Mining Sequential Patterns Using the Integration of Fuzzy Logic and Graph Search Techniques

Pisit Phokharatkul
Department of Computer Engineering, Faculty of Engineering,
Mahidol University, Nakhon Pathom, 73170, Thailand
Email: egpph@mahidol.ac.th

Sukanya Yuenyong
Information Management Department, KASIKORNBANK PCL,
RatBurana, Bangkok 10140, Thailand
Email: sukanya.yu@kasikornbank.com, lukkaew@live.com

Abstract

Sequential pattern discovery is an important problem in data mining. In recent years, the researchers have been to find the new techniques to extract the sequential patterns from a large database. In this research, an effective way of the integrating fuzzy logic and graph search methods to create the fuzzy logic and graph search (FGS) algorithm for sequential pattern mining is proposed. The execution time of the two graph search techniques was compared. It was found that the depth-first search (DFS) takes less execution time than the breadth-first search (BFS). Also, the FGS algorithm takes less execution time than the GST algorithm when the k-sequence is greater than or equal to the 1-sequence (k≥2). The outcomes of the FGS algorithm are more valuable than the GST algorithm because the quantitative values of each transaction are considered. Finally, it was found that the FGS outcomes are substantially lower than the GST outcomes. Sometimes, the reduction is an advantage but it may not be so for all cases.

Key Words: data mining, sequential pattern, fuzzy logic, graph search.

1. Introduction

Nowadays, we use computers to collect data in various formats such as text file, database and XML formats. The advantages of data collection are more than search and review. The knowledge can be extracted from the existing data; call *data mining*. Data mining is the process of extracting interesting information or patterns from large information repositories. There are many types of data mining. Sequential pattern discovery is an important problem in data mining. In recent years there have been and continue to be many researchers trying to find new techniques to extract the sequential patterns from large database. They used difference algorithms such as: DSG algorithm [1], fuzzy algorithm [2], the algorithm mining path traversal pattern [3], and Generalized Sequential Pattern (GSP) [4].

This paper studies the integrating fuzzy logic and graph search algorithm for sequential pattern mining. There are two important things in the sequential pattern mining. First is the performance of execution time. Secondary is the result, how can extract the most useful sequential patterns. This problem can construe into many issues depending on the interest of an individual such as: case of inventory, mining sequential pattern use for predicting the consumer purchasing behavior. The outcome of sequential pattern mining can predict what the next product or group of products will be purchased when the product or group of products already purchased is known.

Although the algorithms always extract the sequential pattern, the user will always desire a better pattern. However, new algorithms must continue to solve the two important constraints of execution time and give a better result.

2. Data mining

Data mining [1] is the process of extracting interesting (non-trivial, implicit, previously unknown and potentially useful) information or pattern from large information repositories such as: relational database, data warehouse, XML repository, etc. Thus, data mining is known as one of the core processes of knowledge discovery in database. The processes of the knowledge discovery method consists the steps is shown in Figure 1.

First, the data source comes from different databases, which may have some inconsistence and duplications. The system cleans the data source by removing some noises or makes some compromises. And the integrated data sources can be stored in the data based system. Second, the related data from the integrated resources are selected and transform them into a format that is ready to be mined. Suppose that we want to find which items are often purchased together in a supermarket, while the database that records the purchase history may contains customer ID, items bought, transaction time, prices, number of each items and so on. But the specific task we only need items bought. After selection of relevant data by data mining techniques, the database will be much smaller. Consequently the whole process will be more efficient.

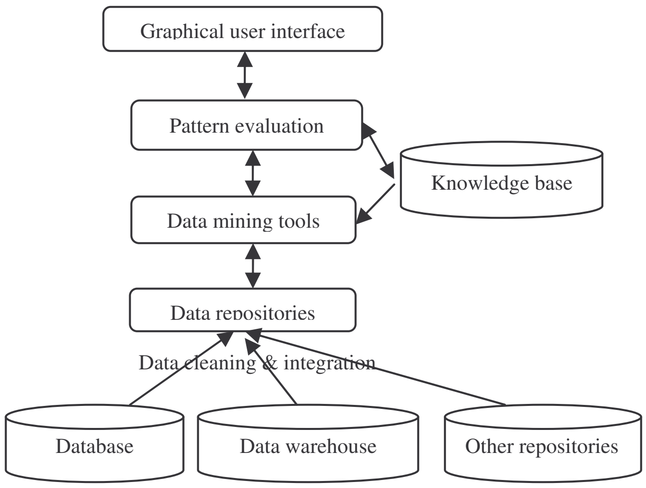

Figure 1. Knowledge discovery in database process.

