Warehousing Massive Mobile Datasets

Zhipeng Liu
College of Information Science and Technology, Nanjing University of Aeronautics and Astronautics, Yudao Street 29, Nanjing, Jiangsu, 210016, P.R. China
Email: liuzhipengcs@139.com

Dechang Pi
College of Information Science and Technology, Nanjing University of Aeronautics and Astronautics, Yudao Street 29, Nanjing, Jiangsu, 210016, P.R. China
Email: dc.pi@nuaa.edu.cn

***Abstract*—Nowadays, scientists can collect and analyze massive mobile data generated by various sensors and applications of smart phones. smart phones have become an important platform for the understanding of social activities, such as community detection, social dynamics and influence. It is extremely important to store and retrieve mobile data efficiently for various data mining tasks. In this paper, we propose Mobile Data Warehouse (*MobileDW*) model which is based on *GraphChi,* a system designed for large-scale graph computation on one PC. We propose multi-shard data structure and Time-based Parallel Sliding Windows (*TPSW*) to store *Social* data such as call logs and SMS. We further propose Mobile Index (*MIndex*) structure and Mobile Position Compression Algorithm (*MPCA*) to warehouse *Position* data such as GPS, Bluetooth etc. The *MIndex* structure can compress *Position* data significantly. The data compression process is based on the following observations: (1) The position of the individual users within a certain period of time often unchanged. (2) A crowd of people tend to move and stay together. Experimental results demonstrate the effectiveness and efficiency of Mobile Data Warehouse.**

***Index Terms*—MobileDW; MIndex; MPCA, TPSW**

I. Introduction

With the development of high-speed telecommunication techniques such as 3G and 4G network, smart phones have been widely deployed. More and more applications have been developed on smart phone platforms. Modern cell phones are embedded with many modules, such as GPS, WLAN and Bluetooth devices. These sensors and applications generate massive mobile data. Mobile data includes call logs, WLAN, GPS, Bluetooth data, etc. These data contains rich topological as well as location information which can be utilized to extract useful information. Scientists have already collected and analyzed massive mobile data. Here, we present two examples:

Example 1: MIT Reality mining project [1]. It has been conducted since 2005. One hundred teachers and students have been involved in this project during data collection period in one year. The number of people remained stable during the experiment. It records almost 450,000 hours of information. The recorded information includes: call logs, SMS logs, Bluetooth data, application usage information and so on. These information has been thoroughly studied and many useful information, such as community structure, friendship relationship and daily behavior can be inferred from it.

Example 2: The Lausanne data collection campaign[2]. This project has been conducted by Nokia Research Center, Idiap and EPFL. It has been conducted since 2009, and it concluded in 2011. Smart phones with data collection software were allocated to 185 volunteers. However, some volunteers quitted during data collection period. The data collection campaign produced a massive dataset with 240,227 calls, 175832 SMS, 37151 photos, 31,013,270 WLAN observations, 2,940 videos, etc. Researches such as next place prediction, semantic place prediction and demographic prediction have already been carried out.

From information above we can observe that it is important for scientists to collect and analyze massive mobile data, and researchers are increasingly interested in mobile data mining to find some interesting patterns, such as inferring dynamic communities in call log graphs, mining structural patterns in time-evolving social networks and spatio-temporal dynamics. In order to support these mining algorithms, it needs a mobile data warehouse to store and retrieve mobile data. However, due to complex data types and large quantities of mobile data, it is difficult to implement mobile data warehouse.

