Variable social vector clocks for exploring user interactions in social communication networks

Ta-Yuan Hsu*

Department of Electrical and Computer Engineering,
University of Illinois at Chicago,
Chicago, IL 60607, USA
Email: thsu4@uic.edu
*Corresponding author

Ajay D. Kshemkalyani

Department of Computer Science,
University of Illinois at Chicago,
Chicago, IL 60607, USA
Email: ajay@uic.edu

Abstract: Social network communication analysis has drawn widespread attention in recent years. Vector clocks can be applied to capture the most recent communication with all other local peers in a social network. A modification of conventional social vector clocks has been previously proposed to deal with the issue of poor scalability without keeping whole temporal views. In this paper, our proposed framework maintains the low bound of how out-of-date each peer can be with respect to others, and also considers the shortest friendship separation to restrict how far information may be transmitted along time-respecting paths. To quantitatively analyse the influence of user interactions over different limitations of friendship distance, we also provide an adaptive incremental updating approach that can exactly recover the real situation in a specified upper bound of friendship distance. Experimental results also show that social vector clocks can be efficiently exploited to improve memory space requirements.

Keywords: vector clocks; social networks; *n*-degree separation; Twitter; social influence; space-based computing.

Reference to this paper should be made as follows: Hsu, T-Y. and Kshemkalyani, A.D. (2015) 'Variable social vector clocks for exploring user interactions in social communication networks', *Int. J. Space-Based and Situated Computing*, Vol. 5, No. 1, pp.39–52.

Biographical notes: Ta-Yuan Hsu is a PhD student in the Department of Electrical and Computer Engineering at the University of Illinois at Chicago, USA. His research interests include distributed algorithms, social networks and causal memory.

Ajay D. Kshemkalyani received his MS and PhD in Computer and Information Science from The Ohio State University in 1988 and 1991, respectively. He is currently a Professor in the Department of Computer Science at the University of Illinois at Chicago. His research interests are in distributed computing, distributed algorithms, computer networks, and concurrent systems. He has served on the editorial board of *IEEE Transactions on Parallel and Distributed Systems*, and the *Elsevier Journal*, *Computer Networks*.

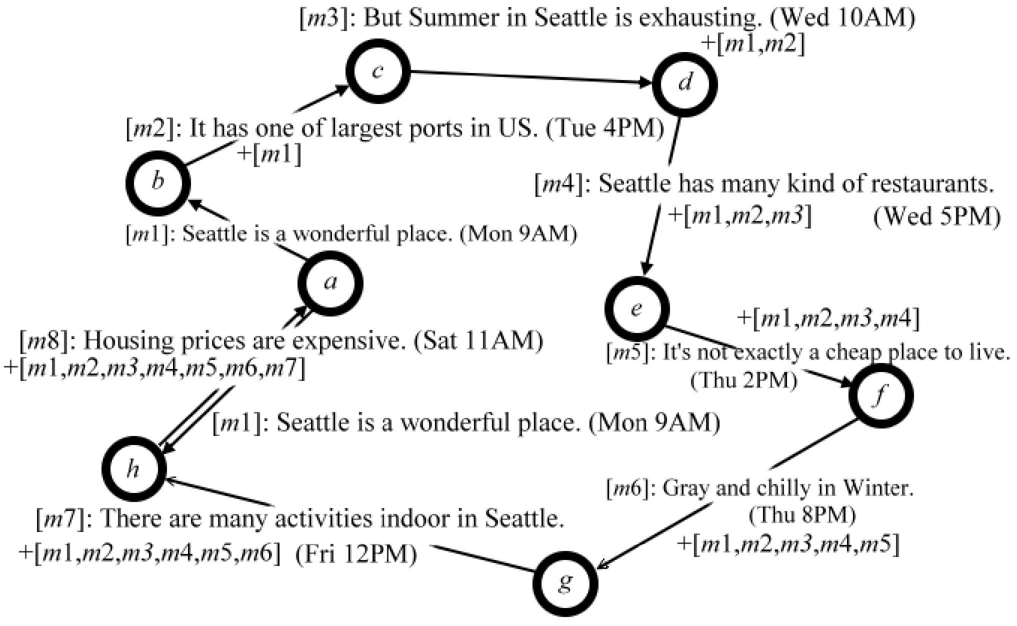
This paper is a revised and expanded version of a paper entitled 'Modeling user interactions in social communication networks with variable social vector clocks' presented at WAINA'2014: International Conference on Advanced Information Networking and Applications Workshops, IEEE, 2014, Victoria, British Columbia, Canada, 13–16 May 2014.

1 Introduction

The use of online communication for large social communities and networks has been a growing concern in a variety of disciplines. For example, larger social networks act as tubes for information flows and messaging communication (Huberman and Adamic, 2004; Granovetter, 1973). Most of the social networking services not only provide a common platform to associate individuals through some characteristic profile, such as ideas, activities, backgrounds, interests, or events, but they also allow them to interact with each other over the internet. A huge number of indications show that social network activities have affected and changed the physical world (Tang et al., 2009; Lewis and Nicholes, 2012; Wright and Hinson, 2012). It also explicates that the potential of social networking usage and social media technologies is deemed very high and quickly grows over time. For the sake of analysing user communication in social networks, it can be visually represented as a dynamic graph (Federico et al., 2011; Berger-Wolf and Saia, 2006; Koylu et al., 2012). Technically, it has been a challenging task to formulate complete data flows for event-driven communication in large-scale networks over a long period of time. The main reason is that event-driven data flows may be highly correlated with the variety of timing and ordering of events. Based on prior knowledge, the fine-grained temporal approach has been proposed to figure out the dynamic inter-communication in a social network through a temporal framework (Holme and Saramäki, 2012). By applying the fine-grained temporal modalities, some researches have begun studies to explore information path latency and indirect message exchanging in social communication networks. They applied the formula of *vector clocks* (VCs) from the field of distributed systems to realise the temporal infrastructure. The notion of VCs was conceptually introduced by Mattern (1988). It can be used to track the lower bound of how recent each process's state is associated with any other one at a given time. Under a partial ordering of events in such a concurrent system, VCs can reflect how a given pair of events are causally related with each other.

Kossinets et al. (2008) figured out how information is diffused in social communication networks. By applying VCs to social networks, they introduced a framework of social vector clocks (SVCs) to mark the latest information available to each actor at each timestamp. As for information diffusion, a two-step information pathway can be faster than a one-step path in real social networks. Through SVCs, communication flows were visualised over time (Harrigan, 2010). They have been used to design a temporal framework to explore the structure of information pathways (Kossinets et al., 2008). The SVC temporal updating scheme can provide an important mechanism capable of maintaining the contents and timestamps of the latest communication for each peer. Without considering the subject matter of information or messages, an ordering of timestamps can present a global view for user interactions. As with Lamport timestamps, inter-process messages contain the local state of the sending process's logical clock.

Figure 1 Illustrative example of social network communication for eight people


[m3]: But Summer in Seattle is exhausting. (Wed 10AM)
+[m1,m2]
c
d
[m2]: It has one of largest ports in US. (Tue 4PM)
+[m1]
b
[m4]: Seattle has many kind of restaurants.
+[m1,m2,m3] (Wed 5PM)
[m1]: Seattle is a wonderful place. (Mon 9AM)
a
e
+[m1,m2,m3,m4]
[m8]: Housing prices are expensive. (Sat 11AM)
+[m1,m2,m3,m4,m5,m6,m7]
[m5]: It's not exactly a cheap place to live. (Thu 2PM)
f
[m1]: Seattle is a wonderful place. (Mon 9AM)
h
[m6]: Gray and chilly in Winter. (Thu 8PM)
+[m1,m2,m3,m4,m5]
[m7]: There are many activities indoor in Seattle.
+[m1,m2,m3,m4,m5,m6] (Fri 12PM)
g

Consider an example as shown in Figure 1. Suppose that there are eight people that talked about the location of their vacation. Figure 1 shows a complete series of direct and indirect communication messages over six days. They can be referred to as user-directed mentions in Twitter. Let member *a* serve as a coordinator for the review programme of travelling issues and make decisions in collecting the other members' comments. Member *a* began by sending message m1 to members *b* and *g* on Monday morning. Member *b* told member *c* *m*2 plus an item of piggybacked information *m*1 at 4 PM on Tuesday afterward. Clearly, the coordinator *a* would finally receive the message *m*8 from member *h* and all the other latest indirect communication information *m*2, *m*3, *m*4, *m*5, *m*6, *m*7 on Saturday morning. Although member *a* did not directly contact with most of all the others, he would learn about the latest opinions in the group at 11 AM on Saturday. This scenario can be realised by indirect message updating using the nature of SVCs. It illustrates that all indirect up-to-date communications between any pair of members can be saved and tracked by SVCs, but with quadratic space requirements. It also causes that some of the indirectly exchanged messages may be insignificant for most members in a huge social network group. Consider a case that member *a* talked to another member *i* something about his professional life irrelevant to travelling issues on Sunday morning. Member *i* would receive all communication messages exhibited in Figure 1, even if he did not have interactions with those members directly. Apparently, it is unrealistic to preserve all peers' indirect updated messages in qualitatively analysing user interactions using the conventional SVCs in social groups.

