Multidimensional Service Weight Sequence Mining based on Cloud Service Utilization in Jyaguchi

Shree Krishna Shrestha, *Member, IAENG,* Yasuo Kudo, Bishnu Prasad Gautam, *Member, IAENG,* and Dipesh Shrestha, *Member, IAENG*

***Abstract*—Research in the area of sequential pattern mining has resulted in the proposal of numerous algorithms. However, there is a lack of research in the field of multidimensional sequential pattern mining which can be used in cloud services for service recommendation. This paper presents algorithms that process a sequential pattern of services in multidimensional space. Beyond mining single-dimensional sequences, we take into account multidimensional information associated with sequential data such as time series patterns, which can have a great impact for many potential service users. Thus, we propose a time based multidimensional sequential pattern mining algorithm. This algorithm constructs sequences by finding the service usage pattern and then characterizes each set of sequences using multidimensional properties based on user id, time series, and usage frequencies that define the multidimensional sequences. These sets of sequences are built around frequent sequential patterns that consider multiple dimensions. Thus, the whole process results in constructing a multidimensional array of sets and characterizing sequential patterns using multidimensional information. This algorithm has implemented in the Jyaguchi cloud system, in which the daily activities of the users and services are mined by considering the time factor in order to analyze the behavior of users.**

***Index Terms*—Service Mining, Multidimensional Pattern Mining, Jyaguchi Service Cloud, Relative and Absolute Service Weight**

I. Introduction

In the last couple of decades, we have witnessed an extensive growth of the information available on the Internet. One of the reasons for this growth is the addition of social network applications to the web. However, this growth of information has created a great deal of complexity for the user. In this context, information filtering functionalities, which can provide tools that support users information acquisition. The objectives of these tools are to rank information according to partly revealed preferences, which are encapsulated in previously undertaken surveys done by marking users' provided information and properties itself. Nonetheless, the presented rank might not satisfy users' expectation.

Generally, frequency of access of an item or duration of its use are considered when mining an item. However, we believe service mining in a cloud environment requires parameters of service usage for high precision in finding frequent patterns. In this research, we propose an algorithm, Time Weight Sequence Mining Algorithm (TWSMA), for mining multidimensional sequences considering service usage time as a service weight parameter.

II. Jyaguchi Cloud System and Its Overview

In order to perform our experiment, we utilize the Jyaguchi cloud [7],[12],[14] system. The term Jyaguchi was introduced by the author [7] and was derived from the Japanese language, in which the term means "an outlet portion of a tube or tap, which has opening and closing valves to regulate the rate of water flow." Accordingly, such a behavior of regulating resources is incorporated in the field of service usage, which was introduced in the Jyaguchi architecture. Jyaguchi proposed a hybrid architectural model because no single architectural model sufficiently provides a solution that is capable of regulating services on a pay per use basis, thereby providing features of SaaS. Furthermore, Jyaguchi is an architectural model for the development of distributed applications that can be extended to an architecture for cloud services and demonstrates how this style can be used to enhance the architectural design of a next-generation service cloud [1]. Fig. 1 below portrays the interaction between service provider and client.

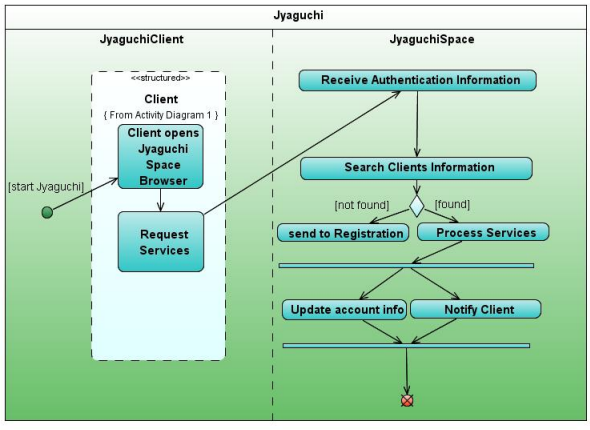

Fig. 1. Jyaguchi Architecture

S. K. Shrestha is studying on Master Course in Muroran Institute of Technology, Muroran, Hokkaido, Japan (e-mail: sthashree@gmail.com).

Y. Kudo is Assoc. Prof. in Muroran Institute of Technology, Muroran, Hokkaido, Japan (e-mail: kudo@csse.muroran-it.ac.jp).

