BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **15**, No 2

Sofia • 2015

Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2015-0032

Privacy-Preserving of Check-in Services in MSNS Based on a Bit Matrix

Chen Wen

School of Mathematics and Computer Science, Tongling College, Tongling, P. R. China
Email: tlxychenwen@163.com

Abstract: *Check-in service, being one of the most popular services in Mobile Social Network Services* (*MSNS*)*, has serious personal privacy leakage threats. In this paper check-in sequences of pseudonym users were buffered, and a bit matrix for buffered check-in sequences was built, which can achieve privacy guarantee of k-anonymity. The method guarantees that the number of lost check-in locations is minimized while satisfying users' privacy requirements. Besides, it also reduces the cost of finding a trajectory k-anonymity set. At last, the results of a set of comparative experiments with* (k, δ)*-anonymity on real world datasets show the method accuracy and efficiency.*

Keywords: *Privacy preservation, location privacy, trajectory privacy, mobile social networks.*

1. Introduction

Check-in service is the mainstream application of mobile social networks. The users use a mobile device with a positioning function to send their geographical position to the mobile social network server, and select the semantic location corresponding to the geographical position for a check-in. However, the privacy disclosure has become the primary obstacle for a check-in service [1]. The trajectories formed by the users' check-in records over a period of time can cause the exposure of the users' tracks [2]. The attackers can analyze the users' behaviour patterns of that day, then speculate on the users' identity, and can even predict the users' future trajectory. Trajectory anonymization [3], by way of dummy location, reconstructs the true tracks for privacy preserving. Data suppression [4, 5] restricts the release of a certain sensitive position on the trajectory or it restricts the trajectory segments

that may lead to disclosure of sensitive information. Generalization is the mainstream privacy preserving technology of trajectory with trajectory *k*-anonymity [6-8] as its representation. The privacy preserving technology of location based on the location service and the privacy preserving technology of the trajectory data have achieved some research results, but they cannot be directly applied to mobile social networks. In recent years researchers have made studies on the privacy in mobile social networks, such as the location inference attack [9] in mobile social networks, the location privacy preserving and the absence privacy preserving [10] in mobile social networks, the location privacy preserving of proximity based services [11, 12], and so on. Making use of the prefix trees, the authors in [13] have achieved privacy protection of trajectory *k*-anonymity. This paper uses the means of the buffer user check-in sequences proposed in [13], constructs a bit matrix to achieve *k*-anonymity privacy preserving. The experiment has proved that the method is more efficient.

2. System structure

This paper uses the central server system structure proposed in [13]. The central server system structure is composed of a client, a privacy preserving server and a service provider. Presuming that the privacy preserving server and users are reliable, only pseudonym users in the system use the privacy preserving service. Before using the check-in service, the users are required to send a registration request to the privacy protection server. The user registration module is in charge of the user registration and storage of the personalized privacy preserving parameters set by the users. These parameters mainly include: privacy preserving parameters k, the longest tolerance time Δt, a sensitive location privacy region. When ready to sign, the user first sends a "pre-sign" command to the privacy preserving server. Then the pre-processing module judges if the location submitted by the user is the exact geographical location or a semantic location. If it is the semantic location, the module will directly buffer the user's check-in location; if it is the geographical location, the module will make an anonymous inquiry to the service provider, gain access to a point of interest nearby and then return it to the user, who can select the proper point of interest for another pre-sign. The preprocessing module is also in charge of deleting the sensitive location privacy region of check-in sequences. According to the user check-in sequences, the privacy preserving module constructs a bit matrix, gains access to *k*-anonymity sequences and signs to the service provider.

3. The generation algorithm of *k*-anonymity check-in sequence

Definition 1. Check-in Sequence. Check-in records of a certain position of a user u_i may be expressed as (L_i, t_i), where L_i indicates ID of the registration location and t_i represents the registration time. The check-in records constitute a check-in sequence of the user u_i in a chronological order.

Definition 2. *k*-anonymity check-in sequences. Given the user u_i, the predefined privacy parameters k and the longest tolerance time Δt, if and only if there are other $k-1$ users which have the same check-in sequence with u_i within Δt time, the check-in sequence is named as k-anonymity check-in the sequence of u_i.