Many data warehouse have been proposed with the emergence of new types of data, such as MoveMine[3], RFID data warehouse[4] and InfoNetOLAP[5]. MoveMine[3] is designed to process movement data, with the emergence and advancement of GPS and wireless technologies. It includes three parts: movement data collection and cleaning, data mining and visualization. The main part of MoveMine is data mining. It includes moving object pattern mining and trajectory mining. Moving object pattern mining includes periodic pattern mining, swarm pattern and movement interactions mining. Trajectory mining includes trajectory clustering, trajectory classification and trajectory outlier detection. These algorithms can be modified and applied to deal with location data collected by mobile phones. RFID data warehouse[4] processes RFID data, which is usually generated by logistics. The trajectories of RFID data can be presented and stored as virtual path among different places, and RFID data warehouse is mainly deals with virtual path compression. However, mobile position data is usually recorded as real paths generated by position data sampling, and RFID data compression algorithms can not directly be applied to mobile position data. InfoNetOLAP [5] deals with complex social networks such as DBLP. It handles informational as well as topological data. Traditional OLAP methods ignore links among data objects. InfoNetOLAP proposes graph cube model to aggregate the network data in all possible dimensional spaces. However, InfoNetOLAP does not handle location data, while mobile data includes position data. We can conclude that these newly proposed as well as traditional data warehouse technologies can not directly deal with mobile data due to its complex data structures and various data types. MobileMiner[6] proposes mobile data warehouse framework. It takes moving record stream and calling record stream as input, uses profile mining platform to perform data cleaning; then it performs novel data mining techniques such as sequence mining, clustering, graph mining and classification with domain knowledge to support the application layer. The application layer of MobileMiner includes mobile customer segmentation, social community discovery, churn prediction and customer re-entry analysis. However, it only proposes the overall framework of mobile data storing and retrieving issues, while the implementation details have not been thoroughly discussed.

II. Mobile Data Warehouse (MobileDW)

Mobile data can be classified into three types: social data, position data and other data. we propose Mobile Data Warehouse (*MobileDW*) model to store and retrieve massive mobile data efficiently and effectively on one PC. The relationship of different types of mobile data in *MobileDW* is depicted in Figure 1.

MobileDW contains a fact table *Link*, that stores cleansed mobile data, such as phone number and personal ID; a dimensional table, *Social*, which is composed of call logs and SMS messages, call logs is composed of phone numbers of both communication terminals, directions and the start and the end time of calls. SMS messages is composed of phone numbers of both communication terminals, directions and time stamp information; a dimensional table, *Position*, which stores mobile position information such as GPS, WLAN and Bluetooth data. GPS and WLAN data records the location data with time stamp information of the related phone number, while Bluetooth data records the proximity information with time stamp information of mobile phone users; *Info*, which stores personal information of mobile users and smart phones. These information is independent of time, and remain stable for a long period of time; *Other* includes information of audios and videos with timestamp, length and format information.

We do not discuss how to store and retrieve *Other* information in this paper.

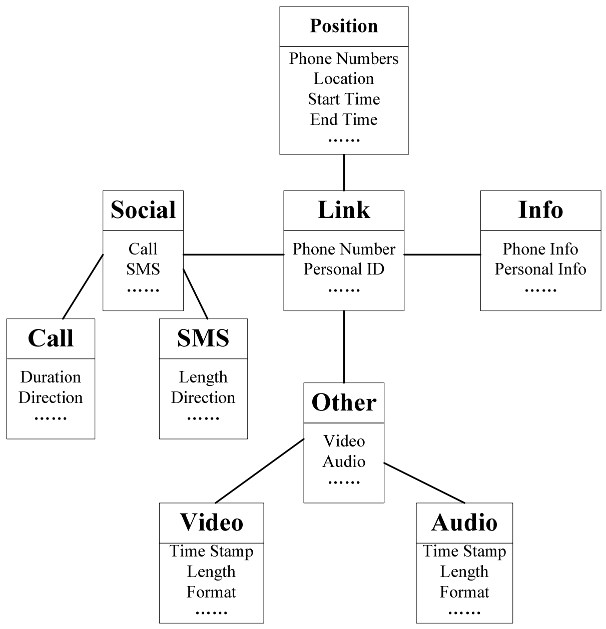

Figure 1. Snowflake schema fact table of *MobileDW*

III. Warehousing Social data

In this section, we present the mechanism of warehousing *Social* data.

A. Social Data Concepts

Definition 1 *Social* network model is composed of three types of network data, where $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3)$. $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ denote call logs, SMS and phone book entries respectively. Each call log entry has a state among: (1)in, which represents incoming call; (2) out, which represents outgoing call; and (3)missed, which represents missed phone call. Each SMS message owns a state among (1)in, which represents incoming SMS message; (2)out, which represents outgoing SMS message; (3)failed, which represents that the destination of the SMS message is unreachable; (4)pending, which represents that the state of the SMS message is undetermined.

Suppose the time span of $\mathcal{G}$ is T . $\mathbb{G}_i = (G_{i1}, G_{i2}, ...G_{im})$ denotes a series of m mobile network stream segments, where $i \in [1..3]$. $G_{ij} = (V_{ij}, E_{ij})$ is a directed static graph, where v and e denote vertexes and edges respectively. $v_{ij} \in V_{ij}$, $e_{ij} \in E_{ij}$. Each vertex v_{ij} is associated with a collection of attributes $\{A_{1j}, A_{2j}, ..., A_{nj}\}$.

