Extraction Method of Frequent File Access Patterns based on Relative Position in Folder Tree

Toshiko Matsumoto and Takashi Onoyama

Abstract—**Toward efficient management of enterprise file server, listing potentially deletable files is expected to be effective. For this purpose, we propose a method of extracting frequent file access pattern to sort out files accesses by operating system or of file system, because such access patterns are supposed to be highly frequent. In our method, each access sequence of one user ID at short time intervals is processed in three steps: 1) path is encoded on the basis of relative position in folder tree structure, 2) accesses are clipped into independent sub-trees, and 3) statistical significance of number of observation is calculated according to occurrence probability. We apply our method to file server access logs of a company and demonstrate that it can extract characteristic access patterns although when multiple processes perform access in parallel.**

Index Terms—**access, file server, folder tree, frequent pattern**

I. Introduction

RECENTLY growing trends of unstructured files in enterprise organizations has received more and more attention, as digitalization of document increases volume of data stored in file servers[1]. Deleting unnecessary files is expected to be an effective way to reduce volume of file servers and to make operational management of file server more efficient. Although a file with old access time is generally considered as deletable [2], [3], value of last access time value is set not only by user-intended access but also by system-driven access: operating system performs crawling access against all objects (files and folders) in a subtree in order to execute a folder access, file system makes cache of a file and divides one access for a large file into multiple accesses, or an application program reads multiple setting files at a time in its start-up process [4]. Since such system-driven accesses are highly frequent, they may bury characteristics of user-intended accesses. Therefore, user-intended accesses are required to be distinguished from system-driven accesses and to be extracted for more convincing candidate list of deletable files. When access time is re-calculated from user-intended accesses, i.e. other than system-driven accesses, the access time accurately represents how recently the file is utilized. We propose the “Relative-Position based Frequent Access Mining (RPFAM) method”, which extracts frequent access patterns from access log of enterprise file server. RPFAM method is effective to distinguish system-driven accesses from user-intended access and can contribute to make list of potentially unnecessary files be more accurate.

Manuscript received June 30, 2014; revised July 23, 2014.

T. Matsumoto is with Hitachi Solutions, Ltd., 4-12-7, Higashishinagawa, Shinagawa-ku, Tokyo, 140-0002, Japan (phone: +81-3-5780-2111; fax: +81-3-5780-2049; e-mail: toshiko.matsumoto.jz@hitachi-solutions.com).

T. Onoyama is with Hitachi Solutions, Ltd., 4-12-7, Higashishinagawa, Shinagawa-ku, Tokyo, 140-0002, Japan (e-mail: takashi.onoyama.js@hitachi-solutions.com).

II. Problems in Extracting Frequent Access Pattern

For convincingly listing deletable files, frequent access patterns are required to be extracted as supposed system-driven accesses, and accesses time is required to be re-calculated from other accesses, i.e. supposed user-intended accesses (Fig. 1.) Several methods have been proposed for mining frequent pattern in sales records or in web access log [5]–[9]. However, there are three technical problems in applying these methods to extracting frequent pattern from access log of enterprise file servers.

First problem is about number of objects. Analysis of web access log deals with relatively small web sites, because it distinguishes each page as an independent object. As opposed to the case of web access log, analysis of file server access log is required to consider up to millions of objects or more. When each file or folder is analyzed as an independent object, number of observation of access could be too small to conclude meaningful results. Furthermore, common access characteristics could not be extracted from absolute file path when a folder subtree is assigned to a department and each user accesses to files in the subtree assigned to his/her department or business role. For example, suppose that a department provides several products, a subfolder is assigned for each one of products, and product folder contains marketing materials, i.e. catalogs, best practice sheet, comparison sheets and so on. When a sales person visits a customer, he/she access folders of products planed to be

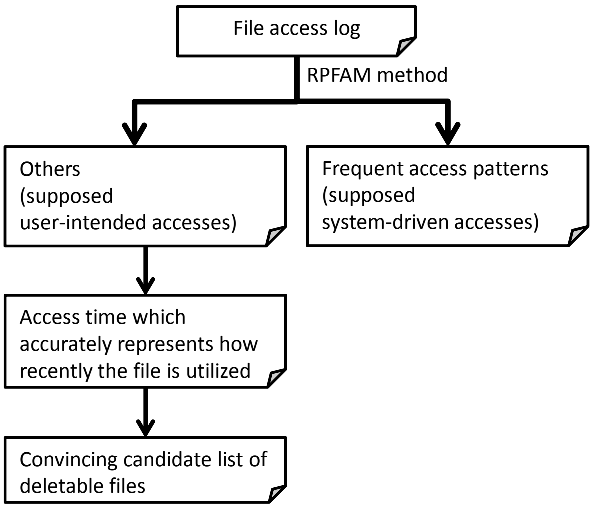
File access log
RPFAM method
Others (supposed user-intended accesses)
Frequent access patterns (supposed system-driven accesses)
Access time which accurately represents how recently the file is utilized
Convincing candidate list of deletable files

