Dynamic Social Feature-based Diffusion in Mobile Social Networks

Xiao Chen[1], Kaiqi Xiong[2]
[1]Department of Computer Science, Texas State University, San Marcos, TX 78666
[2]Florida CyberSecurity Center and College of Arts and Sciences, University of South Florida, Tampa, FL 33620
Email: xc10@txstate.edu, xiongk@usf.edu

***Abstract*—With the wide use of smart mobile devices and the popularity of mobile social networks (MSNs), direct marketing has been adopted by more and more companies to announce the news of their products first to a group of selected profitable customers and let them diffuse the news by "word-of-mouth" to other potential buyers to control the marketing cost. In this paper, we study the diffusion minimization problem whose goal is to select an optimal set of initial nodes to disseminate the information to the whole network as quickly as possible. We tackle the problem by taking advantage of node social features in MSNs. We define dynamic social features to capture nodes' dynamic contact behavior and use social similarity metrics to measure their social closeness. We adopt the community concept in social networks to reduce the complexity of the diffusion minimization problem. We propose novel diffusion node selection algorithms based on these new features to minimize the diffusion time. Simulation results show that our algorithms have lower diffusion times than the existing ones.**

***Index Terms*—diffusion, dynamic social features, mobile social networks, social similarity, static social features**

I. Introduction

With the wide use of smart mobile devices and the popularity of mobile social networks (MSNs) where people move around and contact each other through these devices, *direct marketing* has been adopted by more and more companies to announce the news of their products first to a group of selected profitable customers and then let them disseminate the news by "word-of-mouth" [10] to other potential buyers to control the marketing cost. The communication in MSNs does not solely rely on network infrastructures. In many cases, people communicate *opportunistically* via local wireless bandwidth such as Bluetooth. This makes MSNs similar to the delay tolerant networks (DTNs) [1] where nodes communicate through a *store-carry-forward* fashion. When two nodes move within each other's transmission range, they *contact* each other and when they move out of their ranges, their contact is lost. The message to be delivered needs to be stored in the local buffer until a contact occurs in the next hop.

There are several papers in the literature studying information dissemination by "word-of-mouth" in social networks. Some of them investigate node influence [3], [8], [21] while others focus on node selfishness and privacy in information dissemination [13], [14]. Recently, Lu *et al.* [9] work on the diffusion minimization problem whose goal is to find an optimal set of initial nodes to disseminate information to the whole network as quickly as possible. In the dissemination process, a node will be affected or influenced by the information with an *affect probability* p. The diffusion minimization problem under the probabilistic diffusion model can be formulated as an asymmetric k-center problem which is NP-hard [4]. The best known approximation algorithm for the asymmetric k-center problem has an approximation ratio of $\log^* n$ and a time complexity of $O(n^5)$ [20], where n is the number of nodes in the network and $\log^* n$ is the iterated logarithm of n.

Obviously, the performance and the time complexity of the approximation algorithm are not scalable in large MSNs. To make the algorithm scalable, Lu *et al.* [9] utilize the community structures in the network and identify diffusion nodes in the communities based on the fact that nodes in a community are more likely to meet and influence each other. Their solution to the diffusion minimization problem is based on applying network analysis methods to the social network graph formed by aggregating past node encounters. The social network graphs can show whether two nodes have met in the past but not the frequency of the meetings [6] nor the social information of the nodes. In this paper, we plan to consider these information and tackle the problem from a different perspective inspired by the social feature method used by several routing algorithms in MSNs [11], [16], [22], [23]. The social features $F_1, F_2, F_3, \cdots$ may refer to people's *Nationality*, *City*, *Language*, etc., and $f_1, f_2, f_3, \cdots$ represent the values of these social features. For example, the value of *Language* can be *English*. The user social features are usually provided by users when they fill out their profiles. The diffusion process can take advantage of social features because people having more social features in common tend to contact more frequently as shown by the routing algorithms [11], [16], [22], [23] that use social features. In addition, in information diffusion, it is more likely for someone to be influenced by people with similar social features.

