Distributed expectation-based spatio-temporal cluster detection for pocket switched networks

Matthew Orlinski and Nick Filer
School of Computer Science
University of Manchester
Manchester, M13 9PL, UK
Email: orlinskm,nick@cs.man.ac.uk

***Abstract*—Distributed cluster detection has been widely applied to the problem of efficient data delivery in Pocket Switched Networks (PSNs). However, previously proposed methods can become less efficient over long periods of time as more and more transient, pair-wise connections are used to form clusters which monotonically increase in size. Monotonically increasing clusters, also known as aggregated clusters, suffer an inevitable loss of temporal information. Thus cluster relevance to individual movement patterns at particular times is diminished. This paper applies expectation-based spatio-temporal clustering to the problem of data delivery in PSNs. We will show that spatio-temporal clusters can be detected distributively and can improve the data delivery efficiency by 7% and deliver 6% more packets compared to aggregated methods. Furthermore, 3 data delivery strategies which utilise spatio-temporal clusters are proposed.**

***Index Terms*—Delay Tolerant Networks, Pocket Switched Networks, Distributed cluster detection, Spatio-temporal clustering**

I. Introduction

Mobile Ad-hoc Networks (MANETs) are created from mobile electronic devices forming ad hoc connections between themselves using wireless links. MANETs have been a popular research topic since the 1990s. Consequently there is no exhaustive list of applications for MANETS, nor is there one specific configuration of devices and mobility which is prevalent in the field of MANET research.

Recent research into one particular type of MANET looks at the opportunistic exchange of data during encounters between personal mobile devices. This is sometimes called Pocket Switched Networking (PSN) [1], [2]. As well as providing inter-human data dissemination, PSNs may also include "gateways" to external services such as the Internet.

Reality Mining [3] experimental data can be used to infer the prevalence of PSNs in the real world. In Reality Mining experiments, data is collected on human movement and encounter patterns by devices belonging to or given to participants to use in an unsupervised manner. As encounters between participants are recorded, studies can be undertaken on new techniques for data dissemination in PSNs.

In this paper we have constructed PSNs from empirical data on inter-human encounters belonging to the Infocom, Cambridge [4] and Reality [5] data-sets shown in Table I. We have used these data-sets to conduct data delivery experiments between virtual mobile devices using The One Simulator [6].

TABLE I
Comparison of some Reality Mining data-sets. τB is the inter-probe time of Bluetooth scans in seconds.

	Infocom5	Infocom6	Cambridge	Reality
Duration (Days)	3	3	12	246
Environment	Conference		Campus	
Mobile Devices	41	78	36	97
Device Type	iMote	iMote	iMote	Phone
Number of Connections	22459	128979	10641	102594
Daily Encounter Probability	0.78	0.73	0.24	0.01
τB	120	120	600	300
Location trace	No	No	No	Cell ID

By limiting the size of the spanning tree needed to reach a destination, distributed cluster detection techniques [7]–[10] can result in data delivery in PSNs being more efficient in terms of the number of duplicate packets needed to deliver a message.

However, the traditional definition of a cluster used in distributed clustering only allows for monotonically increasing cluster sizes over time. As the algorithms run, they aggregate many distinct encounter patterns from different times of the day into aggregate clusters. As a result, data delivery efficiencies using aggregated clusters can suffer efficiency losses as obsolete cluster memberships persist. Using the Quality [10] distributed aggregate clustering and data delivery method, Fig. 1 illustrates the increased cost of delivering a message in terms of duplicate packets as cluster size increases.

The transient encounters between devices in PSNs can be represented as vertices and edges respectively in dynamic contact graphs [11]. If data is aggregated into a single graph, then there is an ever increasing likelihood that there will be an edge between any two vertices. When aggregated graphs grow they may grow to a Densification Power Law (DPL) exponent [12], where the number of edges over time raised to some power α is proportional to the number of vertices, $e(t) \propto v(t)^a$. The higher the DPL exponent α is, the faster additional edges are added to vertices in the graph. Fig. 2 illustrates the densification rate of the reality mining data-sets under consideration in this paper. We found that all of the data-sets densify at a fairly consistent rate (DPL exponentα of around 1.25) until 75% of the devices have been added to the graph. This conclusion may be helpful when modelling

