Plankton: An Efficient DTN Routing Algorithm

Xiang Fa Guo
School of Computing
National University of Singapore
Email: xiangfa@comp.nus.edu.sg

Mun Choon Chan
School of Computing
National University of Singapore
Email: chanmc@comp.nus.edu.sg

***Abstract*—In this paper, we present an efficient routing algorithm, *Plankton*, for Delay/Disruptive Tolerant Network (DTN). Plankton utilizes replica control to reduce overhead and contact probability estimates to improve performance. Plankton has two major features. First, it uses a combination of both short-term bursty contacts and long-term association based statistics for contact prediction. Second, it dynamically adjusts replication quotas based on estimated contact probabilities and delivery probabilities.**

Our evaluation on extensive traces shows that Plankton achieves significantly better prediction accuracy than existing algorithms for contact probability prediction. In addition, we show that while Plankton incurs much lower communication overhead compared to Spray-and-Wait, MaxProp and RAPID with savings from 14% to 88%, it can also achieve similar if not better delivery ratios and latencies.

***Index Terms*—delay/disruptive tolerant network (DTN) routing, contact prediction, replication control**

I. Introduction

Mobile nodes with short range wireless interfaces can communicate when they move sufficiently close to each other. While the mobility disrupts the formation of stable connections, it enables nodes to communicate via intermittent connections. Mobile nodes can communicate by carry-and-forward messages without end-to-end connections, which forms a delay/disruptive tolerant network (DTN) [1].

In the last decade since the work by Vahdat and Becker [2], the DTN research community has been focusing on designing routing algorithms that can deliver more messages in short latencies with constrained transmission and buffer capability. Two major approaches adopted in DTN routing algorithms are replica control and contact prediction.

As more message duplications cause more energy consumption, a number of DTN routing algorithms focus on limiting the number of message replicas without considering nodes' contact probabilities. Such algorithms include 'Spray and Wait' (S&W) [3] and many more ([4],[5],[6],[7]). Many routing algorithms do not limit messages' replicas. They instead prioritize messages based on contact probabilities. Some examples include PRoPHET [8], 'Bubble Rap' [9], MaxProp [10], RAPID [11] and many more ([12],[13],[14],[15],[16],[17],[18],[19]).

The two approaches indicate a fundamental trade-off between performance and overhead. Compared to algorithms that limit replicas, algorithms that utilize contact information and duplicate messages without limits tend to perform better but incur more overhead. It is thus natural to seek to combine replica control and the exploitation of contacts prediction. A number of DTN routing protocols have in fact attempted to perform this trade-off ([20],[21],[22],[17]).

In this paper, we present *Plankton*, an efficient DTN routing algorithm that achieves good performance with a novel replica control approach. Plankton is designed based on the idea that in order to achieve good performance, it is sufficient for a link to be classified into either *strong* one or *weak* one. The link strength in this work does not mean the transmission quality of a connection. It instead indicates the delivery probability of messages transmitted over a contact. A message sent via a strong link is likely more deliverable than it is sent via a weak link. Replica control varies with the ability to discover strong links. Once a replica of a message encounters strong links, its replica quotas is reduced substantially.

This paper makes the following contributions. (1) We propose an association based prediction and show that strong links can be identified through these association relationships that can be observed at different time scales. Our evaluation shows that the accuracy of our association based prediction is much higher than the contact predictions used in PRoPHET, MaxProp, RAPID and 'Bubble Rap'. (2) Based on the proposed contact prediction, we design Plankton that controls replicas by contact probabilities and message delivery probabilities. We compare Plankton with S&W, EBR [21], 'Bubble Rap', MaxProp, and RAPID on seven mobility traces. The evaluation results show that Plankton consistently generates significantly less number of replicas, but it achieves similar, if not better, delivery ratios and latencies.

The paper is organized as follows. Section II presents the evaluation of existing contact prediction algorithms. We then present Plankton prediction algorithm and its evaluation in Section III. The initial setting and adjustment of replica quota is described in Section IV and routing evaluation is discussed in Section V. Finally, we present related work in Section VI and summarize the work in Section VII.

II. Contact Prediction Evaluation

A. Evaluation of PRoPHET, MaxProp, and RAPID

First, we evaluate three representative contact prediction algorithms used by PRoPHET, MaxProp, and RAPID. They can be briefly described as follows. Let a node v's probability of encountering a node u be $p_{v,u}$.

- In *PRoPHET*, $p_{v,u}$ additively increases when v has a new contact with u, $p_{v,u} = p_{v,u(old)} + (1 - p_{v,u(old)}) \times p_{init}$,

978-1-4799-0469-3/13/$31.00 ©2013 IEEE

Trace	No. of Nodes	Interface/ Trans. range	Context
RollerNet[23]	62	bluetooth	outdoor rollerblading
Haggle IC06[24]	98	bluetooth	conference
Reality[25]	100	bluetooth	campus
SF taxi[26]	500	50m	city taxi
Seattle Bus[27]	1200	50m	city shuttle bus
RPGM[28]	100	10m	reference point group mobility
Random Way Point (RWP)	100	10m	random and free node movement

TABLE I: Summary of traces used in evaluation.

and multiplicatively decreases by the time since the last contact (k), $p_{v,u} = p_{v,u\,(old)} \times \gamma^k$. We set p_{init} to 0.75 and γ to 0.98 [8].

- In *MaxProp*, $p_{v,u}$'s initial value is zero. $p_{v,u}$ on v halves when v encounters a node other than u, and increases to $(1 + p_{v,u})/2$ when v encounters u.
- *RAPID* uses a prediction algorithm based on exponential inter-contact distribution. The probability u will encounter v in t is computed as $1 - e^{-\lambda t}$, where $1/\lambda$ is the average Inter Contact Length (ICL) between v and u.

When two nodes meet, after meta-data exchange, each node predicts the probabilities of encountering other nodes in the future. Such computations are performed for each contact. The total number of estimates can be up to the product of the number of contracts and the square of the number of nodes in the network.