Similar to [9], we will adopt the community structure to group nodes into different communities based on their social features to speed up information diffusion in MSNs. However, there are several challenging issues to consider before we use social features in information dissemination in MSNs. First, how to use social features. Social features in user profiles, which we refer to as *static social features*, do not show nodes' meeting frequencies either and are not always adequate to reflect users' dynamic contact behavior, especially for an

MSN formed impromptu at some conferences or events. For example, someone who puts *New York* as his home state in his profile may actually attend a conference in *Texas*. Thus, static social features need to be extended to include nodes' contact frequencies in order to be useful in information diffusion. Second, how to compare the social similarity or closeness of nodes to form communities based on their social features. Finally, how to find an optimal set of diffusion nodes from these communities to minimize the diffusion time.

To address the above issues, in this paper, we first put forward the definition of *dynamic social features* to capture nodes' dynamic contact behavior based on nodes' encounter frequency. Then, we present an enhanced definition of dynamic social features to better serve the purpose. Next, we adopt metrics derived from data mining [5] to calculate the social similarity of nodes based on dynamic social features. Moreover, we propose diffusion node selection algorithms to select a set of nodes from communities formed according to nodes' social similarity using the two definitions of dynamic social features. Finally, we feed the selected nodes into the diffusion algorithm to obtain the diffusion time. Simulation results show that our algorithms using dynamic social features have shorter diffusion time than the algorithms based on network analysis, static social features, and random diffusion node selection.

In summary, we make the following contributions in this paper. (1) To the best of our knowledge, this is the first research to study the diffusion minimization problem using social features. (2) We introduce the concepts of dynamic and enhanced dynamic social features to capture nodes' dynamic contact behavior and use social similarity metrics to measure nodes' social closeness. (3) We group nodes into communities based on their social closeness to make our algorithms scalable. (4) We conduct simulations to evaluate the performance of our proposed algorithms.

The rest of the paper is organized as follows: Section II references the related works. Section III defines the problem we want to solve in this paper. Section IV introduces the preliminaries of our solution to the problem. Section V presents our algorithms. Section VI shows the simulation results, and the conclusion is in Section VII.

II. Related works

A. Information dissemination by "word-of-mouth"

In the literature, several papers [3], [8], [13], [14], [21] have been concerned with the "word-of-mouth" advertisement diffusion problem in the network, among which some papers [3], [8], [21] focus on node influences in information dissemination. For example, Domingos *et al.* [3] model customers' influence by their *network value*. Kempe *et al.* [8] work on maximizing the spread of influence through a social network. And Wang *et al.* [21] propose a community-based greedy algorithm for mining top-K influential nodes. Some other papers [13], [14] emphasize on node selfishness and privacy in the diffusion process. For instance, Peng *et al.* [14] design schemes to address users' selfishness and their privacy concerns in information diffusion. Ning *et al.* [13] put forward a Self-Interest-Driven (SID) incentive scheme to stimulate cooperation among selfish nodes for ad dissemination in autonomous mobile social networks. Recently, Lu *et al.* [9] discuss the diffusion minimization problem and propose a community-based algorithm from network analysis.

B. Social analysis-based and social feature-based methods

As social network applications explode in recent years, there are basically two methods that take social factors into account in the study of routing problems. The first is the social analysis method [2], [7], [12], [15], [19], which assesses the message delivery probability of a node by analyzing the social network graph generated by the aggregation of past node contacts. The second is the social feature method [11], [16], [22], [23], which evaluates a node's message delivery probability by looking at the number of common social features shared between the node and the destination. The intuition of this method is that nodes with more common social features are more likely to meet in the future. In this method, routing is treated as a process to resolve social feature differences between a source and a destination. In the study of diffusion minimization problem, Lu *et al.* [9] address it using the social analysis method, identifying communities by analyzing node connections from past encounter history. In this paper, we will solve the problem using nodes' social features and their contact frequencies that are not reflected in the network analysis method. Our approach, as far as we know, has not been proposed in information diffusion before.

