Practical Routing Protocol for Impromptu Mobile Social Networks

Xiao Chen[1], Zhen Jiang [2], Kaiqi Xiong[3], Jian Shen[4]
[1]Department of Computer Science, Texas State University, San Marcos, TX, USA
[2]Department of Computer Science, West Chester University of Pennsylvania, West Chester, PA, USA
[3]College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, NY, USA
[4]Department of Mathematics, Texas State University, San Marcos, TX, USA
Email: xc10@txstate.edu, zjiang@wcupa.edu, kxxics@rit.edu, js48@txstate.edu

***Abstract*—With the popularity of mobile devices, mobile social networks (MSNs) formed by people carrying mobile devices moving around and contacting each other have become a hot research topic these days. In this paper, we study a specific kind of MSNs that is formed impromptu (e.g. when people carrying mobile devices gather at some social events). We refer to them as *Impromptu Mobile Social Networks* (IMSNs). Due to the dynamic nature of such networks, routing poses special challenges. The existing social-based MSN routing algorithms that take advantage of stable social relationships or social features of people in the network may not be suitable for IMSNs. Thus, new routing algorithms that can catch node contact behavior need to be designed for IMSNs. We first propose two statistical-based theoretical routing algorithms named BerRout and PoiRout inspired by the node contact models in several papers and then put forward a practical routing algorithm UpDown which makes routing decisions based on a simple *Counter* capturing the ups and downs of people's relationships formed in an IMSN. We compare our algorithms with the existing social-based MSN routing algorithms by simulations. The results show that the practical algorithm performs close to the two theoretical ones and all of our proposed algorithms outperform the existing ones in terms of performance versus cost in an IMSN environment.**

***Index Terms*—mobile social networks, routing, social analysis, social features, social network graph**

I. Introduction

Due to the rapid development of mobile devices such as smartphones, laptops, and PDAs, *mobile social networks* (MSNs) [15] formed by people carrying mobile devices moving around and contacting each other have become a hot research topic these days. In this paper, we study a specific kind of MSNs that are formed impromptu, which we refer to as *Impromptu Mobile Social Networks* (IMSNs). The IMSNs are formed in situations, for example, when people carrying mobile devices attend a professional conference, event or festival. The IMSNs provide lightweight communication mechanisms that use contact opportunities to allow people to communicate via local wireless bandwidth such as Bluetooth without a network infrastructure. The links in IMSNs are time-dependent and short-term. Thus continuous network connectivity is not guaranteed, which poses special challenges to routing algorithms in IMSNs. Therefore, nodes in IMSNs communicate through a *store-forward* fashion. When two nodes move within each other's transmission range, they *contact* each other and become *neighbors*. When they move out of their ranges, their contact is lost. The message to be delivered need to be stored in the local buffer until a contact occurs in the next hop.

In the literature, several routing protocols have been developed for MSNs [2], [8], [12], [19], [20]. A rudimentary yet costly routing approach in MSNs is to perform a flooding-based route discovery [19] where a message is spread like an epidemic of a disease. To reduce cost and improve routing efficiency, the fundamental idea for a message holder to choose the next best forwarder in its local neighborhood is to predict the future meeting probabilities of the candidate nodes with the destination using the current information known to the message holder. The node that has the highest probability to meet the destination will be selected to forward the message. With the popularity of social networks such as Facebook [4] and LinkedIn [11], researchers find that the meetings of people in daily lives exhibit small world phenomenon [13] and the social relationships of individuals are likely to be long-term and less volatile than node mobility. Therefore, some researchers utilize node connections in social network graphs created by linking nodes with past encounters as the current information and apply social analysis methods to evaluate nodes' *centrality* to predict nodes' future meeting probabilities [2], [8]. The social network graphs can show whether two nodes have met in the past but not the frequency of the meetings. To facilitate discussion, we refer to the routing algorithm that uses this approach as the *social-analysis-based* algorithm. Some other researchers use people's social features in user profiles as the current information and explore the similarity of these social features to predict nodes' future meeting probabilities [12], [20]. Social features can refer to an individual's social attributes such as *nationality, language, affiliation, position, city, and country, etc*. The intuition of this approach is that people having more similar social features tend to meet more often. We refer to the routing algorithm based on this approach as the *social-feature-based* algorithm.