Each estimated contact probability falls between zero and one. Let it be e_i. For each e_i estimated at time t, we can consult contact oracles whether the predicted contact actually occurs before the time $t + d$, where d is the time period of interest. The length of period of interest depends on messages and applications. Based on this answer, we form a tuple of $\langle e_i, a_i \rangle$, where a_i equals zero if the contact does not occur or equals one if it occurs.

These tuples are then placed into b bins by the value of e_i, and each bin is of size $1/b$. The tuples with e_i between $\frac{j}{b}$ and $\frac{j+1}{b}$ are placed into bin j, $0 \leq j < b$. For each bin j, we compute the accuracy A_j as $\frac{\sum_{e_i \text{ in bin } j} a_i}{\text{No. of elements in bin } j}$. We exclude estimates made during warm-up and cool-down periods. Also, if the number of elements in a bin is too small, say less than thirty, we do not consider the result of the bin.

Finally, we use the midpoint of the bin as the value of the estimate. Let this value be E_j. Hence, after the processing, we get a series of tuples, $\langle E_0, A_0 \rangle, \cdots, \langle E_{b-1}, A_{b-1} \rangle$. If the estimate can provide useful information to routing algorithms, we should expect the sequences $E_0, \cdots, E_{b-1}$ and $A_0, \cdots, A_{b-1}$ have strong correlation. This evaluation can also apply to the scheme where the predicted values are utilities instead of contact probabilities, e.g., MaxProp, and PRoPHET.

By the above approach, we only evaluate one-hop contact probability. One-hop probability evaluation is sufficient, since multi-hop probability is based on one-hop probability.

Trace	Exponential	PRoPHET	MaxProp	Plankton
RollerNet	-0.58	0.92	0.60	0.99
Haggle IC06	0.84	0.75	0.37	0.95
Reality	-0.43	-0.35	0.16	0.60
SF taxi	-0.26	0.86	0.68	0.99
Seattle Bus	-0.22	0.65	0.57	0.94
RPGM	-0.44	0.98	0.52	0.87
RWP	0.08	0.04	0.13	0.64

TABLE II: Correlations between tested predictions and the ideal prediction.

Contact predictions were evaluated on seven traces in Table I with different settings of d ($1800s$, $3600s$, and $7200s$) and b (10, 50, 100, and 200). As the results from different settings are similar, we will only show the results where d equals $3600s$ and b equals 100.

First, we present the results of different algorithms on a particular trace, namely the SF taxi trace. Figure 1a shows the plot of E_i versus A_i. The case for ideal prediction, where E_i equals A_i, is included for reference. The results show that all the three algorithms are fairly inaccurate. The Exponential algorithm is the most inaccurate. MaxProp and PRoPHET generate more accurate results than Exponential algorithm, but their predictions are still noisy.

To view the evaluation on different traces, we plot MaxProp's prediction on different traces in Figure 1b. It shows that E_i is often far away from the respective A_i on most traces.

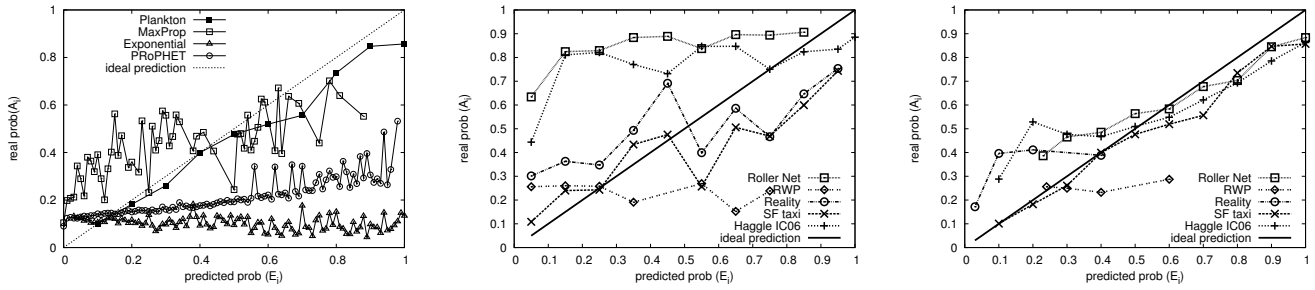
Due to space limits, we cannot show the detail results for all traces. Instead, we quantitatively measure prediction reliability by computing the Pearson correlation between the estimate values and the ideal prediction. A strong positive correlation indicates a reliable prediction. Table II lists the correlations from the three algorithms on all traces. We can make the following observations.

First, while correlations vary among different traces, these prediction algorithms are often unreliable. In many cases, the Exponential algorithm has negative correlations. The reliability of PRoPHET varies significantly from -0.35 (Reality) to 0.98 (RPGM). The results by MaxProp are better, but the correlations still vary from 0.16 (Reality) to 0.68 (SF taxi).

Second, the results indicate that while routing algorithms using these contact prediction schemes can achieve high message delivery ratios and short latencies, the good performance sounds unlikely due to accurate prediction on contact probabilities.

B. Evaluation of 'Bubble Rap'

'Bubble Rap' utilizes nodes' centrality for selecting relays, instead of pair-wise contact probability. In 'Bubble Rap', when u with message m encounters v without m, u either duplicates m to v (v is a relay) or does not duplicate m to v, depending on the result of centrality comparison and community information. For Bubble Rap, we record the duplication decision, and check whether the relay and destination actually meet within a given delay d by off-line contact oracle. A good estimator should make a larger proportion decisions where the relays do meet the destinations. For Plankton, we record the result when



(a) Accuracy of different prediction algorithms on the SF taxi trace.

(b) MaxProp predictions on different traces.

(c) Plankton prediction on different traces

Fig. 1: Prediction accuracy by different algorithms on different traces.

Algorithm	Roller	IC06	Reality	SFtaxi	SeattleBus	RPGM	RWP
'Bubble Rap'	0.24	0.19	0.18	0.12	0.02	0.20	0.25
Plankton	0.29	0.31	0.55	0.30	0.58	0.25	0.25

TABLE III: Contact prediction accuracy: Bubble Rap vs. Plankton.

a link is declared as a strong link and check whether the relay meet the destination by off-line contact oracle.

