ that node i had with the destination node j within time interval T. Assume that node A has m_A code blocks for a given message destined to node C and node A encounters node B. It will send to node B m_B messages where

$$m_B = m_A \cdot \frac{\tau_{B,C}}{\tau_{A,C} + \tau_{B,C}} \tag{1}$$

and $\tau_{i,j} = N_{i,j}/T$.

Consider a simple scenario where node A has encoded a message with erasure coding and replication factor three. Thus, a destination node E should receive the 33.3% of the generated code blocks in order to decode the message. The threshold G is set to the 33.3% of the code blocks. Assume the following information about the number of contacts that nodes A, B, C and D, respectively, had with node E within the time interval T:

Table 1. Number Of Contacts Of Nodes A, B, C And D With Node E

$N_{A,E}$	$N_{B,E}$	$N_{C,E}$	$N_{D,E}$
10	2	4	10

Now let's assume that node A encounters nodes B, C and D at times t1, t2 and t3 correspondingly and re-dispatches its code blocks according to equation (1). The result of this procedure is depicted in Table 2.

According to our sample allocation, at least two out of four relays are required to succeed in order to decode the message successfully. It should be noticed that, in this example, the result accrued from the proportional allocation method matches the gain of the simple allocation method, where blocks are equally distributed among four relays. Furthermore, since only few relays are utilized, the robustness of the erasure coding technique in worst-case delays may be jeopardized.

Table 2. Blocks Allocation According To Equation (1)

	m_A %	m_B%	m_C%	m_D%
t0	100	0	0	0
t1	83.3	16.7	0	0
t2	59.5	16.7	23.8	0
t3	29.75	16.7	23.8	29.75

3.2 The Proposed Allocation Method

According to our method, each node maintains a table with metrics for each other known node in the network. The metric of node i to node j represents the properness of node i to act as a relay for messages destined to node j. As we detail later, each node will eventually apply predefined rules that elaborate on these metrics in order to characterize whether a relay for a given message is *good* or not.

It is certainly desirable to confine the maximum number of relays that can be utilized – spreading code blocks into too many relays would reduce the performance of our algorithm in terms of delivery latency. Thus, we define both a minimum quantity of code blocks Q_{min} and a maximum number of code blocks Q_{max} that a node is obliged, or allowed, respectively, to carry. Q_{max} is set to 1/r of the generated code blocks and it should be a multiple of Q_{min}. Notice that while Q_{min} is a tunable parameter, Q_{max} is inflexible.

For simplicity, two nodes will not exchange any blocks of a given message if both of them have already code blocks of it. Each node that carries more than Q_{max} code blocks, in each contact opportunity, will dispatch Q_{max} code blocks to the other node. Each node *i* that carries N_i code blocks where $Q_{min} < N_i \leq Q_{max}$, upon encountering node *j*, it will use the rule $R^{i,j}_{split}$ to decide whether it is *good* relay for this message or not. If it is a *bad* relay it will dispatch half of its code blocks to node *j*. If node *i* is *good* relay for a message or if it contains Q_{min} code blocks, thus it cannot split blocks any further, if node *j* is considered to be better relay according to rule $R^{i,j}_{forward}$, it will send all its code blocks to *j*.

According to our method, each node initially receives Q_{max} code blocks and decides afterwards whether it should split or not based on rule R_{split}. However, some relays may experience very sparse connectivity or too limited buffer resources; hence, they will not be able to forward the blocks they have received. Thus, a node *i* with $N_i > Q_{max}$ blocks should use the rule $R^{j,i}_{low}$ to determine whether node j is such a *bad* relay. In that case, it will dispatch to j only the minimum quantity of blocks Q_{min}. Correspondingly, nodes with high delivery probability would be more effective on dispatching their blocks. Thus, each node i with $N_i > Q_{max}$ blocks should apply a rule R_{high} to recognize these advanced relays and dispatch to them more than Q_{max} blocks. The pseudo-code of our model is presented below.