The generation algorithm of k-anonymity check-in sequence comprises the following steps:

Step 1. Preprocessing of the check-in sequence: Delete the location belonging to a predefined sensitive location privacy region in the users' check-in sequence;

Step 2. Sort the check-in sequence according to the check-in location ID, build a bit matrix and finally get k-anonymity which could be signed.

Step 1 is implemented by a preprocessing module, not described herein. At Step 2 the paper transforms the k-anonymity problem into a frequent item sets mining problem based on a bit matrix, deletes the check-in position which is less than k and gets k-anonymity check-in sequence. The following check-in sequence examples linked with Fig. 1 (a) are used to introduce the **algorithm steps:**

Step 1. Establish the bit vector BV_{L_i} for each attendance record L_i. If the check-in records L_i appear in the check-in sequence of the user u_j, the value of the bit vector of j-th position is set to 1, otherwise it is 0. The bit vector of the check-in sequence shown in Fig. 1a is

$$\mathrm{BV}_{L_1} = \{11\ 111\ 100\ 000\},\ \mathrm{BV}_{L_2} = \{11\ 100\ 011\ 110\},\ \text{etc.}$$

Step 2. Establish the identity matrix: Create $M \times M$ matrix $\boldsymbol{A}$, M is equal to the number of the users u_i. The matrix is initialized to 0. If the number "1" of $\mathrm{BV}_{L_i} \wedge \mathrm{BV}_{L_j}$ $(i < j)$ is not less than the number of the privacy protection parameters k, then set $\boldsymbol{A}[i, j] = 1$, $\boldsymbol{A}[M, j] = \boldsymbol{A}[M, j]+1$, otherwise $\boldsymbol{A}[i, j] = 0$.

For example, assume that the privacy parameter k is 3, $\mathrm{BV}_{L_1} \wedge \mathrm{BV}_{L_2} = \{11\ 100\ 000\ 000\}$, the number of "1" in the result is not less than k, so $\boldsymbol{A}[1, 2]$ is set to 1, $\boldsymbol{A}[11, 2]$ is set to 1. The matrix constructed according to the above method is expressed in Fig. 1b. In the bit matrix, except for the last line, if $\boldsymbol{A}[i, j]$ is equal to 1, record the values of i and j and then form a sequence which is denoted as S_2. From Fig. 1b we can obtain:

$$S_2 = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{1, 5\}, \{2, 7\}, \{2, 8\}, \{3, 4\}, \{3, 5\}, \{4, 5\}, \{7, 8\}\}.$$

Step 3. Expand the last item of the subsequence in the sequence S_n to generate a sequence S_{n+1}. Let $S_n = \{i_1, i_2, ..., i_n\}$, if $\boldsymbol{A}[i_n, i_u] = 1$ $(i_u > i_n)$, $\boldsymbol{A}[M, i_u] \geq n$, and $\boldsymbol{A}[i_1, i_u] = \boldsymbol{A}[i_2, i_u] = ... = \boldsymbol{A}[i_{n-1}, i_u] = 1$, then $\{i_1, i_2, ..., i_n\}$ is expanded into $\{i_1, i_2, ..., i_n, i_u\}$. If the number of "1" in the result of $\mathrm{BV}_{L_{i1}} \wedge \mathrm{BV}_{L_{i2}} \wedge \ldots \wedge \mathrm{BV}_{L_{in}} \wedge \mathrm{BV}_{L_{iu}}$ is not less than the privacy parameters K, then $\{i_1, i_2, ..., i_n, i_u\}$ is recorded into S_{n+1} sequence. Extend the sequence successively until a new sequence could not be expanded.