Table III shows the utility of the decision in terms of how often the selected relays meet destinations. It reveals that 2% to 25% replication decisions by 'Bubble Rap' result in one-hop delivery, with an average of 17% over all seven traces.

III. Plankton Contact Prediction

Plankton's contact prediction is based on two key ideas. First, links can be classified into either *strong* or *weak* link by how likely a message sent on this link can be delivered to the destination. Second, strong links can be identified through association relationships observed in different time scales.

In the rest of this section, we first discuss how weak and strong links are defined, then how strong links are discovered, and finally how these estimations are combined into a single contact prediction.

A. Weak Links (ρ)

The division between weak links and strong links is relatively. As the names suggest, a message sent via a weak link is less likely to be delivered to the destination than one sent on a strong link. Hence, the first task is to define what a weak link is. The approach taken by Plankton is to classify all links initially as weak links. A link may be reclassified as a strong link later using the association relationships to be defined below. Since all links are initially classified as weak, we approximate the delivery probability of a weak link as ρ, the average contact probability between any two nodes within a time window (win). A good value for win closely depends on remaining lifetime of a message and applications. With the assumption that contacts are independent and identically distributed (i.i.d.), ρ can be approximated as follows:

$$\rho = \frac{\text{avg. encountered nodes within } win \text{ by a node}}{\text{total No. of nodes } - 1}. \quad (1)$$

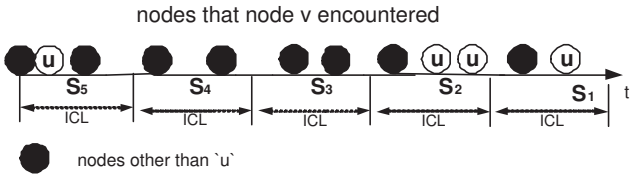


Fig. 2: The prediction by bursty contacts.

The assumption of i.i.d. does not tightly fits real world contacts, as some nodes have more contacts than others and a node encounters some nodes more frequently while it encounters others less fluently. However, with the assumption, the probability ρ can provide a gross remark on different contact chances of different traces. For example, a node in the RollerNet trace can encounter 20% of all nodes within 900s, while a node in the Seattle bus trace can only encounter 1% of all nodes within 900s. The value of ρ can be estimated online or computed off-line through historical information.

B. Prediction by Recent Busty Contacts ($p^b_{v,u}$)

Bursty and self-similar events have been well investigated and exploited in many fields, such as web caching and internet traffic modeling. In real-world mobility pattern, once two nodes are in the same proximity, they have a good chance of being close to each other in the near term. These could be because these nodes are two persons in the same seminar room attending a talk, two passengers on the same subway train, or two vehicles traveling along a road in the same direction. In DTN routing, such bursty contacts have been exploited in last contact based routing [13][20], but they only consider the most recent contact.

Plankton however exploit a number of most recent intervals previous to contact prediction time. Let $p^b_{v,u}$ denote the contact probability between v and u based on recent contacts. To compute $p^b_{v,u}$, v checks its contacts in the most recent n time intervals, where each interval has a duration of average pairwise inter-contact length (ICL). If v encounters u in n_u out of the n intervals, $p^b_{v,u}$ equals n_u/n.

Figure 2 illustrates an example, where n is set as five. The probability $p^b_{v,u}$ computed by v equals $3/5$, as v encounters u in three out of the five intervals.

Symbol	Interpretation
ρ	pairwise average delivery probability
$p_{v,u}$	the contact probability between v and u
$p^a_{v,u}$	the contact probability between v and u based on associated contacts
$p^b_{v,u}$	the contact probability between v and u based on bursty contacts
d^l_m	local delivery probability for m
d^g_m	global (maximum of known local)delivery probability for m
n_m	m's destination node

TABLE IV: Summary of symbols.

The probability $p^b_{v,u}$ provides an estimation based on short-term contact features, and its implementation is straightforward. We use average pairwise ICL, instead of an absolute time length because ICL is adaptable to different traces. ICL is easy to compute by a node with local contact information without causing any overhead on metadata exchange. A parameter Plankton uses is n. The settings of n have little effect on performance as long as it is not too big (larger than twenty) or too small (less than five). We set it to ten in our evaluation.

C. Prediction by Indirect Associations ($p^a_{v,u}$)

The intuition behind prediction by indirect association is as follows. If whenever v encounters w, it often encounters u, then it is likely that once v meets w, the probability of meeting u is high. Such association captures indirect relationship among nodes that occurs over longer time scale. It exploits community relationships among nodes without the need to perform community clustering or search for nodes with high centrality as is done in 'Bubble Rap'.

We write the contact probability between v and u through the indirect association with w as $p^a_{v,w,u}$, which can be computed by the following steps. The node v first locates the recent n non-overlapping intervals with the length of ICL that start with an encounter with w. Next, v counts the intervals that also contain at least one encounters with u and let this number be n_u. $p^a_{v,w,u}$ is computed as n_u/n. Since the association can happen through different nodes, $p^a_{v,u}$ is taken as the maximum value over all indirect associations through the nodes that v recent encounters. The node w is v's recent encounter if the time since the last contact between w and v is less than ICL, which uses the same length as the interval for checking indirect association. Indirect association probability can be computed as follows:

$$p^a_{v,u} = \max_i \{p^a_{v,i,u}\} \ (i \text{ is } v\text{'s recent encounter}).$$

Figure 3 illustrates an example, where n equals five. In the past five non-overlapping intervals starting with an encounter with w, v encounters u in four intervals. Thus, $p^a_{v,w,u}$ equals $4/5$.

By taking the maximum value over all possible associations, we follow the idea taken in popular universal prediction algorithm [29] and mobile nodes location prediction [30], which looks for the longest subsequence.

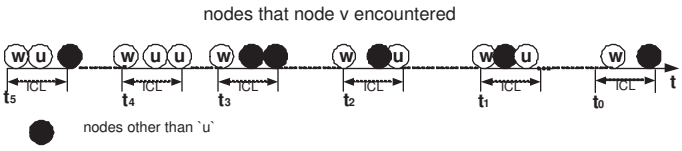


Fig. 3: The prediction by associated contacts.