In Dunbar and Hill (2005), it is demonstrated that cognitive constraints may not impact the size of the whole social network. One modification to the SVCs, in addition to shortest time-respecting paths, has been proposed to formulate the fine-grained temporal features applicable to large-scale social interaction networks with better scalability for link prediction (Lee et al., 2013). The amount of reachable indirect incoming information is subject to the shortest friendship distance μ (i.e., the minimum number of hops) from the source to the target on time-respecting paths. It insinuates that some information may be lost and inconsistent when the upper bound of μ is less than the number of overall members. For example, as shown in Figure 1, if μ is bounded to be '1' such that most of the indirect updates will be discarded, member *a* may acquire inconsistent information. In such a case, member *a* can obtain only one message from member *g*. This observation infers that the information loss rate is apparently dependent on the value of μ.

1.1 Contributions

In this paper, we further extend the modification of SVCs to be variable social vector clocks (VSVCs) and apply VSVCs to quantitatively model the influence of the restriction of μ on the reachable information. Here, we focus on five medium or large social groups from a certain microblogging network. They include a list of UK athletes organised by *The Telegraph*, a list of past and present *MLB* players, a list of members in *SXSWi* created by a commercial platform *mashable*, a group of followers following the latest information posted by *UICnews*, and a list of journalists curated by *TheNewYorkTimes* on Twitter, where participants can explicitly dispatch messages to targeted receivers. Twitter networking has been widely used to deal with several analyses (Kwak et al., 2010; Mao et al., 2011; Cha et al., 2010).

In order to improve the performance of running SVCs with the fixed upper bound of μ, we propose an incremental adaptive approach to update VSVCs. It requires only one running cycle for the complete dataset. It can exactly recover the practical situation in different upper bounds of μ. We present the reasonable compromise of information loss rate and memory space requirement, and then the distribution of separation topology for incoming reachable messages. We also demonstrate that our results for information loss rates and incoming reachable topology distributions are consistent.

1.2 Organisation

This paper is organised as follows. In Section 2, we review the definition and updating algorithm for the conventional VCs and specify our full extension of SVCs. In Section 3, we describe our methodology – how VSVCs can be applied to our experiments. In Section 4, we present our experiment results and evaluation. Finally, we summarise the conclusion and discuss some future research directions in Section 5.

2 Vector clocks

We begin with a brief review of conventional VCs in distributed systems (Mattern, 1988). Then, we present an underlying introduction about the concept of SVCs and the modification of SVCs. Finally, we provide a description of how to apply the framework of VSVCs to social networks.

2.1 Conventional VCs

There are many existing works studying the conventional VCs in the distributed system literature (Mattern, 1988; Raynal and Singhal, 1996; Kshemkalyani and Singhal, 2008). Basically, to establish VCs in the system, each process $P_i \in P$ (P is the set of processes in the system) maintains a VC V_i of n (number of processes in the system) integers, which is updated by the following rules.

1 Before an internal event happens at process P_i, $V_i[i] = V_i[i] + 1$.

2 Before process P_i sends a message, it first executes $V_i[i] = V_i[i] + 1$, then it sends the message piggybacked with V_i.

3 When a process P_j receives a message with timestamp U from P_i, it executes $\forall k \in [1 \dots n]$, $V_j[k] = \max(V_j[k], U[k])$; $V_j[j] = V_j[j] + 1$; before delivering the message.

Conventional VC in a distributed system is used to track the causal relation ($\prec$) between two events that happened in the system by comparing their corresponding VC timestamps, i.e., $e_i \prec e_j \Leftrightarrow V_{e_i} < V_{e_j}$, where $V_{e_i} < V_{e_j}$ means $\forall a \in [1, n]$, $V_{e_i}[a] \leq V_{e_j}[a]$ and $\exists b \in [1, n]$ such that $V_{e_i}[b] < V_{e_j}[b]$.

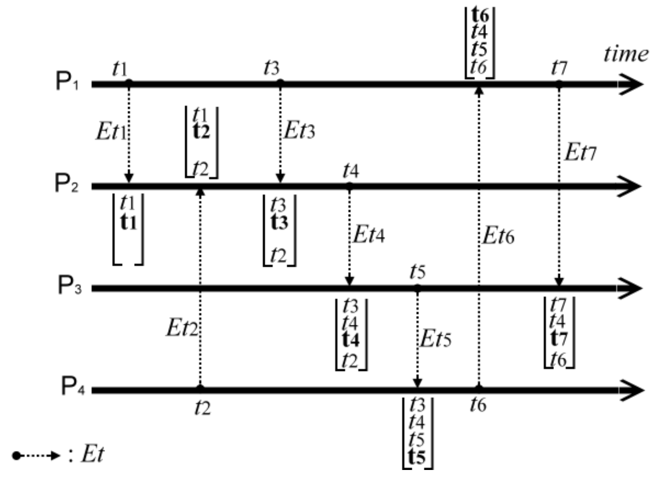
Furthermore, the conventional VC can also be expressed as a multivariate function on a three-tuple (t, s, r), which is (time, sender, receiver). P_i's temporal view of every process in the system at time t is defined as a function $\phi_{i,t} = (\phi_{i,t}(j)$: $j \in P)$ where $\phi_{i,t}(j)$ is P_i's temporal view of a particular process P_j at time t. Here, $\phi_{i,t}(j)$ corresponds to the j^{th} entry of P_i's local vector timestamp V_i when $V_i[i] = t$.

2.2 Traditional SVCs and the modification

Based on the structure of our social communication data, the data are expressed as the following: Given a set of N peers in a social network, we pre-define a time interval $[0 \sim T]$, over which a complete sequence of communication events are organised in terms of the global time ordering. Each event is composed of a multivariate function on a three-tuple (t, s, r) as per the definition given in Section 2.1. Suppose that there is no communication delay and there exists one global synchronous time. Note that the above two constraints are not applicable to the framework of asynchronous distributed systems.

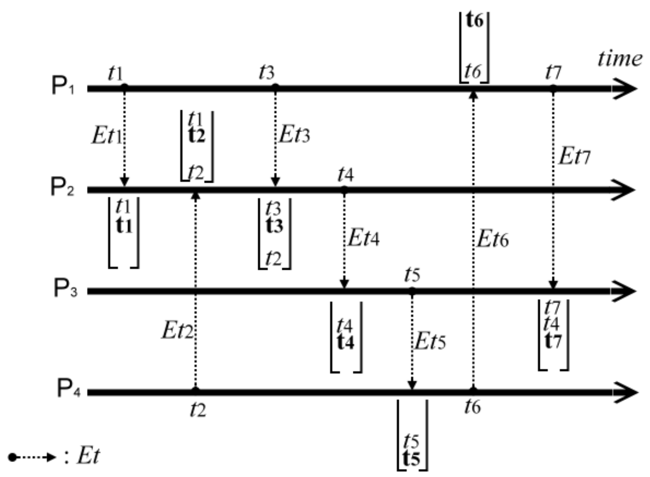
The traditional SVC updating approach practically follows the mechanism of the conventional VCs. Due to the assumption of no propagation delay in social networks, when receiving an incoming communication event (E_t) sent in time slice t from peer i, the timestamp of the receiver peer j's temporal view of the i^{th} entry is set to peer j's VC on the j^{th} entry and the sending timestamp of E_t. Figure 2 illustrates how to update the entries on SVCs. For instance, when P_4 receives E_{t5} sent from P_3 at t_5, $V_3[3] = t_5$ and $V_3[4] = t_5$ as well. By the rule 3 in the conventional VCs, $V_3[1]$ and $V_3[2]$ are set to t_3 and t_4, respectively.

Figure 2 Illustrative example of social network communication for four peers with the conventional SVCs



For N peers in a social system, traditional SVCs take up $O(N^2)$ globally to support the latest fine-grained temporal patterns. Under the piggyback system, each peer will soon get a large number of indirect updating messages from other peers, most of whom the receiver does not have any direct communication with ever, or has too far social-connection steps in between them. Therefore, it may not be efficient and practical to straightforwardly utilise the function of SVCs in modelling user social interactions; otherwise, huge amounts of impractical information will be generated in steady state.