For example, in connection with the bit matrix in Fig. 1b, analyze the specific process of the sequence extension. Assume that the privacy parameter k is 3, $M = 11$, when $n = 2$, the subsequence $\{1, 2\}$ of S_2 is extended to S_3. $\boldsymbol{A}[2, 3] = 1$, $\boldsymbol{A}[11, 3] = 2$ is not less than n, $\boldsymbol{A}[1, 3] = \boldsymbol{A}[2, 3] = 1$. $\mathrm{BV}_{L_1} \wedge \mathrm{BV}_{L_2} \wedge \mathrm{BV}_{L_3} = \{11\ 100\ 000\ 000\}$ the number of "1" is not less than the number of the privacy

parameters k, thus $\{1, 2, 3\} \subset S_3$. Extend each subsequences, the final sequence set obtained is as follows:

$$S_3=\{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 3, 5\}, \{1, 4, 5\}, \{3, 4, 5\}, \{2, 7, 8\}\},$$
$$S_4=\{\{1, 2, 3, 4\}, \{1, 3, 4, 5\}\},$$
$$S_5=\Phi .$$

Step 4. Restore the sequence in S_n ($n \geq k$) into the check-in sequence S'_n. Compare the check-in sequence L_i and S'_n, get the k-anonymity sequence which could be signed by the longest common subsequence.

Definition 3. The longest common subsequence. Given two sequences S_1 and S_2, if there is a sequence S_{sub} which meets $S_{sub} \subset S_1$ and $S_{sub} \subset S_2$, and there is not a subsequence $S'_{sub} \subset S_{sub}$ meeting the above conditions, then S_{sub} is called the longest common subsequence of S_1 and S_2.

For example, $k = 3$, S_3, S_4 are reverted to check-in sequences:

$$S'_3=\{\{L_1, L_2, L_3\}, \{L_1, L_2, L_4\}, \{L_1, L_3, L_4\}, \{L_1, L_3, L_5\}, \{L_1, L_4, L_5\}, \{L_3, L_4, L_5\}, \{L_2, L_7, L_8\}\},$$
$$S'_4 = \{\{L_1, L_2, L_3, L_4\}, \{ L_1, L_3, L_4, L_5\}\}.$$

Traverse the sign sequence in Fig. 1, select S'_n having the same length subsequence according to the check-in sequence length of u_i to make a comparison and get the longest common subsequence. For instance, to get the longest common subsequence between the check-in sequence of u_6 and S'_4, the result is u_6: $L_1 \rightarrow L_3 \rightarrow L_4 \rightarrow L_5$; u_7's check-in sequence length is 3, to get the longest common subsequence with S'_3, the result is u_7: $L_2 \rightarrow L_7 \rightarrow L_8$. Ultimately, the sign obtained on the k-anonymous sequence is shown in Fig. 1c.

u_1:	L_1	L_2	L_3	L_4	L_5	0	1	1	1	1	0	0	0	0	0	0
u_2:	L_1	L_2	L_3	L_4		0	0	1	1	0	0	1	1	0	0	0
u_3:	L_1	L_2	L_3	L_4		0	0	0	1	1	0	0	0	0	0	0
u_4:	L_1	L_3	L_4	L_5		0	0	0	0	1	0	0	0	0	0	0
u_5:	L_1	L_3	L_4	L_5		0	0	0	0	0	0	0	0	0	0	0
u_6:	L_1	L_3	L_4	L_5	L_6	0	0	0	0	0	0	0	0	0	0	0
u_7:	L_2	L_7	L_8			0	0	0	0	0	0	0	1	0	0	0
u_8:	L_2	L_7	L_8			0	0	0	0	0	0	0	0	0	0	0
u_9:	L_2	L_7	L_8	L_9		0	0	0	0	0	0	0	0	0	0	0
u_{10}:	L_2	L_7	L_{10}			0	0	0	0	0	0	0	0	0	0	0
u_{11}:	L_3	L_4	L_5			0	1	2	3	3	0	1	2	0	0	0

(a) (b)

u_1:	L_1	L_2	L_3	L_4
u_2:	L_1	L_2	L_3	L_4
u_3:	L_1	L_2	L_3	L_4
u_4:	L_1	L_3	L_4	L_5
u_5:	L_1	L_3	L_4	L_5
u_6:	L_1	L_3	L_4	L_5
u_7:	L_2	L_7	L_8	
u_8:	L_2	L_7	L_8	
u_9:	L_2	L_7	L_8	
u_{11}:	L_3	L_4	L_5	

(c)

Fig. 1. Check-in sequence (a); bit matrix (b); k-anonymized check-in sequences (c)

The following description is given based on a bit matrix for k-anonymity check-in sequence algorithm.