Various data mining techniques are applied to the data source; different knowledge comes out as the mining result. That knowledge is evaluated by the certain rules, such as the domain knowledge or concepts. After the evaluation as shown in Figure 1, if the result does not satisfy the requirements or contradicts with the domain knowledge, some processes have repeated some processes until getting the right results. Depending on the evaluation result the user may to repeat the mining or modify his requirements. After we get the knowledge, the final step is to visualize data cubs or 3D graphics. This process tries to make the data mining results easier to be used and more understandable.

3. Sequential Pattern mining with the Integration of Fuzzy Logic and Graph Search Techniques

In this section, the researcher proposes a mining sequential pattern with the integration of fuzzy logic and graph search techniques. The method includes three points: (1) applies with sequential patterns (2) define the interval extended sequence using fuzzy sets, (3) searches sequential patterns using graph search techniques.

3.1 Sequential patterns

Sequential Pattern [1] is a sequence of item sets that frequently occurred in a specific order, all items in the same item sets are supposed to have the same transaction time value or within a time gap. Usually all the transactions of a customer are together viewed as a sequence, usually called customer-sequence, where each transaction is represented as an item sets in that sequence, all the transactions are list in a certain order with regard to the transaction-time.

3.2 Sequential Pattern Mining

Sequential Pattern Mining [1] is the process of extracting certain sequential patterns whose support exceeds a predefined minimal support threshold. Since the number of sequences can be very large, and users have different interests and requirements, to get the most interesting sequential patterns, usually a minimum support is pre-defined by users. By using the minimum support we can prune out those sequential patterns of no interest, consequently make the mining process more efficient. Obviously a higher support of sequential pattern is desired for more useful and interesting sequential patterns.

The miner begins to use the fuzzy membership functions for transform the quantitative values. Next to mine the sequential patterns by two graph search algorithm: depth-first search (DFS) and breadth-first search (BFS).

The methods of sequential patterns mining step by step as follow;

3.2.1 Prepare the sample data

The generation of the synthetic data is generated by Advanced Data Generator program into the transaction database. The notation C, I, and D represent the sum of customers, the sum of kind of items, and the sum of transactions, respectively. For example, C500.I5.D5000 represents the simulation environment with 500 customers who purchased items, 5 kinds of item, and 5000 transactions. We use the various experimental data to evaluate the output. The structure of the data table (1-sequences) as in Table 1.

Table 1. Example of the data 1-sequence

Purchased Items	Customer Id	Transaction Times	Quantities
C	1000	20/3/2550	10
A	1000	14/4/2550	9
E	1000	30/3/2550	7
….	…..	……	…..
C	1001	15/12/2550	10
A	1001	16/12/2550	9

3.2.2 Assume the fuzzy membership function

In this work, the quantitative purchased are divide into three fuzzy regions: Low, Middle and High. The example membership functions are show is shown in Figure 2.

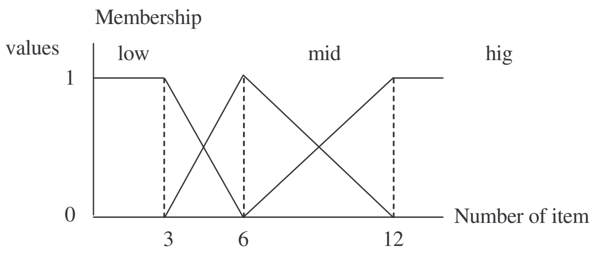

Figure 2. Membership functions

$$\left.\begin{aligned}\mu_{low(x)} &= \max(\min((6-x/6-3), 1), 0)\\ \mu_{mid(x)} &= \max(\min(x-3/6-3, 12-x/12-6), 0)\\ \mu_{hig(x)} &= \max(\min(x-6/12-6, 1), 0)\end{aligned}\right\} \quad (1)$$

The membership function associates each element u of U with a real number $\mu_{\tilde{A}}(u)$, in the interval [0, 1]. The main difference of fuzzy set theory from classical set theory is that different "degrees of membership" are allowed. In classical set theory, any element u of U either belongs to subset (membership value 1) or does not belong to the subset (membership value 0). The fuzzy membership functions define carefully. The number of memberships and how many items in each membership are depend on each issue.