Though the social-analysis-based and the social-feature-based methods work well in their own MSN scenarios, they may not be applicable to IMSNs. The social-analysis-based approach relies on the stability of long-term social relationships of individuals in daily lives. But in IMSNs, the links between

nodes are established during the time frame of a conference or other social gathering, which makes node connections to be short-term and dynamic. Similarly, the social-feature-based method assumes the stability of node social features and the consistency of user behavior and their social features. It may not be a good fit for IMSNs either, because someone who puts *New York* as his *State* in his profile may actually attend a conference in Texas. His social connections in New York may not be very helpful in making routing decisions in an IMSN formed at a Texas conference. Besides, not everyone participating in an event likes to answer their social feature questions in their profiles. Therefore, we need to design new routing algorithms specifically for IMSNs.

In this paper, we explore new routing algorithms for IMSNs in two steps. First, inspired by the assumptions in several papers [5], [9], [10] which, in their respectively applications, assume the meetings of distinct node pairs are independent and follow a Poisson process in the MSNs which, we think, resemble the IMSNs we discuss in this paper. Using these theoretical models of how people contact as the current information, we make our first attempt to put forward two statistical-based routing algorithms called *BerRout* based on the assumption that node meetings are independent and *PoiRout* based on the assumption that node meetings follow a Poisson process in IMSNs. We also prove theoretically that these two algorithms are related by the Poisson limit theorem [1] when the time interval is large and the node meeting probability during the time interval is small.

Next, we propose a practical routing algorithm that makes routing decisions based on the new information that can capture the encounters of people as the IMSN is being formed. As pointed out by [7], it is potentially possible to deliver information relying only on encounters. Therefore, the new information should be obtained solely from the encounters of people in the current IMSN and not the nodes' previous connections and social features. Furthermore, it should be simple for the sake of the distributed and dynamic environment. It should not only reflect the meetings of people but also the meeting frequency. And it should be time-related so that it can catch people's dynamic relationships. By considering all of these, we propose a practical routing algorithm called *UpDown* that uses a $Counter$ which captures the *growing* and *decaying* phases of people's relationships in the IMSN as the current information to make routing decisions. More specifically, our UpDown algorithm maintains an array of $Counters$ for each node that records the node's meetings with other nodes. The encounter of two nodes in a time period will increment their counters by a certain amount and no encounter in a time period will decrement their counters by some amount. The amount to increase or decrease reflects the growing or decaying of a relationship. When making a routing decision, the message holder will select a neighbor who has the largest $Counter$ with the destination. Despite the simplicity of the UpDown algorithm, it captures the natural social behavior of people: if people's relationships grow, they tend to meet more and all that is recorded by the increment of the $Counters$. If people's relationships decay, they will meet less and all that is recorded by the decrement of the $Counters$. The $Counter$ contains the meeting frequency of nodes which is missing in the social network graphs of the social-analysis-based algorithm and does not have the inconsistency problem as in the social-feature-based algorithm because the $Counter$ reflects people's actual meeting behavior.

To evaluate the performance of our UpDown algorithm, we compare it with the theoretical algorithms BerRout and PoiRout, the existing social-analysis-based and social-feature-based algorithms, and with Flooding using a real conference trace - Infocom06 [16] which represents an IMSN formed at the Infocom 2006 conference. Simulation results show that our proposed algorithms can achieve comparable delivery rates as the optimal case in Flooding but at a much lower cost. The practical algorithm UpDown is close to the two theoretical ones, which allows us to discuss its properties theoretically through the other two if the direct analysis is difficult. All of our proposed algorithms, namely BerRout, PoiRout and UpDown, outperform the existing social-based algorithms in terms of performance versus cost in an IMSN environment.

The rest of the paper is organized as follows: Section II references the related works; Section III presents our routing algorithms; Section IV compares our algorithms with the existing ones using simulations; and conclusion is drawn and future works are mentioned in Section V.

II. Related works

In this section, we introduce the related routing protocols for MSNs in the literature. These routing protocols, suitable for their MSN scenarios, use different predictors for a message holder to locally identify a forwarder which is most likely to deliver a message to the destination based on the different information currently available to the message holder.

A. Flooding-based algorithms

A rudimentary routing approach in MSNs is to perform a flooding-based route discovery as in [19] where the spread of the message is like the epidemic of a disease. The flooding-based routing and its derivatives use multiple copies of a single message to find independent paths to the destination so as to improve routing efficiency and robustness. However, flooding has nonneglectable drawbacks [17]: it consumes a high amount of bandwidth and energy; may result in poor performance because of high contention for shared resources. As the average node degree increases, it is not scalable in memory size needed and number of transmissions performed.