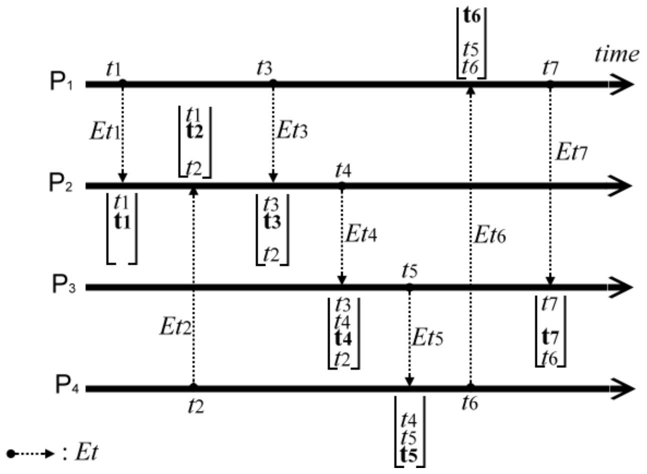
Figure 3 Illustrative example of social network communication for four peers with μ being 1



A further modification of the updating framework for SVCs has been addressed in Lee et al. (2013), where a parameter μ is considered as the minimum number of hops between a pair of sending peer and receiving target along time-respecting paths and included in the framework of the traditional SVCs. The semantics of three major different values assigned to μ are as follows:

- $\mu = 1$: This case is concerned with direct friendship communication. A receiver can update a component of the local SVC based on the incoming message if and only if the corresponding sender for that component ever directly interacts with the receiver (an incoming communication is a message with the VC piggybacked on the message). The same communication example in Figure 2 is modified as shown in Figure 3.
- $\mu = 2$: This case considers friendship-of-friendship indirect communication. A receiver j can update the k^{th} component of the local SVC based on the k^{th} component of the piggybacked timestamp directly sent from a peer i if and only if the corresponding indirect sender k has ever directly interacted with the sender i. The illustrative example in Figure 2 is replaced with Figure 4.
- $\mu = \infty$ (it is enough for μ being N–1 in a N-peer social group): it is equivalent to the conventional SVC updating approach, considering unlimited indirect communication spread without self-looping updating, as show in Figure 2.

Figure 4 Illustrative example of social network communication for four peers with μ being 2

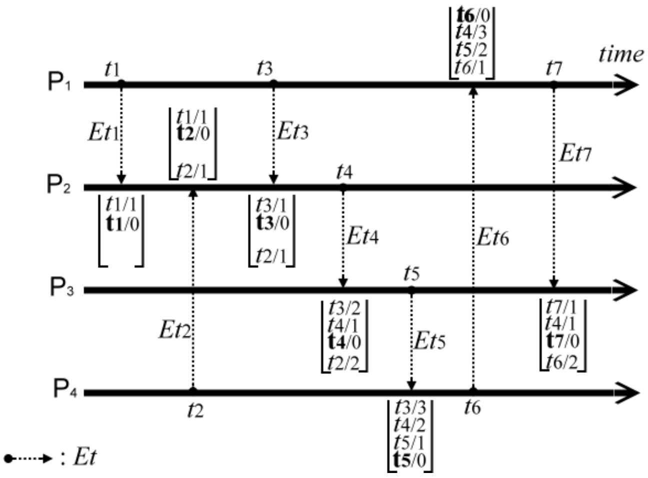


2.3 Variable social VCs

The above three major social relationships were presented in Lee et al. (2013). In order to consider all different reachable distances of friendship, we define a universal framework of the VSVCs in social networks, as follows (Hsu et al., 2014).

Without loss of generality, assume that when $\mu = c$, peer *j* sent a direct message to peer *i*. Peer *i* can receive an indirect update on the k^{th} component of the local SVC based on the k^{th} component of the piggy back timestamp via a direct update from a peer *j*, if and only if the maximum number of hops from peer *k* to peer *j* is $c - 1$.

Figure 5 Illustrative example of social network communication for four peers with VSVCs



Whenever a receiver updates its own latest temporal scopes, it also computes the shortest friendship distances with respect to all the other peers. $V_i[j]$, the j^{th} entry of peer *i*'s SVC, needs to maintain two types of data elements.

1 $V_i[j]$.*time* captures the latest timestamp of peer *j* at peer *i*.

2 $V_i[j]$.*dist* measures the shortest friendship-respecting path.

As shown in Figure 5, each entry in a peer is represented by |*t*/*d*|. In the j^{th} entry of P_i, *t* means $V_i[j]$.*time* and *d* denotes $V_i[j]$.*dist*. When an incoming event E_t occurs sent from peer *j* to peer *i* at time slice *t*, then

1 $V_i[j].time = E_t.timestamp$

2 $V_i[k].dist = \min(V_j[k].dist + 1, V_i[k].dist)$ when $k \neq j$.

Figure 5 illustrates an example of applying VSVCs to the case in Figure 2. In this paper, we use the framework of VSVCs to analyse how the social communication is affected by the upper bound of the minimum number of hops (μ).

3 Methodology

Our evaluation seeks to investigate the impact of the hop-constrained paths on social communication. The scope of this research lies in the characteristics of user interactions with different limitation friendship distances (μ). However, the traditional SVC updating approach has redundant computation complexity $O(mn)$ (where *n* and *m* represent the number of vertices and communication events, respectively). It is also impractical to run the entire series of communication events for each different μ. We design an efficient accumulative VSVC updating algorithm keeping track of the minimum number of hops the information travels between a pair of sender and receiver targets. It only takes time $O(m)$. Whenever one communication event E_t happens, triggered from a sender peer (E_t.*sender*) to a receiver target (E_t.*receiver*), the receiver target's VC needs to be updated. For each update, the shortest friendship distance from the sender to the receiver needs to be computed as well.

We now describe our evaluation methodology of manipulating VSVCs with the shortest friendship distances, by first presenting the conceptual basis of *information loss rate* and *reachable incoming distribution*, and then presenting the updating process proposed by the methodology. The algorithm pseudocode of *VariableSocialVectorClocks* is shown in Algorithm 1 and explained in Section 3.3.

3.1 Information loss rate

Initially, we define the following terms:

- *Message*: It is a direct piece of communication sent from a sender to a receiver.
- *Information*: Each component on the VSVC piggybacked with a sending message is defined as a piece of information.

VariableSocialVectorClocks described in Section 3.3 is an incremental algorithm to update both the latest timestamp and the shortest distance at each time slice *t*. In an *N*-peer social network group, when μ is bounded to be *N*–1, it is equivalent to the conventional SVC algorithm with unlimited communication spread. Intuitively, the number of pieces of updating information in peer i with μ being k can be defined as $DIST_i[k]$.

When the shortest friendship distance is k, the number of pieces of receiving and updating information is:

$$DIST[k] = \sum_{i=0}^{N-1} DIST_i[k] \tag{1}$$

The total number of pieces of receiving information without losing any communication in a social group is:

$$N_M = \sum_{k=1}^{N-1} DIST[k] \tag{2}$$

Only when k is equal to or less than μ will this information be received. We can compute the information loss rate R_L corresponding to the specified value of μ as:

$$R_L(\mu) = 1 - \frac{\sum_{k=1}^{\mu} DIST[k]}{N_M} \tag{3}$$

$R_L(\mu)$ can also be referred to as the space saving rate. Interestingly, it decreases as the value of μ increases. With this view, we can gain insight into how to determine the upper bound of μ to efficiently utilise memory space without losing reasonable communication in a social network.

