Efficient Discovery of Highly Interrelated Users in One-Way Communications

Jihwan SONG†(1, Student Member, Deokmin HAAM†, Yoon-Joon LEE†, and Myoung-Ho KIM†, Nonmembers

SUMMARY In this paper, we introduce a new sequential pattern, the Interactive User Sequence Pattern (IUSP). This pattern is useful for grouping highly interrelated users in one-way communications such as e-mail, SMS, etc., especially when the communications include many spam users. Also, we propose an efficient algorithm for discovering IUSPs from massive one-way communication logs containing only the following information: senders, receivers, and dates and times. Even though there is a difficulty in that our new sequential pattern violates the Apriori property, the proposed algorithm shows excellent processing performance and low storage cost in experiments on a real dataset.

Key words: data mining, sequential pattern, Apriori property violation

1. Introduction

Recently, various one-way communication services (e.g., e-mail, SMS) have been used in connection with crime. To prevent, investigate, detect, or prosecute serious crimes that use such services, law enforcement agencies (e.g., police, FBI) are legally permitted to access electronic records of the communications, often called logs. The logs usually include senders, receivers, dates and times, and locations; however, communication content is excluded owing to privacy or technical issues*.

Unfortunately, the process of manual log analysis is both time-consuming and labor-intensive because logs are usually massive and, moreover, include a lot of meaningless information generated by spam users.

To effectively find highly interrelated users from massive one-way communication logs, we introduce a new sequential pattern, the Interactive User Sequence Pattern (IUSP). First, an Interactive User Sequence (IUS) is a sequence of users that communicate interactively; for example, supposing a user A sent a message to a user B and then B back to A, a user sequence (AB) can be formed from the interactive communication between A and B. (The formal definitions with the time constraints and more details are given in Sect. 2.) The sequence (AB) is called an IUS. As shown Fig. 1 (a), if an IUS is derived from frequent interactive communications, it is called an IUSP. Here, frequent interactive communications mean that the frequency of communication is greater than or equal to a user-specified value. Generally, users included in an IUSP can be considered to be highly interrelated because frequent interactive communications between users indicates the close relationship of those users. In contrast, if a user unilaterally sends messages to others, as shown Fig. 1 (b), the recipients can hardly be said to be highly interrelated to the sender. Typical examples of this scenario are email or SMS spams; i.e., most users getting spam messages ignore them and might not reply to the spam senders.

However, most existing sequential pattern mining algorithms[1]–[5] may not be able efficiently to discover IUSPs from the massive logs because the downward-closure property of sequential patterns, a.k.a. the Apriori property**, does not hold for IUSPs. In general, the Apriori property violation causes performance degradation because the existing algorithms directly or indirectly use the property to increase their performance. Figure 1 (c) shows an example of the Apriori property violation for IUSPs. Depending on the counting scheme of applications, the IUS (ABC) has from at least one to at most four interactive communications; i.e., at most, it can be derived from the following four interactive communications: A ⇄ B ⇄ C, A ⇄ B > C, A ⇄ B < C, and A ⇄ B ≥ C. (Here, u1 ⇄ u2 means u1 sent a message to u2 at time t1 and then u2 back to u1 at time t2.) However, the IUS (BC), the sub-sequence of (ABC), has only one interactive communication B ⇄ C. Hence, the Apriori property does not always hold for IUSPs.

In this paper, we propose an efficient algorithm, the IUSPMiner, for discovering IUSPs from a massive one-way communication log. To improve performance, our algorithm reduces search space by pruning infrequent interactive communications of IUSs that do not violate the Apriori property in the middle of the process. To do this, IUSPMiner, first, finds some part that may cause a violation of the Apriori property violation for IUSPs.

Fig. 1 Examples of (a) an IUSP (AB) derived from frequent interactive communications, (b) non-interactive (unilateral) communication, and (c) the Apriori property violation for IUSPs.