B. Social-based algorithms

As social network applications explode in recent years, social network graphs that link nodes with past encounters can be created and analysis of these graphs shows that some nodes are the common acquaintances of other nodes and act as communication hubs [14], [18]. Therefore, one promising way of predicting future contact probability is to analyze these networks and use metrics such as node *centrality* to

make forward decisions [2]. To accurately calculate a node's centrality in a graph, all nodes are required to have a global view of the social network, which is difficult and costly to obtain in a distributed mobile environment. To avoid a global view, researchers resort to the local view to estimate a node's centrality in a graph [5]. Then the routing decision of a message holder in a social-analysis-based method is made based on the centralities of the nodes connecting to it in the social network graph. The neighbor that has the highest centrality will be chosen. The social-analysis-based method relies on the social network graph which can represent node past encounters, but the degree of node meeting frequency cannot be represented. As pointed out by [6], [21], the "static" social graph has the tradeoff between time-related information lost and predictive capability. Therefore, the social-analysis-based method is applicable to MSNs where the social relationships of individuals are likely to be long-term and less volatile than node mobility.

Some other works utilize people's social features in their user profiles as the current information and compare node social similarities based on these social features to predict nodes' future contact probabilities with the destination to guide routing [12], [20]. Social features can be a person's *nationality, language, affiliation, city*, *state,* etc. Researchers find that individuals with similar social features tend to meet more often in MSNs. The initial idea of social-feature-based routing is proposed in [12]. Then the authors in [20] propose a novel multi-path routing based on the structural property of hypercubes where each forwarding is guided by reducing social feature differences between the source and the destination. In the more detailed implementation of [20], a social feature vector is attached to each node representing in which social features it has the same values as D ("1"s are put in the places of these social features) and in which social features it has different values from D ("0"s are put in those places). Destination D's social feature vector is always set to $(1, 1, 1, 1, 1, 1)$ if six social features are considered since it is the target. Suppose the source has a social feature vector of $(1, 0, 0, 0, 0, 0)$, meaning it only has the same value as D in the first social feature, e.g. they have the same *nationality*. The routing process should find the intermediate nodes that can resolve the differences in the rest of the five social features between the source and the destination. So a possible routing path can be: $S(1, 0, 0, 0, 0, 0) \rightarrow B_1(1, 1, 0, 1, 0, 0) \rightarrow B_2(1, 1, 1, 1, 0, 0) \rightarrow B_3(1, 1, 1, 1, 0, 1) \rightarrow B_4(1, 1, 1, 1, 1, 1)$. In each hop, the message is forwarded to some node which is more socially similar to D by having more common social features with D. When the message gets to B_4 which has the same values in all of the social features as D, it will be delivered to D directly when B_4 meets D. The social-feature-based method works well in MSNs where people's social features are stable and consistent with their behavior.

III. Our Routing Algorithms

In this section, we first propose two statistical-based routing algorithms BerRout and PoiRout based on the assumptions of node contact model and then put forward a practical new routing algorithm UpDown based on node encounters in the IMSNs. Though these three algorithms use different predictors, they use the same routing algorithm framework below.

Routing Algorithm Framework

1: Let $u_1, u_2, \cdots, u_{N-1}$ be nodes in the network, d be the destination, and $Pr_{u_i d}$ be the probability that node u_i will meet d in the future. The calculations of $Pr_{u_i d}$ in the BerRout, PoiRout and UpDown algorithms are in Formulas (6), (3) and (5), respectively.
2: Each node u_i has quality which is its $Pr_{u_i d}$ and level τ_i.
3: INITIALIZE $\forall i : \tau_i \leftarrow Pr_{u_i d}$
4: On contact between message holder u_i and node u_j:
5: **if** u_j is the destination d **then**
6: u_i forwards the message to u_j and the algorithm is terminated
7: **else if** $\tau_i < Pr_{u_j d}$ **then**
8: $\tau_i \leftarrow Pr_{u_j d}$
9: **if** u_j does not have the message **then**
10: u_i forwards the message to u_j
11: **end if**
12: **end if**

Fig. 1. The routing algorithm framework

A. Routing Framework of Our Algorithms

Our algorithms BerRout, PoiRout and UpDown use the same routing framework as shown in Fig. 1 to deliver messages from a source to a destination. The difference between them is that they use a different predictor $Pr_{u_i d}$ to make routing decisions. In the routing framework, we adopt delegation forwarding [3], which is proved by the authors to bring down the expected cost of message delivery from $O(N)$ to $O(\sqrt{N})$, where N is the number of nodes in the network. In delegation forwarding, each node u_i is assigned a quality $Pr_{u_i d}$ which in our algorithms indicates u_i's probability to meet destination d in the future based on different predictors, and a level value τ_i. Initially, the level of each node is equal to its quality. In each hop of the delegation forwarding, a message holder u_i only considers forwarding the message to a node u_j which has a higher quality than u_i's level hoping that u_j has a better chance to deliver the message to the destination. At the same time, node u_i improves its level to the quality of u_j. In the rest of the routing process, each message holder does the same thing until the destination receives the message. The essence of delegation forwarding is that a copy is transferred to a newly encountered node if the node is "closer" to the destination based on a certain predictor than other nodes that the current node has already met.