3.2 Reachable incoming topology distribution

Figure 6 gives an illustrative example of an incoming communicating pattern for peer P. If μ is one, only five peers having ever sent direct messages to the central peer P can communicate with it. As the value of the limitation of μ increases, the number of peers whose updating messages can reach peer P will increase. In our methodology, we also make an analysis of the reachable incoming topology distribution in the steady state. We would learn about the number of peers (N_p) whose sending information can reach one peer p (at the latest time slice T) with respect to the limitation of μ based on the up-to-date value of $V_p[i].dist$ (using the VCs $\{V_0, \ldots, V_{N-1}\}$ in Algorithm 1). $V_p[i].dist$ means the shortest friendship distance from peer i to peer p. By summing up the value of x_i (x_i: a unit step function)

$$N_p(\mu) = \sum_{i=1}^{N} x_i \begin{cases} x_i = 1 & \text{if } V_p[i].dist \le \mu;\ p \ne i \\ x_i = 0 & \text{otherwise} \end{cases} \tag{4}$$

If $x_i = 1$, a sending message from peer i can reach peer p; otherwise, it is unreachable even though there exists a directed connection path. Because of the variation of individual peers, we normalise the value of N_p for each peer:

$$N_{avg}(\mu) = \frac{\sum_{p=0}^{N-1} N_p(\mu)}{N} \tag{5}$$

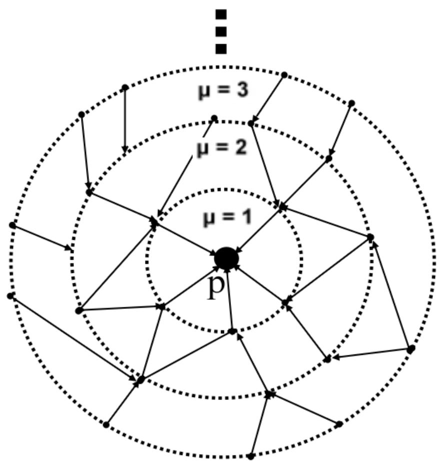
We refer to $N_{avg}(\mu)$ as the 'average reachable incoming distribution number'. Apparently, the value of N_{avg} positively depends on the upper bound of μ. Therefore, the average reachable incoming rate can be defined as:

$$R_I(\mu) = \frac{N_{avg}(\mu)}{N} \tag{6}$$

If social communication is in a steady-state, $R_I(\mu)$ should be close to $1–R_L$. On the other hand, the reachable incoming rate is highly dependent on the social interaction strength in a group. Naturally, there must be a certain number of people who did not send or receive any message to and from others. $N_{\max}(\mu)$ is defined as the maximum number of peers from whom one peer has received information with μ in the same group. We use $N_{\max}$ to denote $N_{\max}(\infty)$. Therefore,

$$R_{\max}(\mu) = \frac{N_{\max}(\mu)}{N} \tag{7}$$

Figure 6 Illustrative example of reachable incoming with different limitation distances μ



Furthermore, the normalised average reachable incoming rate is defined as follows:

$$R_{I'}(\mu) = \frac{N_{avg}(\mu)}{N_{\max}} \tag{8}$$

We also use $N_{avg}(\mu)$ to evaluate the degrees of separation for individual social networks in our experiments based on *the six degrees of separation*.

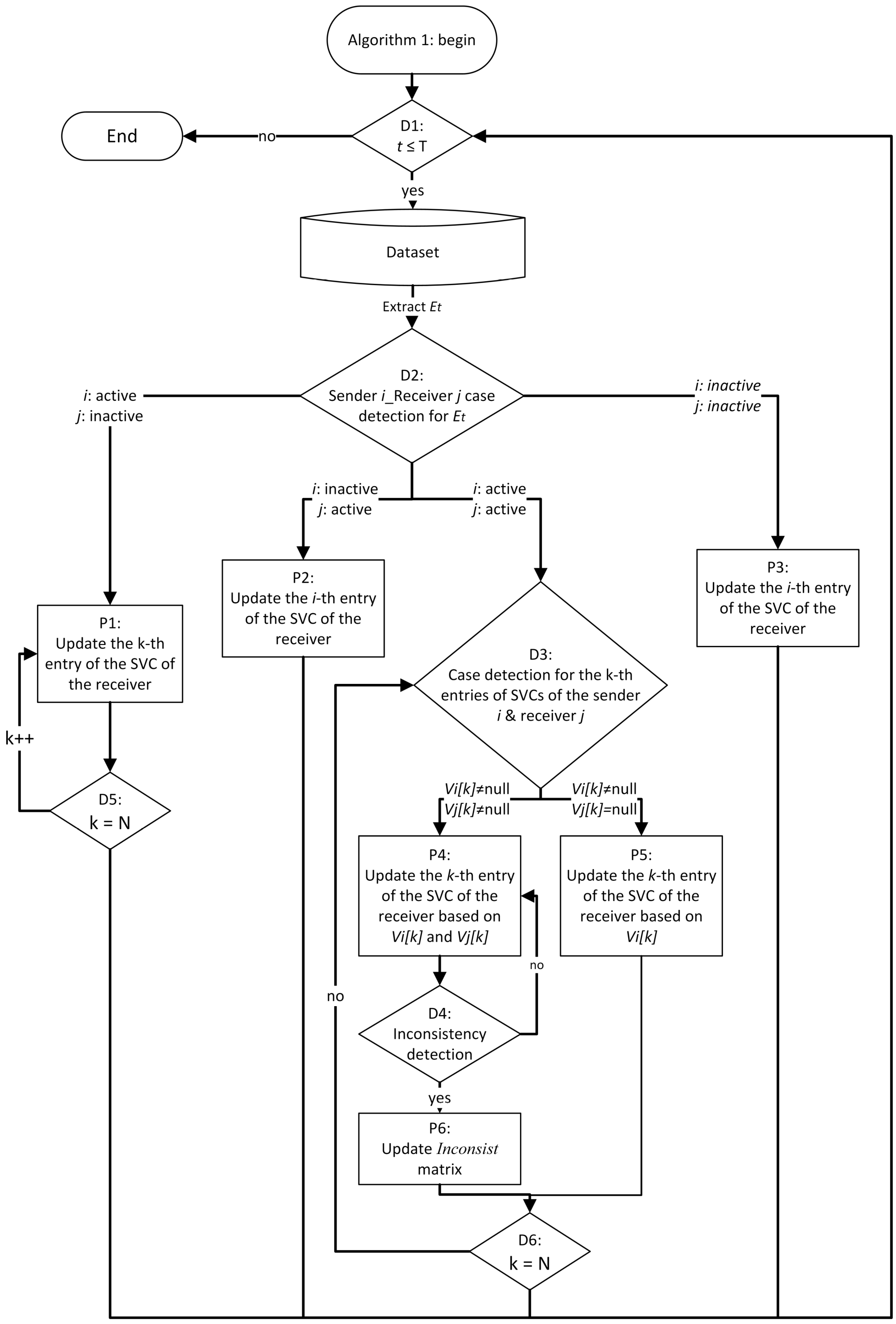
3.3 Algorithm 1 – adaptive approach

A sequence of E_t ($0 \le t \le T$) is treated as the input. In lines 1 to 4, the timestamp vector ($V_i.time$) and distance vector ($V_i.dist$) for each peer i associated with any other peer are initialised. When index i is '0', it means the first peer. When index i is $N - 1$, it indicates the Nth peer. Each peer maintains its own VC with the latest information with respect to all other peers. The value of $DIST_i[k]$ gives how many pieces of information are indirectly/directly delivered to peer i in the minimum number of hops of k, where $k \ge 1$ is the shortest distance between a pair of peers. $V_x[y].time$

represents the latest temporal view of peer x on peer y; $V_x[y].dist$ maintains the shortest distance from y to x. Each peer i needs to maintain a 2D matrix of $Inconsist_i$. Without it, the information loss rate for each μ would be incorrect in some cases. We will illustrate the reason later.

There are four different cases for updating V_j and $DIST_j$. Lines 7 to 22 deal with the first case when both sender's and receiver's VCs have been active. 'Active' implies that a peer has received at least one direct incoming message from any other peer. This case occurs most often, such as E_{t7} shown in Figure 3. P_1 becomes 'active' after time t_6 and P_3 is 'active' after time t_4. Lines 23 to 29 correspond to the second case, such as E_{t2}, where P_4 has not been active until t_5 but P_2 has become active since t_1. Lines 30 to 32 treat the third case like E_{t6} contrary to the second case. Lines 33 to 35 illustrate the fourth case, such as E_{t1}, where neither of P_1 and P_2 has been active at t_0.