III. Problem definition

In an MSN network with n nodes, information diffusion is a process as follows: First, a set of diffusion nodes are selected and given the information to spread. Then, these *affected* diffusion nodes will spread the information when they encounter *unaffected* nodes. An unaffected node will become affected with an affect probability p. The diffusion process terminates when all of the nodes are affected.

Let D be the set of k selected diffusion nodes. The *diffusion time* $T(D)$ of the selected node set D is defined as the time interval from the start of information spreading by the diffusion nodes to the time when all of the nodes have accepted the information (affected). To solve the diffusion minimization problem using social features, we need node encounter history H in an MSN because static social features in user profiles are not adequate to capture users' dynamic contact behavior. Thus, our problem can be formulated as: Given node static social features F and their encounter history H in an MSN, and given the diffusion set size k and the affect probability p, we want to find a diffusion set D to minimize $T(D)$.

IV. The preliminaries

In this section, we introduce the preliminaries of our solution to the diffusion minimization problem. We first give the definition of dynamic social features and its enhancement, then show how to calculate the social similarity of two nodes based on their dynamic social features.

A. Definitions of dynamic social features

Suppose we consider m social features $\langle F_1, F_2, \cdots, F_m \rangle$ in an MSN. We associate each node with a vector of its social feature values. Thus, a node is denoted by a vector, x, consisting of m components $\langle x_1, x_2, \cdots, x_m \rangle$. Based on nodes' encounter history H, we define x_i as follows to capture nodes' contact behavior:

(1). Dynamic social features by frequency

One definition of x_i is the frequency of node x meeting nodes with the same f_i out of all of the nodes it has met in the history we observe. That is,

$$x_i = \frac{M_i}{M_{total}} \tag{1}$$

In Definition (1), M_i is the number of times that x has met nodes with the same f_i in the history we observe and M_{total} is all of the nodes that x has met in that interval. For example, if f_i refers to $Student$ and if x has met 20 $Students$ out of a total of 100 people, then $x_i = 20/100 = 0.2$.
Therefore, a node x's dynamic social features are defined by its vector, which is

$$< x_1, x_2, \cdots, x_m > = \left\langle \frac{M_1}{M_{total}}, \frac{M_2}{M_{total}}, \frac{M_3}{M_{total}}, \cdots \frac{M_m}{M_{total}} \right\rangle$$

Nevertheless, one problem with the frequency definition of x_i can be shown in the following example. Assume node x has met 1 $Student$ out of 2 people it has met in total in the history we observe. Node y has met 5 $Students$ out of 10 people it has met in total. Using Definition (1), both of their frequencies are 0.5 in meeting $Students$. So which one is more likely to meet $Students$ in the future? From the intuition, node y should be given a higher priority because it is more actively meeting people. To deal with this kind of case, we have the following enhanced definition by focusing on M_i.

(2). Enhanced dynamic social features by focusing on M_i

If we focus on M_i, x_i can be calculated as:

$$\begin{aligned} x_i &= (\tfrac{M_i+1}{M_{total}+1})^{p_i} (\tfrac{M_i}{M_{total}+1})^{1-p_i} \\ &= (M_i+1)^{p_i} \tfrac{M_i^{1-p_i}}{M_{total}+1} \end{aligned} \tag{2}$$

In Definition (2), $p_i = M_i/M_{total}$. This definition predicts x_i by looking at the next meeting probability of node x with another node having the same social feature value f_i. In the next time, the total meeting times will be $M_{total}+1$. The first part $(\frac{M_i+1}{M_{total}+1})^{p_i}$ means that there will be p_i probability that x will have a "good" meeting with another node having the same social feature value f_i next time. In this case, M_i will also be incremented by 1. The second part $(\frac{M_i}{M_{total}+1})^{1-p_i}$ means that there will be $1-p_i$ probability for x not to meet a node with the same social feature value f_i next time. In that case, M_i will remain the same. The definition for x_i then takes the geometric mean of the two parts.