3.2.3 Transform the quantitative values

The quantitative values of each transaction datum are transformed by the given membership functions mention above. The result of each membership function called *'degree of truth'* [2]. So the maximum value means the maximum truth.

For example: The quantity of item C is 5

$$\begin{aligned}\mu_{Low\,(5)} &= \max(\min((6-5/6-3), 1), 0)\\ &= \max(0.33, 0)\\ &= 0.33\end{aligned}$$

$$\begin{aligned}\mu_{Mid\,(5)} &= \max(\min(5-3/6-3, 12-5/12-6), 0)\\ &= \max(0.67, 0)\\ &= 0.67\end{aligned}$$

$$\begin{aligned}\mu_{Hig\,(5)} &= \max(\min(5-6/12-6, 1), 0)\\ &= \max(-0.17, 0))\\ &= 0\end{aligned}$$

$$\begin{aligned}\text{The maximum degree of truth} &= \max(\mu_{Low\,(x)}, \mu_{Mid\,(x)}, \mu_{Hig\,(x)})\\ &= \max(0.33, 0.67, 0)\\ &= 0.67\end{aligned}$$

Thus, the quantitative value of item C is transformed to 'Mid' because middle function gives the maximum degree of truth.

However, the support count value of each transaction in the table must be greater than or equal to the minimum support.

The support count value defines as the fraction of customer who supports this sequence.

$$Support(C) = \frac{\underline{\textit{Number of support customers}}}{\textit{Total number of customers}} \quad (2)$$

For example: There are 5 customers. If we define the minimum support at 20% that means the kind of item in each transaction must purchase by customers at least

$$20\% = \frac{\text{Number of support customers}}{5}$$

$$\begin{aligned}\text{Number of support customers} &= 20\% * 5\\ &= 0.2*5\\ &= 1 \text{ person}\end{aligned}$$

So, we prune away the kind of item that have a unique customer purchased its less than one person.

3.3 Create of the 2-sequences from the 1-sequence table

The structure of the data as follows: the 2-sequences are created from the 1-sequence table. The first item must be purchase before the other. However, the support count value of 2-sequences in the table must greater than or equal to the minimum support and time space between two sequences must less than the maximum interval value (time constraint). For example, we define the max-interval value is 30 days. That means we want to find the sequential patterns it's have a time space between each item in the patterns less than or equal to the maximum interval value (<= 30 days).

The time space of (C-hig) (B-low), (A-hig) (B-low) and (E-mid) (B-low) are more than max-interval that means they are uninteresting sequences.

Next, we prune away the 2-sequence that has support count value less than the minimum support. The data structure table as shows in Table 2-6.

Table 2. The quantitative values after transformation

Purchased Item	Customer Id	Transaction Time	Quantities
C-hig	1000	20/3/2550	10
A-hig	1000	14/4/2550	9
E-mid	1000	30/3/2550	7
……	……	…….	…..
C-hig	1001	15/12/2550	10

Table 3. The quantitative values of customer id '1000' after transformation

Purchased Item	Customer Id	Transaction Time	Quantities
C-hig	1000	20/3/2550	10
A-hig	1000	14/4/2550	9
E-mid	1000	30/3/2550	7
B-low	1000	24/5/2550	2

Table 4. The 2-sequence of customer id '1000'

Sequence	Customer Id	Start Time	End Time	Time Space
(C-hig)(A-hig)	1000	20/3/2550	14/4/2550	25
(C-hig)(E-mid)	1000	20/3/2550	30/3/2550	41
(C-hig)(B-low)	1000	20/3/2550	24/5/2550	35
(A-hig)(B-low)	1000	14/4/2550	24/5/2550	40
(E-mid)(A-hig)	1000	30/3/2550	14/4/2550	14
(E-mid)(B-low)	1000	30/3/2550	24/5/2550	24

Table 5. The 2-sequence data

Sequence	Customer Id	Start Time	End Time
(C-hig)(A-hig)	1000	20/3/2550	14/4/2550
(C-hig)(E-mid)	1000	20/3/2550	30/3/2550
(E-mid)(A-hig)	1000	30/3/2550	14/4/2550
(C-hig)(A-hig)	1001	15/12/2550	16/12/2550
(C-hig)(A-hig)	1003	31/10/2550	16/11/2550

Note: If the start time equal to the end time (purchased in the same day), we separate the item name with space such as (C-hig A-mid).