978-1-4673-4404-3/12/$31.00 ©2012 IEEE

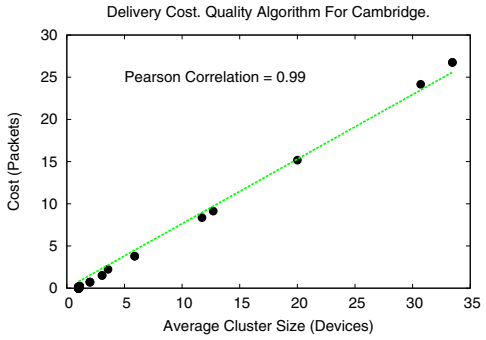

Fig. 1. Cost of data delivery in duplicate packets as a function of cluster size for the Quality algorithm in the Cambridge data-set.

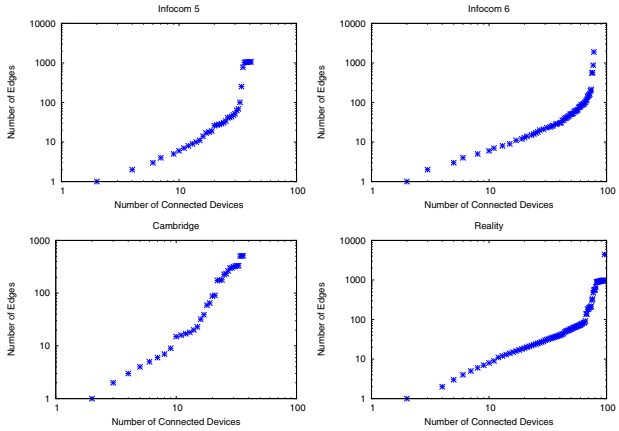

Fig. 2. Densification of PSNs over time.

human behaviour in the future. In the Infocom data-sets α is 1.27, and in the Cambridge and Reality data-sets α is 1.25 and 1.24 respectively.

A. Related work

Early epidemic label propagation methods [8] for distributed aggregate clustering suffered the *monster cluster* problem where a single cluster evolves to dominate the entire network. SHARC [9] prevents monster clusters but suffers from the *wandering cluster* problem in dynamic networks caused when a large number of devices and propagate their cluster labels elsewhere.

The Bubble clustering and data delivery algorithm for dynamic networks [13] is a hybrid routing protocol in that it combines aggregate cluster detection and centrality measurements. In Bubble, data is passed to devices with higher global centrality values until it reaches a cluster containing the destination node, at which point local centrality rankings within clusters take over. Local rankings within clusters are calculated using a temporal mechanism Cumulative Window (C-Window) which recalculates centrality every 6 hours.

Experiments in how distributed aggregate clustering methods partition a network [10] showed that the larger the clusters in device memory, called the local clusters, the more suitable they are for data delivery. With cluster size of 30% of the total data-set size being adequate to replicate the data delivery rates of Epidemic at a lower duplicate packet cost.

Work by Borgia et al [14] proposed a temporal adaptation to the Simple distributed clustering mechanism [7] used in Bubble. Their proposal called AD-Simple, relies on pruning clusters of obsolete members using a timer which counts down from the moment devices are entered into local clusters. AD-Simple successfully maintains *home* clusters, whilst removing information on obsolete or rarely visited clusters.

II. Distributed Expectation-Based Spatio-Temporal Clustering

Distributed Expectation-Based Spatio-Temporal Clustering (DEBT) will address concerns around monster and wandering clusters whilst allowing home cluster membership to vary over time.

The restrictions imposed on DEBT are similar to that of other distributed clustering mechanisms such as Bubble and Quality. Firstly, devices can only communicate with other nearby devices during pair-wise, ad-hoc wireless connections. Secondly, devices each keep a record of their own local cluster. Lastly, a calculation is performed on connected devices to decide if they should include each other in their local clusters. In this section, how clusters are created and maintained in DEBT is addressed. Data delivery mechanisms are considered a separate problem and explored later in Section III.