Algorithm. k-anonymized check-in sequences generation algorithm

Input: Check-in sequences L_i, privacy preserving parameters k

Output: k-anonymized check-in sequences

For (j=1; j<=M; j++)

/* M is equal to the number of users */

for all item i in L_j do

set the j-th bit of BV_{L_j} to 1;

for all i in L_i do

for every two items i, j ($i < j$) do
if (the number of 1 in $BV_{L_i} \wedge BV_{L_j}$)>=k
{$\boldsymbol{A}[i, j]$=1; $\boldsymbol{A}[M, j]$= $\boldsymbol{A}[M, j]$+1; $S_2 = S_2 \cup \{i, j\}$;}
/* Expand the last item of the subsequence in the sequence S_n to generate a sequence S_{n+1}*/
n=2;
for all sequences $\{i_1, i_2, ..., i_n\} \subset S_n$ do
{
n=n+1;
if (i_n<M)
for i_u= i_n +1 to M do
if ($\boldsymbol{A}[i_n, i_u]$=1&& $\boldsymbol{A}[M, i_u]$>=n)
if ($\boldsymbol{A}[i_1, i_u]$= $\boldsymbol{A}[i_2, i_u]$=…=$\boldsymbol{A}[i_{n-1}, i_u]$=1)
if ((the number of 1 in
$BV_{L_{i1}} \wedge BV_{L_{i2}} \wedge \ldots \wedge BV_{L_{in}} \wedge BV_{L_{iu}}$)>=$k$)
$S_{n+1} = S_{n+1} \cup \{i_1, i_2, ..., i_n, i_u\}$
}
If (n>=k)
transform S_n to S'_n
for all L_i do
L'_i=LCS(L_i, $S'_{|Li|}$)
/* LCS is the longest common subsequence function, $|L_i|$ is the length of check-in sequences, L'_i is the k-anonymity check-in sequence*/

4. Experimental results and analysis

4.1. Experimental data

The algorithm of this paper uses Java, running on E5800 3.2 GHz processor and Windows XP platform with 2 G memory which also use true check-in data on Brightkite of [14]. The data sets are collected data from 24 months and the data set properties are shown in Table 1. We use k-anonymity algorithm [6] as a comparison algorithm.

Table 1. Dataset properties

Check-in number of locations	Total number of users	Regional area, km^2	User density
541 169	9435	443 556	0.02
The average number of check-in	POI average number of check-in	Average check-in Interval, h	Average check-in distance interval, km
57.36	6.10	56.81	15.39

The parameters of the experiment include the privacy protection parameters k, the check-in sequence Length ***len*** and the maximum tolerable time Δt. The check-in success rate c_s reflects the proportion of the users' original check-in location

contained in the *k*-anonymity check-in sequence and uses the next formula to measure:

$$c_s = \frac{|k\mathrm{ChS}_i\mathrm{L}| - ||k\mathrm{ChS}_i\mathrm{L}| - |\mathrm{ChS}_i\mathrm{L}||}{|k\mathrm{ChS}_i\mathrm{L}|}, \tag{1}$$

where $|k\mathrm{ChS}_i\mathrm{L}|$ denotes the number of the check-in locations of *k*-anonymity check-in sequence; $|\mathrm{ChS}_i\mathrm{L}|$ indicates the number of check-in locations of the original check-in sequence.

The experiment randomly selects the check-in sequence in the data set from 4000 users and does not consider the changes in the maximum tolerance time, while testing the influence on the check-in success rate from the privacy protection parameters *k* and the sequence Length ***len***. This algorithm is denoted by BMC (Bit Matrix Check-in privacy preserving algorithm). Fig. 2 shows the check-in success rate of the algorithm.

4.2. Result analysis

4.2.1. Privacy protection parameters and check-in success rate

Fig. 2 shows the impact on the check-in success rate by the privacy parameters *k*. The values of the privacy protection parameters *k* increases from 5 up to 12. As it can be seen from Fig. 2a, the check-in success rate decreases with the increase of the privacy parameters *k*. The increases in *k* cause more losses of the check-in location, which results in the decrease of the check-in success rate. Meanwhile, the increases in the check-in sequence due to a failure of anonymity of (*k*, *δ*)-anonymity algorithm also lead to a decline in the check-in success rate. From the comparison experiment we can get the idea that the success rate of the algorithm in this paper is much higher.