B.P. Gautam is Asst. Prof. in Wakkanai Hokusei Gakuen University, wakkanai, Hokkaido, Japan. (e-mail: gautam@wakhok.ac.jp).

D. Shrestha was affiliated with Wakkanai Hokusei Gakuen University, Wakkanani, Hokkaido, Japan. He is now working as a System Engineer with DynaSystem Co., Ltd., 1-14, North 6, West 6, Kita-ku, Sapporo, Hokkaido, Japan (e-mail: d_shre@dynasystem.co.jp)

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

We chose to use Jyaguchi in our experiment because there is a rapid development of cloud computing technology as well as diverse methods of obtaining data. Consequently, the requirement for data mining in this infrastructural environment has dramatically increased. At present, however, there are few data analysis tools to process large-scale data that float in the cloud service environment. Also, data mining technology is gradually emerging in the backdrop of such circumstances. In fact, massive data mining over cloud services could be a very important guide to scientific research and business decision making. In order to propose a prototype data mining technique in cloud services utilizing sequential pattern mining, we have made use of the Jyaguchi architecture, in which the authors have substantial experiences.

III. Background and Related Works

The problem of sequential pattern mining was first introduced by Srikant and Agrawal in [1]. Since then, numerous studies have addressed the issue of efficient mining of sequential patterns [6], [9], [16], [18], [19], [20], [21], [22]. Gu et al. [15] were the first to come up with the notion of time interval in sequential pattern mining and proposing time-interval sequential pattern mining, which reveals the order of items as well as time intervals between successive items. Han et al.[17] have further enhanced this concept with partial periodic patterns in time series databases for mining multiple periods. Chang [11] proposed the framework of TiWS pattern mining, in which the weight of each sequence is first obtained from the time intervals of elements in sequence, and subsequently TiWS patterns are found using this weight. Yun et al. [3], Ahmed et al. [5], have used weighted frequent pattern mining techniques for efficient mining. Yun et al. [3] developed an algorithm with maximal weight and experimentally showed the technique to be superior to earlier methodology that did not consider weights. Multidimensional sequence mining was introduced by Pint et al. [2] who suggested three methods for mining: uniseq, seq-dim and dim-seq. Songram et al. [8] also discussed the mining of closed multidimensional sequences. None of them considered service usage time, which makes all these algorithms difficult to use for the purpose of service mining.

Similarly, the techniques used in pattern mining for items by Ahmed et al. [5] can be used for service mining; however, while mining services, an additional dimension of data should be considered, i.e., the time of service usage. This is the time-duration difference between two consecutive services, provided that the user does not sign out from the vendor's system. Service usage, generally, is directly proportional to the popularity of the service. In our proposed algorithm, we consider this factor of time series pattern usages for service mining of cloud services. Experiments similar to those in this paper are performed by Zhou et al. [23] and Yap et al. [13] in which a sequential pattern mining algorithm showed evidence of a sufficient solution for data mining but ignored the time series pattern of usages.

The following are reasons why this area is less studied:

-- Despite of maturity, services provided on the cloud platform are less comparable than the web based services.

-- The availability of user data is rare because vendors try to consolidate their user's transaction for their own purpose.

-- Service mining is an emerging concept and hence more research is yet to be done.

IV. Algorithm: Service Mining for Cloud Users

A. Problem Definition

Multidimensional sequential pattern mining is the process of mining sequential patterns along with one or more dimensions of information in which the order of dimension values is not important.

Let $I = \{i_1,i_2,i_3,\ldots,i_n\}$ be a set of services. A service-set X is a subset of services, i.e., $X \subseteq I$. A sequence S is denoted by $(s_1,s_2,s_3..s_l)$ where s_j is a service-set, i.e., $s_j \subseteq I$ for $1 \le j \le l$. s_j is also called an element of the sequence. A service usage time sequence *ST* is denoted by $(st_1,st_2,..st_k)$, where st_m is a set of pairs of service (s_j) and service usage time ($t_{s_j,o,p}$) in sequence o and position p, i.e., $st_m=(s_j, t_{s_j,o,p})$

Definition 1 (Unit Time): Unit time is a unit for measuring service usage *time* periods. In this algorithm, unit time is manually set.