D. Combining Different Predictions

Based on the previous discussion, v has three estimates for the contact probability between v and u, based on the three predictions used, namely ρ, $p^b_{v,u}$ and $p^a_{v,u}$. Given that $p^b_{v,u}$ and $p^a_{v,u}$ are meant to be the stronger links, they are only useful or meaningful when their values are larger than ρ.

Next, while it is possible to combine the values of $p^b_{v,u}$ and $p^a_{v,u}$ by assuming that they are independent, we choose not to take this assumption for the following reasons. First, the two estimations are for the same pair of nodes. We do not find reason to believe that the estimates are independent. Second, we prefer conservative estimates due to our approach to contact probability based dynamic quota adjustment, as we will explain later. The quota adjustment can tolerate false negative to some extent, but we need low false positive for high contact probability. We thus compute the final contact probability between v and u, $p_{v,u}$, as:

$$p_{v,u} = \max\{\rho, p^b_{v,u}, p^a_{v,u}\}. \quad (2)$$

E. More Discussion on Contact Estimators

These three contact estimators are complementary. The first estimator, ρ, explores the contacts that are difficult to predict, e.g., the contacts having few past contacts of respective node pair to exploit. While these contacts are 'unreliable', they are plentiful. For example, more than 80% of all contacts happen between node pair that have two or less contacts in one day in SF taxi trace. Many existing algorithms either simply ignore such contacts by setting the delivery probability to zero or incorrectly estimate the probability because of the lack of past contact information. These contacts however can significantly contribute to messages dissemination [31] as they are abundant. By exploiting average contact chances by ρ, Plankton can efficiently utilize these contacts without historical contact information of specific nodes pairs.

The other two estimators capture different features of contact patterns provided past contacts provided sufficient contacts information is available. Recent contacts based prediction capture burstiness of contacts by investigating short term contacts, while indirect association based prediction captures contact association by investigating long term contacts. The two estimators exploit different association behaviors and can cover different set of nodes.

F. Evaluation

We evaluate the accuracy of Plankton prediction and compare it to other contact estimator presented in Section II. Results in Figure 1a and Figure 1c show that Plankton can achieve much more accurate predictions than the other

three estimators. From Table II, we can see that Plankton outperforms the other three estimators on all traces. It achieves correlation equal to or larger than 0.87 on all traces except Reality (0.60) and RWP (0.64). Even for these two traces, Plankton achieves much higher correlations than the other estimators.

As 'Bubble Rap' does not make contact prediction but instead make binary decision on whether a message should be forwarded to another node, we compare the utility of the decision of 'Bubble Rap' and Plankton based on the ratio of message delivery for every message forwarded. Table III compares the utilities of the forwarding decision in Bubble Rap and Plankton. It shows that the utility of the Plankton is always better or equal to 'Bubble Rap'. The improvement varies from 20% for RollerNet to 2800% for the Seattle Bus trace except for RWP where there is no difference, .

IV. Plankton

A. Overview

To deliver a message m, the source node first assigns m an initial replica quota by ρ and target delivery probability. As m is duplicated, the quota is divided among the replicas. A novel feature of Plankton is that it decreases the replica quota when m is duplicated to a node that has high contact probability or the replicas of m have generated sufficiently high delivery probability.

B. Initial Quota Setting

The initial replica quota $\mathcal{Q}$ defines a message's maximum delivery probability. When $\mathcal{Q}$ is fully utilized, i.e., $\mathcal{Q}$ replicas are transferred onto $\mathcal{Q}$ relays, the message gains maximum delivery probability. In order to accurately set $\mathcal{Q}$, the source needs to know which nodes will be the relays and their contact probabilities to the destination. This however is generally infeasible in DTNs. Plankton estimates $\mathcal{Q}$ using the delivery probability of weak link ρ.

Suppose that $\mathcal{X}$ replicas of m are duplicated onto $\mathcal{X}$ relays $(x_1, x_2, \cdots, x_{\mathcal{X}})$, whose delivery probabilities (p_{x_i,n_m}) may be smaller or larger than ρ. By the definition of ρ in Equation 1, the following approximation holds when $\mathcal{X}$ is large:

$$\sum_{i=1}^{\mathcal{X}} p_{x_i,n_m} \approx \mathcal{X} \times \rho \tag{3}$$

The contact probabilities of p_{x_i,n_m} are for different node pairs. We simply assume that they are independent. We use $\mathcal{P}$ to symbolize the delivery probability when m is duplicated to $x_1, x_2, \cdots, x_{\mathcal{X}}$. By Equation 3, we can get the following inequality:

$$\mathcal{P} = 1 - \prod_{i=1}^{\mathcal{X}} \left(1 - p_{x_i,n_m}\right) \quad \geq 1 - (1-\rho)^{\mathcal{X}}$$

By above inequality, we can get $\mathcal{X} \leq \log(1-\mathcal{P})/\log(1-\rho)$. Since the actual quota $\mathcal{Q}$ cannot be larger than $\mathcal{N}$, the number of total nodes, we can get:

$$\mathcal{Q} = \min\{\log(1-\mathcal{P})/\log(1-\rho), \mathcal{N}\}. \tag{4}$$

Note that $\mathcal{Q}$ is a conservative approximation (larger than necessary) without the information of online contact probability when replicas are generated. Thus, replica quotas are accordingly reduced once a node duplicates m to a relay that has high delivery probability, which we discuss as follows.

C. Dynamic Quota Adjustment Based on Contact Probability

Plankton defines two types of link, strong links and weak links. A node pair (u, v) has a *strong link* if $p_{u,v}$ is larger than ρ. Otherwise, it is a *weak link*. The typical scenario of checking strong links is as follows. The node u has a message for the destination v. The node u encounters w, and then u checks whether w has a strong link with v or not.