Figure 7 Cognitive information processing


Algorithm 1: begin
End
no
D1: t ≤ T
yes
Dataset
Extract Et
D2: Sender i_Receiver j case detection for Et
i: active j: inactive
i: inactive j: inactive
i: inactive j: active
i: active j: active
P1: Update the k-th entry of the SVC of the receiver
k++
D5: k = N
P2: Update the i-th entry of the SVC of the receiver
P3: Update the i-th entry of the SVC of the receiver
D3: Case detection for the k-th entries of SVCs of the sender i & receiver j
Vi[k]≠null Vj[k]≠null
Vi[k]≠null Vj[k]=null
P4: Update the k-th entry of the SVC of the receiver based on Vi[k] and Vj[k]
P5: Update the k-th entry of the SVC of the receiver based on Vi[k]
no
D4: Inconsistency detection
yes
P6: Update Inconsist matrix
D6: k = N

Algorithm 1 VariableSocialVectorClocks

```
Input: $E_0, \ldots, E_T$
Output: $V_0, \ldots, V_{N-1}$, $DIST[]$ and $Inconsist[][]$
1  for $i \leftarrow 0$ to $N-1$ do
2      $V_i.time \leftarrow [\perp, \ldots, \perp]$; $V_i.dist \leftarrow [\perp, \ldots, \perp]$;
3      $DIST_i \leftarrow [0, \ldots, 0]$;
4      $Inconsist_i[][] \leftarrow [0, \ldots, 0]$;
5  while $t \le T$ do
6      $j \leftarrow E_t.receiver$; $i \leftarrow E_t.sender$;
7      if $V_i$ and $V_j$ have been active then
8          $V_j[j].time \leftarrow E_t.time$; $V_j[i].time \leftarrow E_t.time$;
9          $V_j[i].dist \leftarrow 1$;
10         for $k \leftarrow 0$ to $N-1$ but $k \neq j, i$ do
11             if $V_i[k] \neq \perp$ and $V_j[k] \neq \perp$ then
12                 if $V_i[k].dist < V_j[k].dist$ then
13                     $V_j[k].dist \leftarrow V_i[k].dist + 1$;
14                 if $V_i[k].time < V_j[k].time$ then
15                     $V_j[k].time \leftarrow V_i[k].time$;
16                     $DIST_j[V_j[k].dist]$++;
17                     if $V_i[k].dist > V_j[k].dist$ then
18                         $Inconsist[V_i[k].dist][V_j[k].dist]$++;
19             else if $V_i[k] \neq \perp$ and $V_j[k]$ is $\perp$ then
20                 $V_j[k].time \leftarrow V_i[k].time$;
21                 $V_j[k].dist \leftarrow V_i[k].dist + 1$;
22                 $DIST_j[V_j[k].dist]$++;
23     else if $V_i$ is active but $V_j$ has not been active then
24         $V_j[j].time \leftarrow E_t.time$; $V_j[i].time \leftarrow E_t.time$;
25         $V_j[i].dist \leftarrow 1$; $DIST_j[1]$ + +;
26         for $k \leftarrow 0$ to $N-1$ but $k \neq j, i$ do
27             $V_j[k].time \leftarrow V_i[k].time$;
28             $V_j[k].dist \leftarrow V_i[k].dist + 1$;
29             $DIST_j[V_j[k].dist]$++;
30     else if $V_j$ is active but $V_i$ has not been active then
31         $V_j[j].time \leftarrow E_t.time$; $V_j[i].time \leftarrow E_t.time$;
32         $V_j[i].dist \leftarrow 1$; $DIST_j[1]$ + +;
33     else if both $V_i$ and $V_j$ have not been active then
34         $V_j[j].time \leftarrow E_t.time$; $V_j[i].time \leftarrow E_t.time$;
35         $V_j[i].dist \leftarrow 1$; $DIST_j[1] \leftarrow 1$;
36     $t$++;
37 for $i \leftarrow 0$ to $N-1$ do
38     $DIST[] \leftarrow DIST[] + DIST_i[]$;
39     $Inconsist[][] \leftarrow Inconsist[][] + Inconsist_i[][]$;
```

The corresponding cognitive information processing of *VariableSocialVectorClocks* is shown in Figure 7. Whenever some timestamp t in D1 applied to line 5 in Algorithm 1 is valid, the event E_t would be extracted from the dataset. D2 determines which process would be used to update the VSVC of the receiver of E_t according to the situation of the sender and the receiver of it. P4 or P5 would deal with the first case as lines 7 to 22 do. D3 would first determine whether the k^{th} entries of the sender's and the receiver's VSVCs are undefined or not. Then, P4 or P5 would update this entry of the receiver's VSVC based on the result of D3. D4 refers to line 17. If an inconsistent case occurs, P6 needs to update *Inconsist* matrix. P1, which is equivalent to lines 23 to 29, would process the second case to update each entry of the receiver's VSVC based on the sender's VSVC. P2 and P3 respectively represent the processes to solve the third case and the fourth case defined before. When t is equal to T, Algorithm 1 will terminate.

Note that when a receiver j completes updating timestamps and friendship distances, $DIST_j[d]$ also needs to be updated. Whenever a more recent indirect/direct message is received and the shortest friendship distance from the sender peer to the receiver j is d, the value of $DIST_j$ in the index d should be added by one. Let us consider the example of E_{t2} as shown in Figure 5.

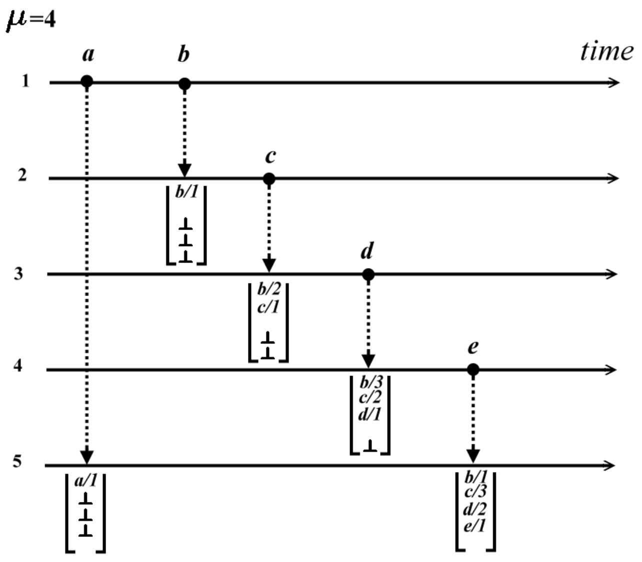
1 After receiving E_{t1}, $V_2 = [t1/1, t1/0, \perp, \perp]$.

2 Based on line 32, $DIST_2[1, 2, 3] = [1, 0, 0]$.

3 When receiving E_{t2}, $V_2 = [t1/1, t2/0, \perp, t2/1]$.

4 Again, based on line 32, $V_2[3].dist$ = '1'. $DIST_2[1]$++ will run one time, then $DIST_2[1, 2, 3] = [2, 0, 0]$.

Figure 8 An illustrative communication example with VSVCs by μ being 4



Lines 37 to 39 accumulate $DIST_i[]$ and $Inconsist_i[][]$ from each peer into a global $DIST[]$ and a global $Inconsist[][]$. After $t2$, $DIST[1, 2, 3] = [2, 0, 0]$. It indicates that the numbers of updating messages in friendship distance 0 and 1 are two and three, respectively. As mentioned before, Algorithm 1 can improve the computing performance against the conventional SVC updating method.

However, without using *Inconsist* matrix to record the number for each inconsistent case during updating, the real information loss rate cannot be correctly computed. Consider an illustrative communication example with five peers as follows.

‘ ’ indicates the latest receiving information for the receiver itself. Since these ‘ ’ outcomes are trivial for our study, we use the symbol of ‘ ’ to mark them. After timestamp *e*, *DIST*[1, 2, 3] = [6, 3, 2] in Figure 8. *DIST*[1] = 6 presents that the number of pieces of updating information with the receiving friendship distance being 1 is 6.

If the upper bound of μ is 2, the information loss rate computed in terms of the definition of equation (4)

$$R_L(2) = 1 - \frac{6+3}{6+3+2} = \frac{2}{11} \tag{9}$$

Again, using the same communication events as Figure 8 with a fixed upper bound of μ being 2, the information loss rate in Figure 9 is

$$R_L(2) = 1 - \frac{5+3}{11} = \frac{3}{11} \tag{10}$$

Obviously, the two outcomes are inconsistent. There are two major reasons leading to this inconsistency issue. First, the case in real situation with a fixed upper bound of μ will discard some indirect updating messages of corresponding friendship distances that are larger than μ. Second, the framework of variable SVCs allows to keep track of the latest timestamp and the shortest friendship distance from different senders. When a receiver *r* obtains an indirect piece of updating information from source *i* via a direct sender s, the receiver will compare the timestamp (S.T) and friendship distance (S.D) of this message with those (R.T and R.D) of its own existing message coming from the same source *i*. As shown in Figure 10, peer *r* receives an indirect updating message from peer *i* via peer *s*. By updating rule,

$$X = \max(S.T, R.T); \quad Y = \min(S.D+1, R.D) \tag{11}$$

max means that *X* will choose the latest from S.T and R.T and keep that one. In equation (9), there are nine cases (*S.T* >,=,< *R.T* and *S.D* + 1 >,=,< *R.D*). However, the inconsistency issue will happen in S.T < R.T and *S.D* > *R.D*. In other words, if the upper bound of μ is less than *S.D*, and not less than *R.D*, it may result in the inconsistency issue.

$$S.D > \mu \geq R.D; S.T < R.T \tag{12}$$

Therefore, it is necessary to use *Inconsist*[][] to record any inconsistent situation. The data structure of *Inconsist* looks like a lower triangular matrix as shown in Figure 7. In Algorithm 1, the real information loss rate in the upper bound of μ is defined as

$$R_L(\mu) = 1 - \frac{\sum_{k=1}^{\mu} DIST[k] - \sum_{k=\mu}^{N-1} Inconsist[k][\mu]}{N_M} \tag{13}$$

The above analysis will be observed in the next section.