Where DEBT differs from aggregate clustering methods, is that each device splits the passage of time into discrete time frames of length t seconds in order to compare communication metrics between frames. Based on the comparison between frames, devices can then decide to add or remove devices from their local clusters. An illustration of how the data-sets look when using this stratified sampling is given by plotting the total connection times within hourly time frames in Fig. 3. We found that all of the data-sets have long periods of inactivity. Moreover, in the conference data-sets between 42-48% of hour long frames have a higher connection time than previous frames. In the campus and city wide environments values are similarly grouped, with between 36-37% of hourly frames having a higher connection time than previous frames.

A. Local spatio-temporal cluster building

The first step in calculating expectation-based spatio-temporal clusters is the calculation of the expected connection baseline. Cumulative connection duration within frames is used as this is easy to measure and allows us to detect the best connected pairs of devices despite connections being frequently disrupted [15].

When calculating the baseline, one could choose to *ignore*, *stratify* or *adjust* for the seasonal effects which can give rise to repeating patterns in the data. The purpose of DEBT is to detect any encounters between devices which are out of the

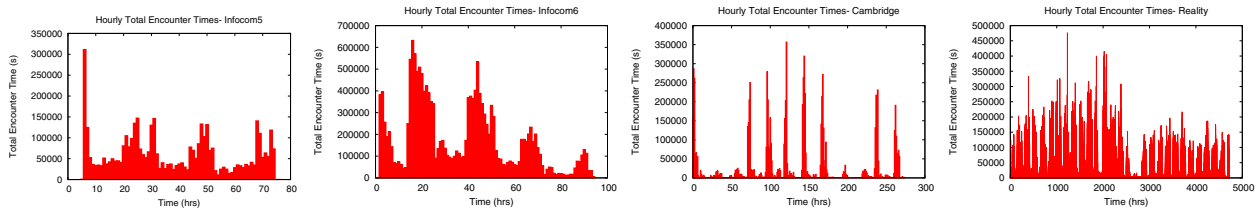

Fig. 3. Hourly total contact times in the Infocom5, Infocom6, Cambridge and Reality data-sets.

ordinary compared to seasonal trends. Therefore, the specific time series analysis method used to compute the baseline is the mean of means from the last w frames.

The Baseline is renewed by each device at the end of each time frame. To do this, a device d_i calculates the mean connection duration, $a_i^{f_n}$, from all connected devices at the end of time frame f_n. The mean of the results from current and previous $w-1$ time frames is then taken to produce the baseline for d_i, referred to as $b_i^{f_n}$. This process can be summarised in Equation 1.

$$b_i^{f_n} = \frac{a_i^{f_n} + a_i^{f_{n-1}} + \cdots + a_i^{f_{n-(w-1)}}}{w} \quad (1)$$

Once the expected baseline has been calculated at the end of frame f_n, it is compared with cumulative connection times for individual devices within frame f_n. For d_i to consider d_j for inclusion in d_i's local cluster, the cumulative connection time x between d_i and d_j within time frame f_n (denoted $x_{d_i d_j}^{f_n}$) should be higher than a coefficient, C_{up} multiplied by the baseline for d_i as shown in Equation 2. E.g. if the new baseline on device d_i is 20, and the cumulative connection time between d_i and d_j in time frame f_n was 30. Then with $C_{up} = 1$, Equation 2 is true and d_j will be added to d_i's local cluster. However, if $C_{up} = 2$, then Equation 2 is false and d_j will not be included. So that devices can be added to local clusters at any point during a time frame, DEBT will also perform the comparison in Equation 2 with the existing baseline as cumulative connection times are increased.

$$x_{d_i d_j}^{f_n} > (b_i^{f_n} * C_{up}) \quad (2)$$

The final stage of the cluster building process is choosing what data the devices should exchange. In DEBT, each device keeps a local cluster table which contains devices it's currently got in it's local cluster called the *neighbour* column, and a copy of each neighbour's local cluster table stored in the *branch* column. When devices meet they exchange their local cluster tables, and once Equation 2 has been evaluated to true for a remote device, the information can be entered into the local cluster table on the local device.

With neighbour and branch data stored in local cluster tables, each device can identify clusters which may be multiple hops in size. However, as ad-hoc connections are transient, connections to all of the devices in the local cluster table can not be guaranteed, but connections to neighbours can be assumed to be more stable than those indicated by the branch data. Furthermore, devices in the branch column are not unique, a device can belong to many branch fields, a property which will later be used in Section III-B to detect paths which may contain routing loops.