The judgement behind the division of strong links and weak links is the reliability of contact prediction. Labeling two nodes with weak links does not mean their contacts is useless in message forwarding, but it indicates that it is difficult to reliably predict their contacts. Another important implication is that two weak links may have minor difference in contact chances, but it is difficult and likely unworthy to differentiate them. Strong links have opposite properties. Their contacts can be reliably predicted and two strong links can have significant different contact chances, which is knowable and worthy to know. Thus, weak links and strong links have different contributions to routing performance. Message duplications over weak links help to seek strong links. Large number of replicas duplicated via weak links themselves certainly can generate considerably large chances for message delivery. Strong links mainly contribute to direct deliver messages. Therefore, Plankton wields weak links and strong links by different policies as follows.

Suppose that node v with m encounters u. If u and n_m have a weak link, v performs duplication procedure similar to binary split S&W. If u and n_m have a strong link, v always duplicates m to u when v's quota for m is larger than zero. v adjusts the quota by computing the number of weak links that equal the strong link between v and n_m, measured by the chance of delivering m. Let h be the number of weak links. We can get $p_{u,n_m} = 1 - (1-\rho)^h$, and $h = \log(1 - p_{u,n_m})/\log(1-\rho)$. Let q denote m's quota on v. v has a quota of one and u has a quota of q', as follows:

$$q' = \max(1, q - h) \tag{5}$$

The replica quota adjustment and send-list generation is listed in algorithm 1. We would like to highlight the follows. First, the above quota adjustment scheme requires low false positive in contact prediction. This explains our preference for conservative contact probability estimates by Equation 2. Second, the number of strong links do not have to be large. A small number of strong links is sufficient.

D. Dynamic Quota Adjustment based on Delivery Probability

1) Efficient Delivery Probability Estimate: Besides reducing replica quotas through strong links, Plankton stops duplicating replicas for a message when it detects that replicas

generated are sufficient to meet the achievable delivery probability adapted to constrained DTN resources.

Delivery probability of m is computed based on the contact probability on all relays of m. Plankton does not utilize multiple-hop delivery probability for three reasons. (1) DTNs are small world network [32][9], which means messages can be delivered in a small number of hops. (2) Our measurements and evaluations have indicated that multiple-hop delivery probabilities are often highly inaccurate. (3) Multiple-hop delivery probability estimation requires expensive exchange pairwise contact probabilities. Thus, we compute delivery probability from multiple relays by direct delivery. This may underestimate the true delivery probability by ignoring multiple-hop delivery chances, but we can have a conservative estimate from which we can safely reduce quota. Also, note that while the multi-hop delivery probability is not utilized, Plankton allows multiple-hop transmission.

The local delivery probability of a message m computed by a node u is:

$$d_m^l = 1 - \prod_{(v \in V)} \big(1 - p_{v,n_m}\big),$$

where V is the set of nodes that u has encountered and known to have buffered m.

Since local delivery information of nodes having small number of neighbors is limited, each node also records the maximum local delivery probability known so far, which we name as d_m^g. Plankton uses d_m^g as m's delivery probability. While d_m^g may be not accurate global delivery probability, the use of d_m^g as delivery probability has the advantages that d_m^g can propagate quickly in the network with a very small amount of meta-data exchange.

2) Estimating Achievable Delivery Ratio: Replica control by online feedbacks can quickly adapt to traffic and connections change. The key idea is that if the network is under utilized, we can aim for higher delivery ratio and performs more replication. However, if the network is congested, then set a lower threshold and replicates less. Otherwise, with too many replicates, performance can be worse.

Plankton uses online feedbacks to compute the achievable delivery ratio, p_{ach}. On receiving the acknowledgement for m for the first time, the receiver memorizes its d_m^g. The node averages all known d_m^gs of delivered messages as p_{ach}.

A node simple sets the quota for a message m as *zero* after it knows d_m^g is larger than p_{ach}, and it never duplicates m to any other nodes except the destination.

Sometimes, the target delivery ratio cannot be met. This can be due to node specific behavior (e.g., node is isolated) or network congestions which would make the increase in replication a bad option. A node running Plankton computes the achievable delivery probability

Input: v's buffer list
Output: sorted send-list for v: $list$
foreach m *in* u*'s buffer, not in* v*'s buffer* **do**
  if $v = n_m$ **then**
    $gain(m) \leftarrow 1$, add m to $list$
  else if $m's\ quota > 0$ **then**
    if $p_{v,n_m} \leq \rho$ *and* $m's\ quota > 1$ **then**
      $gain(m) \leftarrow (1 - d_m^g) \times p_{v,n_m}$ add m to $list$, evenly split m's quota
    end
    if $p_{v,n_m} > \rho$ **then**
      $gain(m) \leftarrow (1 - d_m^g) \times p_{v,n_m}$
      add m to $list$, compute quota by Eqn. 5
    end
  end
end
decreasingly sort $list$ by $gain(m)$

Algorithm 1: Node u generates duplication list for v.

V. Performance Evaluation

We evaluate Plankton with an event-based simulator adopted from ONE [33]. We simulate a wide range of buffer and bandwidth settings on extensive traces to ensure that the simulation results are more realistic [34]. Message lifetime is set to 1,000s for the Roller net trace, and 20,000s for the Reality trace, and 7,200s for all other traces. Message sizes are uniformly distributed between 1KB and 100KB. Warm-up and cool-down periods are set and the metadata communication overhead are simulated.

We compare Plankton with MaxProp, RAPID, S&W, EBR, and 'Bubble Rap'. For fair comparison, acknowledgement packets are flooded for all algorithms since MaxProp and RAPID use acknowledgements. The quota for S&W and EBR is set to be the same as the initial quota used by Plankton.