Figure 9 The same communication example with μ being 2, as in Figure 4

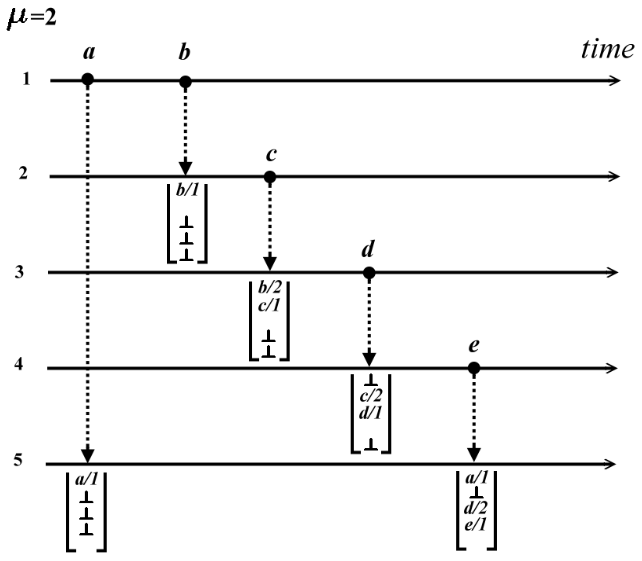
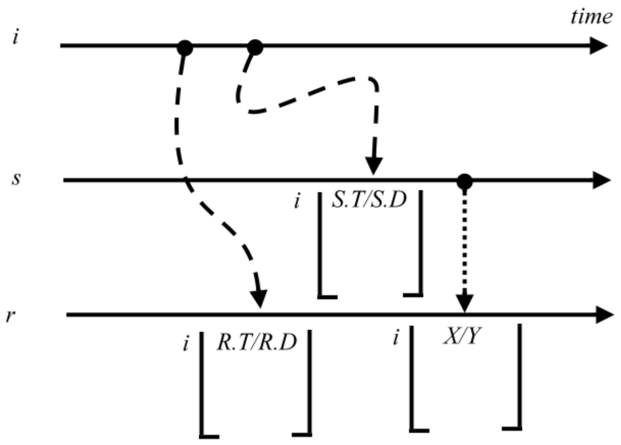


Figure 10 Illustrative example of inconsistent updating



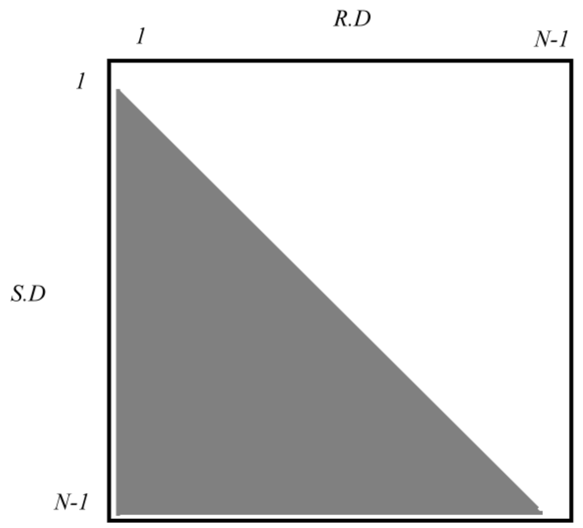
4 Results and evaluation

This section provides an overview of the system architecture in experimental dataset and framework configuration. The overview is followed by the analysis of the results and discussion.

4.1 Experimental design

The underlying infrastructure is constructed along with the entities outside the system that it interacts with. All the data we crawled was collected from Twitter. Twitter’s API provides straightforward interfaces to retrieve data for most Twitter functionality based on certain filters. We consider collecting communication data from Twitter into our dataset. Twitter 4J is an open-source library for Twitter API which is released under BSD license. It can easily integrate Java applications with the Twitter service. We extract tweets into our dataset using its REST functional implementation through the twitter4j package. To support our methodology, *VariableSocialVectorClocks* would be applied to several datasets to illustrate the influence of the friendship separation distance constraint on social communication.

Figure 11 The data structure of *Inconsist*



4.2 Datasets

In the Twitter data that we analyse here, we only consider the explicit user interactions where one sender refers to an indicated receiver target. After preprocessing, filtering them and removing self loop, two targeted forms of communication are preserved. The first one occurs in the form of retweets (where one user rebroadcasts another user tweets). The second one happens in the form of user mentions (where the @ symbol is used to explicitly refer to a specific user). If there exist more than one user in a tweet, it is transformed into multiple communication events, each of which corresponds to a pair of sender and receiver represented in the tweet.

- *London 2012 UK Olympics Data*: The Olympics dataset covers Twitter communication among a set of 492 UK Olympic athletes over the course of the four years until now, including about 940,000 tweets. It is based on a LIST of UK athletes organised by *The Telegraph* (twitter.com/#!/Telegraph2012/london2012). A Twitter 'LIST' is a curated group of Twitter users. Users can create their own lists or subscribe to lists created by others. Viewing a list timeline will show a stream of Tweets from only the users on that list. Also, we remove all tweets that are not the forms of user mentions or retweets between the core set of 492 users. Finally, some users who do not send and receive any messages are also eliminated from the core set. The number of core users is reduced to 459.
- *Twitter MLB data*: The MLB dataset includes Twitter communication among a list of 563 past and present Major League Baseball players in 2013 on Twitter, containing about 660,000 tweets. It is based on a list of Major League Baseball players organised by MLB (twitter.com/MLB/lists/players). We pre-process all data as in the previous setting. Statistically, the number of core users is reduced to 554.
- *Twitter SXSWi data*: This dataset comes from targeted communication among 480 speakers and attendees to stay in the know on the latest. These members are subscribed to a Twitter List of 'SXSWi' organised by Mashable (mashable.com). It is a leading source for news, information and resources for the connected generation. Mashable reports on the importance of digital innovation and how it empowers and inspires people around the world. Mashable's 34 million monthly unique visitors and 14 million social media followers have become one of the most engaged digital networks in the world. In total, after preprocessing, this dataset covers 458 users and 17,292 users' mentions and retweets among them.
- *Twitter UICnews data*: It is based on tweets gathered from followers subscribing to *UICnews* where it shows the latest from the University of Illinois at Chicago. Followers on Twitter are people who receive tweets issued by their following social target. There are 4,869 followers for *UIC's NEWS*. This dataset includes 3,765,054 tweets ranging from March 2009 to May 2014. However, after removing users who did not have any interactions with others in following *UICnews*, there are only 1,719 users and 26,809 users' mentions and retweets among them.
- *Twitter NY Times Journalist Data*: The *nyt-journalists* dataset is based on a public list of Twitter users, including reporters, editors, photographers and producers, in this case curated by *The New York Times*. After the process of purifying, there are 671 users and 97,457 users' mentions and retweets coming from around 1,142,000 tweets.

4.3 Experiment evaluation and results

For the UK dataset, the total number of pieces of information that can be captured without any friendship separation constraints is 905,702. The maximum friendship separation distance from a sender to a receiver is 18. In other words, when $\mu > 18$ the number of pieces of information indirectly updated is zero. Obviously, it means that no information travels along time-respecting paths bounded by the minimal number of hops more than 18. For the MLB dataset, the total number of messages delivered is 2,656,976. The maximum friendship separation distance across senders and receivers is 16. In other words, when $\mu > 16$ the number of pieces of information indirectly updated is zero. The SXSWi, UICnews, and NYtimes have about 205 k, 322 k, and 4353 k pieces of information, respectively. The maximum friendship separation distances from a sender to a receiver for the three datasets are 12, 15, and 14, respectively.

Figure 12 shows the relationship between the minimum number of hops (μ) and the Information loss rate (R_L). The detailed results with respect to R_L are shown in Table 1.