3.4 Design and construct the relational graph table.

The relational graph table is designed using the important properties. The structure of the data table as follows:

From_ vertex	To_ vertex	Edge_ type	Custo mer Id	Start Time	End Time

Note: the data type of 'Edge type' is a Boolean, set true if start time not equal to end time.

3.5 Search the sequential patterns from the relational graph.

The sequential patterns are searched from the relational graph table by two graph search algorithm. They are depth-first search (DFS) and breadth-first search (BFS) algorithm.

DFS follows the following rules:

Step 1: Select an unvisited node, visit it, and treat as the current node

Step 2: Find an unvisited neighbor of the current node by compare the start time of an unvisited neighbor with the end time of current node, visit it, and make it the new current node.

Step 3: If the current node has no unvisited neighbors, backtrack to its parent, and make that the new current node then repeat the above two steps until no more nodes can be visited.

Step 4: If there are still unvisited nodes, repeat from step 1.

BFS follows the following rules:

Step 1: Select an unvisited node s, visit it, have it be the root in a BFS tree being formed. Its level is called the current level.

Step 2: From each node x in the current level, in the order in which the level nodes were visited, visit all the unvisited neighbors of x. The newly visited nodes from this level form a new level that becomes the next current level.

Step 3: Repeat the previous step until no more nodes can be vsisted.

Step 4: If there are still unvisited nodes, repeat from Step 1.

Table 6. The relational graph table

From_ vertex	To_ vertex	Edge_ type	Custom-er Id	Start Time	End Time
C-hig	A-hig	T	1000	20/3/2550	14/4/2550
C-hig	E-mid	T	1000	20/3/2550	30/3/2550
E-mid	A-hig	T	1000	30/3/2550	14/4/2550
C-hig	A-hig	T	1001	15/12/2550	16/12/2550
C-hig	A-hig	T	1003	31/10/2550	16/11/2550

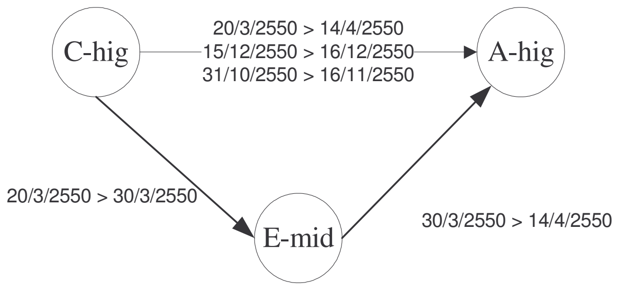

Figure 3. the relational graph.

Table 7. the sequential patterns table

Sequential Patterns	Customer Id
C-hig > A-hig	1000
C-hig > A-hig	1001
C-hig > A-hig	1003
C-hig > E-mid > A-hig	1000

3.6 Calculate the confidence values of sequential patterns

Each pattern has difference confidence values [5]. We calculate the confidence value from equation (3) as shows in Table 8.

$$S - conf(S) = \frac{Min_{1 \le m' \le m, 1 \le k' \le length(s_{m'})}\{\mathrm{sup}port(\{x_{m'k'} \subseteq s_{m'}\})\}}{Max_{1 \le m'' \le m, 1 \le k'' \le length(s_{m''})}\{\mathrm{sup}port(\{x_{m''k''} \subseteq s_{m''}\})\}} \quad (3)$$

The support of C-hig, A-hig and E-mid are 0.6, 0.6 and 0.2 in ordered. So the confidence value of C-hig > A-hig = (0.6/0/6) = 1 and the confidence value of C-hig > E-mid > A-hig = (0.2/0.6) = 0.33

Table 8. The sequential patterns with confidence values

Sequential Patterns	Confidence
C-hig > A-hig	1
C-hig > E-mid > A-hig	0.33

3.7 Evaluate the performance of algorithm.

We explore the execution time and the sequential patterns from the integration of fuzzy logic and graph search algorithm under different simulation environments and minimum support.