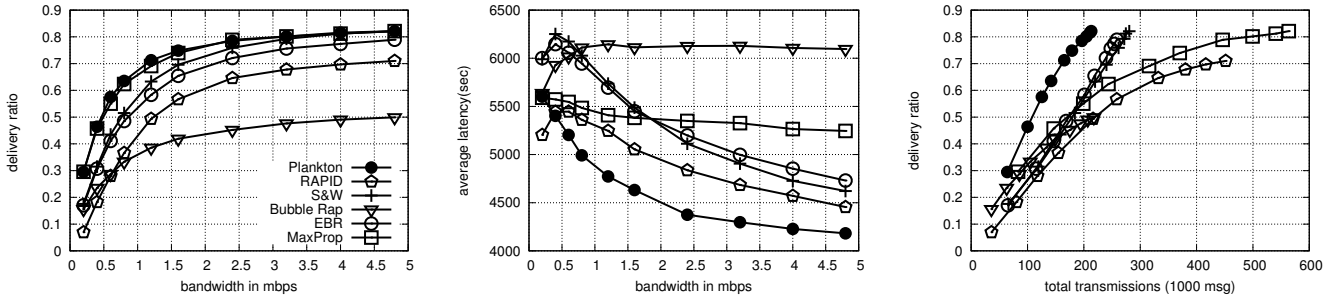
DTN routing performance is often measured by delivery ratio (DR) and average latency. DR is the ratio of delivered messages to unique messages to be delivered. Average latency is the average latencies of delivered messages. We measure delivery overhead by dividing the number of replicas generated for all messages by the number of delivered messages.

A. Evaluation on the SF taxi trace

1) Performance of Varying Bandwidth and Buffer: Figure 4 shows the results when bandwidth is varied. Figures 4a and 4b show that Plankton achieves much higher DR (up to 80% higher) and lower latencies (up to 70% lower). Plankton achieves similar DR as MaxProp but with much lower latencies and less overhead (saving up to 60%).

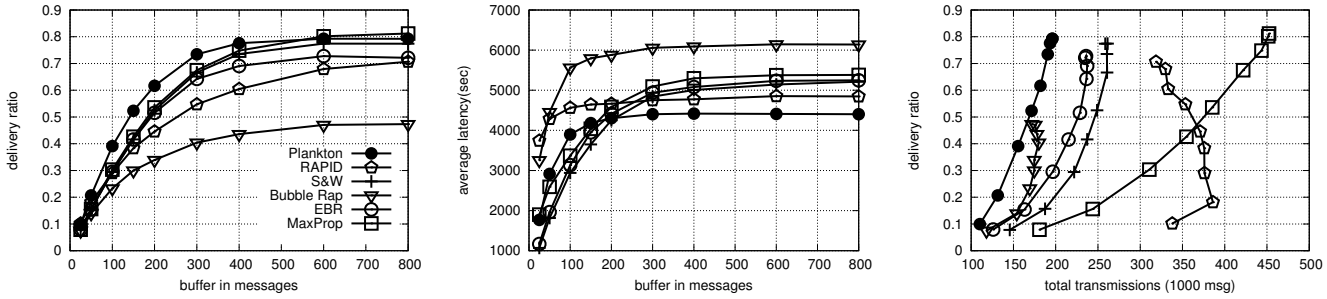
Figure 5 shows the results when buffer size is varied. Figure 5a shows that Plankton achieves higher DRs than all other algorithms. Figure 5b indicates that when buffer size is more than 200 messages, Plankton achieves the lowest latencies among all algorithms. When the buffer size is less than 200, Plankton delivers significantly more messages with slightly higher latencies compare to EBR, S&W, and MaxProp.

2) Overhead Comparison: Figures 4c and 5c compare the delivery overhead when buffer or bandwidth is varied. For the same delivery ratio, Plankton incurs the lowest delivery overhead. Plankton can save up to 60% replicas compared to



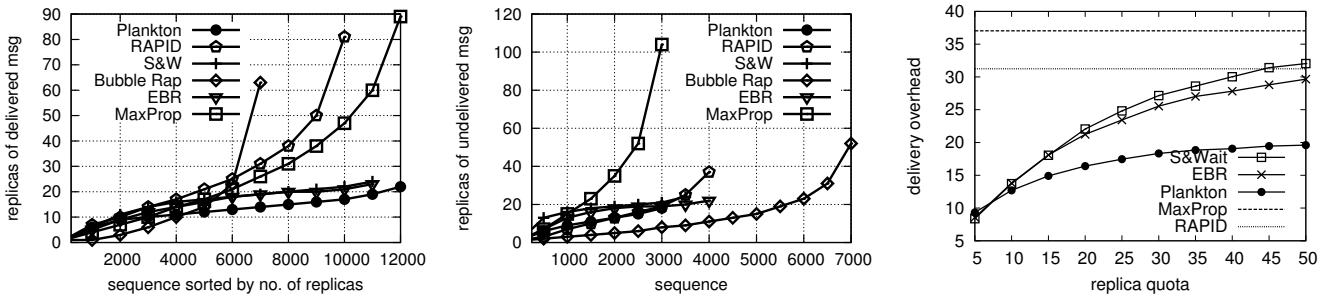
(a) Bandwidth vs. delivery ratio. (b) Bandwidth vs. latency. (c) Transmissions vs. delivery ratio.

Fig. 4: Routing performance for varying bandwidth.



(a) Buffer size vs. delivery ratio. (b) Buffer size vs. transmission latency. (c) Transmissions vs. delivery ratio.

Fig. 5: Routing performance for varying buffer size.



(a) Replicas of delivered messages. (b) Replicas of un-delivered messages. (c) Quota vs. delivery overhead.

Fig. 6: Routing overhead comparison with bandwidth of 2.4MBps and the buffer size of 500 messages.

MaxProp and RAPID that do not limit on replicas. Meanwhile, Plankton can also achieve savings (up to 40%) over efficiency oriented algorithms, such as EBR and S&W, which limit replicas. The results from Figures 4c and 5c show that the Plankton can efficiently control replicas so as to achieve good DR while using the least number of replicas. The result of delivery latency is similar and is not discussed here.

We also examine the distributions of the number of replicas generated for both delivered and undelivered messages. Figure 6a shows that MaxProp and Rapid generate many more replicas than Plankton for delivered messages, thus consuming more resources. These are the overhead Plankton attempts to save. Similarly, Figure 6b shows that 1/3 of undelivered messages are really difficult to deliver, as MaxProp fails to deliver them after generating replicas onto more than 10% (40 out of 400 nodes) of total nodes. One possible reason is that the destinations are kind of isolated. Plankton saves resources in these cases by avoiding generating too many replicas.