Definition 2 (Service Usage Time): Service usage time ($t_{s_j,o,p}$) in a sequence $ST = (st_1, st_2, \ldots, st_k)$ where $st_m= (s_j, t_{s_j,o,p})$ is a usage time of service s_j in sequence o and position p.

Definition 3 (Absolute Service Weight): Absolute Service Weight (ASW_{u_i,s_j}) is a weight of service s_j for user *i*. Absolute Service Weight is the quotient of total service time usage of services s_j by user *i* to total service usage time of user *i* in the system.

$$ASW_{u_i,s_j} = \frac{\sum_{q=1,r=1}^{z,y} t_{s_j,q,r}}{\sum_{a=1,q=1,r=1}^{n,z,y} t_{s_a,q,r}}, \quad (1)$$

where n = total no. of service used by user i,
z = total no. of sequences
y = total no. of positions in sequence

Definition 4 (Service Used Count): Service Used Count ($SC_{s_j,o,p}$) of service s_j in sequence o and position p is the Service Usage Time ($t_{s_j,o,p}$) per unit time (u).

$$SC_{s_j,o,p} = \frac{t_{s_j,o,p}}{u} \quad (2)$$

Definition 5 (Relative Service Weight): Relative Service Weight ($RSW_{s_j,o,p}$) is defined as the weight of service in each position of each sequence. Relative Service Weight in each position of each sequence is calculated by multiplying Service Used Count ($SC_{s_j,o,p}$) by Absolute Service Weight (ASW_{u_i,s_j}).

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$$RSW_{s_j,o,p} = SC_{s_j,o,p} * ASW_{u_i,s_j} \quad (3)$$

Definition 6 (Sequence Database Weight): Sequence Database Weight (*SDW*) is the sum of all Relative Service Weights along the whole sequence database.

$$SDW = \sum_{j=1,q=1,r=1}^{n,z,y} RSW_{s_j,o,p} \quad (4)$$

Definition 7 (Minimum Weight): Minimum Weight (W_m) is the minimum weight that a service should have in order to be a frequent service and is the minimum support percentage of the sequence database weight.

$W_m = SDW *$ min_support (5)

Definition 8 (Mean Weight): Mean Weight (*MW*) of a subset or subsequence or sequence is the mean of the weights of each service in the sequence.

$$MW = \frac{\sum_{i=1}^{n} RSW_{s_j,o,p}}{n}, \quad (6)$$

where n = total no. of services in the sequence

A sequence database *S* is a set of tuples <*sid*,*sw*>, where *sid* is a sequence id and *sw* is a set of pairs of service id and relative service weight i.e., $sw = (s_j, RSW_{s_j,o,p})$. The relative service weight differs with time of service usage in that sequence and service user.

Further, the dimension of user_id with service weight sequence *SW* forms a multidimensional database with schema like (*SID*, U_i,*SW*), where *SID* is the unique sequence id, U_i is the User Id dimension and *SW* is in the domain of sequences. The multidimensional sequence will be of the form (u_i,*sw*), where $u_i \in (U_i \cup \{*\})$ for $(1 \le i \le m)$ and *sw* is a sequence. A multidimensional sequence $Q = (u_i, sw)$ is said to match tuple $t = (x_i, sw_t)$ in the multidimensional sequence database if and only if, for($1 \le i \le m$), either $u_i = x_i$ or $u_i = *$, and sw_t lies entirely within *sw*. The support of *Q* is the number of tuples in the database matching multidimensional sequence *Q* and is denoted by support (*Q*). The multidimensional sequence *Q* is called a multidimensional sequential pattern if and only if support (*Q*) > min_support, where min_support is the given minimum support threshold.

B. Algorithm description

A complete set of algorithms for making a multidimensional sequential input file and mining for frequent patterns is proposed here. The whole algorithm will be described in two parts: the first one covers creation of multidimensional service weight sequences and the second one covers mining multidimensional sequences. In a later part of this section, the algorithm will be explained with an example.

1) Create Multidimensional service-weight sequence

In this section of algorithm, whole log database is read and relative service weight is calculated for each service in each position to generate sequences of pairs of services and relative service weights are created from the log database.