4.2.2. Length of the check-in sequence and check-in success rate

The value of the sequence length of the location fluctuates between 5 and 30; it increases by 5 each time. The value of the location sequence length relates to the dataset properties. Selecting the check-in sequence randomly from 4000 users, the experiment intercepts and complements the value randomly that satisfies the length requirement. From Fig. 2b, when *k* equals 10, the increase of the check-in sequence length leads to the decrease of the algorithm check-in success rate. The reason is that the longer the location sequence is, the less the identical location sequence will be, which may result in a greater loss of location. Due to the limited impact of the increase of the sequence length on the trajectory distance and clustering results, the growth of the sequence length barely affects the check-in success rate of (*k*, *δ*)-anonymity.

4.2.3. The longest tolerance time and check-in success rate

Fig. 2c indicates the impact of the longest tolerance time on the check-in success rate. The longest tolerance time is the time of a cache check-in sequence. In the analysis of the data set attributes, the paper gets the time interval of the check-in

data set through an experiment as Table 1 shows. The value of the longest tolerance Δt is 1-5 times of the average check-in time. In the experiment, the value of k is 10, there are no specific restrictions on the check-in sequence length. In Fig. 2c the increase of the check-in success rate of the algorithm is accompanied by the growth of the longest tolerance time. Besides, the more the cache of the check-in sequence is, the easier the k-anonymous check-in sequence is generated; while the less the check-in sequence is, the greater loss in the check-in location k-anonymous sequence is generated. So does the method of (k, δ)-anonymity: the more the cache of the location sequence is, the more possible is the search for a k-anonymous set through a cluster.

According to the experiments, the method introduced in the paper significantly outperforms the method of (k, δ)-anonymity in the success rate of the check-in.

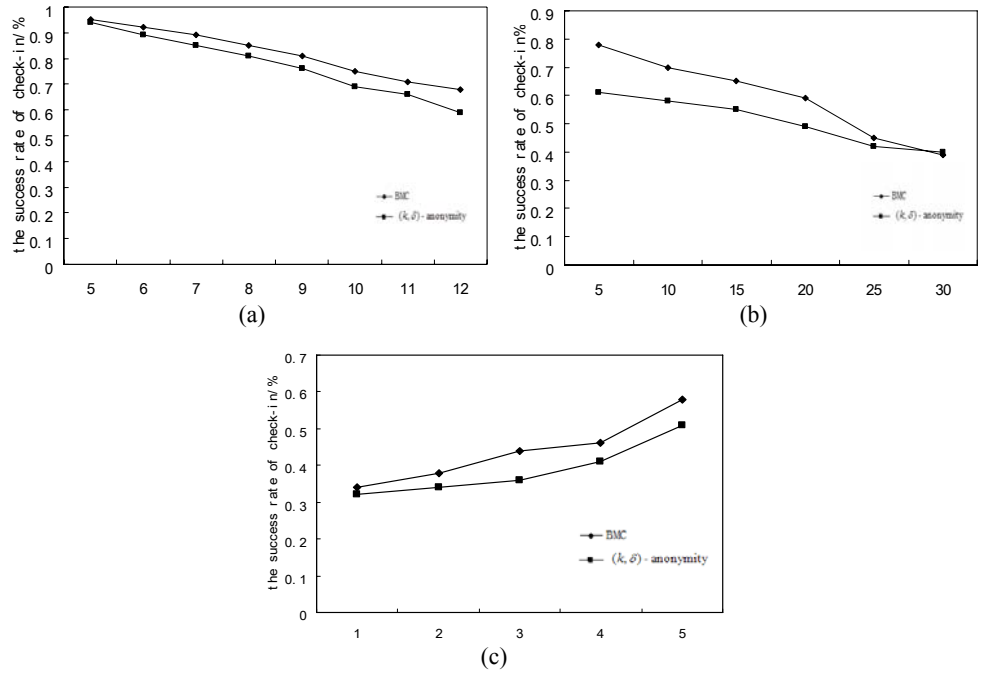

Fig. 2. Experimental results: the parameter k (a); the length of check-in sequences (b); the longest tolerance time t (c)

5. Conclusions

Connected with the problem of privacy disclosure of the location and trajectory of pseudonym users in mobile social networks, the paper presents tools for privacy protection based on a bit matrix which ensures both the trajectory privacy of the users and a higher success probability of the check-in. Through transformation of the trajectory k-anonymity into k-frequent item sets mining based on the bit matrix, the method deletes the check-in location whose support degree is less than k and gets the check-in sequence of k-anonymity. The method not only simplifies the

searching process of the trajectory *k*-anonymity set and saves the storage by a bit matrix, but also reduces the calculation and improves the privacy protection.