Fig. 1. Process overview of extraction of frequent access pattern toward candidate list of deletable files.

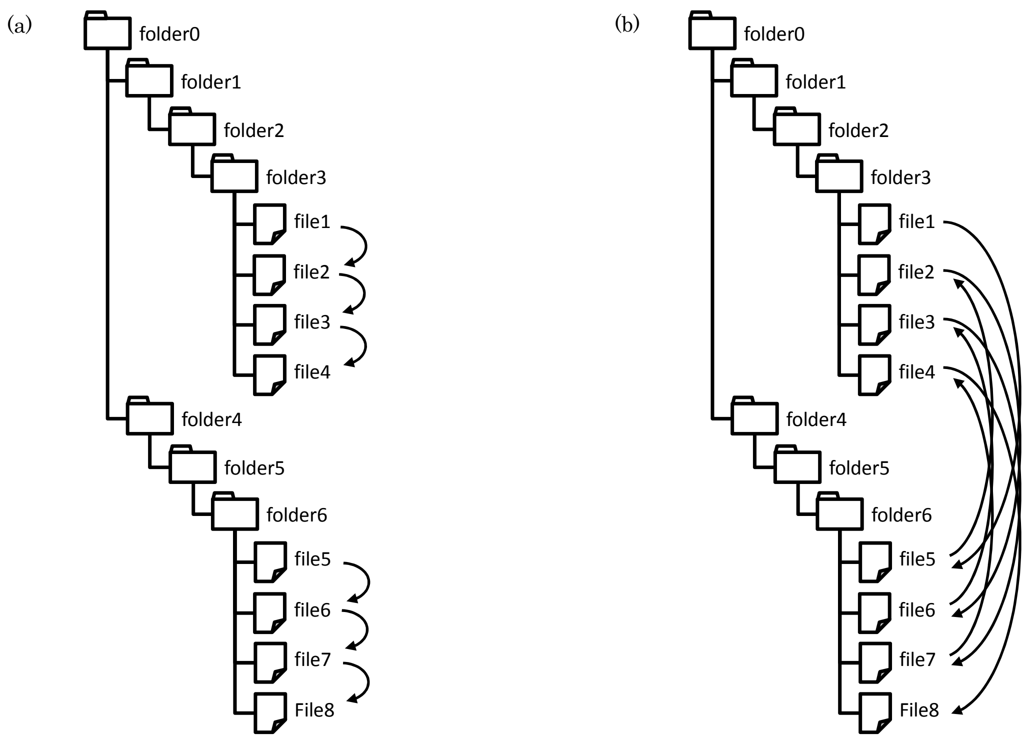

Fig. 2. Examples where two crawling-type accesses occur (a) separately and (b) simultaneously.

proposed. The accesses make operating system cache contents in the folders, and therefore sequentially read accesses would occur for files of the products. In this situation, access tendency "sequential access to multiple files in a folder" can not be extracted, because each product folder is treated as a distinctive object in absolute path.

Second problem is caused by parallel access. For analysis of web access log, we can utilize session ID to distinguish successive accesses. Since access log of file server only contains user ID (lacks session ID-like information), simultaneously multiple programs can mixed access sequences. Fig. 2 shows an example where two crawling access sequences are mixed. In case (b), the access log can not be considered as a crawling in a direct manner.

Third problem is about variation in length of access sequences. Pattern mining methods generally extracts patterns if their number of observation is larger than a threshold value "minimum support" [5]–[9]. Since number of observation decreases monotonously as length of pattern grows, threshold of number of observation suppress long pattern. Extraction method for long pattern is required, where file access characteristics is expected to be observed clearly.

III. The Proposed Extraction Method

In this section, we describe RPFAM method, which extracts frequent access pattern from access log of file server. We use the following notations.