TABLE II
DEPT PARAMETERS CHOSEN BY THE USER.

Parameter	Description
t	Length of time frames in seconds.
w	Number of previous frames used in baseline calculation.
C_{up}	Neighbour promotion coefficient.
C_{down}	Neighbour demotion coefficient.

B. Cluster maintenance

Without a mechanism to remove devices from local clusters, DEBT would suffer from the monotonically increasing cluster sizes seen by aggregated methods. Devices could adopt a large number of policies when deciding to remove others from local cluster tables. Some of the issues to be aware of in DEBT are:

1) Removing devices at the end of frames as soon as cumulative connection times fall below the baseline keeps local clusters relevant to the previous time frame but means that cluster membership may not accurately reflect periods longer than the time frame length.
2) Not removing devices often enough may cause large message delivery overheads as messages are exchanged between devices who are no longer spatially related.
3) Due to the possible long wait between encounters, devices should be able to act independently in order to delete neighbours.

To guard against these concerns, devices who previously had each other in their local clusters but have failed the test in Equation 3 are not deleted immediately but are marked for deletion at the end of the next frame. Devices who are marked for deletion are only removed from the local cluster table, and their corresponding branch data deleted if Equation 2 is not satisfied by the end of the next frame.

$$x_{d_i d_j}^{f_n} \leq (b_i^{f_n} * C_{down}) \quad (3)$$

C. Spatio-temporal cluster analysis

Having a mechanism to delete devices from local cluster tables prevents the monster and wandering clusters seen in

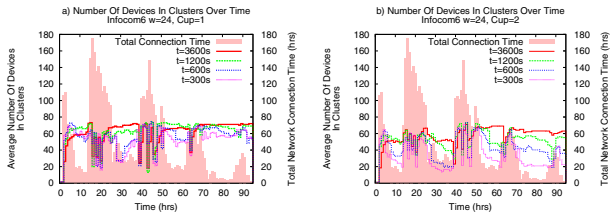

Fig. 4. Distributed cluster timings for Infocom6.

epidemic label propagation methods. Also with DEBT, the monotonic increase in cluster sizes seen in [7], [9], [10] are not an issue if reasonable coefficient values are chosen.

The coefficients which govern how much greater or lower metrics should be than the baselines before considering devices for inclusion or deletion from local clusters are called C_{up} and C_{down} respectively. These coefficients govern how quickly clusters are built up and destroyed. For example, when C_{up} is set to 1, clusters are built up from devices whose connection times exceed the baseline multiplied by 1. In the case of the Infocom6 data-set, a C_{up} of 1 is too lenient. In other words as soon as devices are removed they are likely to be re-added to local clusters in the next frame. Setting C_{up} to 2 instead creates cluster sizes which respond to the temporal connectivity metrics as the difference between Fig. 4a and Fig. 4b shows.

Identifying suitable time frame length is critical to matching the rate of change in network structure [16]. It will be useful to present the level of change between time frames of different lengths, but due to lack of space, only t values of 300, 600, 1200, and 3600 s are shown in Fig 4. The effect of frame sizes effects the ability of expectation-based spatio-temporal clustering to track changes in the data. In most cases, when frame size is large, the ability of DEBT to react to changing behaviours is diminished as there is less metric differences between large frames. However, in a sparse data-set such as Reality, short frame sizes of 300 s don't make much sense as every connection becomes significant. Therefore for Reality, frames 1 hour in length are used in the data delivery results detailed in the next section. For all the other data-sets, DEBT uses a frame size of 300 s.

III. Multi-hop Data Routing

Besides Bubble, few data delivery methods utilise distributed cluster detection to provide a balance between data delivery success and efficiency in dynamic networks. A large proportion of packets can be delivered using intra-cluster duplication if local clusters are large enough [10], but this can create large numbers of redundant data packets. In this section, three new data delivery methods for use with DEBT are evaluated.

The transfer and storage of remote local clusters as branch data acts as a mechanism for local clusters to be merged and inter-cluster path information to be preserved. This allows for inter-cluster delivery as in Fig. 5. In this example, a message

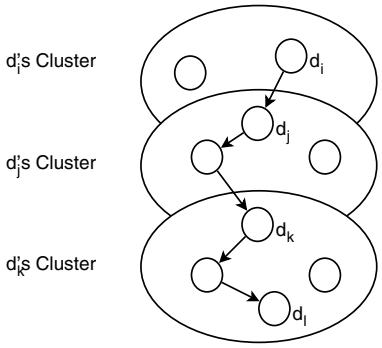

Fig. 5. Inter-cluster data delivery overview.