Algorithm The allocation method

Description: Node i which has N_i code blocks of a message encounters node j which does not have any code blocks.

```
1:  if ($N_i > Q_{max}$)
2:      if $\left((R^{j,i}_{high})\&\&(!\,R^{i,j}_{high})\right)$
3:          $N_j = N_i - Q_{max}$
4:      else if $\left((R^{j,i}_{high})\&\&(R^{i,j}_{high})\right)$
5:          $N_j = \left\lfloor N_i / (2 \cdot Q_{min}) \right\rfloor \cdot Q_{min}$
6:      else if $(R^{j,i}_{low})$
7:          $N_j = Q_{min}$
8:      else
9:          $N_j = \min(Q_{max}, N_i - Q_{max})$
10: else if $\left((Q_{min} < N_i \le Q_{max})\&\&(R^{i,j}_{split})\right)$
11:     $N_i = \left\lfloor N_i / (2 \cdot Q_{min}) \right\rfloor \cdot Q_{min}$
12: else if $(R^{i,j}_{forward})$
13:     $N_j = N_i$
14: $N_i = N_i - N_j$
```

It should be noticed that each node in the DTN will carry $n \cdot Q_{min}$ code blocks, with $\{n \in \mathbb{Z} \mid 0 \le n \le Q_{max}/Q_{min}\}$. Thus, the number of relays n_r that will be utilized is confined as: $r \le n_r \le r \cdot Q_{max}/Q_{min}$.

3.3 Rules Definition

The effectiveness of our algorithm is partially based on the selection of proper rules; indeed the rules determine the ability of a node to act as relay for a message - and hence, inherently affect the efficacy of the proposed scheme.

We assume that nodes follow a mobility pattern according to which nodes that have been previously encountered it is likely to be encountered again. This assumption does not hold for all types of mobility; however, it is valid for most scenarios since users do have mobility patterns that reflect repetitive behaviors or mobility preferences. Thus, a node can decide about its properness to act as relay for a message based on the history of its encounters. According to our model, each node retains a table with probabilities to meet each other node in the network. We use the same model as PROPHET protocol [10] to compute and update these probabilities.

According to PROPHET, when node a encounters node b, it updates its delivery probability for node b according to equation (2):

$$P_{(a,b)} = P_{(a,b)old} + \left(1 - P_{(a,b)old}\right) \times P_{init} \tag{2}$$

where P_{init} is an initialization constant.

The probabilistic metric is also assumed to have a transitive property according to which node a updates its delivery probability for node c according to equation (3):

$$P_{(a,c)} = P_{(a,c)old} + (1 - P_{(a,c)old}) \times P_{(a,b)} \times P_{(b,c)} \times \beta \tag{3}$$

where β is a constant.

Finally, equation (4) is used to decrease the probability of node a to encounter node b if these two nodes have not been encountered for a period of time:

$$P_{(a,b)} = P_{(a,b)old} \times \gamma^{k} \tag{4}$$

where γ is an aging constant and k indicates the number of time slots that have elapsed since the last update of the metric.

We should also incorporate into our decision the available storage resources of communicating nodes that act as potential relays. In order to compare nodes i, j in terms of their storage availability we define the *Storage* metric as described in (5):

$$Storage_{(i,j)} = \frac{FreeSpace_i}{\max\left(BufferSize_i, BufferSize_j\right)} \tag{5}$$

Finally, we define the rules $R^{i,j}_{split}$, $R^{i,j}_{forward}$, $R^{i,j}_{low}$ and $R^{i,j}_{high}$, according to which node i dispatches to node j a portion or all of its code blocks that are destined to node d, as follows:

$$R^{i,j}_{split} = \left(P_{(i,d)} < T_{prob}\right) \quad \vee \quad \left(Storage_{(i,j)} < T_{buf}\right) \tag{6}$$

$$R^{i,j}_{forward} = \left(P_{(i,d)} < P_{(j,d)}\right) \wedge \left(Storage_{(j,i)} > T_{buf}\right) \tag{7}$$

$$R^{i,j}_{low} = \left(4 \times P_{(i,d)} < T_{prob}\right) \vee \left(Storage_{(i,j)} < T_{buf}\right) \tag{8}$$

$$R^{i,j}_{high} = \left(P_{(i,d)} > 4 \times T_{prob}\right) \wedge \left(Storage_{(i,j)} > T_{buf}\right) \tag{9}$$

where T_{prob} and T_{buf} are predefined thresholds.

4 Simulation Results

We evaluate our *probabilistic erasure coding* scheme (*ProbEC*) in terms of delivery delay by performing a set of simulations. We compare our scheme with other forwarding algorithms such as *spray and wait* (*SnW*) [11] and *erasure coding* (*EC*) [12]. In order to demonstrate the impact of erasure coding alone, we also examine the performance of *ProbEC* when we set $Q_{min} = Q_{max}$, allowing each node to carry exactly the number of code blocks that are required for the message's reconstruction. This strategy differs from the simple replication algorithm as each node that encounters a node with better delivery predictability, will forward all of its blocks. We name this forwarding policy as *probabilistic replication* (*ProbRep*).