$A = \{a_i\}_{i=1,\dots,N}$: access log composed of ordered list of N accesses

$a_i =< p_i, u_i, t_i >$: i th access composed of path, user, and timestamp

$p_i =< d_{i,0}, d_{i,1}, \dots, d_{i,l_i} >$: path of i th access a_i .

$d_{i,j}$: j th-depth object (file or folder) of path of i th access: $d_{i,0}$ is the root folder for all i

u_i : user of i th access

t_i : timestamp of i th access

$t_{threshold}$: threshold value of time interval to decide two access are performed sequentially or not

$\subseteq$: inclusion relation of two paths: path $p_i =< d_{i,0}, d_{i,1}, \dots, d_{i,l_i} >$ is included in path $p_j =< d_{j,0}, d_{j,1}, \dots, d_{j,l_j} >$, i.e. $p_i \subseteq p_j$, when $d_{i,l} = d_{j,l}$ for all l of $l_i \le l_j$ and $0 \le l \le l_i$

First, RPFAM method separates original sequence of accesses into sequence where user is identical and all time intervals are less than $t_{threshold}$. Next, it encodes path of each access in separated sequence based on relative position in folder tree structure, clips them into independent subtrees, and calculates occurrence probabilities of them. We denote sequence of encoded accesses as "access pattern." Finally RPFAM method outputs access patterns with number of observation larger than expected value based on occurrence probability. Three technical problems described in section 2 are resolved by encoding based on relative position in folder tree structure, clipping accesses into independent subtrees, and calculating statistical significance of number of observation based on calculating occurrence probability. Detailed description is given for each step of RPFAM method in the following subsections.

A. Encoding paths based on relative position in folder tree structure

Sequence of accesses can be analyzed not based on absolute path but based on relative position of folder tree structure. Path of first access is encoded as $< 0, 0, \dots, 0 >$ and succeeding access is encoded by as follows:

Suppose that there are sequence of access $a_1, a_2, ..., a_n$, and $e_{i,l}$, encoded result of $d_{i,l}$, is given as:

$e_{i,l} = 0$, when $< d_{i,0}, d_{i,1}, ..., d_{i,l-1} > \not\subset p_j$, $1 \le \forall j < i$

$e_{i,l} = e_{j,l}$, when $1 \le \exists j < i$ s.t. $< d_{i,0}, d_{i,1}, ..., d_{i,l} > \subseteq p_j$

$e_{i,l} = \max\limits_{1 \le j < i, < d_{i,0}, d_{i,1}, ..., d_{i,l-1} > \subset p_j} e_{j,l} + 1$, when $1 \le \exists j < i$ s.t. $< d_{i,0}, d_{i,1}, ..., d_{i,l-1} > \subset p_j$ and $< d_{i,0}, d_{i,1}, ..., d_{i,l} > \not\subseteq p_k$, $1 \le \forall k < i$

Table 1 shows example of encoded results. Files in folder "folder1/folder3" are encoded not in alphabetical order but in occurrence order. With this encoding, we can describe sequential access for files in a folder. Encoded result of third access represents that the access is for first file in different folder of first-depth, because first-depth folder object is encoded as "1" and the other folder file objects are encoded as "0."

B. *Clipping independent subtrees*

Sequence of accesses in table 1 contains accesses for two independent subtrees as shown in Fig. 3. Independence is implied in encoding results: "1" and "0" is observed in first-depth and only "0" is observed in second-depth folders. For $a_1, a_2, ..., a_n$, RPFAM method examines whether $\exists l_1$ and $\exists l_2$ s.t. $l_2 > l_1$, $e_{i,l_1} > 0$, and $e_{i,l_2} = 0$, $1 \le \forall i < n$. If there exists such l_1 and l_2 , RPFAM method treats subtrees of depth of l_1 or more as independent for possible minimum value of l_1 . This clipping step enables to treat each sequence of accesses are independent one when mixed sequence of accesses by multiple programs.