Now we can break the tie in the example above. For node x, $M_i = 1, M_{total} = 2, p_i = 0.5$, and for node y, $M_i = 5, M_{total} = 10, p_i = 0.5$. Using Definition (2), $x_i = (1+1)^{0.5} * \frac{1^{(1-0.5)}}{2+1} = 0.4714$ and $y_i = (5+1)^{0.5} * \frac{5^{(1-0.5)}}{10+1} = 0.4979$. These two results are close to the result from Definition (1), yet they tell us that y is better because it has met more nodes with the intended social feature value $Student$ and it will be more likely doing so in the future.

Dynamic social features, as shown in the definitions, not only record if a node has certain social features, but also predict the probability of this node meeting other nodes with the same social features. Unlike the static social features, dynamic social features change as user activities change over time so that they can better reflect users' contact behavior.

Next is the definition of the *mean dynamic social features* which will be used in the later algorithms.

Mean dynamic social features

For n nodes $u_1, u_2, \cdots, u_n$ in a network, assume their associated dynamic social features are: $u_1 = \langle u_{11}, u_{12}, \cdots, u_{1m} \rangle$, $u_2 = \langle u_{21}, u_{22}, \cdots, u_{2m} \rangle$, $\cdots$, $u_n = \langle u_{n1}, u_{n2}, \cdots, u_{nm} \rangle$. The mean dynamic social features of these nodes is defined as: $u_{mean} = \left\langle \frac{\sum_{i=1}^n u_{i1}}{n}, \frac{\sum_{i=1}^n u_{i2}}{n}, \cdots, \frac{\sum_{i=1}^n u_{im}}{n} \right\rangle$.

B. Calculation of social similarity

With the defined dynamic social features of nodes, we can use the social similarity calculation algorithm in Fig. 1 to calculate the social similarity $S(x, y)$ of nodes x and y based on their dynamic social feature vectors. The first few steps of the algorithm are to obtain the dynamic social features of x and y from the recorded static social feature set F and the contact history H. In calculating dynamic social features, we should combine all of their social feature values in their vectors. If a node does not have a value for, say f_i, then the x_i for that f_i is 0. After getting their dynamic social features, in the last step of the algorithm, we apply the following metrics derived from data mining [5] to calculate their social similarity. In these metrics, x and y represent the dynamic social feature vectors of nodes x and y. All of these metrics are normalized to the range of $[0, 1]$.

(1). Euclidean similarity

After normalizing the original Euclidean similarity to the range of $[0, 1]$ and subtract it from 1, it is now defined as $S(x, y) = 1 - \frac{\sqrt{\sum_{i=1}^m (y_i - x_i)^2}}{\sqrt{m}}$.
For example, suppose we consider four social features $\langle City$, $Language$, $Position$, $Affilation \rangle$. Node x's values in these social features are: $\langle NewYork$, $English$, $Student$, *New York State Univ.*$\rangle$ and y's values in these social features are: $\langle NewYork$, $English$, $Student$, *Texas State Univ.*$\rangle$. According to Fig. 1, we create a vector $\langle NewYork$, $English$, $Student$, *New York State Univ.*, *Texas State Univ.*$\rangle$ containing all of x and y's social feature values. Then we obtain x and y's dynamic social features by filling x_i and y_i in these fields according to nodes' contact history H. Suppose x's dynamic

Algorithm: Social Similarity $S(x,y)$ Calculation

```
Require: $m$: a set of social features we consider; $F$: a set recording the static social features of nodes; $H$: a data set containing the encounter history of the $n$ nodes in the network.
1: Obtain the static social feature values of $x$ and $y$ from $F$: $\langle f_{1x}, f_{2x}, \cdots, f_{mx}\rangle$ and $\langle f_{1y}, f_{2y}, \cdots, f_{my}\rangle$.
2: /* create a vector of social feature values that is the union of the social feature values of $x$ and $y$ */
3: $\langle f_1, f_2, \cdots, f_l\rangle = \langle f_{1x}, \cdots, f_{mx}\rangle \bigcup \langle f_{1y}, \cdots, f_{my}\rangle$.
4: calculate the dynamic social features of $x$ and $y$ by filling $x_i$ and $y_i$ in these fields using dynamic social feature definitions (1) or (2). If $x$ or $y$ does not have a value in a field, put a 0 there.
5: apply one of the similarity metrics in Section IV-B to the dynamic social features of $x$ and $y$ to calculate their social similarity.
```