Input: (1) Service Log database D
(2) Unit time
Output: (1) Multidimensional service-weight sequence
begin

$t_{s_j,o,p} = 0$, $ASW_{u_i,s_j} = 0$, $RSW_{s_j,o,p} = 0$, unit_time=*u*,
$SC_{s_j,o,p} = 0$, $SW =$ ‘’, $S =$ ‘’;
for *i*< size(*S*);
if in_array(session_id,exists_session_id) then
p=p+1
else
exists_session_id.add(session_id)
o=o+1;
end if
serviceUsedByUserPerService[u_i][s_j] += $t_{s_j,o,p}$
$SC_{s_j,o,p} = t_{s_j,o,p} / u$
serviceUsedByUser[u_i]+= $t_{s_j,o,p}$;
end for
foreach(serviceUsed = size(serviceUsedByUserPerService))
for (j<size(serviceUsed))
ASW_{u_i,s_j} = round(serviceUsedByUserPerService[u_i][s_j]/serviceUsedByUser[u_i],3);
end for
end for
foreach j=size(o)
foreach k=size(p)
$RSW_{s_j,o,p} = SC_{s_j,o,p} * ASW_{u_i,s_j}$
$SW=(s_j, RSW_{s_j,o,p})$
S=(session_id, *SW*)
end for
end for
end

2) Mining Multidimensional Sequence

Mining a multidimensional sequence [2] is done in three steps: 1. Mining Sequential Pattern 2. Forming Projected Database and 3. Mining MD-patterns. The whole algorithm is named TWSMA. We implemented service weight in the prefixSpan [4] algorithm such that service weight will be a factor for the service to become frequent. The mean weight of services in a subset is used to check if the subset is frequent. The following explains the details of the algorithm:

Input: (1) Multidimensional Sequence Database: M-SDB;
(2) Minimum support
Output: The complete set of multidimensional sequential patterns
Method:
A.Mining Sequential Pattern
1. First Scan
a: Calculate sequence database weight
b: Calculate min_weight
min_weight = min_support% of Sequence Database Weight(SDW)
c: Call PrefixSpan($\alpha, l, S \mid \alpha$)
Subroutine PrefixSpan($\alpha, l, S \mid \alpha$)
The parameters are 1) α is a sequential pattern; 2) l is the length of α and 3) $S \mid \alpha$ - is the α -projected database if $\alpha \neq <>$ otherwise, it is the sequence database S.

Method:

1. Scan $S|\alpha$ once, find each frequent item, b, such that

(a) b can be assembled to the last element of α to form a sequential pattern; or

(b) <b> can be appended to α to form a sequential pattern.

2. for each frequent item b, append it to α to form a sequential pattern α'

3. if(mean weight of $\alpha' \geq$ min_weight) output α'.

4. else break

5. end if

6. for each α, construct α'-projected database $S|\alpha'$, and

call PrefixSpan($\alpha', l+1, S|\alpha'$)

d: End if no frequent pattern is found or at end of database

B. Form MD- Projected Database

The dimension of the multidimensional sequence will make an MD-pattern. As we are considering only user_id as dimension it will make an MD-pattern of * or user_id. Then all the sequences in tuples containing MD-pattern P = (user_id) are collected which will form the multidimensional Projected Database or MD-Projected database for p, as denoted by $SDB|_{p.}$

C. Mine MD-patterns from MD-Projected Database

Then mining of the projected database is done using the aprioriall [10] algorithm with given minimum support which gives the user-based frequent MD-pattern.

The same algorithm will be explained with the help of an example in the following section. We took a service log database of 27 rows with 3 users and 4 services. The multidimensional table with sequence id, user id, and the sequence is created as shown in the table below. The sequence consists of pairs of service and service usage time. The services used in a single session are considered as a single sequence.

Table I is created from the log database whose entries consist of the user_id and a set of pairs of service and corresponding service usage time. From this table, we calculate the relative service weight of each service in each sequence and position, after which we create a multidimensional service weight sequential database.

We take the case of user 10 and service 2 for following calculations:

Time of usage of service 2 at position 1 and sequence 1 is $(t_{2,1,1})$ = 6 min.