B. The BerRout algorithm

Several papers [9], [10] have the assumption that the meetings of two distinct pairs are independent in MSNs. Following that, the probability of node u meeting node v k times in an interval t can be expressed as in the Bernoulli experiment:

$$P\{X=k\} = \binom{t}{k} p_{uv}^k (1-p_{uv})^{t-k}, k = 0, 1, 2, \cdots, t \quad (1)$$

In the formula, variable X is the meetings of two nodes u and v. Notation p_{uv} $(0 \leq p_{uv} \leq 1)$ is the frequency of u meeting v. It can be calculated as the average of the meeting frequencies in the past intervals. The probability that u does not meet v in interval t is $P\{X=0\} = (1-p_{uv})^t$. Then the probability that u meets v at least once in interval t is: $1-P\{X=0\} = 1-(1-p_{uv})^t$. We can use this formula as the utility function to predict the future meeting probability Pr_{ud} of u and d as shown in Formula (2). We refer to the routing algorithm using this predictor as the *BerRout* algorithm as it is derived from the Bernoulli experiment.

$$Pr_{ud} = 1 - (1-p_{ud})^t \quad (2)$$

Since $(1-p_{uv})$ is between 0 and 1 and when t gets very large, $(1-p_{uv})^t$ will become 0 due to computation precision. Thus we cannot differentiate the meeting probabilities of candidates. In this case, without affecting the result of comparison, we can drop 1 and take the ln of the utility function in Formula (2) and change it to Formula (3).

$$Pr_{ud} = -tln(1-p_{ud}) \quad (3)$$

C. The PoiRout algorithm

In [5], the authors formulate the contact process of node pairs in MSNs as a Poisson process and conducted χ^2 tests on Infocom06 trace. They found that when enough test intervals (≥ 10) are used, over 85% of the contacted node pairs pass the test. Following this, the probability that u meets v k times in interval t can be expressed as:

$$P\{X=k\} = \frac{\lambda^k e^{-\lambda}}{k!} \quad (4)$$

In the formula, variable X is the meetings of two nodes u and v. Parameter $\lambda = tp_{uv}$, where t is the time interval and p_{uv} is the meeting frequency of u and v which can be estimated as the average of the frequencies in the past intervals. From the formula, we know that the probability that u does not meet v is: $P\{X=0\} = e^{-\lambda}$. Then the probability that u meets v at least once is: $1-P\{X=0\} = 1-e^{-\lambda}$. We can use this formula as the utility function to predict the future meeting probability Pr_{ud} of u and d as shown in Formula (5). We refer to the routing algorithm that uses this predictor as the *PoiRout* algorithm as it is derived from the Poisson distribution.

$$Pr_{ud} = 1 - e^{-\lambda} \quad (5)$$

The following theorem shows that the BerRout utility function in Formula (3) is close to the PoiRout utility function in Formula (5) if t is large and p is small.

Theorem 1: The BerRout utility function and the PoiRout utility function are close if $t \to \infty$ and $p \to 0$.

Proof According to the Poisson limit theorem [1], if $t \to \infty, p \to 0$ and $\lambda = tp$, then $\binom{t}{k} p^k (1-p)^{t-k} \to \frac{\lambda^k e^{-\lambda}}{k!}$. If $k = 0$, then $(1-p)^t \to e^{-\lambda}$. Then the result of Formula (3) is close to the result of Formula (5). □

In practical calculations, when $t \geq 10, p \leq 0.1$, the results of Formula (3) and Formula (5) are already very close.

D. The UpDown algorithm

The above two algorithms are based on theoretical assumptions about node contact. In this section, we propose a practical routing algorithm UpDown that uses a $Counter$ to predict the probability Pr_{ud} of a node u meeting destination d in the future. That is,

$$Pr_{ud} = Counter_{ud}, \quad (6)$$

where $Counter_{ud}$ is the counter based on the times that u met d during the time intervals we observe. Its detailed calculation will be explained in the next paragraph. Normally the probability should be a value in the range of $[0, 1]$. But for convenience's sake, we directly use $Counter_{ud}$ to represent the likelihood of u meeting d in the future without normalizing it to the $[0, 1]$ range since the higher the $Counter_{ud}$, the higher the probability they will meet.