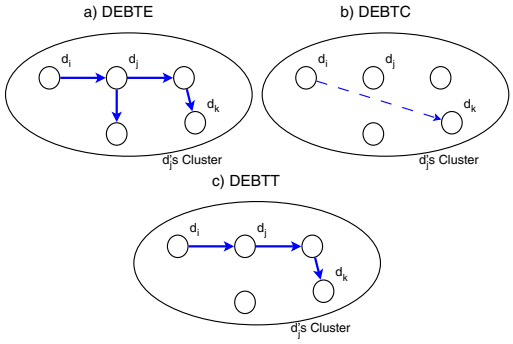

Fig. 6. Example intra-cluster delivery scenarios. Generally DEBTE will try to flood clusters with duplicate packets, where DEBTC attempts intra-cluster delivery by waiting for an encounter between message carrier and the destination.

is generated by d_i to be delivered to d_l. The message can be passed to d_j if d_j knows of d_l via branch data in it's local cluster table.

Our work has resulted in three suggested data delivery strategies to provide inter and intra-cluster routing via the local cluster tables. The first two protocols DEPT Epidemic (DEBTE) and DEBT Clustering (DEBTC) look at the branch data within the local cluster tables as non-hierarchical sets. The third method called DEBT Trees (DEBTT), preserves path information within the branch data as trees and offers the ability to spot routing loops at the expense of some additional processing. Here are some brief observations which were used to inform our choices for our data delivery methods but are not fully elaborated on due to space.

1) Most of the duplicate messages using DEBT are caused by inter-cluster data duplication. An issue Bubble addresses with its pre-cluster "bubbling" using centrality.
2) If intra-cluster message duplication is stopped at devices which have branches that contain the transmitting device and the destination, then data delivery success rates suffer.

A. Epidemic based delivery (DEBTE)

DEBTE makes use of the neighbour and branch data to forward data as given in the example in Fig. 6a. Simply put, if the message destination is anywhere in an encountered devices' local cluster table then the message is copied to the encountered device.

B. Cluster based delivery (DEBTC)

DEBTC makes more conservative decisions regarding the data delivery compared with DEBTE. If a message can be delivered via an encountered device, but the encountered device also has the transmitting device in the same branch field as the destination, then the message will not be forwarded unless there is another route which does not contain the transmitting device.

In the example illustrated in Fig. 6b, the packet may not be passed between d_i and d_j if either: d_j got it's only information about the destination from d_i. Or all of the branches on d_j which contain the destination, also contain d_i.

C. Tree based delivery (DEBTT)

The aim of DEBTT is to address the case from the example of DEBTC where there may be a path to the destination contained within the branch data, but a message is not transfered because it may contain a routing loop. Routing loops can not be identified in the branch data if the information is processed as a non-hierarchical set of other devices, so DEBTT constructs a tree in the branch field by preserving who received local cluster tables from whom and by sharing this information using a tree data structure.

IV. DEBTE vs. DEBTC vs. DEBTT

Fig. 7 shows that the data delivery results of all of the methods using DEBT. The general pattern is that DEBTE will produce many duplicate packets in an attempt to flood clusters until a message reaches it's destination; DEBTC will not flood clusters and wait for an intra-cluster encounter between devices; DEBTT falls in between DEBTE and DEBTC in terms of message delivery success ratio.

It may be reasonable to suggest that DEBTT might deliver the same number of packets as DEBTE with less duplication (see Fig. 6c). However this is not the case in temporal contact graphs as there is no guarantee that pair-wise connections will be established between neighbours in the order needed to deliver all packets. As a result DEBTT delivered 10% fewer packets than DEBTE overall.