C. *Calculating statistical significance of number of observation for of access pattern based on occurrence probability*

We denote encoded result of i th access as $b_i = < q_i, u_i, t_i >$. RPFAM method calculates occurrence probability of access pattern $b_1, b_2, ..., b_n$ on the basis of probability of going up/down in folder tree structure and distribution of number of objects in a folder. Suppose that there is a sequence of accesses $b_1, b_2, ..., b_n$ and path of target object b_i and b_{i+1} are $q_i = < e_{i,0}, e_{i,1}, ..., e_{i,l_i} >$ and $q_{i+1} = < e_{i+1,0}, e_{i+1,1}, ..., e_{i+1,l_{i+1}} >$, respectively $(1 \le i < n)$. RPFAM method find l where

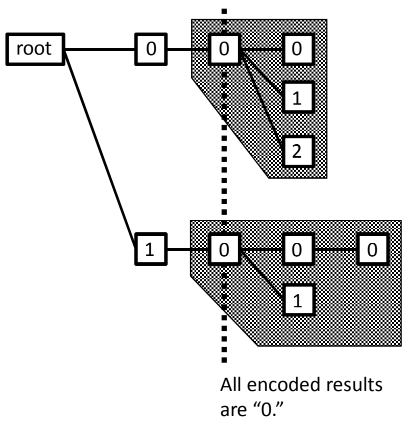

Fig. 3. Example of independent sub-trees in access sequence.

TABLE I
EXAMPLES OF RELATIVE POSITION-BASED ENCODING

No.	Path	Encoded Path
1	<folder1, folder3, file3>	<0, 0, 0>
2	<folder1, folder3, file2>	<0, 0, 1>
3	<folder2, folder4, folder0, file5>	<1, 0, 0, 0>
4	<folder1, folder3, file1>	<0, 0, 2>
5	<folder2, folder4, file6>	<1, 0, 1>

$e_{i,0} = e_{i+1,0}$, $e_{i,1} = e_{i+1,1}$, …, $e_{i,l} = e_{i+1,l}$, $e_{i,l+1} \ne e_{i+1,l+1}$, $1 \le l < l_i$, and $1 \le l < l_{i+1}$. Relative position of q_i to q_{i+1} can be described as "go up $(l_i - l + j)$ depth(s), go down $(l_{i+1} - l + j)$ depth(s), and select one object from objects in the subfolder for each step of going down" for $0 \le \forall j < l$. We denote $upward(l)$ and $downward\ (l)$ as probability of going up l depth(s) and as going down l depth(s) for two neighboring accesses ($upward(0) + upward(1) + ... = 1$ and $downward\ (0) + downward\ (1) + ... = 1$.) We also define $sib(n)$ as probability of having n child objects ($sib(1) + sib(2) + ... = 1$.) With this definitions, probability of selecting a child object that encoded as m can be expressed as following $child(m)$:

$child(m) = 1$, when no child object has been selected from the parent folder

$child(m) = \sum_{n=m}^{\infty} \frac{n - m + 1}{n} sib(n)$, when one or more child object have been selected, but none of them is encoded as m

$child(m) = \sum_{n=m}^{\infty} \frac{1}{n} sib(n)$, when one or more child objects are selected and encoded as m

More specifically, when no child object has been selected yet from the object before, encoded result is always "0." When no m encoded child object has been selected from the object, the probability of selecting an object which is encoded as m can be described as probability of "the parent object has more than m child objects and a new child object is selected." On the other hand, when one or more m encoded object have been selected, the probability of selecting an object which is encoded as m can be described as probability of "the parent object has more than m child object and an existing child object is selected again."

With these probabilities, conditional probability of observing access b_{i+1} given access pattern $b_1, b_2, ..., b_i$ can be described as following $prob(b_{i+1} | b_1, b_2, ..., b_i)$:

$$prob(b_{i+1} | b_1, b_2, ..., b_i) = \sum_{j=0}^{l} (upward(l_i - l + j) \cdot downward(l_{i+1} - l + j) \cdot \prod_{k=l-j+1}^{l_{i+1}} child(e_{i+1,k}))$$

Probability of observing access pattern $b_1, b_2, ..., b_n$ can be calculated by multiplying the conditional probabilities as follows:

$$prob(b_1, b_2, ..., b_n) = 1 \cdot \prod_{i=2}^{n} prob(b_i | b_1, ..., b_{i-1})$$

Number of observation of focused access pattern accords with Poisson distribution with the probability $prob(b_1, ..., b_n)$

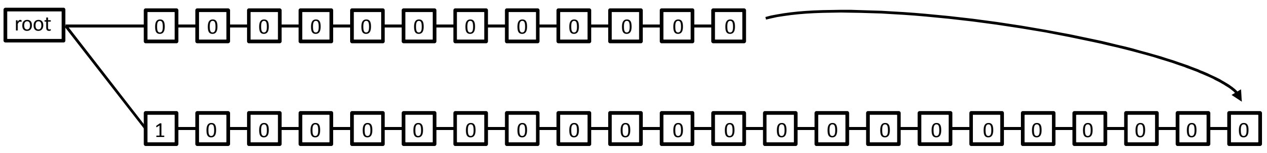

Fig. 4. An example of statistically significantly frequent access pattern (sequential accesses to two files in distant folders.)