The basic idea of the $Counter$ is that when two nodes meet during an interval, their $Counters$ will be incremented. If they do not meet during an interval, their $Counters$ will be decremented. So the $Counters$ will go up and down. Thus we name our algorithm using the predictor based on the $Counters$ the UpDown algorithm. The detailed $Counter$ calculation is shown in Fig. 2. More specifically, first, time is divided into equal length intervals t which is initialized to 1 and keeps incrementing as long as the algorithm does not terminate. Every node has an array of $Counters$ that records its meetings with other nodes. When node u meets node v for the first time, it will set up a new $Counter_{uv}$ with v. If u and v continue meeting each other in the next few intervals, their counters will be incremented by $f(\alpha, t_1) = \alpha^{t_1}$ $(t_1 \geq 1)$, where α (> 1) is the value incremented and t_1 is the number of time intervals they continue meeting. Function $f(\alpha, t_1)$ is increasing with the increment of t_1 and $\alpha > 1$. The intuition is that if two nodes meet continuously, the increment to their counters will be larger and larger, meaning that they are more and more likely to meet in the future. On the other hand, if u and v do not meet in the next few intervals, their counters will be decremented by $f(\beta, t_2) = \beta^{t_2}$ $(t_2 \geq 1)$, where β (> 1) is the value decremented and t_2 is the number of time intervals they discontinue to meet. $f(\beta, t_2)$ is also an increasing function with the increment of t_2 and $\beta > 1$. The intuition is that if two nodes do not meet in the next several intervals, the decrement to their counters will be larger and larger, meaning that it is less and less likely for them to meet in the future. The counters of u and v will be updated like this until the counters are below a defined low threshold (LTH), indicating

The $Counter$ calculation in the UpDown algorithm

1: If u meets v for the first time, initialize nodes u and v's counters $Counter_{uv}$ and $Counter_{vu}$ to 0; Initialize interval $t = 1$.
2: **repeat**
3: $t_1 = 1$;
4: **while** u and v meet in interval t **do**
5: $Counter_{uv} += \alpha^{t_1}$; $Counter_{vu} += \alpha^{t_1}$;
6: t_1++; $t++$;
7: **end while**
8: $t_2 = 1$;
9: **while** u and v do not meet in interval t **do**
10: $Counter_{uv} -= \beta^{t_2}$; $Counter_{vu} -= \beta^{t_2}$;
11: t_2++; $t++$;
12: **end while**
13: **until** $Counter_{uv} < LTH$
14: u and v remove each other's counters from their records.

Fig. 2. The $Counter$ calculation in the UpDown algorithm. Parameter t_1 is the number of time intervals u and v continue meeting each other and t_2 is the number of time intervals they discontinue to meet. LTH means low threshold.

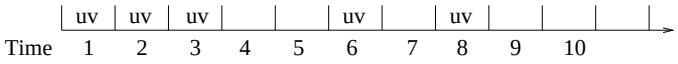


Fig. 3. The meeting history of nodes u and v in 10 time intervals

that they will not likely meet again in the future. Then u and v can remove each other's counters from their records.

Let us look at an example of how $Counters$ are calculated. Suppose the meetings of u and v are shown in Fig. 3. The numbers $1, 2 \cdots 10$ represent the 10 time intervals in the history we observe. The appearance of uv in an interval means that they met in that interval. As we can see, they start to meet each other in interval 1. They set up $Counter_{uv}$ ($Counter_{vu}$) for each other. The values of their counters in the next 10 intervals are shown in Table I.

IV. Simulations

We compared our algorithms with the existing social-based algorithms using a self-written simulator in Matlab. Flooding was included as a benchmark. To fit the legend in each figure, we attached a short name beside each algorithm.

1) The Flooding Algorithm (Flooding) [19]
2) The Social-analysis-based Algorithm (Analysis) [5]
3) The Social-Feature-based Algorithm (Feature) [20]
4) The UpDown Routing Algorithm (UpDown)
5) The Bernoulli-based Routing Algorithm (BerRout)
6) The Poisson-based Routing Algorithm (PoiRout)

The Feature algorithm takes the idea from [20] and converts it into a single-copy scheme for fair comparison where routing is guided by resolving social feature differences between a source and a destination.

In the Analysis algorithm, to calculate centrality locally in a distributed environment, we adopt the centrality metric in [5] as a utility function to predict the future contact probability of a node u based on its local contacts with its neighbors.

$$Pr_u = 1 - \frac{1}{N-1} \sum_{all v, v \neq u} e^{-\lambda_{uv} t} \quad (7)$$

N is the total number of nodes in the network. λ_{uv} is the meeting frequency of nodes u and v. Pr_u indicates the average probability that a randomly chosen node v in the network is contacted by u within time t.