V. Results

In this section the data delivery methods introduced will be compared against 2 aggregate clustering delivery methods, Bubble [13] and Quality [10]. For Bubble, both the K-clique and Simple [7] clustering techniques were used as a basis for the aggregate clustering. In total, 5 random message generation patterns for each data-set/protocol combination were used to send small (1KB) messages between random devices with a

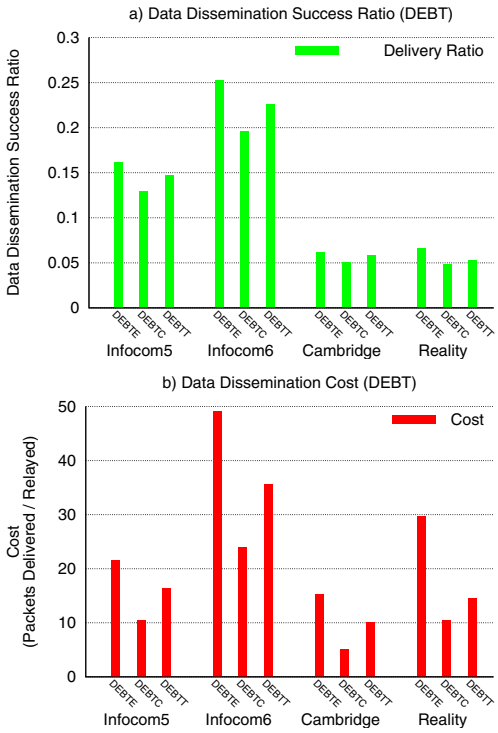

Fig. 7. Distributed clusters data delivery rates and cost for DEBT data delivery methods.

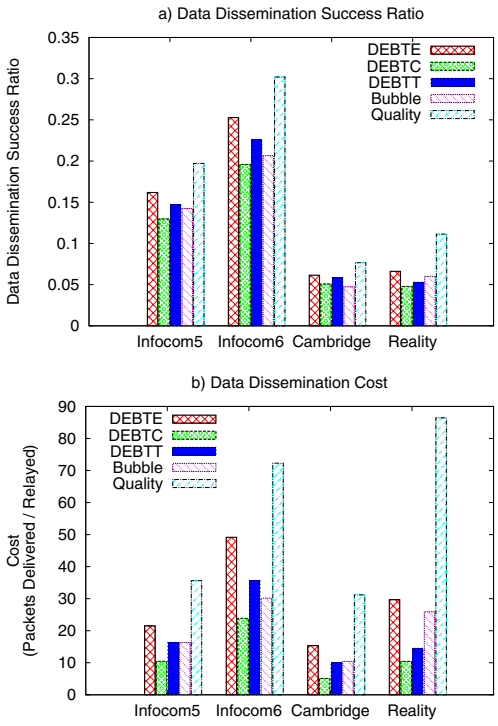

Fig. 8. Data delivery rates compared.

TABLE III
FINAL AVERAGE DATA DELIVERY RESULTS ACROSS ALL EXPERIMENTS.

Method	Data Delivery	Overheads
Bubble K-Clique	0.1141	20.6972
Quality	0.1717	56.3885
DEBTE	0.1356	28.9228
DEBTC	0.106	12.4691
DEBTT	0.1213	19.1464

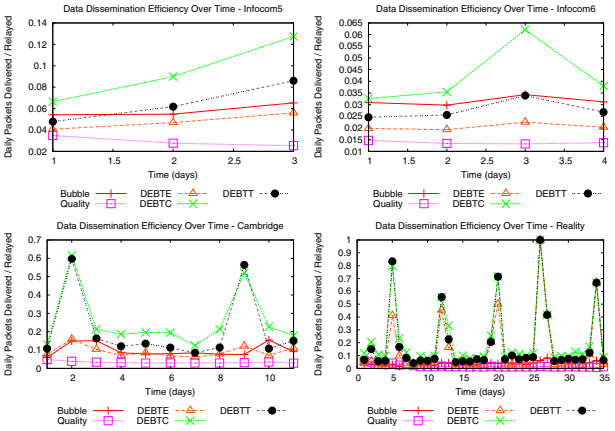

Fig. 9. Data delivery efficiency over time.

time to live of 1 hour increased to 1 day in the Reality data-set due to the sparsity of connections.

The results presented in Fig. 8a show that the data delivery success ratios of Quality tend to be higher than the other methods. This is expected as Quality effectively floods networks with duplicate packets after constructing very large clusters.