Figure 2 depicts the transition of a $\mathbb{G}_1$ network snapshots $G_{11} \sim G_{13}$. $V_{11} \sim V_{15}$ denote 5 users in the call log graph, and the directed edges represent phone calls among the users.

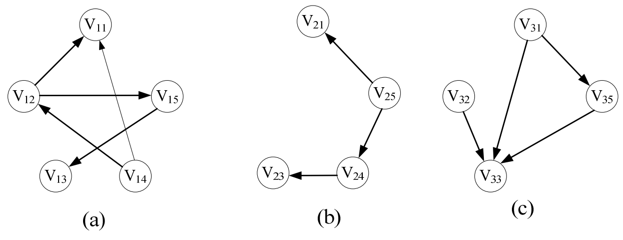

Figure 2. Snapshots of $G_{11} \sim G_{13}$

Definition 2 The informational dimensions of *Social* network model $\mathbb{G}_i$ is composed of informational attributes $\{A_{1j}, A_{2j}, ..., A_{nj}\}$ associated with V_{ij}.

Definition 3 The topological dimensions of *Social* network model $\mathbb{G}_i$ is composed of a series of snapshots $G_{i1}, G_{i2}, ..., G_{ij}$ with attributes.

B. Social Data Compression Solutions

GraphChi [7] is a disk-based system which can perform large-scale graph computation on one PC. It is based on vertex-centric computation model. It partitions large graphs into small ones by a data structure named *shard*. It uses Parallel Sliding Windows(*PSW*) to process graph data efficiently. *PSW* is composed of 3 parts: loading the graph, parallel updates and updating graph to disk. It has already been able to execute many data mining algorithms, such as PageRank, triangle counting and matrix factorization. We extend *GraphChi* to store and index *Social* data in two ways. First, we propose multi-shard data structure to store and retrieve *Social* data efficiently. Second, we proposed Time-based Parallel Sliding Windows (*TPSW*) which is based on *PSW* to support high-level computations.

Multi-shard Structure

The design of multi-shard data structure is based on *shard* in *GraphChi*. Given a specified time interval t, we partition $\mathbb{G}$ into $U(U=\lceil T/t \rceil)$ time intervals. Note the length of the last time interval may be less than t. For each interval I_w ($w \in [1,U]$), Three shards S_{w1}, S_{w2} and S_{w3} are allocated to store call logs, SMS messages and phone book entries respectively. Each shard stores all edges that have destination in I_w. Multi-shard structure is depicted in Figure 3.

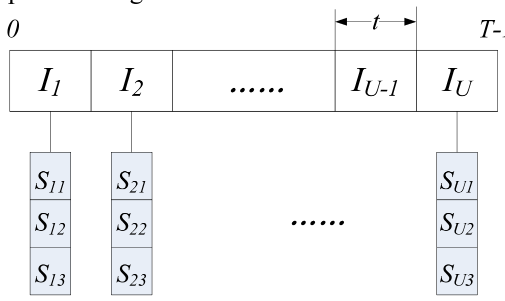

Figure 3. Multi-shard data structure

Since mobile data conforms to power-law distribution, we then arrange high in-degree nodes and low in-degree nodes in different shards. Intervals are chosen to balance the number of edges in each shard. The construction of graph $\mathbb{G}$ is depicted in Algorithm 1.

Algorithm 1 The construction of graph $\mathbb{G}$

```
CreateGraph(G, t)
        G:      The file of original graph
        t:      Time span of each interval
        M :     The amount of system memory
        I :     The list of intervals
1.   U=⌈T / t⌉
2.   for i := 1 to U
3.   //Partition Interval(i). $I_{Max}$ and $I_{Min}$ are two Intervals in main memory.
4.   // $I_{Max}$ for high in-degree nodes, $I_{Min}$ for low in-degree nodes.
5.          if Indegree( $G_{ij}$ ) > threshold
6.             append( $G_{ij}$ , $I_{Max}$ )
7.          else
8.             append( $G_{ij}$ , $I_{Min}$ )
9.          if $I_{Max} + I_{Min} >= M$
10.            temp := $I_{Max} >= I_{Min}$ ? $I_{Max}$ : $I_{Min}$
11.         for j := 1 to 3
12.            Process temp, construct multi-shard.
13.         end for
14.  end for
15.  for i :=1 to U
16.         BuildIndexes(I[i])
17.  end for
```

An example of *Social* data compression solution is depicted in Figure 4. S_{ij} denotes the *j* type shard in the *i*-th interval. Index structures are built for these shards. $Node_1$ to $Node_3$ stores information from I_1 to I_P respectively. The node *root* always resides in main memory to facilitate search speed. $Node_1$ to $Node_3$ is implemented as a buffer tree[8] by procedure *BuildIndexes*.