Plankton, S&W, and EBR use quotas to control message replicas. Their efficiency varies with the quota used. We examine their energy efficiencies by varying quota. Results in Figure 6c show that Plankton incurs significantly less overhead for all quota settings evaluated. We can see that Plankton generates the smallest increase in replicas among the three algorithms when the quotas increase. This demonstrates the effectiveness of the dynamical quota adjustment in Plankton.

The saving on replicas closely relates with the encountered strong links. Figure 7 shows that the number of strong links

Trace/ ρ /$\mathcal{Q}$	Algorithms	DR	Delivery overhead	Latency (sec)
RollerNet/ 0.23/ 9	*Bubble Rap*	0.66	9.82	516
	Plankton	**0.90**	**10.34**	**291**
	EBR	0.83	12.33	432
	S&W	0.86	12.70	387
	MaxProp	0.88	13.93	322
	RAPID	0.90	15.19	314
Haggle IC06/ 0.1/ 22	*Bubble Rap*	0.49	13.61	3890
	Plankton	**0.73**	**17.25**	**1639**
	S&W	0.73	22.58	1800
	EBR	0.74	17.85	1840
	MaxProp	0.76	37.53	1491
	RAPID	0.76	40.64	1571
Reality/ 0.03/ 75	*Bubble Rap*	0.34	7.07	9052
	Plankton	**0.56**	**14.01**	**7815**
	S&W	0.52	16.11	8538
	EBR	0.52	18.57	8621
	MaxProp	0.55	23.95	8793
	RAPID	0.54	28.15	8569
SF taxi/ 0.1/ 20	**Plankton**	**0.79**	**16.00**	**3939**
	S&W	0.76	21.19	4625
	EBR	0.71	24.77	5473
	Bubble Rap	0.45	24.80	6126
	RAPID	0.65	31.88	4451
	MaxProp	0.79	36.22	4818
SeattleBus/ 0.04 / 55	**Plankton**	**0.90**	**39.74**	**2955**
	EBR	0.89	51.61	3043
	S&W	0.90	53.44	2919
	Bubble	0.76	100.16	3898
	MaxProp	0.94	331.15	2759
	RAPID	0.90	341.50	2611
RPGM/ 0.22/ 9	**Plankton**	**0.81**	**14.09**	**1709**
	EBR	0.72	17.86	2413
	Bubble Rap	0.22	19.02	2383
	S&W	0.73	19.68	2218
	MaxProp	0.82	48.72	2122
	RAPID	0.85	53.92	2035
RWP/ 0.24/ 8	**Plankton**	**0.85**	**8.56**	**2155**
	EBR	0.69	9.91	2845
	Bubble Rap	0.42	10.64	2944
	S&W	0.80	12.61	2788
	RAPID	0.82	14.18	2034
	MaxProp	0.80	19.24	2608

TABLE V: Performance comparison on different traces.

encountered over different time periods. Over a duration of 1800s, 55% of the messages encounter at least one strong link. For a longer duration of 7200s, about 70% of the messages encounter at least one strong link. Encounters with strong links allow Plankton to dramatically reduce the replica quota with minimum impact on performance.

B. Results for Different Traces

We present the results on seven traces in Table V. ρ is computed using Equation 1, and $\mathcal{Q}$ is computed via Equation 4 with $\mathcal{P}$ set to 0.9. Bandwidth is set to 2.4Mbps and buffer size is 500 messages. We compare Plankton to five other algorithms. For each mobility trace, we rank the algorithms in ascending order of delivery overhead, with the algorithm that incurs the least overhead listed first.

Plankton incurs the lowest communication overhead per delivered message in four of the seven traces, namely: SF taxi, Seattle Bus, RPGM, and RWP with saving from 14% to 88%. For the other three traces, Plankton occurs larger overhead

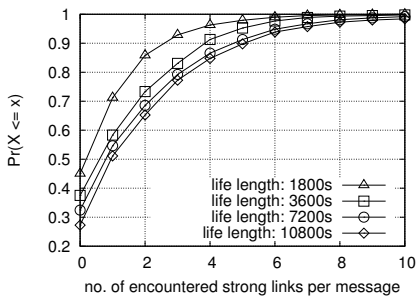


Fig. 7: The No. of encountered strong links in SF Taxi Trace

than only Bubble Rap, but has significantly higher DRs and shorter latencies.

In all mobility traces, Plankton has the highest or close to the highest DR among all algorithms evaluated. For Haggle trace, DR of Plankton is 73% and DR of RAPID is 76%. The 3% loss in delivery ratio is compensated by 57% gains in delivery overhead. The result is similar for latency. Plankton has the lowest latency in four out of seven traces.

We have two interesting observations. (1) MaxProp transmits more replicas and performance better than RAPID. One reason is that MaxProp generates much less metadata overhead than RAPID. (2) Performance of Bubble Rap is not as good as other algorithms. Our observation suggests two reasons. (1) Bubble Rap renders too few replicas as it is very selective in choosing relays, which lowers delivery overhead but loses in the DR and latency; (2) Bubble Rap selects relays by 'centrality' that reflects nodes' dissemination capability. Such a scheme can cause congestions at the central nodes.

VI. Related Work

DTN routing can be roughly divided into three types: (a) non-prediction based routings, (b) prediction based local optimal routings, and (c) prediction based global optimal routings.

Early efforts mainly focused on (a), which limits the number of replicas without investigating the delivery probabilities. These work include (1) the schemes that use the fixed number of replicates, e.g., 'random single copy' [4] and S&W [3]; (2) the schemes that approximately control replicates by the number of nodes that the source node can encounter, such as 'two hop routing' [5] 'gossip based routing' [6]; (3) the schemes that balance the number of replicas of different messages, e.g., SCP [7]. These work simply control resource usage without exploring contact probabilities.

One weakness of these approaches is the rigid control on the number of message replicas. DTNs however may have very different transmission and buffer capability, traffic loads. Some messages may require fewer replicas to deliver while others may require many more replicas. Approaches using a fixed quota for all messages all the time fail to be aware of these diversities. Nevertheless, by placing a limit on resource usage, these approaches tend to be more resource efficient.

More recent works in (b) facilitate message exchanges by predicting future contacts from past contacts. They include (1)

last contact based routing [12][13] [20], (2) multiple history contact based routing [8][10][16] [14][21][17], and (3) social network based routing [15][9][35][19].