To compare the performance of these algorithms, we used a real-life trace - Infocom06 trace [16], which has recorded conference attenders' contact history in an IMSN using Bluetooth devices (iMotes) for three days at IEEE Infocom 2006 in Miami. The trace data set consists of two parts: *contacts* between the iMote devices that are carried by participants and *social features* of the participants, which are the statistics of participants' information from a questionnaire form. Six social features are extracted from the data set: *nationality, language, affiliation, position, city*, and *country*.

We define three important performance metrics to evaluate algorithms:

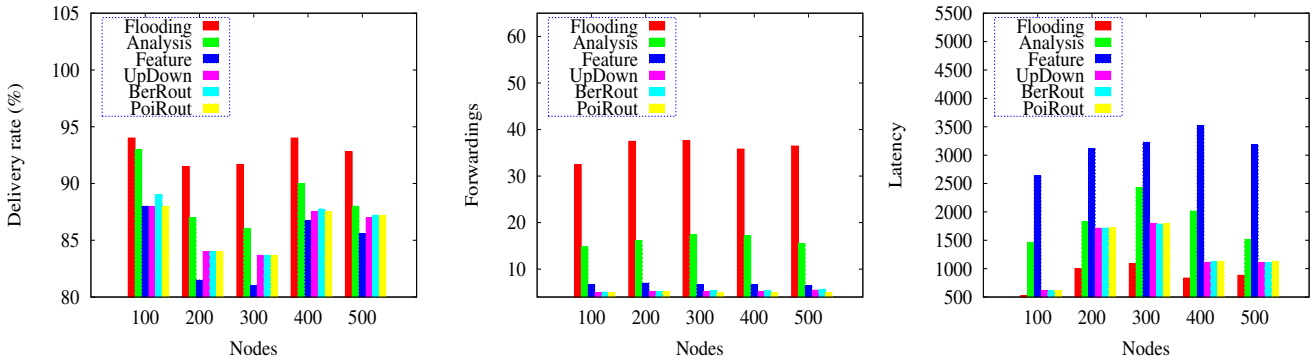
1) *delivery rate*: the fraction of generated messages that are correctly delivered to the destination within a given time period.
2) *number of forwardings*: the number of forwardings needed to successfully deliver a packet.
3) *delivery latency*: the time between when a message is generated and when it is received.

We have the following settings for the algorithms we compare. For the Feature algorithm, in the Infocom06 trace we study, we found that 17 out of 79 iMotes carried by people have no or partial social features, which excludes them from being used in the social-feature-based routing. Thus, the actual number of iMotes used was 62. For the UpDown, BerRout, and PoiRout algorithms, we used the past 5000 length of history divided up into 5 intervals to predict the future meeting probabilities of nodes. In the UpDown algorithm, α and β can take any values greater than one. We set α to 6 and β to 2 to favor nodes' meetings. We randomly generated 100 to 500 source-destination pairs to send packets. To compare fairly, the same source-destination pairs were applied to all of the algorithms. And to avoid 100% delivery rate of the algorithms, the time-to-live of all of the packets (except for those created in the Flooding simulation) was set to 9, meaning that a given packet can be transferred at most nine times. The three performance metrics were calculated and averaged.

The simulation results are shown in Fig. 4. In the experiment, the delivery rates of all of the algorithms with node numbers from 100 to 500 are above 80% (see Fig. 4(a)). Flooding has the highest delivery rate and Feature has the lowest. Analysis has an up to 5% improvement in delivery rate than UpDown, BerRout and PoiRout but at the cost of about 3 times their forwardings as shown in Fig. 4(b). As expected, Flooding has the highest number of forwardings since it spreads copies epidemically. Feature also has higher