Copyright © 2011 The Institute of Electronics, Information and Communication Engineers
ori property; then, the interactive communications related to the violation part are protected from being pruned. This is because IUSs derived from protected interactive communications may violate the Apriori property. Our performance study over a real dataset shows that IUSPMiner outperforms the Depth-First-Search (DFS)-based IUSPMiner, which is a variation of a DFS-based sequential pattern mining algorithm to discover IUSPs. Note that DFS-based sequential pattern mining algorithms such as PrefixSpan [4] show the best performance in many general cases.

2. Interactive User Sequence Patterns

This section formally defines an IUSP and its related terminology. First, a communication event, denoted by \((u_i, u_j, t_n)\), consists of two distinct users \(u_i\) and \(u_j\) and a date and time \(t_n\), which means \(u_i\) sent a message to \(u_j\) at \(t_n\). A communication log is a set of communication events and can be represented in the text-based (left) or graphic-based (right) form, as shown in Fig. 2 (a).

An Interactive Communication Pair (ICP) \(p\), denoted by \((u_i, u_j, t_m, t_n)\), means a pair of communication events \((u_i, u_j, t_m)\) and \((u_i, u_j, t_n)\) such that \(0 < t_n - t_m \leq W_{\text{max}}\). Here, \(u_i\) denoted by \(\text{SRC}(p)\), \(u_j\) by \(\text{DST}(p)\), \(t_m\) by \(\text{ST}(p)\), and \(t_n\) by \(\text{ET}(p)\) are called the source, destination, starting time, and ending time of \(p\), respectively. Also, \(t_n - t_m\) denoted by \(\text{RT}(p)\) and \([t_m, t_n]\) by \(\text{TI}(p)\) mean the response time and time interval of \(p\), respectively. Especially, \(W_{\text{max}}\) is a user-specified maximum response deadline; i.e., we assume that the two communication events \((A, B, t_1)\) and \((B, A, t_2)\) are totally unrelated to each other if \(t_2 - t_1 > W_{\text{max}}\). For \(W_{\text{max}} = 20\), Fig. 2 (b) shows all ICPs in the communication log of (a).

An Interactive Communication Sequence (ICS), denoted by \(\langle p_0, p_1, \ldots, p_{n-1} \rangle\), is a sequence of \(n\) ICPs \(p_0, p_1, \ldots, p_{n-1}\) such that (1) \(\text{DST}(p_i) = \text{SRC}(p_{i+1})\), (2) \(\text{SRC}(p_i) \not\in \bigcup_{j=1}^{i-1} \{\text{SRC}(p_j)\} \cup \{\text{DST}(p_{n-1})\}\), (3) \(\text{TI}(p_i)\) completely includes \(\text{TI}(p_{i+1})\), and (4) \(\text{RT}(p_i) \leq \text{UpperRespDeadLine}(p_{i+1}) + \alpha\), where \(\text{RT}(p_{n-1}) \leq \alpha\) and \(\text{UpperRespDeadLine}(p) = \alpha \times \lceil \frac{\text{RTC}(p)}{\alpha} \rceil\). Here, \(\alpha\) is a user-specified two-party response deadline; i.e., we assume \(p_i\) and \(p_{i+1}\) are related to each other if \(\text{RTC}(p_i) \leq \text{UpperRespDeadLine}(p_{i+1}) + \alpha\), where \(\text{RT}(p_{n-1}) \leq \alpha\). When \(\alpha = 5\), all ICSs from the ICPs of (b) are shown in Fig. 2 (c).

Lastly, an Interactive User Sequence (IUS), denoted by \(\langle u_0 u_1 \ldots, u_{n-1} \rangle\), is a sequence of \(n\) users \(u_0, u_1, \ldots, u_{n-1}\) such that at least one ICS \(\langle p_0, p_1, \ldots, p_{n-2} \rangle\) exists, where \(\text{SRC}(p_0) = u_0, \text{SRC}(p_1) = u_1, \ldots, \text{SRC}(p_{n-2}) = u_{n-2}\), and \(\text{DST}(p_{n-2}) = u_{n-1}\). Since an IUS can be derived from one or more ICSs, the frequency of an IUS is defined as the number of its ICSs; i.e., the frequency of an IUS \(u_i\) denoted by \(\text{Freq}(u_i)\), is \(k\) if \(u_i\) is derived from \(k\) ICSs. An Interactive User Sequence Pattern (IUSP) is an IUS \(u_i\) such that \(\text{Freq}(u_i) \geq \text{minsup}\). Here, \(\text{minsup}\) is a user-specified minimum support. Figure 2 (d) shows all IUSs in the ICSs of (c) with their frequencies. (The number after the “:” sign indicates the frequency of corresponding IUSs.) If \(\text{minsup} = 3\), IUSs \((C\ D), (B\ C\ D)\), and \((A\ B\ C\ D)\) become IUSPs, since only these IUSs have three or more ICSs.