Total time of usage of service i for user j, $ST_{i,j}$, is sum of times of use of service i by user j, then the total time of usage of service 2 for user 10 is

TABLE I
MULTI-DIMENSIONAL SEQUENCE WITH SERVICE USAGE TIME

seq. id	user_id	Sequence
1	10	(2,6),(123,16),(456,31),(2,33),(456,35)
2	10	(2,21),(2,20),(2,22),(1,22),(2,21)
3	16	(2,1),(123,9),(456,1),(123,1),(456,15)
4	15	(456,19),(456,24)(234,24),(456,43)
5	15	(234,20),(234,11),(234,30),(456,38)
6	16	(456,19),(123,39),(456,30),(234,30)

$ST_{2,10}$ = (6 + 33 + 21 + 20 + 22 + 21) min = 123 min.

Total service usage time for user j, T_j, is sum of all times of user j, then the total service usage time for user 10 is

T_{10} = (6 + 16 + 31 +… + 21) min = 227 min.

Weight of service 2 for user 10 is $(ASW_{2,10})$ = 123/227 = 0.542 by (1).

For unit time (u_t) 5 min, service usage count for service 2 at position 1 and sequence 1 is $(SC_{2,1,1})$ = 6/5 = 1.2 by (2).

Consequently, the weight of service 2 at sequence 1 and position l is $(RSW_{2,1,1})$ = 1.2 * 0.542 = 0.650 by (3).

Similarly, relative weight of service 2 at sequence 1 and position 4 is 3.577. From this result, we know that, for the same service and same user also the service usage time makes a difference in the weight of service.

After calculating the relative service weight of each service in each position following the above method, we can create the multidimensional service weight sequential database as shown in Table II. The total weight of sequence database is *SDW* = (0.650+0.224+1.804+...+1.242)=49.83 by (4).

For minimum support 5%, minimum weight, $W_{m,}$ is W_m = 49.83 * 5% = 2.4915 by (5).

For first scanning of sequence, if the following condition holds, we regard the service as a frequent service:

Total weight of service in whole database $\geq W_m$.

The total weight of service 2 in the whole database = 0.650 + 3.577 + 2.276 + 2.168 + 2.385 + 2.276 + 0.001 = 13.333.

Since the total weight of service 2 is greater than the minimum weight, service 2 is a frequent service.

This will make the 3 projected databases with service 2: (123, 456, 2, 456), (2, 2, 1, 2), and (123, 456, 123, 456). By scanning the <2>-projected database once, its locally frequent services are generated checking that the sum of weight of service is higher than min_weight, and all the length-2 sequential patterns prefixed with <2> will be found.

The repetition of this process for all frequent services will give the frequent pattern from the frequent service and user_id dimension.

TABLE II
MULTI-DIMENSIONAL SEQUENCE WITH RELATIVE SERVICE WEIGHT

seq. id	user_id	Sequence
1	10	(2,0.650),(123,0.224),(456,1.804),(2,3.577),(456 ,2.037)
2	10	(2,2.276),(2,2.168),(2,2.385),(1,0.427),(2,2.276)
3	16	(2,0.0014),(123,0.608),(456,0.089),(123,0.068),(456,1.344)
4	15	(456,2.253),(456,2.846)(234,1.954),(456,5.1)
5	15	(234,1.628),(234,0.895),(234,2.442),(456,4.507)
6	16	(456,1.702),(123,2.636),(456,2.688),(234,1.242)

V. Experiments, Results and Evaluation

A. Experiments and Results

To verify the efficiency of the new algorithm in mining services for cloud users, it is tested in the service cloud system, Jyaguchi. The real log sets of Jyaguchi Cloud users are used as required data for mining. The real logs of the Jyaguchi Cloud have a starting timestamp and an ending timestamp of service usage, which will give the service usage time for each service. The services used in a single session are considered as a single sequence and user id as dimension. All

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

experiments were performed on a 2.9 GHz Pentium machine with 4 GB of main memory, Windows operating system, and all programs were implemented in Java.

The experiment was done in the system on the set of 11 services and 10 users. The total number of rows in the database was varied to investigate processing time and memory usage. The value of minimum support was also varied to find the appropriate minimum support to get sufficient frequent patterns to verify the effectiveness of the proposed algorithm. Finally the proposed algorithm was compared with the Multidimensional sequence mining algorithm seq-dim [2] in order to verify the effectiveness of our algorithm.

Figs. 2–4 show the basic performance of the weight-based time sequence mining method on the data set of the Jyaguchi Cloud system. Fig. 2 shows the processing time per number of sequence with various values for minimum support. The graph shows that the process time increaseswith the number of sequences and decreases with the size of minimum support. Fig. 3 shows the memory used per number of sequence for varied minimum support. This figure shows memory usage consistent with number of sequence and decrease in minimum support.