4.1 Simulation Model

We implemented our model in the Opportunistic Network Simulator, ONE [16]. The simulation area size is 3600×3600m^2. Nodes follow a restricted random waypoint movement model similar to that of [15]. We divide the whole area in four equal square subareas A, B, C and D. We randomly set 40 waypoints in each subarea. Each node in the network has a given set of 30 waypoints. Nodes choose randomly a destination from their set of waypoints and a random velocity within an allowed interval. After arriving at the destination they wait for a random time interval and then they pick another destination. We consider four groups of nodes, each one containing 20 mobile nodes. We define different groups of nodes in order to allow nodes belonging to the same group to move within a specific subarea with higher probability. In Table 3 we present the percentage of waypoints within a specific area that nodes have according to the group that they belong to. It becomes apparent that our simulation scenario matches real-life scenarios and is aligned with the probabilistic policy that was incorporated in equations (2), (3) and (4).

Table 3. Waypoints Distribution Per Group Of Nodes

	Waypoints in area A	**Waypoints in area B**	**Waypoints in area C**	**Waypoints in area D**
1st Group	80%	10%	10%	0%
2nd Group	0%	80%	10%	10%
3rd Group	10%	0%	80%	10%
4th Group	10%	10%	0%	80%

Each group of nodes contains nodes with different velocity ranges. More precisely, each group is consisted of 15 slow moving nodes whose velocities range from 0.5m/s to 2m/s and 5 fast moving nodes with velocity range from 4m/s to 6m/s.

According to our simulation model, a message is created every 30 seconds. The source and the destination of the message are randomly selected among the 80 nodes of the network. Hence, the deterministic manner used for generating messages is enhanced by the random location of communication, which in terms results in non-deterministic generation of messages at each communication sub-area. We allow nodes to move within the simulation area without creating any messages for the first 12000 seconds in order to initialize their delivery probability tables. Afterwards, messages are created for the next 46000 seconds. The whole simulation time of each experiment is set to 216000 seconds, a duration that is deemed sufficient for the algorithms to exploit their potential.

Thresholds T_{prob} and T_{buf} have been set to 0.015 and 0.1 correspondingly and parameters P_{init}, β and γ to 0.7, 0.3 and 0.98. The message size is set to 2000 bytes and the block size 250 bytes. We examine the performance of our algorithm using replication factor 4. Thus, we generate 32 blocks per message. In the following experiments, we set the Q_{max} parameter of *ProbEC* to 8 blocks and the Q_{min} to 2 blocks. Thus, the minimum number of utilized relays is 4 and the maximum 16. Respectively, according to the examined EC scheme, code blocks will be equally split to 16 relays. Although we repeated the experiments with various parameters (e.g., message sizes or rates) we only present here representative results for each scenario.

4.2 Experiments in General Network Scenarios

In this section, we evaluate the performance of our algorithm in scenarios with adequate buffer resources. In this scenario, 5 out of 15 slow moving nodes in each group have 160KB buffer, while all the other nodes have 360KB buffer.

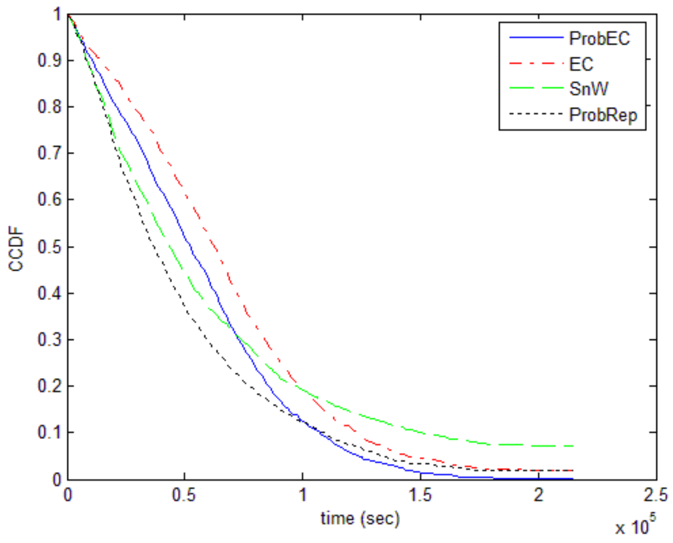

Fig. 1. CCDF of messages delivery latency for ProbEc, EC, SnW and ProbRep routing algorithms in scenario with adequate buffer resources