Fig. 1. The social similarity calculation algorithm

social feature vector is: $\langle 0.7, 0.93, 0.41, 0.30, 0\rangle$, meaning in the history we observe, x has met New Yorker 70% of the time, people who speak English 93% of the time, students 41% of the time, people from New York State University 30% of the time, and no one from Texas State University. And suppose y's social feature vector is: $\langle 0.23, 0.81, 0.5, 0, 0.2\rangle$. Applying the Euclidean similarity metric on x and y's dynamic social feature vectors, their social similarity $S(x,y) = 0.73$.

(2). Tanimoto similarity

It measures the similarity of x and y as: $S(x,y) = \dfrac{x \cdot y}{x \cdot x + y \cdot y - x \cdot y}$. The notation $x \cdot y$ is the product of the two vectors.

(3). Cosine similarity

It measures the similarity of x and y as: $S(x,y) = \dfrac{x \cdot y}{\sqrt{(x \cdot x)(y \cdot y)}}$.

(4). Weighted Euclidean similarity

In addition to the basic Euclidean similarity mentioned above, we also employ the weighted Euclidean similarity to favor the social features that are more influential to the delivery of the packet. To determine the weight of a social feature, we use the Shannon entropy [18] which quantifies the expected value of the information contained in the feature [22]. The Shannon entropy for a given social feature is calculated as: $w_i = -\sum_{i=1}^{k} p(f_i) \cdot log_2(f_i)$, where w_i is the Shannon entropy for feature F_i, vector $\langle f_1, f_2, \cdots f_k\rangle$ contains the possible values of feature F_i, and p denotes the probability mass function of F_i. The weighted Euclidean similarity normalized to the range of $[0,1]$ is: $S(x,y) = 1 - \dfrac{\sqrt{\sum_{i=1}^{m} w_i \cdot (y_i - x_i)^2}}{\sqrt{\sum_{i=1}^{m} w_i}}$.

Algorithm: Diffusion Node Selection

```
Require: $k$: the number of diffusion nodes; $F$: a set recording the static social features of nodes; $H$: a data set containing the encounter history of the $n$ nodes in the network.
1: arbitrarily choose $k$ nodes from the network as the diffusion nodes and form $k$ clusters with each cluster containing one diffusion node;
2: calculate the dynamic social features for each node using Definition (1) or (2) according to $F$ and $H$;
3: repeat
4:     (re)assign each node to a cluster whose center, defined by the mean dynamic social features of the nodes in the cluster, is most similar to that node based on some similarity metric in Section IV-B;
5:     after all of the nodes are assigned to clusters, update each cluster center, that is, recalculate the mean dynamic social features of the nodes in each cluster;
6: until no more changes;
7: pick the node which is most similar to its cluster center as the diffusion node of that cluster.
8: return a set of diffusion nodes $D$.
```

Fig. 2. The diffusion node selection algorithm

V. Diffusion node selection and diffusion algorithms

With the above preliminaries, we present our algorithm to select k diffusion nodes in Fig. 2 inspired by the k-means algorithm in data mining [5]. The idea of the algorithm is as follows: first arbitrarily choose k nodes from the network as the diffusion nodes and form k clusters with each cluster containing one diffusion node. Then calculate the center of each cluster which is defined as the mean dynamic social features of that cluster. Assign each node to a cluster whose center is most socially similar to that node. After all of the nodes are allocated to the k clusters, recalculate the center of each cluster. Repeat this process until there are no more changes in node allocation. Then in each cluster, pick the node that is most similar to the cluster center as the diffusion node of that cluster and return all of these nodes as the diffusion nodes of the network.