3. IUSP Mining

We propose an algorithm, the IUSPMiner, to efficiently discover IUSPs from the massive communication log. IUSPMiner consists of three phases: (1) Overlap Detection, (2) ICSFrag Protection, and (3) IUSP Generation.

3.1 Overlap Detection

The first phase is to detect overlaps between ICPs, which are likely to cause violations of the Apriori property. First, the algorithm finds all the ICPs from the given communication log and then divides them into groups called ICPGroups according to their response times. Given \(W_{\text{max}}\) and \(\alpha\), ICPGroups, \(g_0, g_1, \ldots, g_{n-1}\), are created such that each has ICPs whose response times are between interval \((i\times\alpha, (i+1)\times\alpha]\), where \(n = \lceil \frac{W_{\text{max}}}{\alpha} \rceil\). Here, \(i\) is called a group number of \(g_i\), denoted by \(\text{GN}(p)\), where \(p \in g_i\). After the ICP discovery, the algorithm finds overlaps between ICPs. As shown in Fig. 3, two ICPs \(p\) and \(p’\) overlap one another if the following conditions hold: (1) \(\text{TI}(p)\) overlaps with \(\text{TI}(p’)\), (2) \(\text{SRC}(p) = \text{SRC}(p’)\) and \(\text{DST}(p) = \text{DST}(p’)\), (3) \(\text{GN}(p) - \text{GN}(p’) \leq 1\), and (4) \(\text{MAX}(\text{GN}(p), \text{GN}(p’)) > 0\). For each of the overlaps between ICPs \(p\) and \(p’\), the algorithm forwards a tuple \((o, M, d)\) called ICP Overlap Information (IOInfo) to the next phase, where \(o\) is an overlapped time interval (OTI) \([\text{Min}(\text{ST}(p), \text{ST}(p’)), \text{Max}(\text{ET}(p), \text{ET}(p’))]).
M is \( \text{MAX}(GN(p), GN(p')) \), and \( d = \text{DST}(p) \) or \( \text{DST}(p') \).

### 3.2 ICSFrag Protection

In the second phase, the algorithm protects some ICS fragments (ICSFrags) using the IOInfo forwarded from the first phase. Here, ICSFrags can be part of ICSs if several conditions hold, and, particularly, ICSs including protected ICSFrags are likely to cause Apriori property violation. Suppose that \( g_i \) is a non-empty ICPGroup, an ICSFrag \( f \) is an ordered list of ICPs, \( (p_0, p_1, \ldots, p_n) \), where \( p_j \in g_i \), if the following conditions hold: (1) \( n = 1 \) or (2) \( n > 1 \), \( \text{DST}(p_j) = \text{SRC}(p_j+1) \) and \( \text{TI}(p_j) \) completely includes \( \text{TI}(p_{j+1}) \), and \( \text{SRC}(p_j) \notin \bigcup_{k=j+1}^{n} [\text{SRC}(p_k) \cup [\text{DST}(p_{n-1})] \). Here, \( p_0 \) is denoted by \( \text{FirstICP}(f) \) and \( i \) is called a group number of \( G_i \), denoted by \( GN(f) \), where \( f \in G_i \). First, the algorithm finds all ICSFrags from each of the non-empty ICPGroups and then divides them into groups called ICSFragGroups according to their origins; i.e., ICSFragGroup \( G_i \) has only the ICSFrags discovered from ICPGroup \( g_i \). After the discovery of ICSFrags, as shown Fig. 4, for each tuple \( (o, M, d) \) of all IOInfo, the algorithm protects an ICSFrag \( f \) such that \( \text{TI}(\text{FirstICP}(f)) \) is completely included in \( o \), \( |GN(f) - M| \leq 1 \), and \( d = \text{SRC}(\text{FirstICP}(f)) \).