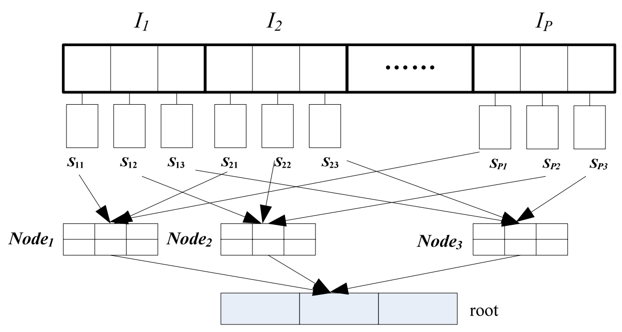

Figure 4. An example of *Social* data compression solution

Time-based Parallel Sliding Windows (*TPSW*)

The main idea of *GraphChi* is based on Parallel Sliding Windows (*PSW*). It can process graph data efficiently on one PC. However, *PSW* can not be directly adopted to store *Social* data for two reasons. First, *PSW* can not handle all kinds of mobile data types, such as *Social* data. Second, *PSW* does not take the temporal aspects of edges of mobile data into consideration. We propose Time-based Parallel Sliding Windows (*TPSW*) to store and index *Social* data.

For each edge of $G_{ij} = (V_{ij}, E_{ij})$, *TPSW* uses compressed sparse row (*CSR*) $e_{ij} = < v_i, v_j, w_{ij}, ts_{ij}, d_{ij} >$ to store the edges of the graph on disk. If e_{ij} represents call log, ts_{ij} and d_{ij} denote the start time and duration. If e_{ij} represents SMS message, ts_{ij} denotes the timestamp and d_{ij} = 0. w_{ij} denotes the weight between v_i and v_j.

Three rules are used to arrange edges in multi-shard structure by *TPSW* model.

Rule 1: In each multi-shard data structure, *CSR*s are sorted by v_i in ascending order.

Rule 2: For each node v_i, *CSR*s are sorted by v_j in ascending order.

Rule 3: If there are multi-edges exist between v_i and v_j, *CSR*s are sorted by ts_{ij} using last generated first stored method. This means recent communication event has greater impact on mobile users.

IV. Warehousing Position data

In this section, we present the general framework of warehousing *Position* data. The compression methods are based on the MoveMine data warehousing scheme[3].

A. Mobile Index Structure (MIndex)

MIndex is an indexing data structure to facilitate the search speed of *Position* data. *MIndex* is composed of phone number (denoted by *PN*) which can be used to identify one user uniquely. The contents are composed of *Info*s and time. *Info* field stores information of the associated phone number, which does not change constantly with time. This information can not be stored in first level index due to bulk memory usage. Time-dependent position information is stored in *P* field with specified time. If *m* is large, the node is split into two nodes with approximately equal number of sampling time points. The *MIndex* data structure is depicted in Figure 5.

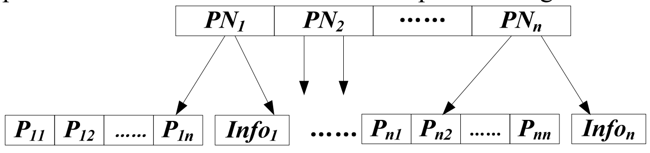

Figure 5. *MIndex* data structure

B. Mobile Position Compression Algorithm (MPCA)

We propose *MPCA* for single user in mobile network. The intuition behind the algorithm is based on following observation: The position of a mobile phone user can remain the same for a period of time. For example, the *Position* information of a user at home or during work place.