5. Experimental Results

5.1 Experiment 1:

We explore the execution time of graph search technique (GST) and the integration of fuzzy logic and graph search (FGS). In this experiment we use two graph search techniques that are depth-first search (DFS) and breadth-first search (BFS). We experiment under the different simulation environment and minimum supports, as shown in Fig. 4.1. Here we assume the fuzzy membership function are low <= 3, middle = 10 and high >= 18 and the max-interval = 30 days. As result, we found the DFS take the execution time less than BFS. Beside, the FGS algorithm takes the execution time less than the GST algorithm when k-sequence more than 1-sequence (k>=2). Although we change the value of total customers and the total kinds of item, the FGS algorithm always take the execution time less than the GST algorithm.

Finally, we also found the high support value use the few execution times because support value of k-sequence always less than the minimum support as shows in Figure 4-6.

5.2 Experiment 2:

We explore the outcomes of each algorithm are shown as table 4.1. Here the experiment label is C250.I10.D5000 and we assume the fuzzy membership function are low <= 3, middle = 10 and high >= 18, the max-interval = 30 days and the minimum support = 20%, the FGS algorithm and the GST algorithm take the execution time are 101.05 and 3,652.34 second in ordered and the outcomes of the FGS algorithm given more feature than the GST algorithm. It inform about the quantitative of each items in the sequential patterns. That gives more valuable outcomes in many issues such as; catalog design, store layout, cross marketing strategies, cross inventory strategies, effectiveness of promotional campaigns etc. Beside, the FGS algorithm can reduce the number of outcomes patterns from 7860 to 6 patterns as well as shows in Table 9-10.

5.3 Experiment 3:

We explore the execution time of the FGS algorithm under the different simulation environment and minimum supports, as shown in Fig. 4.2. Here the number of transactions is varied from 5000 to 20000 and minimum support from 20% to 100%. When the number of transaction increases, the execution time increases as well as shows in Figure 6.

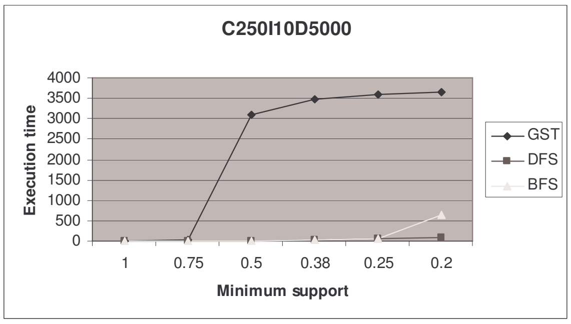

Figure 4. The environment C250.I10.D5000

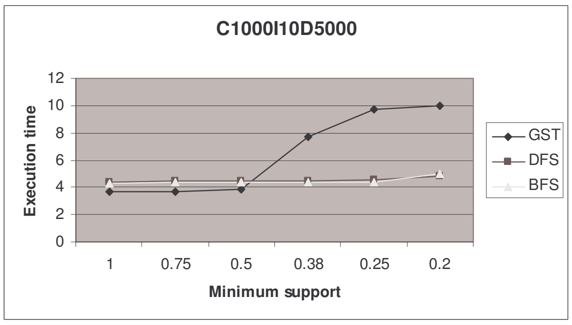

Figure 5. The environment C1000.I10.D5000

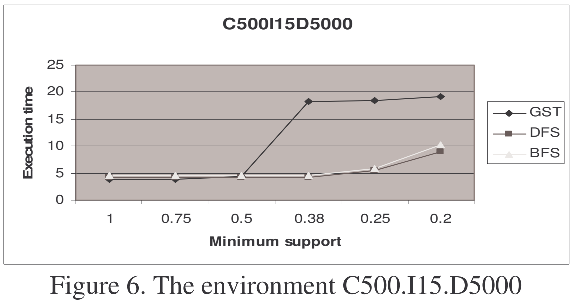

Figure 6. The environment C500.I15.D5000

Table 9. The outcomes of the FGS algorithm total 6 Patterns

Sequential Patterns	S-Confidence
B-hig -> G-hig	0.9794
I-hig -> A-hig	0.9695
F-hig -> A-hig	0.8781
B-hig -> A-hig	0.8719
F-hig -> A-hig -> A-hig	0.5737
B-hig -> G-hig -> G-hig	0.5697