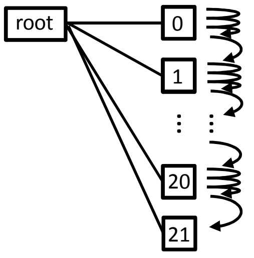

Fig. 5. Another example of statistically significantly frequent access pattern.

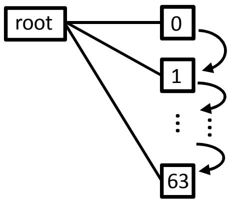

Fig. 6. Yet another example of statistically significantly frequent access pattern.

TABLE II
FREQUENCY OF ACCESS PATTERNS

	Without clipping independent sub-trees	With clipping independent sub-trees
Concentrated accesses	180,949 times (10%)	397,954 times (6%)
Crawling accesses	865,816 times (50%)	4,446,145 times (71%)
Non-typical accesses	696,280 times (40%)	1,405,512 times (22%)

TABLE III
TYPES OF ACCESS PATTERNS

No.	Name	Description
1	Concentrated accesses	More than four accesses for one object
2	Crawling accesses	One-by-one accesses for more than four object
3	Non-typical accesses	Others

and number of accesses in access log. Since statistical significance of number of observation of access pattern can be calculated by comparing number of observation and the distribution, we can confirm statistical significance of number of observation independently of length of access pattern.

IV. EXPERIMENT AND EVALUATION

In order to evaluate RPFAM method, we applied it to access log of enterprise file server. The access log contains 242,699 accesses of two hours. RPFAM method extracted 189,096 statistically significantly frequent access patterns at the significance level 0.05. 188,727 access patterns remain statistical significance after using Bonferroni adjustment[10]. Fig. 4, 5, and 6 shows examples of especially significant access patterns. Access pattern of Fig. 4, 5, and 6 has observed 2 times, 34 times and 6 times respectively.

Now we evaluate RPFAM method in regards to three problems described in section 2. First problem is about number of objects. For access sequences of length 1 (i.e. $a_1,...,a_n$ of $n=1$), number of unique path was 76,286 in the original access log. With the encoding step of RPFAM method, number of unique path was reduced to 1, because object in the first access is encoded as 0 by definition. For access sequences of length 2 (i.e. $a_1,...,a_n$ of $n=2$), number of unique pair of paths was 145,249 in the original access log. After the encoding step, the number of unique pairs was reduced to 245. These reductions demonstrate that number of unique access sequences can be dramatically suppressed by encoding path based on relative position of folder tree structure.

Second problem is caused by parallel accesses. Table 2 shows comparison result of number of observation of access patterns with or without clipping independent subtrees or the basis of types of access patterns described in table 3. Although crawling accesses are frequently observed when system-driven access is performed, they look like non-typical access when multiple system process simultaneously access file server. Table 2 demonstrates that clipping independent subtrees decrease percentage of non-typical accesses and increase percentage of crawling accesses. This tendency indicates that many of crawling accesses look like non-typical accesses because multiple process simultaneously access to file server, and that they can be correctly as crawling accesses by clipping independent subtrees.

Third problem is about variation in length of access sequences. We compare access patterns ordered by statistically significance of number of observation and ordered by absolute number of observation. In table 4, since value of correlation coefficient is 0.985, most of access patterns have similar order in both ways. However, 645 access patterns with top 10% number of observation has bottom 10% of statistical significance, their frequency is overestimated by simple absolute number of observation. We present two access patterns as examples of the gap between two ways. First example is an access pattern composed of single access. Although number of observation is 243,591 and it looks large enough, occurrence probability is 1 and the number of observation has no statistical significance. Second example is a crawling access in Fig. 6. Although it is