Interval	$Counter_{AB}(Counter_{BA})$	Interval	$Counter_{AB}(Counter_{BA})$
1	α	6	$\sum_{t_1=1}^{3}\alpha^{t_1} - \sum_{t_2=1}^{2}\beta^{t_2} + \alpha$
2	$\sum_{t_1=1}^{2}\alpha^{t_1}$	7	$\sum_{t_1=1}^{3}\alpha^{t_1} - \sum_{t_2=1}^{2}\beta^{t_2} + \alpha - \beta$
3	$\sum_{t_1=1}^{3}\alpha^{t_1}$	8	$\sum_{t_1=1}^{3}\alpha^{t_1} - \sum_{t_2=1}^{2}\beta^{t_2} + \alpha - \beta + \alpha$
4	$\sum_{t_1=1}^{3}\alpha^{t_1} - \beta$	9	$\sum_{t_1=1}^{3}\alpha^{t_1} - \sum_{t_2=1}^{2}\beta^{t_2} + \alpha - \beta + \alpha - \beta$
5	$\sum_{t_1=1}^{3}\alpha^{t_1} - \sum_{t_2=1}^{2}\beta^{t_2}$	10	$\sum_{t_1=1}^{3}\alpha^{t_1} - \sum_{t_2=1}^{2}\beta^{t_2} + \alpha - \beta + \alpha - \sum_{t_2=1}^{2}\beta^{t_2}$

TABLE I
THE VALUES OF COUNTERS



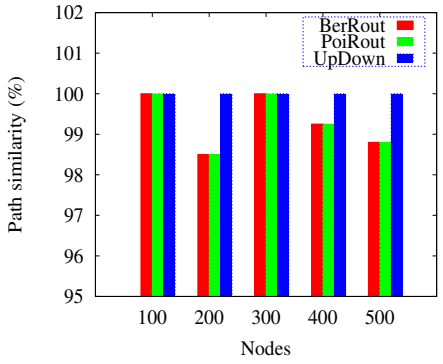
(a) Results of average delivery rate (b) Results of average number of forwardings (c) Results of average latency

Fig. 4. Comparison of UpDown, BerRout, PoiRout with Flooding and Social algorithms

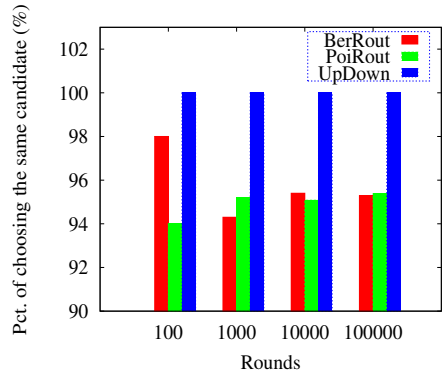
forwardings than UpDown, BerRout and PoiRout. In terms of latency (see Fig. 4(c)), Flooding has the lowest because of its epidemic nature. Feature has the highest latency followed by Analysis and then the three proposed ones. In short, in terms of performance versus cost, our proposed algorithms UpDown, BerRout and PoiRout are better than the existing social-analysis-based and social-feature-based algorithms.

In all of the three metrics, algorithms UpDown, BerRout, and PoiRout have similar results. There is no doubt about the similarity between BerRout and PoiRout as they are related by the Poisson limit theorem when we looked at a long time interval (5000) and the probability of two nodes meeting each other is less than 0.1 according to the trace. We want to find out why UpDown also has similar results as them. We compared the path similarity of the three algorithms. Using the UpDown algorithm as a benchmark, we found that over 98% of the paths selected by the BerRout and PoiRout algorithms to send a message from a source to a destination are the same as those selected by the UpDown algorithm (see Fig. 5 (a)). It is hard to find a mathematical relationship between UpDown and BerRout or PoiRout as they have so different utility functions. Therefore, we resorted to simulations to study their relationships. We randomly generated contact histories of two candidate nodes, say u and v, in the past 5000 time units and applied the utility functions of the three algorithms. We ran the program 100, $1k$, $10k$ and $100k$ times. Again using UpDown as a benchmark, we found that over 94% of the time, the utility functions of BerRout and PoiRout chose the same candidate as that of UpDown (see Fig. 5 (b)). Though without mathematical support yet, their utility functions give consistent results in choosing the same candidates most of the time. That explains the similarity in the selected paths and therefore the similarity in the performance of the three algorithms.

In conclusion, the practical algorithm UpDown is a better algorithm for IMSNs than the existing social-analysis-based and social-feature-based algorithms. In a dynamic and distributed network formed impromptu, the previous connections in social network graphs and the social features in user profiles cannot provide much help in making right routing decisions as can be inferred from the simulation results. Routing in IMSNs needs to be based on information collected right there as the network is being formed. UpDown is such an algorithm that is simple and only relies on the information of node encounters gathered in IMSNs. The node encounter information not only reflects the meetings of people but also their meeting frequency that is missing in the social-analysis-based algorithm. It is time dependent in that it catches the growing and decaying of people's relationships as time goes on, thereby it can catch node contact behavior more accurately than the static information in the social network graphs and



(a) Path similarity between BerRout, PoiRout, and UpDown algorithms



(b) Percentage of choosing the same candidate in BerRout, PoiRout, and UpDown algorithms

Fig. 5. Comparison of UpDown with BerRout and PoiRout algorithms

in the user profiles. This result confirms the claim by [7] that it is potentially possible to deliver information relying only on encounters. The two statistical-based theoretical routing algorithms BerRout and PoiRout also beat the social-based ones, indicating that the node contact patterns they assume are reasonable in the trace we use. The similar performance of UpDown to the two theoretical ones allows us to discuss the properties of UpDown theoretically through them if the direct analysis is difficult.