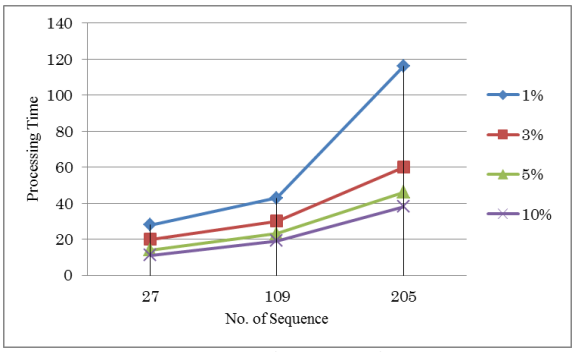

Fig. 2. Process time with no. of sequences for varied minimum support

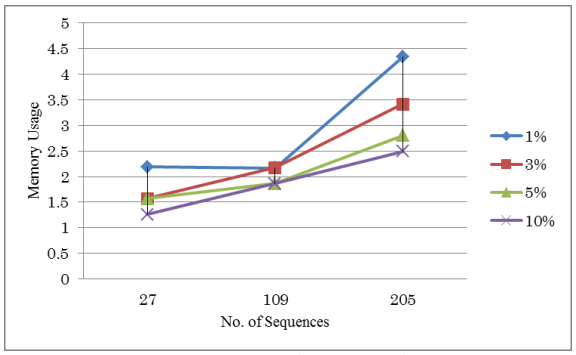

Fig. 3. Memory Usage with no. of sequences for varied minimum support

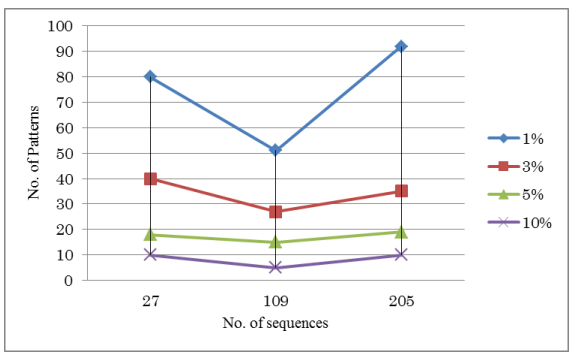

Fig. 4. No. of patterns with no. of sequences for varied minimum support

Fig. 4 shows the number of frequent patterns per number of sequences with varied minimum support. The result shows that the number of frequent patterns is not linear with the number of sequences but that it depends on the nature of sequences. However, for all sequences, the number of frequent patterns was inversely related to the minimum support. From the experiment, minimum support of 3% is found to be an appropriate value to get a sufficient number of frequent patterns.

Figs. 5–7 compares the number of frequent patterns, process time, and memory usage for the seq-dim algorithm and the TWSMA algorithm for data sequence 205. The figures reveal that the process time and memory usage for our algorithm is not much higher than that, for the original seq-dim algorithm. So, the conclusion can be drawn that for almost the same processing time and memory usage as the seq-dim algorithm, the proposed algorithm will mine the sequence with the service usage time.