Time complexity of the diffusion node selection algorithm. It is easy to see that the time complexity of the diffusion node selection algorithm is $O(nkt)$, where n is the total number of nodes in the network, k is the number of diffusion nodes, and t is the number of iterations. Normally, $k << n$ and $t << n$. Therefore, the method is relatively scalable and efficient in large mobile social networks.

After the k diffusion nodes are selected, they are fed into the diffusion algorithm in Fig. 3 to spread the message in the network. The selected k diffusion nodes are 'affected' nodes and whenever an affected node meets an unaffected node, there is a p probability to affect that node. The process continues until all of the nodes in the network are affected.

Algorithm: Diffusion

```
Require: $D$: a set of diffusion nodes from the Diffusion Node
    Selection Algorithm; $p$: node affect probability.
1: set the states of all of the nodes in set $D$ to be 'affected'.
2: repeat
3:   if an affected node encounters an unaffected node then
4:     the affected node will send the message to the unaffected node.
5:     if the unaffected node accepts the message with an affect probability $p$ then
6:       change the state of the unaffected node to 'affected'.
7:     end if
8:   end if
9: until all of the nodes are affected;
```

Fig. 3. The diffusion algorithm

VI. Simulations

We conducted simulations to compare the diffusion times of our algorithms and the existing ones using a custom simulator written in Matlab.

A. Algorithms compared

We compared the following algorithms.

1) *The Diffusion Algorithm using Static Social Features* (Static): where diffusion nodes are selected from communities formed using nodes' static social features in user profiles.
2) *The Diffusion Algorithm using Network Analysis Methods* (Analy) [9]: where diffusion nodes are selected from communities formed using network analysis on network graphs.
3) *The Diffusion Algorithm using Random Diffusion Nodes* (Rand): where diffusion nodes are selected randomly.
4) *The Diffusion Algorithm using Dynamic Social Feature Definition (1)* (DSF1): our algorithm where diffusion nodes are selected from communities formed using the Euclidean social similarity metric and nodes' dynamic social feature Definition (1).
5) *The Diffusion Algorithm using Dynamic Social Feature Definition (2)* (DSF2): our algorithm where diffusion nodes are selected from communities formed using the Euclidean social similarity metric and nodes' dynamic social feature Definition (2).

B. Simulation trace

To compare the performance of these algorithms, we used a real-life Infocom06 trace [17], which has recorded conference attenders' contact history in an MSN using Bluetooth devices (iMotes) for four days at IEEE Infocom 2006 in Miami. The trace data set consists of two parts: *contacts* between the iMote devices carried by participants and *social features* of the participants, which were collected using a questionnaire form. Six social features were extracted from the data set: *nationality, language, affiliation, position, city*, and *country*.

C. Settings

We had the following settings for the algorithms we compared. In the Infocom06 trace we studied, we found that 17 out of 79 iMotes carried by people have no or partial social features, which excluded them from being used in the diffusion algorithms. Thus, the actual number of iMotes used was 62. In our simulations, we utilized the first one day of the data as the initial history for the algorithms and performed the diffusion algorithms on the remaining three days.

We tried the cases where the number of diffusion nodes k was 5 and 10, and nodes' affect probability went from 0.4 to 1 to ensure a reasonable affect probability so that the message will be diffused and accepted by all of the nodes within the duration of the trace. We ran each algorithm 300 times and averaged the diffusion times of the compared algorithms.

To find the best fit for our simulations, we compared Euclidean, Tanimoto, Cosine, and Weighted Euclidean similarity metrics by adopting them in the diffusion process. Results show that all of the metrics produced similar diffusion times. Here, we present the results of using the Euclidean metric in our algorithms as an example as the metric does not require the calculation of additional weighting values and performs slightly better than Tanimoto and Cosine in diffusion time.

D. Simulation results

We first compared the diffusion times of DSF1 and DSF2. Results in the selection of both 5 and 10 diffusion nodes show that DSF2 has a shorter diffusion time than DSF1. This confirms that Definition (2) of the dynamic social features is more accurate than Definition (1) as it breaks the tie cases in Definition (1).