V. Conclusion

In this paper, we designed a practical routing algorithm specifically for IMSNs where node connections are time-dependent and short-term. We first proposed two statistical-based theoretical routing algorithms named BerRout and PoiRout inspired by the node contact models in several papers and then put forward a practical routing algorithm UpDown which makes routing decisions based on a simple *Counter* capturing the ups and downs of people's relationships formed in an IMSN. We compared our algorithms with the existing social-based algorithms by simulations. The results showed that the practical algorithm performed close to the two theoretical ones and all of our proposed algorithms outperformed the existing ones in terms of performance versus cost in an IMSN environment. In the future, we will explore more accurate models to describe social interactions of people to guide message delivery in IMSNs and use data sets from various sources to further validate our results.

VI. Acknowledgement

This paper is supported in part by NSF CNS grant 1305302.

References

[1] http://en.wikipedia.org/wiki/Poisson_limit_theorem.
[2] E. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant MANETs", *Proc. of IEEE MobiHoc*, 2007, pp. 32-40.
[3] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding", *Proc. of IEEE MobiHoc*, 2008, pp. 251-260.
[4] Facebook, http://www.facebook.com.
[5] W. Gao, Q. H. Li, B. Zhao, and G. H. Cao, "Multicasting in delay tolerant networks: a social network perspective", *Proc. of IEEE MobiHoc*, 2009, pp. 299-308.
[6] T. Hossmann, T. Spyropoulos, and F. Legendre,"From contacts to graphs: Pitfalls in using complex network analysis for dtn routing", *IEEE Int. Workshop on Network Science For Communication Networks*, 2009, pp. 260-265.
[7] W. J. Hsu, and A. G. Helmy, "On Nodal Encounter Patterns in Wireless LAN Traces", *Proc. of 4th Int. Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2006, pp. 03-06.
[8] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks", *Proc. of IEEE MobiHoc*, 2008, pp. 241-250.
[9] S. Ioannidis, A. Chaintreau, and L. Massoulie, "Optimal and scalable distribution of content updates over a mobile social network", *Proc. of IEEE Infocom*, 2009, pp. 1422-1430.
[10] U. Lee, S.-Y. Oh, K.-W. Lee, and M. Gerla, "RelayCast: Scalable multicast routing in delay tolerant networks", *Proc. of IEEE ICNP*, 2008, pp. 218-227.
[11] LinkedIn, http://www.linkedin.com.
[12] A. Mei, G. Morabito, P. Santi, and J. Stefa, "Social-aware stateless forwarding in pocket switched networks," *Proc. of IEEE Infocom*, 2011, pp. 251-255.
[13] S. Milgram, "The small world problem", *Psychology Today*, Vol. 1, No. 1, May 1967, pp. 60-67.
[14] M. Motani, V. Srinivasan, and P. Nuggehalli, "PeopleNet: engineering a wireless virtual social network", *Proc. IEEE MobiCom*, 2005, pp. 243257.
[15] J. Ott, E. Hyytia, P. Lassila, T. Vaegs, and J. Kangasharju, "Floating content: information sharing in urban areas", *Proc. of IEEE PerCom*, 2011, pp. 136-146.
[16] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD trace cambridge/haggle/imote/infocom2006 (v.2009-05-29)", Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle/imote/infocom2006, May 2009.
[17] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Single-copy routing in intermittently connected mobile networks," *Proc. of Secon*, 2004, pp. 235-244.
[18] V. Srinivasan, M. Motani, and W. Ooi, "Analysis and implications of student contact patterns derived from campus schedules", *Proc. of IEEE MobiCom*, 2006, pp. 8697.
[19] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *Technical Report*, Department of Computer Science, Duke University, 2000.
[20] J. Wu and Y. Wang, "Social feature-based multi-path routing in delay tolerant networks," *Proc. of IEEE Infocom*, 2012, pp. 1368-1376.
[21] S. Yang and J. Wu, "Adaptive Backbone-based Routing in Delay Tolerant Networks," *Proc. of IEEE MASS*, 2013, pp. 356-364.