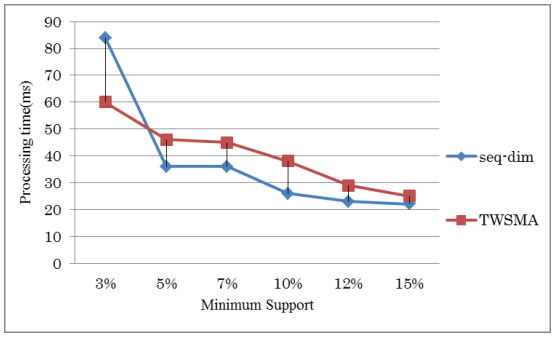

Fig. 5. Processing Time seq-dim vs TWSMA

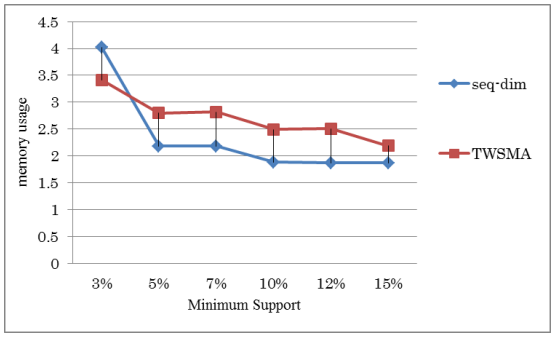

Fig. 6. Memory Usage seq-dim vs TWSMA

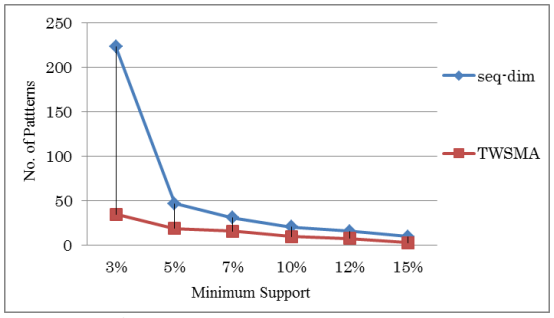

Fig. 7. No. of patterns seq-dim vs TWSMA

B. *Evaluation*

The dataset of Table I was used to evaluate the efficiency of our proposed algorithm. The dataset was used as input for the original seq-dim algorithm and the TWSMA algorithm. For minimum support of 20%, the total output number of frequent patterns from the seq-dim algorithm was 25 and from TWSMA algorithm was 15. The frequent patterns from

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

seq-dim include sequence (2,123), (2,123,456), (456,123,456). In table I, service 2 is used for only 6 min and 1 min before service 123. Although service 2 occurs twice, the service usage time is too low in the 2,123 sequence to be regarded as a frequent service. The case in the (2,123,456) and (456,123,456) sequences is similar. The proposed algorithm in the paper well excludes these sequences whose service usage time is low. This exclusion gives fewer but more efficient frequent patterns which are beneficial to use in recommending services.

VI. Future Work and Conclusion

A. Future Work

One major problem in this method comes from the fact that during the construction of a multidimensional sequence pattern, we need to formulate a tree structure in order to reduce searching and constructing time for the sets of sequences. However, we have just utilized a prefix span algorithm based approach during our search of the sequence. This approach is well suited while there are few dimensions; however, a hierarchical tree structure or graph algorithm need to be applied in order to formulate and effectively construct our multidimensional sequence pattern. This would improve performance during the search and construction of multidimensional sets, but it would be costly to set up. Nevertheless, it is recommended that this tradeoff needs to be investigated further to find the optimal type of searching algorithm. An additional performance gain could be achieved through utilizing parallelized processing in the database of multidimensional sequence sets. Furthermore, we have identified service category, user category as other dimensions to increase number of dimensions as future task. We will also be focused on calculating appropriate unit time and distributed behavior mining in our research.

B. Conclusion

In this paper, we have proposed an algorithm for mining cloud services through multidimensional sequence mining in the Jyaguchi Cloud Environment by utilizing multidimensional pattern mining with relative service weight as an additional parameter of the sequence. Subsequently, a process to get relative service weight through service usage time and frequency of service is also presented. The pair of service and related service weight is prepared for mining. This algorithm can be realized by modifying one of the multidimensional sequential pattern mining algorithms, seq-dim algorithm, in order to adjust the concept of service weight. Successful implementation of this algorithm is done to mine frequent services in the Jyaguchi Cloud Environment. The proposed algorithm will increase precision finding of frequent services by considering the usage time of services in a cloud environment.

Acknowledgment

This work was supported by KAKENHI (23700244).

References

[1] R. Agrawal and R. Srikant, “Mining Sequential Patterns,” Proc. of the Eleventh International Conference on Data Engineering, pp. 3-14, 1995.

[2] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen and U. Dayal, “Multidimensional Sequential Pattern Mining,” Proc. of the tenth international conference on Information and knowledge management (CIKM '01), pp. 81-88, 2001.

[3] U. Yun, H. Shin, K. H. Ryu and E. C. Yoon, “An efficient mining algorithm for maximal weighted frequent patterns in transactional databases,” Knowledge-Based Systems, Vol. 33, pp. 53-64, 2012.

[4] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal and M. C. Hsu, “Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach,” IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 10, pp.1424-1440, 2004.