If μ reaches more than nine, the information loss rates will be less than 1%. When the upper bound of μ is limited to 'two', the memory space requirements can be reduced up to about 60%~80% except for the *NY times* social list. In many social networks, this case has been considered and shown to be significant to the impact of information brokerage activities (Burt, 2007). Based on the observations, it can been seen that the VSVC updates can effectively improve the memory space utilisation and can more closely approximate how far information should be tracked in real social networks. When μ is bounded to be 6~7, the values of R_L (space saving rate) for all the datasets are about 1%~15%. This observation shows that more than 90% of direct or indirect communication messages may be maintained. It implies that when μ is equal to about 6 or 7, the modification of SVCs almost acts as the conventional ones.

Table 1 Information loss rates $R_L(\mu)$

μ	*London*	*MLB*	*SXSWi*	*UICnews*	*NYtimes*
1	0.8248	0.9342	0.8305	0.8562	0.9037
2	0.6327	0.6927	0.6732	0.7739	0.4200
3	0.3571	0.2664	0.4331	0.6300	0.1207
4	0.1793	0.0946	0.2402	0.4479	0.0432
5	0.0947	0.0402	0.1253	0.2774	0.0183
6	0.0533	0.0191	0.0622	0.1546	0.0080
7	0.0302	0.0092	0.0282	0.0783	0.0033
8	0.0167	0.0043	0.0113	0.0359	0.0013
9	0.0088	0.0019	0.0041	0.0149	0.0005
10	0.0043	0.0008	0.0013	0.0055	0.0002

Note: μ is the minimum number of hops.

Figure 12 Information loss rate R_L

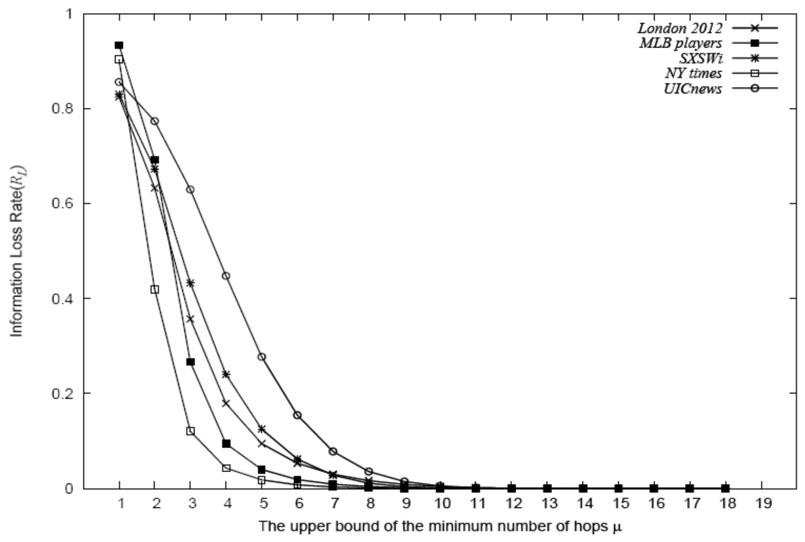


Figure 13 The maximum reachable incoming distribution rate $R_{\max}$

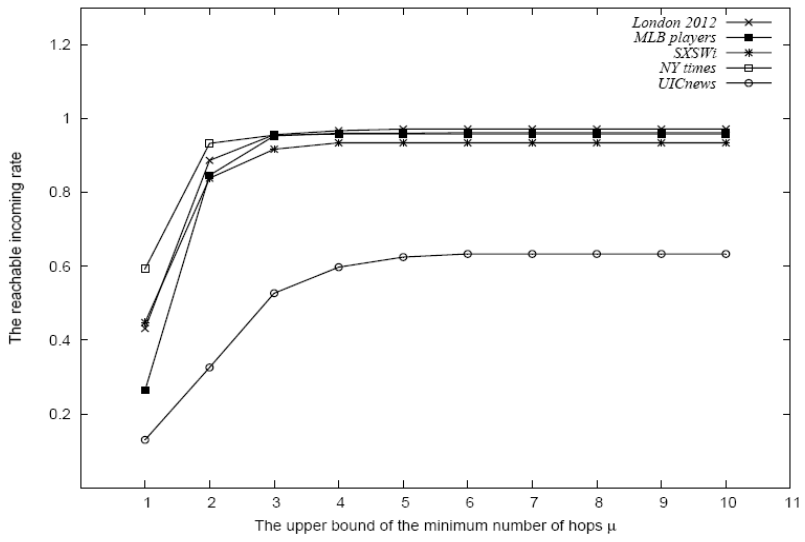
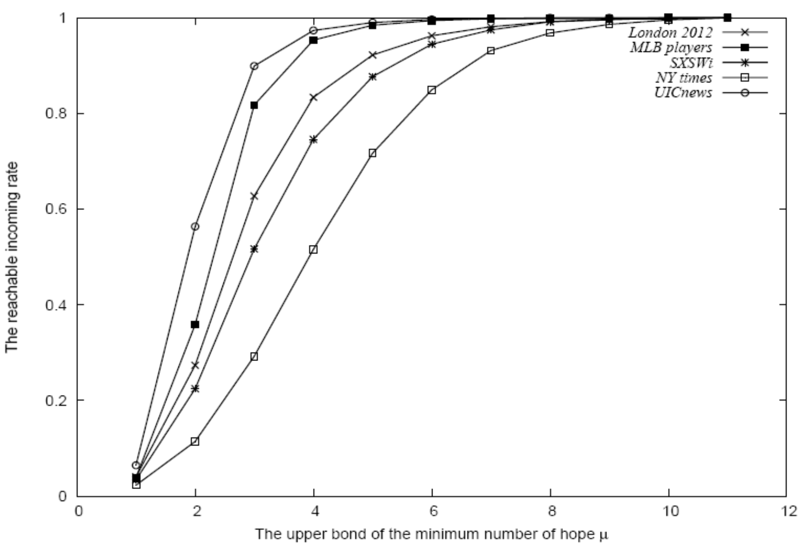


Table 2 The average reachable in-degree distribution number ($N_{avg}(\mu)$) and maximal rate ($R_{\max}(\mu)$)

μ	*London 2012*		*MLB players*		*SXSW i*		*UICnews*		*NY times*	
	$N_{avg}(\mu)$	$R_{max}(\mu)$	$N_{avg}(\mu)$	$R_{max}(\mu)$	$N_{avg}(\mu)$	$R_{max}(\mu)$	$N_{avg}(\mu)$	$R_{max}(\mu)$	$N_{avg}(\mu)$	$R_{max}(\mu)$
1	15	0.4314	20	0.2635	8	0.4476	5	0.1297	39	0.5931
2	102	0.8867	185	0.8466	53	0.8384	25	0.3258	341	0.9329
3	234	0.9564	422	0.9531	122	0.9170	64	0.5271	544	0.9553
4	311	0.9673	492	0.9603	176	0.9345	113	0.5974	589	0.9583
5	344	0.9717	508	0.9603	207	0.9345	157	0.6248	599	0.9583
6	359	0.9717	513	0.9621	223	0.9345	186	0.6335	603	0.9583
7	366	0.9717	515	0.9621	230	0.9345	204	0.6335	604	0.9583
8	370	0.9717	515	0.9621	234	0.9345	212	0.6335	605	0.9583
9	372	0.9717	515	0.9621	235	0.9345	216	0.6335	605	0.9583
10	372	0.9717	516	0.9621	236	0.9345	218	0.6335	605	0.9583

Figure 14 The normalised average reachable incoming distribution rate $R_{I'}$



For the analysis of reachable incoming topology distributions ($N_{avg}(\mu)$), as per the definition in Section 3, Figure 13 presents the maximum number of peers that communicate with each other under the different constraints of the shortest friendship separation (the limitation of the minimum number of hops) in the five social groups. In the case of Lee et al. (2013), their VC framework only included direct friendship (μ = 1) and friendship-of-friendship (μ = 2). Several social networks, such as Facebook, Twitter, and Google+, also focus on allowing direct friends and friends-of-friends to communication with each other. Based on the results shown in Table 2, when μ is bounded to be three, the maximum reachable incoming rates $R_{\max}$ can reach up to about 90%, other than for the group of *UICnews*. If the upper bound of μ is six, the reachable incoming topology can get to a steady state for the five social groups. Figure 13 shows that the distribution rate $R_{\max}$ of *UICnews* is obviously different from that of other groups. No matter what the upper bond of μ is, *London 2012* presents the highest reachable incoming rate but *UICnews* has the lowest one. Interestingly, members in *London 2012*, *MLB players*, *SXSWi*, and *NY times* are highly homogeneous in terms of their professions or certain interests. Their interactions in these four groups could be reasonably strong. However, followers in *UICnews* that just follow the News of UIC have much more diverse backgrounds. Other than *UICnews*, others have stronger communication interactions. Based on the observation in Figure 13, the social connection topology pattern in *UICnews* is weaker than other groups. Therefore, the reachable incoming distribution rate can help detect the strength of social interaction patterns. The higher the reachable incoming distribution rate, the stronger the social communication pattern will be. In order to observe the general reachable incoming topologies corresponding to different datasets, we normalise the data $N_p(\mu)$ by the maximum reachable incoming number in each social group. As shown in Figure 14, if μ is bounded to be two, R_{avg} in *London*, *MLB*, *SXSWi* is close to 20%~30%. The rates of R_{avg} in *UICnews* and *NY times* are about 80% and 40% respectively. The results are both consistent with the observations of the information loss rate in Figure 12.