The results from each experiment are collated into Table III. Overheads were calculated as the number of messages relayed, minus number of messages delivered, divided by the number of messages delivered. From these results we see that compared to Quality, the data delivery success ratio of DEBTE is 21% lower, but for half the cost. However, compared to Bubble, the cost of the DEBTT algorithm is 7% lower with 6% more packets delivered.

Finally, Fig. 9 shows the efficiency of all the protocols over time. Efficiency in these charts is calculated daily by dividing the number of delivered packets by the number of relayed packets. Measured in this way the effectiveness of Quality is very poor, and the spikes in the charts show that the DEBT protocols are reacting to times in the data-sets where new devices cluster.

VI. CONCLUSIONS

We have shown that spatio-temporal clusters can be detected using distributed algorithms, and that by choosing the method carefully, data delivery efficiency can be improved when doing so. The choice of time frame size, introduced in Fig. 4 was not fully explored in this paper due to lack of space. In summary it was found that frame sizes t of 3600 s for Reality and 300 s for the other data-sets not only gave clusters which closely represent the pattern of the data, but also gave efficient data delivery results. An expectation-based approach for PSNs would not work with frame sizes longer than a day as human movement is diurnal [17] and cumulative connection times would not differ greatly between frames. This shows the importance of calculating adequate frame size to the performance of the DEBT protocols.

In future work it may be possible to detect spatio-temporal clusters without artificially splitting time into discrete frames. It may also be possible to predict future cluster formations in order to further improve data delivery success without sacrificing efficiency.

We believe there is no panacea for all conceivable PSN use cases, and that more effort will be needed in the future from academics and engineers to specify and design for more tightly defined scenarios. However, as with Bubble we have presented a general solution and methods which can be tweaked and applied to more specific uses cases as they arise.

REFERENCES

[1] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, and C. Diot, "Pocket switched networking: Challenges, feasibility and implementation issues," *Autonomic Communication*, pp. 1–12, 2006.

[2] A. Lindgren, C. Diot, and J. Scott, "Impact of communication infrastructure on forwarding in pocket switched networks," in *SIGCOMM*, 2006, pp. 261–268.

[3] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," *Personal and Ubiquitous Computing*, pp. 255–268, 2006.

[4] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, *CRAWDAD Trace*, Jan. 2006.

[5] N. Eagle and A. Pentland, *CRAWDAD data set mit/reality (v. 2005-07-01)*, Jul. 2005.

[6] A. Keranen, J. Ott, and T. Karkkainen, "The ONE simulator for DTN protocol evaluation," in *SIMUTools*, 2009, p. 55.

[7] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *MobiArch*, 2007, p. 7.

[8] I. X. Y. Leung, P. Hui, P. Lio, and J. Crowcroft, "Towards real-time community detection in large networks," *Physical Review E*, vol. 79, no. 6, 2009.

[9] G. Herbiet and P. Bouvry, "SHARC: community-based partitioning for mobile ad hoc networks using neighborhood similarity," in *WoWMoM*, Jun. 2010, pp. 1–9.

[10] M. Orlinski and N. Filer, "Quality distributed community formation for data delivery in pocket switched networks," in *SIMPLEX*, 2012.

[11] A. Panisson, A. Barrat, C. Cattuto, W. Broeck, G. Ruffo, and R. Schifanella, "On the dynamics of human proximity for data diffusion in ad-hoc networks," *CoRR*, 2011.

[12] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *SIGKDD*. ACM, 2005, pp. 177–187.

[13] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE rap: Social-based forwarding in delay tolerant networks," *Mobile Computing*, vol. 6, no. 1, pp. 1576–1589, Jan. 2007.

[14] E. Borgia, M. Conti, and A. Passarella, "Autonomic detection of dynamic social communities in opportunistic networks," in *Med-Hoc-Net*, 2011, pp. 142–149.

[15] S. Grasic, E. Davies, A. Lindgren, and A. Doria, "The evolution of a DTN routing protocol-PRoPHETv2," in *Proceedings of the 6th ACM workshop on Challenged networks*, 2011, pp. 27–30.

[16] R. Sulo, T. Berger-Wolf, and R. Grossman, "Meaningful selection of temporal resolution for dynamic networks," in *MLG*, 2010, pp. 127–136.

[17] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," *Computer Networks*, vol. 52, no. 14, pp. 2690–2712, 2008.