DTN routing algorithms in (c) include RAPID [11] and Max-Contribution [18]. A node computes optimal transmission utilities by collecting the global information on replicas distribution and pairwise delivery probabilities.

A common weakness of routing schemes in (b) and (c) is that message replicas can be many more than necessary. In addition, for algorithms like RAPID that operate on 'global' information, the meta-data of 'who has what' and 'who encounters whom' is substantial.

'multi-phase routing'[17] explicitly divides replicas generation process into different phases, and the process stops once the generated replicas can likely deliver the message within deadline. The feedback can add in delays to replica quota adjustment. 'Retiring replicants' [22] controls traffic congestion by node-based replicas management, without the information on forwarding decision.

Plankton controls replicas in two novel ways. First, its contact prediction algorithm is novel and we have shown that it gains much more accurate contact prediction than existing algorithms. Next, Plankton dynamically adjusts the replica quota needed based on contact probability and delivery probability, which is not done by any other DTN algorithms.

VII. Conclusion

Our analysis and simulation results suggest that efficient routing schemes for DTNs can be achieved through controlling message replicas based on reliable contact predictions. Besides this main findings, we have two interesting observations that might be helpful for DTN research community. First, many existing contact estimators are not accurate or reliable enough. Second, our performance evaluation shows that substantial overhead reduction can be achieved without loss in delivery ratios and latencies. Our work provides a technique that integrates highly reliable contact predictions and replica controls. We believe that our work can arouse more research interesting in contact prediction quality and its utilities in designing practical DTN routing algorithms.

Acknowledgements

This work was partially supported by the NRF Singapore through the FUM SMART (R-252-002-430-592) program.

References

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *SIGCOMM '03*. ACM.

[2] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Duke University, Tech. Rep., 2000.

[3] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *WDTN '05*. ACM.

[4] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Single-copy routing in intermittently connected mobile networks," in *SECON '04*.

[5] M. Grossglauser and D. N. C. Tse, "Mobility increases the capacity of ad hoc wireless networks," *IEEE/ACM Trans. Netw.*, vol. 10, 2002.

[6] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE/ACM Trans. Netw.*, 2006.

[7] B. D. Walker, J. K. Glenn, and T. C. Clancy, "Analysis of simple counting protocols for delay-tolerant networks," in *CHANTS '07*. ACM.

[8] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *SIGMOBILE*, 2003.

[9] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, 2011.

[10] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks," *INFOCOM '06*.

[11] A. Balasubramanian, B. Levine, and A. Venkataramani, "Replication routing in dtns: A resource allocation approach," *Networking, IEEE/ACM Transactions on*, 2010.

[12] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, "Age matters: efficient route discovery in mobile ad hoc networks using encounter ages," in *MobiHoc '03*. ACM.

[13] M. Grossglauser and M. Vetterli, "Locating nodes with ease: last encounter routing in ad hoc networks through mobility diffusion," in *INFOCOM '03*.

[14] Q. Yuan, I. Cardei, and J. Wu, "Predict and relay: an efficient routing in disruption-tolerant networks," in *MobiHoc '09*. ACM.

[15] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *MobiHoc '07*. ACM.

[16] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding," in *MobiHoc '08*. ACM.

[17] E. Bulut, Z. Wang, and B. Szymanski, "Cost-effective multiperiod spraying for routing in delay-tolerant networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 5, pp. 1530–1543, 2010.

[18] K. Lee, Y. Yi, J. Jeong, H. Won, I. Rhee, and S. Chong, "Max-contribution: On optimal resource allocation in delay tolerant networks," in *INFOCOM'10*.

[19] J. Wu and Y. Wang, "Social feature-based multi-path routing in delay tolerant networks," in *INFOCOM '12*.

[20] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," in *PerCom Workshops '07*.

[21] S. Nelson, M. Bakht, and R. Kravets, "Encounter-based routing in dtns," in *INFOCOM'09*. IEEE.

[22] N. Thompson, S. Nelson, M. Bakht, T. Abdelzaher, and R. Kravets, "Retiring replicants: congestion control for intermittently-connected networks," in *INFOCOM'10*. IEEE.

[23] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. D. de Amorim, and J. Whitbeck, "The accordion phenomenon: Analysis, characterization, and impact on dtn routing," in *INFOCOM'09*.

[24] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD trace cambridge/haggle/imote/infocom2006."

[25] N. Eagle and A. S. Pentland, "CRAWDAD trace mit/reality/blueaware/devicespan."

[26] M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, "A Parsimonious Model of Mobile Partitioned Networks with Clustering," in *COMSNETS '09*.

[27] J. G. Jetcheva, Y.-C. Hu, S. PalChaudhuri, A. K. Saha, and D. B. Johnson, "CRAWDAD data set rice/"ad hoc city" (v. 2003-09-11)."

[28] X. Hong, M. Gerla, G. Pei, and C.-C. Chiang, "A group mobility model for ad hoc wireless networks," in *MSWiM '99*. ACM.

[29] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *Information Theory, IEEE Transactions on*, 1977.

[30] A. Bhattacharya and S. K. Das, "Lezi-update: an information-theoretic approach to track mobile users in pcs networks," in *MobiCom'99*. ACM.

[31] P. Csermely, *Weak Links: The universal key to the stability of networks and complex systems*. Springer Verlag, 2009.

[32] A. Chaintreau, A. Mtibaa, L. Massoulie, and C. Diot, "The diameter of opportunistic mobile networks," in *CoNEXT'07*. ACM.

[33] A. Keränen, J. Ott, and T. Kärkkäinen, "The one simulator for dtn protocol evaluation," in *Simutools '09*. ICST.

[34] N. Ristanovic, G. Theodorakopoulos, and J.-Y. Le Boudec, "Traps and Pitfalls of Using Contact Traces in Performance Studies of Opportunistic Networks," in *INFOCOM'12*.

[35] E. Bulut and B. Szymanski, "Friendship based routing in delay tolerant mobile social networks," in *GLOBECOM'10*. IEEE.