MPCA traverses the trajectories of each user, finds out the intervals in which position information are within range ε. The distance between two time intervals is calculated by Euclidian distance. If the time length of each interval is greater than *thres*, then *MPCA* compresses position data within the range using the average position value. ε and *thres* are two user predefined values. the *MPCA* algorithm is depicted in Algorithm 2.

Algorithm 2 Mobile Position Compression Algorithm (*MPCA*)

```
1. $cnt$ :=0
2. $maxc$ :=0
3. for $data$ in range $P_{i,1} .. P_{i,n}$
4.     $middle$ := center point of $P_{i,j}$ and $P_{i,j+1}$
5.     if $data_{ij}$ is in $\varepsilon$-range of $middle$
6.         ++$cnt$;
7.     if $data_{ij}$ not in $\varepsilon$-range of $middle$
8.         if $maxc < cnt$
9.             $maxc := cnt$
10.            $maxp := j - cnt + 1$;
11.        $cnt = 0$;
12.    if $maxc > thres$
13.        $N$:= uildNode($average(P_{i,maxp} \sim_{maxp+maxc})$)
14.        Replace nodes from $P_{i,maxp}$ to $P_{i,maxp+maxc}$ with $N$
```

V. Experimental Evaluation

We developed *MobileDW* based on the data structures and algorithms described in previous sections. Our experiments were conducted on a Pentium(R) D 3.0 GHz PC with 1 GBytes of main memory, running on CentOS 4.5 operating system. We implement our algorithm in C++ using GCC 4.5.4. *MobileDW* is based on *GraphChi* 0.2.1. It can perform various graph computation efficiently[9-11]. For the performance comparison experiments, we use the MIT Reality [1] and Nodobo [12] datasets. The preprocessing process transforms original data to multi-shards. Table 1 depicts data preprocessing of experimental graphs. The time interval is defined as 1 month in our experiments. Call logs, SMS messages and GPS data are used to evaluate the performance of *MobileDW*.

TABLE I.

Data preprocessing

Graph name	Nodes	*U*	Preproc. (min)
Reality	100	5	3
Nodobo	27	3	18

A. System Performance

Here, we demonstrate that *MobileDW* can handle large scale mobile data on one PC. We use influence model in [13] to test the performance. We implement the influence algorithm in two ways: (1) STL-based. The mobile data is stored in STL containers. (2) *MobileDW*. Figure 6 shows the performance of influence model on Reality dataset. The scale of Reality dataset is relatively small, so it can fit into main memory without many paging operations. The performance of *MobileDW* improves 30.0~34.4%. Figure 7 shows the performance of influence model on Nodobo dataset. Since the Nodobo dataset contains more data than Reality dataset, the STL-based version need much more time than *MobileDW* version due to the frequent paging operations. Thus, the performance of *MobileDW* improves 70.1~73.5%.

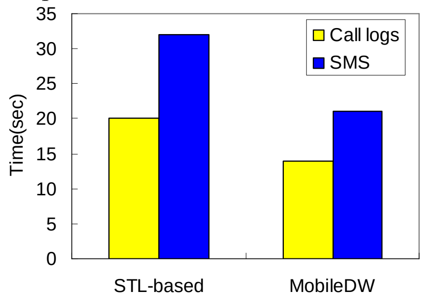

Figure 6. Performance of Influence model on Reality dataset

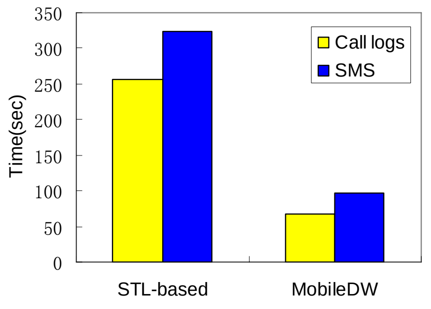

Figure 7. Performance of Influence model on Nodobo dataset

B. MPCA

We choose 5 mobile users in Nodobo dataset to test the performance of *MPCA*. These people are randomly chosen and sorted according to data scale. The parameter list is ε=10，*thres* = 18. Figure 8 shows the effect of *MPCA* on different mobile users. *MPCA* can compress original *Position* data (represented by Raw) very efficiently, ranging from 33.3~80%. Since *MPCA* uses ε -neighborhood criterion during compression process, MPCA is a lossy compression algorithm.