According to the theory of six degrees of separation, any individual in the real world can be connected to any other individual on the planet through a chain of acquaintances with six or fewer intermediaries. As a result, when the upper bound of the shortest friendship distance is six, everyone can communicate with all the people in a social networking community. Intuitively, ‘six’ is applicable to evaluate our experimental results in the reachable in-degree distributions. Interestingly, as shown in Figure 14 when the upper bound of μ is 6, R_{avg} is close to or over 90% (if the upper bound of μ is 7, the R_{avg} for all the datasets is definitely more than 90%). It reasonably implies that a common peer *p* can almost communicate with most of the maximum number of peers from which peer *p* can receive indirect updated messages within 6 or 7 steps for the same social group.

5 Conclusions

Among several researches for social networking, the fine-grained temporal view has been shown to be very useful and applicable to measure the potential for information pathways and event-driven communication. With this view, a sequence of timestamps could provide social networks with a global view of the communication flows. As discussed earlier, however, poor scalability is a major drawback of conventional VCs, although it captures all exchanging messages and indirect communication. As a matter of fact, a tremendous amount of information communication does not seem to occur when two people talk to each other; the number of cognitive peers does not scale with the size of the whole social network. Based on this observation, the modification of SVCs (Kossinets et al., 2008) was proposed to reduce memory space requirements noticeably. An interesting question is raised in regard to what the reasonable value of μ is.

In this paper, we quantitatively analysed the influence of different upper bounds of the shortest friendship separation on the information loss rates and reachable incoming topology distributions using several Twitter social network groups. We also proposed an adaptive approach to systematically updating VSVCs. It improves the efficiency in exploring the above issues we concentrate on. This solution is universal. It can be easily applied to any social network group if and only if all event-driven communication with timestamps in the group could be preserved. The major reason is that our solution follows fine-grained temporal mechanism. Another advantage for our framework is that it can be operated in distributed systems.

The friendship policy in several social networks allows both direct friends and friends-of-friends to communicate with each other. Therefore, we first focus on the minimum number of hops (μ) ≤ 3 in our experiment evaluation. The results of R_L show that when μ is bounded to be ‘two’, in general, the memory space requirements can be effectively reduced up to 60%~80% except for the group having a behaviour of very strong connections, such as *NY times*. Taken together with this restriction, the VC updates can become more efficient for the memory space utilisation and more closely approximate how far information would be tracked along time-respecting paths. When the upper bounds of μ are ‘6’ and ‘7’, the corresponding information loss rates are lower than 15% and 10% for all groups, respectively. In other words, most of communication information would be retained. When the upper bound of μ is ‘10’, R_L will be obviously lower than 1%. It has hardly lost any communication information. When $\mu >$ ‘10’, the information loss rate and the normalised average reachable incoming distribution rates are both in the steady state.

On the other hand, when the upper bound of μ is ‘1’, the normalised average reachable incoming topology distribution rates (R_{avg}) are about 3%~6%. When μ is bounded to be 6~7, the rates of R_{avg} in all the social networking groups are statistically close to or more than 90%. It implies that when μ is 6 or 7, the number of peers from which a general one can exchange incoming information is almost saturated in a social group. It is consistent with the results of R_{avg}. Furthermore, when μ is 2 and 3, the Information loss rates (R_L) are also in line with the corresponding normalised average reachable incoming topology distribution rates (R_{avg}).

In our future work, we plan to integrate preferential connectivity into social communication aggregation analysis with VSVCs and develop a reasonable user interaction model. We want to look for an efficient approach to divide a social group into some subgroups, between which there is no communication.

References

Berger-Wolf, T.Y. and Saia, J. (2006) ‘A framework for analysis of dynamic social networks’, in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ‘06*, pp.523–528, ACM, New York, NY, USA.

Burt, R.S. (2007) ‘Second-hand brokerage: evidence on the importance of local structure for managers, bankers, and analysts’, *Academy of Management Journal*, Vol. 50, No. 1, pp.119–148.

Cha, M., Haddadi, H., Benevenuto, F. and Gummadi, K.P. (2010) ‘Measuring user influence in twitter: the million follower fallacy’, in *ICWSM ‘10: Proceedings of International AAAI Conference on Weblogs and Social Media.*

Dunbar, R.M. and Hill, R. (2005) ‘Social network size in humans’, *Human Nature*, Vol. 14, No. 1, pp.53–72.

Federico, P., Aigner, W., Miksch, S., Windhager, F. and Zenk, L. (2011) ‘A visual analytics approach to dynamic social networks’, *I-KNOW*, Vol. 47, pp.1–47:8.

Granovetter, M.S. (1973) ‘The strength of weak ties’, *Am. J. Sociol.*, Vol. 78, No. 6, pp.1360–1380.

Harrigan, M. (2010) ‘Using vector clocks to visualize communication flow’, in Memon, N. and Alhajj, R. (Eds.): *ASONAM*, pp.241–247, IEEE Computer Society.

Holme, P. and Saramäki, J. (2012) ‘Temporal networks’, *Physics Reports*, Vol. 519, No. 3, pp.97–125.

Hsu, T.Y., Kshemkalyani, A.D. and Shen, M. (2014) Modeling user interactions in social communication networks with variable social vector clocks', in *COLLABES '14: Proceedings of IEEE International Conference on Advanced Information Networking and Applications*.

Huberman, B.A. and Adamic, L.A. (2004) 'Information dynamics in the networked world', in Ben-Naim, E., Frauenfelder, H. and Toroczkai, Z. (Eds.): *Complex Networks*, Vol. 650, pp.371–398, Springer.

Kossinets, G., Kleinberg, J. and Watts, D. (2008) 'The structure of information pathways in a social communication network', in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08*, ACM, New York, NY, USA, pp.435–443.

Koylu, C., Guo, D. and Kasakoff, A. (2012) 'Mapping social relationships across space and time', in *AutoCarto International Symposium on Automated Cartography. Cartography and Geographic Information Society*.

Kshemkalyani, A. and Singhal, M. (2008) *Distributed Computing*, Cambridge University Press, New York, USA.

Kwak, H., Lee, C., Park, H. and Moon, S. (2010) 'What is twitter, a social network or a news media?', in *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, ACM, New York, NY, USA, pp.591–600.

Lee, C., Nick, B., Brandes, U. and Cunningham, P. (2013) 'Link prediction with social vector clocks', in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13*, ACM, April, pp.784–792.

Lewis, B.K. and Nicholes, C. (2012) 'Social media and strategic communication: a two-year study of attitudes and perceptions about social media among college students', *Public Relations Journal*, Vol. 6, No. 4, pp.1–20.

Mao, H., Shuai, X. and Kapadia, A. (2011) 'Loose tweets: an analysis of privacy leaks on twitter', in *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society, WPES '11*, ACM, New York, NY, USA, pp.1–12.

Mattern, F. (1988) 'Virtual time and global states of distributed systems', *Proceedings of the Parallel and Distributed Algorithms Conference*, pp.215–226.

Raynal, M. and Singhal, M. (1996) 'Logical time: capturing causality in distributed systems', *Computer*, Vol. 29, No. 2, pp.49–56.

Tang, J., Sun, J., Wang, C. and Yang, Z. (2009) 'Social influence analysis in large-scale networks', in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09*, ACM, New York, NY, USA, pp.807–816.

Wright, D.K. and Hinson, M.D. (2012) 'Examining how social and emerging media have been used in public relations between 2006 and 2012: a longitudinal analysis', *Public Relations Journal*, Vol. 6, No. 4, pp.1–40.