***Acknowledgements*:** This work was supported by funds from the Universities Key Fund of Anhui Province for Young Talents of China under Grant 2013SQRL082ZD and Natural Science Research Universities Key Project of Anhui Province of China under Grant KJ2014A256.

References

1. Gruteser, M., D. Grunwald. Anonymous Usage of Location-Based Services through Spatial and Temporal Cloaking. – In: Proc. of 1st International Conference on Mobile Systems, Applications and Services. ACM, 2003, pp. 31-42.
2. Xiao-Feng, H. U. O. Z. M. A Survey of Trajectory Privacy-Preserving Techniques. – Chinese Journal of Computers, Vol. **10**, 2011.
3. Huo, Z., X. Meng, H. Hu, et al. You Can Walk Alone: Trajectory Privacy-Preserving through Significant Stays Protection. – In: Proc. of Database Systems for Advanced Applications. Springer Berlin Heidelberg, 2012, pp. 351-366.
4. You, T. H., W. C. Peng, W. C. Lee. Protecting Moving Trajectories with Dummies. – In: Proc. of Mobile Data Management, 2007 International Conference on. IEEE, 2007, pp. 278-282.
5. Terrovitis, M., N. Mamoulis. Privacy Preservation in the Publication of Trajectories. – In: Proc. of Mobile Data Management. (MDM'08) 9th International Conference on IEEE, 2008, pp. 65-72.
6. Abul, O., F. Bonchi, M. Nanni. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. – In: Proc. of Data Engineering (ICDE'2008) IEEE 24th International Conference on IEEE, 2008, pp. 376-385.
7. Nergiz, M. E., M. Atzori, Y. Saygin. Towards Trajectory Anonymization: A Generalization-Based Approach. – In: Proc. of SIGSPATIAL ACM GIS'2008 International Workshop on Security and Privacy in GIS and LBS. ACM, 2008, pp. 52-61.
8. Yarovoy, R., F. Bonchi, L. V. S. Lakshmanan et al. Anonymizing Moving Objects: How to Hide a MOB in a Crowd? – In: Proc. of 12th International Conference on Extending Database Technology: Advances in Database Technology. ACM, 2009, pp. 72-83.
9. Sadilek, A., H. Kautz, J. P. Bigham. Finding Your Friends and Following Them to Where You Are. – In: Proc. of 5th ACM International Conference on Web Search and Data Mining. ACM, 2012, pp. 723-732.
10. Freni, D., C. Ruiz Vicente, S. Mascetti et al. Preserving Location and Absence Privacy in Geo-Social Networks. – In: Proc. of 19th ACM International Conference on Information and Knowledge Management, ACM, 2010, pp. 309-318.
11. Mascetti, S., D. Freni, C. Bettini et al. Privacy in Geo-Social Networks: Proximity Notification with Untrusted Service Providers and Curious Buddies. – The VLDB Journal – The International Journal on Very Large Data Bases, Vol. **20**, 2011, No 4, pp. 541-566.
12. Mascetti, S., C. Bettini, D. Freni et al. Privacy-Aware Proximity Based Services. – In: Proc. of Mobile Data Management: Systems, Services and Middleware, 2009. MDM'09. Tenth International Conference on IEEE, 2009, pp. 31-40.
13. Huo, Z, X. F. Meng, Y. Huang. PrivateCheckIn: Trajectory Privacy-Preserving for Check-in Services in MSNS. – Jisuanji Xuebao, Chinese Journal of Computers, Vol. **36**, 2013, No 4, pp. 716-726.
14. Cho, E, S. A. Myers, J. Leskovec. Friendship and Mobility: User Movement in Location-Based Social Networks. – In: Proc. of 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2011, pp. 1082-1090.