TABLE IV
COMPARISON BETWEEN ACCESS PATTERNS EVALUATED BY OCCURRENCE PROBABILITY AND EVALUATED BY FREQUENCY OCCURRENCE

		EVALUATED BY FREQUENCY OCCURRENCE									
		~10%	~20%	~30%	~40%	~50%	~60%	~70%	~80%	~90%	~100%
Evaluated by occurrence probability	~10%	38,633	44,954	146	0	0	0	0	0	0	0
	~20%	37,010	33,008	13,716	0	0	0	0	0	0	0
	~30%	7,446	5,590	69,693	1,005	0	0	0	0	0	0
	~40%	0	0	0	82,729	1,005	0	0	0	0	0
	~50%	0	0	0	0	82,729	1,005	0	0	0	0
	~60%	0	0	0	0	0	82,729	1,005	0	0	0
	~70%	0	0	0	0	0	0	82,729	1,005	0	0
	~80%	0	0	0	0	0	0	0	82,729	1,005	0
	~90%	0	0	0	0	0	0	0	0	82,729	1,005
	~100%	645	182	179	0	0	0	0	0	0	82,729

observed only 6 times, very small occurrence probability makes the number of observation be statistically significantly frequent. These examples indicate that calculating statistical significance based on occurrence probability is suitable to detect characteristics in system-driven accesses.

Above three evaluations demonstrates that RPFAM method can extract frequent accesses pattern from access log of file server and that RPFAM method is suitable for system-driven accesses. They are preferable features for distinguishing system-driven accesses from user-intended accesses.

V. CONCLUSION

In this paper, we have proposed RPFAM method as a extraction method of frequent access pattern. RPFAM method encodes paths of sequence of accesses based on relative position in folder tree structure, clips them into independent subtrees, and calculates statistical significance of their number of observation by occurrence probability calculated with probability of going up/down layer(s) in folder tree structure and with distribution of number of child objects. These steps can cope with large number of access target objects, multiple simultaneous accesses, and various lengths of access sequences. Our experimental result demonstrates RPFAM method can extract frequent pattern from access log of file server.

RPFAM method is expected to be effective for distinguish system-driven accesses from user-intended and for refining potentially deletable file list. When access time is re-calculated from user-intended accesses, we can convincingly construct candidate list of deletable files. Such a candidate list is expected to contribute for reduction of file server volume and to achieve efficient operational management of enterprise file servers. As a further improvement of RPFAM method, kind of file or connection origin of user can be utilized for more accurate extraction of system-driven accesses.

REFERENCES

[1] J. Gantz and D. Reinsel, "Extracting Value from Chaos," IDC IVIEW, 2011. Available: http://www.emc.com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf

[2] T. Tanaka, R. Ueda, T. Aizono, K. Ushijima, I. Naitoh, and N. Komoda, "Proposal and Evaluation of Policy Description for Information Lifecycle Management," Proceedings or the 2005 International Conference on Computational Intelligence for Modelling, Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, vol. 1, pp.261-267, 2005.

[3] C. Ludescher, T. Carroll, J. Murphy, and M. Zarnstoff, "File Storage Management for TFTR Physics Data," Proceedings of 14th IEEE/NPSS Symposium on Fusion Engineering, vol. 2, pp.856-859, 1991.

[4] S. N. T. Rao, E. V. Prased, and N. B. Venkateswarlu, "A Critical Performance Study of Memory Mapping on Multi-Core Processors: An Experiment with k-means Algorithm with Large Data Mining Data Sets," International Journal of Computers Applications, vol. 1, no.9, p.1-9, 2010.

[5] R. Agrawal and R. Srikant, "Fast Algorithm for Mining Association Rules in Large Databases," Proceedings of International Conference on Very Large Data Bases 1994, pp.487-499, 1994.

[6] R. Agrawal, H.Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo, "Fast Discovery of Association Rules," Advances in Knowledge Discovery and Data Mining, pp.307-328, 1996.

[7] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New Algorithms for Fast Discovery of Association Rules," Proceedings of Knowledge Discovery in Database 1997, pp.283-286, 1997.

[8] R. J. Bayardo Jr., "Efficiently Mining Long Patterns from Databases," Proceedings of ACM Special Interest on Managing Data, pp.85-93, 1998.

[9] J. Pei, J. Han, B. Pinto, Q. Chen, U. Dayal, and M. Hsu, "PrefixSpan: Mining Sequential Patterns by Prefix-Projected Growth," Proceedings of IEEE International Conference on Data Engineering 2001, pp. 215-224, 2001.

[10] G. R. Miller, "Simultaneous Statistical Inference 2nd Edition," Springer-Verlag, New York, 1981.