### 3.3 IUSP Generation

Lastly, the algorithm merges ICSs and ICSFrags to make new ICSs and derives IUSPs from the newly created ICSs, as shown in Fig. 5. In the figure, \( l \)-ICSs and \( l \)-ICSFs mean ICSs and ICSFrags with length \( l \); i.e., they all consist of \( l \) ICPs. First, the algorithm makes 1-ICSs from 1-ICSFrags in ICS-FragGroup \( G_0 \), since 1-ICSFrags themselves in \( G_0 \) are all 1-ICSs, that is, they satisfy all the conditions for ICSs. Then, the algorithm derives 1-IUSs from the 1-ICSs and picks out IUSPs. At this point, the algorithm prunes unnecessary ICSs that both are not deduced to IUSPs and do not include protected ICSFrags, which removal leads to performance improvement. To generate \( l \)-ICSs \( (l > 1) \), the algorithm merges \( j \)-ICSFrags in \( G_i \) with \( (l - j) \)-ICSs, where \( 0 < j < l \) and \( 0 < i < \text{MIN}(l, m) \). After generating \( l \)-ICSs, the algorithm derives IUSPs and prunes unnecessary ICSs. This process is repeated until no more ICSs are newly created. Figure 6 describes the pseudo code of Algorithm IUSP_Generation. Note that procedure DeriveAndPrune(temp, ResultSet, minsup) in Step 21 performs deriving of IUSs and pruning of unnecessary ICSs as follows: (1) it derives IUSs from \( \text{temp} \) that stores ICSs, (2) it adds IUSPs to \( \text{ResultSet} \), and (3) it prunes unnecessary ICSs.

### 4. Experiments

In the experiments, we compared IUSPMiner with a Depth-First-Search (DFS)-based IUSPMiner (DFS-IUSPMiner). Since no previous work for discovering IUSPs exists, we made the DFS-IUSPMiner find IUSPs by modifying an existing sequential pattern mining algorithm based on depth-first-search, generally following the pattern of the best-performing algorithm [4] in sequential pattern mining; it works like a DFS-based sequential pattern mining algorithm but does not prune any ICSs that are not deduced to IUSPs because of the Apriori property violation of IUSPs.

We created a real communication log generated from the Enron e-mail dataset [6], which is a rich source of information showcasing the internal working of an actual corporation over the period of 1998 to 2002. In the midst of Enron’s legal troubles in 2002, the Federal Energy Regulatory Commission in the United States made the dataset available to the public. The communication log contain at most 650,000 communication events as generated from the information of senders, receivers, and dates and times of the Enron e-mail dataset between 1 Oct. 2000 and 30 Sep. 2001. To compare the performance of the two algorithms, we measured their running times and peak memory usages. All experiments were repeated ten times, and the average was taken.

First, we present the results of scale-up experiments;
From the results so far, we can confirm that IUSPMiner needed less time and consumed less memory than DFS-IUSPMiner.

5. Conclusion

This paper introduces a new sequential pattern called an IUSP to group highly inter-related users in one-way communications. The distinct characteristic of the new pattern is to violate the Apriori property. This feature leads to lower performance for existing sequential pattern mining algorithms for discovering IUSPs, since unnecessary intermediate results cannot be pruned. Our proposed algorithm IUSPMiner overcomes this difficulty by protecting part of the Apriori property violation; i.e., the algorithm prunes unnecessary intermediate results that both are not deduced to frequent patterns and do not include protected parts. This new protecting and pruning strategy yields better performance and will help other algorithms mine some sequence patterns that disobey the Apriori property.

Acknowledgments

This work was supported (i) by the 2007 ETRI contract research, titled “Analyses of User Communication Patterns in Wireless Internet”, and (ii) by Basic Science Research Program through the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2009-0089128).

References