[5] C. F. Ahmed, S. K. Tanbeer, B. S. Jeong, Y. K. Lee and H. J. Choi, “Single-pass incremental and interactive mining for weighted frequent patterns,” Expert Systems with Applications, Vol. 39, Issue. 9, pp. 7976-7994, 2012.

[6] K. Kaneiwaa and Y. Kudo, “A sequential pattern mining algorithm using rough set theory,” International Journal of Approximate Reasoning, Vol. 52, Issue 6, pp. 881-893, 2011.

[7] B. P. Gautam, “An Architectural Model for Time Based Resource Utilization and Optimized Resource Allocation in a Jini Based Service Cloud,” Master Thesis, Shinshu University, Nagano, Japan, 2009.

[8] P. Songram, V. Boonjing and S. Intakosum “Closed Multidimensional Sequential Pattern Mining,” Proc. of the Third International Conference on Information Technology: New Generations (ITNG '06), pp.512-517, 2006.

[9] Y. Hirate and H. Yamana, “Generalized Sequential Pattern Mining with Item Intervals,” Journal of Computers, Vol. 1, No. 3, pp. 51-60, 2006.

[10] T. Wang and P. He, “Web Log Mining by an Improved AprioriAll Algorithm,” World Academy of Science, Engineering and Technology, Vol. 4, pp. 591-594, 2007.

[11] J. H. Chang, “Mining weighted sequential patterns in a sequence database with a time-interval weight,” Knowledge-Based Systems, Vol. 24, Issue 1, pp.1-9, 2011.

[12] B. P. Gautam and D. Shrestha, “A Model for the Development of Universal Browser for Proper Utilization of Computer Resources Available in Service Cloud over Secured Environment,” Proc. of the International MultiConference of Engineers and Computer Scientists 2010 (IMECS2010), Vol I, 2010.

[13] G. Yap, A. Tan and H. Pang, “Dynamically-Optimized Context in Recommender Systems,” Proc. of the 6th International Conference on Mobile Data Management, pp. 265-272, 2005.

[14] B. P. Gautam, S. K. Shrestha and D. R. Paudel, “Utilization of Jyaguchi Architecture for development of Jini Based Service Cloud,” Wakkanai Hokusei Gakuen University Journal, No. 11, pp. 7-21, 2011.

[15] C. K. Gu and X. L. Dong, “Efficient mining of local frequent periodic patterns in time series database,” Proc. of 2009 International Conference on Machine Learning and Cybernetics, Vol. 1, pp.183- 186, 2009.

[16] Y. L. Chen, M. C. Chiang and M. T. Ko, “Discovering time-interval sequential patterns in sequence databases”, Expert Systems with Applications, Vol. 25, Issue 3, pp. 343-354, 2003.

[17] J. Han, H. Cheng, D. Xin and X. Yan, “Frequent pattern mining: current status and future directions,” Data Mining and Knowledge Discovery, Vol. 15, Issue 1, pp. 55-86, 2007.

[18] R. Agrawal and R. Srikant, “Fast Algorithms for Mining Association Rules,” Proc. of the 20th International Conference on Very Large Data Bases (VLDB ’94), pp. 487-499, 1994.

[19] H. Mannila, H Toivonen and A. Inkeri Verkamo, “Discovery of Frequent Episodes in Event Sequences,” Data Mining and Knowledge Discovery, Vol. 1, Issue 3, pp. 259-289, 1997.

[20] M. J. Zaki, “Efficient Enumeration of Frequent Sequences,” Proc. of the Seventh International Conference on Information and Knowledge Management (CIKM ’98), pp. 68-75, 1998.

[21] S. Ramaswamy, S. Mahajan and A. Silberschatz, “On the Discovery of Interesting Patterns in Association Rules,” Proc. of the 24th International Conference on Very Large Data Bases (VLDB ’98), pp. 368-379, 1998.

[22] Y. H. Hu, T. C. K. Huang, H. R. Yang and Y. L. Chen, “On mining multi-time-interval sequential patterns,” Data & Knowledge Engineering, Vol. 68, Issue 10, pp. 1112-1127, 2009.

[23] B. Zhou, S. Hui and A. Fong, “Efficient Sequential Access Pattern Mining for Web Recommendations,” International Journal of Knowledge Based and Intelligent Engineering Systems, Vol. 10, No.2, pp. 155-168, 2006.