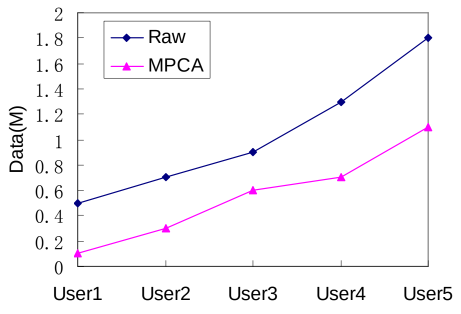

Figure 8. Performance of MPCA

VI. CONCLUSIONS

Mining information in mobile social network becomes a hot topic. It is important to warehouse mobile data for different mining tasks. In this paper, we proposed *MobileDW*, for storing and retrieving mobile data. Mobile data is classified into two main categories: *Social* and *Position* data. We proposed Multi-shard data structure and Time-based Parallel Sliding Windows method to warehouse *Social* data. We also proposed Mobile Index structure and Mobile Position Compression Algorithm to compress *Position* data. Experimental results demonstrated that *MobileDW* can store and retrieve mobile data efficiently and effectively. We are currently investigating into the detailed issues as a further study.

ACKNOWLEDGMENT

This research is sponsored by Qing Lan Project, the 333 project of Jiangsu Province， the Aeronautical Science Foundation of China (No. 20111052010) and 2012 Jiangsu Graduates Innovation Project (CXZZ12_0163).

REFERENCES

[1] N. Eagle, A. Pentland, “Reality mining: sensing complex social systems,” *Journal Personal and Ubiquitous Computing* Vol.10, pp. 255 - 268. March 2006.
[2] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and Laurila J “Towards rich mobile phone datasets: Lausanne data collection campaign,” *Proc ICPS*, Berlin 2010.
[3] Z. Li, M. Ji, JG. Lee, LA. Tang, Y. Yu, J. Han, and R. Kays, “MoveMine: mining moving object databases,” In *ACM*, pp. 1203-1206, 2010.
[4] J. Han, H. Gonzalez, X. Li, and D, Klabjan, “Warehousing and mining massive RFID data sets,” *Advanced Data Mining and Applications*, pp. 1-18, 2006.
[5] C. Li, PS. Yu, L. Zhao, Y. Xie, and W. Lin, “InfoNetOLAPer: Integrating InfoNetWarehouse and InfoNetCube with InfoNetOLAP”, *Proc of VLDB*, vol. 4, 2011.
[6] T. Wang, B. Yang, J. Gao, D. Yang, S. Tang, H. Wu, K Liu, and J. Pei, “Mobileminer: a real world case study of data mining in mobile communication,” In *ACM*, pp. 1083-1086, 2009.
[7] A. Kyrola, G. Blelloch, and C. Guestrin, “GraphChi: Large-scale graph computation on just a PC,” In *OSDI*, 2012.

[8] L. Arge, "The buffer tree: A new technique for optimal I/O-algorithms," *Algorithms and Data Structures*, pp.334-345, 1995.

[9] C. Cheng, Z. Chunhong, Q. Xiaofeng, and J. Yang, "The Study of the Improvement Mechanism of the Efficiency of Social Network," *Journal of Computers*, vol. 8, pp. 3230-3237, 2013.

[10] J. Weitao, T. Huixian, "A Study towards Application-driven Social Network Analysis," *Journal of Computers*, vol. 9, pp. 134-145, 2014.

[11] Y. Zong-chang, "User-Online Load Movement Forecasting for Social Network Site Based on BP Artificial Neural Network," *Journal of Computers*, vol. *8*, pp. 3176-3183, 2013.

[12] S. Bell, A. McDiarmid, and J. Irvine, "Nodobo: Mobile phone as a software sensor for social network research". In *IEEE*, pp. 1-5, 2011.

[13] A. Java, P. Kolari, T. Finin, and T. Oates, "Modeling the spread of influence on the blogosphere," *Pro. of WWW*, pp.22-26, 2006

Zhipeng Liu was born in 1980. He received the B.S. and M.S. degree in computer science and technology from Nanjing University of Posts and Telecommunications, in 2002, 2005, respectively. Now, he is a Ph.D. candidate in Nanjing University of Aeronautics and Astronautics. His research interests are data mining and distributed systems.

Dechang Pi was born in 1971. He was a Ph.D. of Nanjing University of Aeronautics and Astronautics (NUAA) of China and now he is a professor and Ph.D. supervisor in NUAA. His research interests are data mining and database systems etc.