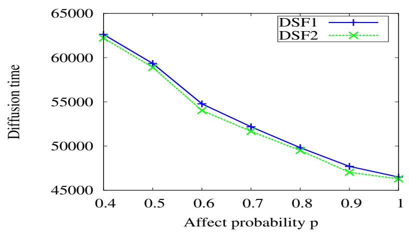
We then compared DSF1 and DSF2 with the Analy, the Static, and the Rand algorithms. With both 5 and 10 diffusion nodes, DSF1 and DSF2 algorithms consistently have shorter diffusion times than these algorithms. In most cases, the Static algorithm has a higher diffusion time than the Rand algorithm, especially when there are 10 diffusion nodes. These results verify the advantages of the dynamic social feature method over the network analysis method by including nodes' contact frequencies and social features, and over the static social feature method to more accurately capture users' dynamic contact behavior.

When nodes' affect probability increases from 0.4 to 1, the diffusion times decrease in all of the algorithms. This means that if nodes are more easily affected, then the diffusion time will be reduced.

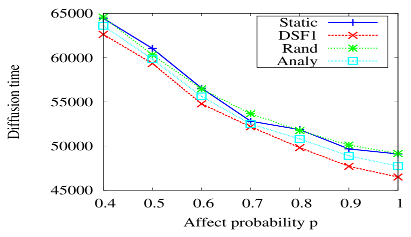
In summary, simulation results show that our algorithms based on dynamic social features have lower diffusion times than the algorithms that use network analysis methods, static social features or random selection. Furthermore the minimization of the diffusion time depends on the accuracy of the model to capture users' dynamic contact behavior.

VII. Conclusion

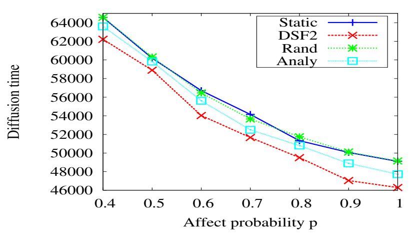
In this paper, we tackled the diffusion minimization problem by proposing new diffusion algorithms where diffusion



(a) Comparison of DSF1 and DSF2

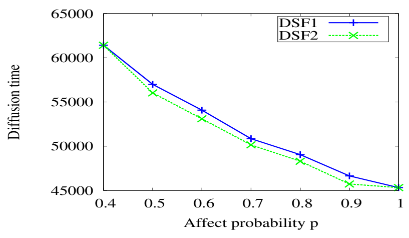


(b) Comparison of Analy, Static, DSF1, and Rand

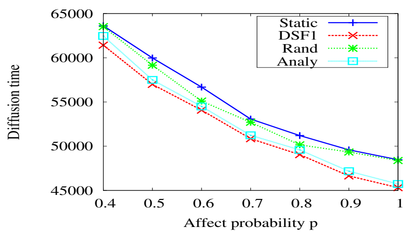


(c) Comparison of Analy, Static, DSF2, and Rand

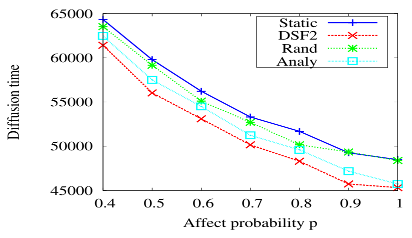
Fig. 4. Comparison of algorithms with 5 diffusion nodes



(a) Comparison of DSF1 and DSF2



(b) Comparison of Analy, Static, DSF1, and Rand



(c) Comparison of Analy, Static, DSF2, and Rand

Fig. 5. Comparison of algorithms with 10 diffusion nodes

nodes were selected based on dynamic social features and community structure in MSNs. Simulation results show that our algorithms have shorter diffusion times than the existing algorithms. Currently, the diffusion node selection algorithm uses the idea from the k-means algorithm to group nodes into communities. This is not the only algorithm that we can use. There are many community detection algorithms in the literature. For example, to make the algorithm more scalable, we can use the microclustering idea, which first groups nearby nodes into "microcluster" and then performs k-means clustering on the microcluster. But as our first attempt, our main focus in this paper is to explore the feasibility of using social features to minimize information diffusion in MSNs. We will leave the task of finding better community detection algorithms and evaluate them on Global Environment for Network Innovations (GENI) to the future.