In Fig. 1 we present the simulation results of data latency distribution in complementary CDF (CCDF) curves for ProbEC, EC, SnW and ProbRep. From Fig. 1, it is clear that ProbEC has improved performance compared with EC both for small and for worst delay performance cases. While SnW still outperforms our algorithm for small delay performance cases, we achieve significant improvement for worst-case delays. We note, however, that small delays do not really pose a challenge; longer delays, instead may annoy users and impact quality of service significantly. ProbRep, acts as an enhanced spray and wait forwarding scheme, which enables nodes to forward their messages if they encounter a node with better delivery predictability: it performs very well both in small and in worst delay performance cases and owes its performance to the strategy of allowing bad relays to re-forward their messages instead of just waiting until encountering the final destination. However, we observe that ProbEC outperforms the other approaches in worst-case delays; this is expected, since it utilizes more relays. In Table 4 we provide numerical results extracted from the previous curves, in order to allow an easy comparison of the examined forwarding schemes. According to this table, ProbEC manages to deliver the 90% of the generated messages faster than all the other algorithms and overall, it delivers more messages until the end of the experiment.

Table 4. Delivery Latency

Algo-rithm	**Delivery Delay (seconds)**				**Successfully Delivered Messages**
	25% of mes-sages	50% of mes-sages	75% of mes-sages	90% of mes-sages	
ProbEC	27000	52000	79000	106000	99.68%
EC	35000	62000	90000	122000	98.07%
SnW	19000	43000	83000	148000	94.21%
ProbRep	18000	41000	74000	110000	97.94%

4.3 Experiments in Scenarios with Limited Storage Resources

In this experiment, we evaluate the performance of our algorithm in scenarios with limited storage resources. We set the buffer capacity of 5 out of 15 slow moving nodes of each group equal to 20KB, while we retain the buffer size of all other nodes to 360KB.

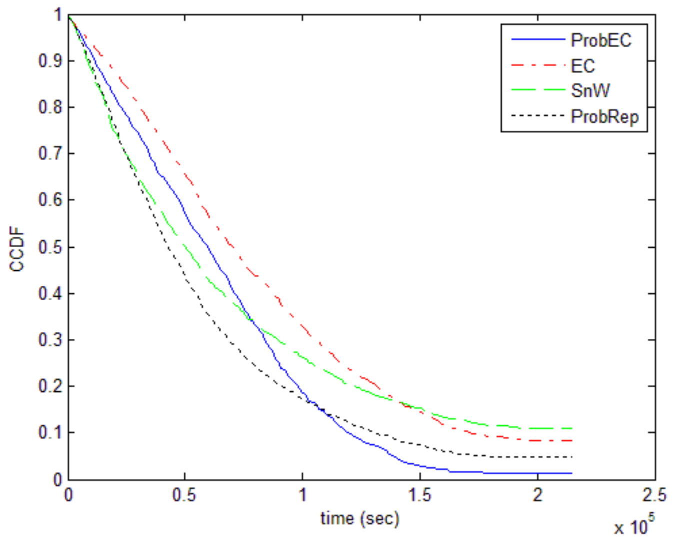

Fig. 2. CCDF of messages delivery latency for ProbEc, EC, SnW and ProbRep routing algorithms in scenario with limited buffer resources

It is clear both from Fig. 2 and Table 5 that while the performance of EC and SnW is decreased significantly compared to their performance in scenarios with adequate storage resources, ProbEC and ProbRep have only a slight degradation in their performance. That is due to their attitude to forward further their messages when storage is exhausted. ProbEC still outperforms all the other algorithms and manages to deliver the 98.71% of the generated messages even in this extreme scenario.

While in the previous scenario we observe that the performance of the ProbEC scheme, compared to the ProbRep, is slightly improved, in this scenario the impact of adopting erasure coding is more obvious. In this experiment, ProbEC manages to deliver 2.5% more messages than ProbRep while in the previous experiment it delivers 1.74% more messages. Moreover, an interesting observation is that among the examined schemes, EC is the most vulnerable to data loss due to storage exhaustion; in this experiment it delivers 6.43% less messages than in the previous one, while SnW delivers 5.14% less messages correspondingly. EC scheme exemplifies significant performance degradation in experiments with limited storage resources because of its poor small delay performance; long delay in message delivery results in increased need of storage resources.