Table 10. Example of the outcomes of the GST algorithm 22 of 7860 patterns

SequentialPatterns	S-Confidence
A -> A -> A -> A -> A -> A -> A -> A -> H -> G	0.8964
D -> A -> A -> B -> B -> B -> B -> B -> C -> A -> E	0.8924
D -> A -> A -> C -> C -> C -> C -> A -> A -> C -> I	0.8924
D -> A -> B -> B -> E -> G -> A -> B -> B -> J	0.8924
E -> A -> A -> C -> C -> D -> G -> I -> F -> H -> G	0.8725
E -> A -> B -> C -> C -> C -> A -> A -> E -> H	0.8725
B -> A -> A -> A -> C -> C -> E -> G -> F -> G	0.8645
B -> A -> A -> B -> B -> B -> B -> B -> C -> A -> E	0.8645
B -> A -> C -> D -> E -> C -> A -> B -> H -> I -> I	0.8645
B -> A -> D -> C -> A -> A -> A -> D -> D -> G -> H	0.8645
B -> A -> F -> D -> A -> A -> C -> C	0.8645
B -> B -> A -> A -> A -> A -> A -> A -> A -> A -> H -> G	0.8645
B -> B -> A -> A -> C -> C -> D -> G -> I -> F -> H -> G	0.8645
E -> B -> A -> A -> A -> A -> A -> A -> A -> A -> H -> G	0.8645
C -> I -> C -> E -> E -> C -> F -> D -> G -> I	0.8566
C -> I -> E -> D -> G -> A -> A -> J -> H -> H	0.8566
C -> H -> D -> A -> A -> A -> A -> A -> A -> A -> A -> H -> G	0.8406
C -> H -> D -> A -> A -> B -> B	0.8406
C -> H -> E -> G -> D -> D -> D	0.8406
F -> A -> A -> C -> C -> D -> G -> I -> F -> H -> G	0.8367
F -> A -> B -> D -> E -> C -> B -> E -> C	0.8367
F -> B -> A -> A -> C -> C -> D -> G -> I -> F -> H -> G	0.8367

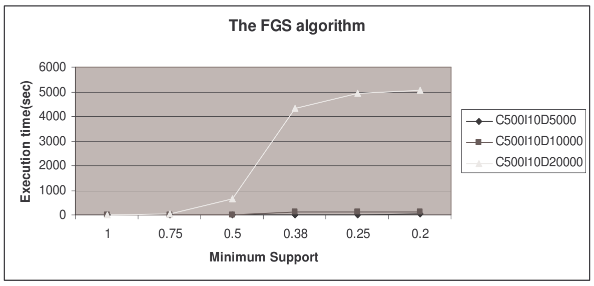

Figure 6. The environment C500.I20.D5000

6. Discussion and Conclusions

We propose the integration of fuzzy logic and graph search (FGS) algorithm to find sequential patterns from a transaction database. Through the experiments, we found the FGS algorithm is superior to the graph search technique (GST) algorithm. The advantage of the FGS algorithm over the GST algorithm as follows:

- The FGS algorithm takes less the execution time than the GST algorithm when k-sequence is more than 1-sequence (k>=2).
- The outcomes of FGS algorithm are more valuable than GST algorithm because we consider the quantitative value of item in each transaction.

The other advantage of the FGS algorithm follows:

- The FGS algorithm can generate k-sequence (k>=3) without following from (k-1)-sequences
- The outcomes of FGS algorithm have more valuable because we consider time space between each sequence.

The disadvantage of the FGS algorithm follows:

- The FGS algorithm takes more the execution time than the GST algorithm when 1-sequence.
- The FGS algorithm abundantly reduces the outcome patterns. The reduction may make us miss the useful or important patterns.

7. References

[1] Z. Qiankun, and S.B. Sourav, “Sequential Pattern Mining: A Survey,” Nanyang Technological Univeristy, Singapore, 2003.

[2] T.P Hong, K.Y. Lin, and S.L. Wang, “Mining fuzzy sequential patterns from quantities trasactions,” Soft Comput (2005).

[3] J. Yang, W. Wang, and P. S. YU, “Infominer: mining surprising periodic patterns,” In Proceeding of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, 2001, pp.395-400.

[4] H. Yu, and Y. Hayato, “Generalized Sequential Pattern Mining with Item Intervals,” Journal of Computers, VOL. 1, NO. 3, June 2006, pp. 51-60.

[5] Y. Unil, “Mining Sequential Support Affinity Patterns with Weight Constraints,” Electronics and Telecom-munications Research Institute, Telematics & USN Research Division, Telematics Service Convergence Research Team, Korea.