Acknowledgments

This research was supported in part by NSF under CNS1305302, ACI1440637, and CNS1065665, and NSF/BBN under CNS1346688 for project 1936.

References

[1] Delay-tolerant networking. http://en.wikipedia.org/wiki/Delay-tolerant_networking.

[2] E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *ACM MobiHoc*, 2007.

[3] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of ACM SIGKDD*, 2001.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Company, 1979.

[5] J. W. Han, M. Kamber, and J. Pei. *Data Mining: concepts and techniques*. Morgan Kaufmann, MA, USA, 2012.

[6] T. Hossmann, T. Spyropoulos, and F. Legendre. From contacts to graphs: Pitfalls in using complex network analysis for dtn routing. In *IEEE Int. Workshop on Network Science For Communication Networks*, 2009.

[7] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: social-basedforwarding in delay tolerant networks. In *MobiHoc*, 2008.

[8] D. Kempe, Kleinberg J., and É. Tardos. Maximizing the spread of influence through a social network. In *ACM SIGKDD*, 2003.

[9] Z. Q. Lu, Y. G. Wen, and G Cao. Information diffusion in mobile social networks: The speed perspective. In *IEEE INFOCOM*, 2014.

[10] H. Ma, H. Yang, M. R. Lyu, and I. King. Mining social networks using heat diffusion processes for marketing candidates selection. In *ACM CIKM*, 2008.

[11] A. Mei, G. Morabito, P. Santi, and J. Stefa. Social-aware stateless forwarding in pocket switched networks. In *IEEE INFOCOM*, 2011.

[12] M. Motani, V. Srinivasan, and P. Nuggehalli. Peoplenet: engineering a wireless virtual social network. In *MobiCom*, pages 243–257, 2005.

[13] N. Ning, Z. Yang, H. Wu, and Han Z. Self-interest-drive incentives for ad dissemination in autonomous mobile social networks. In *IEEE INFOCOM*, 2013.

[14] W. Peng, F. Li, X. Zhou, and J. Wu. A privacy-preserving social-aware incentive system for word-of-mouth advertisement dissemination on smart mobile devices. In *IEEE SECON*, 2012.

[15] A. Pietilainen and C. Diot. Dissemination in opportunistic social networks: the role of temporal communities. In *ACM MobiHoc*, 2012.

[16] D. Rothfus, C. Dunning, and X. Chen. Social-similarity-based routing algorithm in delay tolerant networks. In *IEEE ICC*, pages 1862–1866, 2013.

[17] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. Crawdad trace cambridge/haggle/imote/infocom2006 (v.2009-05-29). http://crawdad.cs.dartmouth.edu/cambridge/haggle/imote/infocom2006, May 2009.

[18] C. Shannon, N. Petigara, and S. Seshasai. A mathematical theory of communication. *Bell System Technical Journal*, 27(1):379–423, 1948.

[19] V. Srinivasan, M. Motani, and W. Ooi. Analysis and implications of student contact patterns derived from campus schedules. In *MobiCom*, pages 86–97, 2006.

[20] S. Vishwanathan. An o(log*n) approximation algorithm for the asymmetric p-center problem, 1996.

[21] Y. Wang, G. Cong, G. Song, and K. Xie. Community-based greedy algorithm for mining top-k influential nodes in mobile social networks. In *ACM SIGKDD*, 2010.

[22] J. Wu and Y. Wang. Social feature-based multi-path routing in delay tolerant networks. In *IEEE INFOCOM*, 2012.

[23] Y. Xu and X. Chen. Social-similarity-based multicast algorithm in impromptu mobile social networks. In *IEEE GLOBECOM*, 2014.