Table 5. Delivery Latency

Algorithm	**Delivery Delay (seconds)**				**Successfully Delivered Messages**
	25% of messages	50% of messages	75% of messages	90% of messages	
ProbEC	29000	59000	89000	118000	98.71%
EC	37000	70000	116000	169000	91.64%
SnW	19000	50000	103000	-	89.07%
ProbRep	20000	42000	82000	129000	96.21%

5 Conclusions and Future Work

We presented a novel routing algorithm for opportunistic networks, which combines probabilistic routing with erasure coding. According to our scheme, a set of rules prescribes the number of code blocks that a node should carry based on its suitability to act as relay for the specific message. Our goal is to exploit the potential of erasure coding scheme to improve the worst-case delay without damaging performance in small delay cases. Our simulation results demonstrate that our algorithm outperforms the simple erasure coding scheme both for small and worst delay performance cases.

The selection of appropriate value for threshold T_{prob} is very important. T_{prob} should vary within the range of node probabilities. Currently, we define T_{prob} statically. However, we plan to investigate the use of an adaptive threshold whose computation will be based on information exchanged by nodes. Finally, it will be interesting to also investigate in more detail the impact of T_{prob} selection.

Acknowledgments. The research leading to these results has received funding from the European Community's Seventh Framework Programme ([FP7/2007-2013_FP7-REGPOT-2010-1, SP4 Capacities, Coordination and Support Actions) under grant agreement **n° 264226** (project title: Space Internetworking Center-SPICE). This paper reflects only the authors views and the Community is not liable for any use that may be made of the information contained therein.

References

1. Delay tolerant networking research group, `http://www.dtnrg.org`
2. Wang, Y., Wu, H.: Dft-msn: The delay fault tolerant mobile sensor network for pervasive information gathering. In: IEEE Infocom (2006)
3. Heidemann, J., Ye, W., Wills, J., Syed, A., Li, Y.: Research challenges and applications for underwater sensor networking. In: Proceedings of IEEE WCNC (2006)
4. Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J., Diot, C.: Pocket switched networks and human mobility in conference environments. In: WDTN 2005: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking (2005)
5. LeBrun, J., Chuah, C., Ghosal, D., Zhang, M.: Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. In: Proceedings of the Vehicular Technology Conference, vol. 4, pp. 2289–2293 (2005)
6. Vahdat, A., Becker, D.: Epidemic Routing for Partially-connected Ad hoc Networks. Technical Report CS-2000-06, Duke University (2000)
7. Burgess, J., Gallagher, B., Jensen, D., Levine, B.N.: MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In: Proceedings of IEEE Infocom, pp. 1–11 (April 2006)
8. Tseng, Y.-C., Ni, S.-Y., Chen, Y.-S., Sheu, J.-P.: The broadcast storm problem in a mobile ad hoc network. Wireless Networks 8(2/3), 153–167 (2002)
9. Chen, X., Murphy, A.L.: Enabling disconnected transitive communication in mobile ad hoc networks. In: Proceedings of Workshop on Principles of Mobile Computing, Colocated with PODC 2001 (August 2001)
10. Lindgren, A., Doria, A., Schelen, O.: Probabilistic routing in intermittently connected networks. SIGMOBILE Mobile Comput. Commun. Rev. 7(3) (2003)
11. Spyropoulos, T., Psounis, K., Raghavendra, C.: Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In: WDTN 2005: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, pp. 252–259. ACM Press, New York (2005)
12. Wang, Y., Jain, S., Martonosi, M., Fall, K.: Erasure-coding based routing for opportunistic networks. In: WDTN 2005: Proceeding of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, pp. 229–236. ACM Press, New York (2005)
13. Chen, L.-J., Yu, C.-H., Sun, T., Chen, Y.-C., Chu, H.H.: A Hybrid Routing Approach for Opportunistic Networks. In: CNANTS 2006: Proceedings of the 2006 SIGCOMM Workshop on Challenged Networks (2006)
14. Jain, S., Demmer, M., Patra, R., Fall, K.: Using Redundancy to Cope with Failures in a Delay Tolerant Network. In: ACM SIGCOMM Workshop on Delay Tolerant Networks (2005)
15. Liao, Y., Tan, K., Zhang, Z., Gao, L.: Estimation based Erasure-Coding Routing in Delay Tolerant Networks. In: IWCMC 2006: Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing (2006)
16. Keranen, A., Ott, J., Karkkainen, T.: The ONE simulator for DTN protocol evaluation. In: Simutools 2009: Proceedings of the 2nd International Conference on Simulation Tools and Techniques (2009)