Hindawi Publishing Corporation
Applied Computational Intelligence and Soft Computing
Volume 2012, Article ID 786387, 12 pages
doi:10.1155/2012/786387

Research Article

Discovery of Characteristic Patterns from Transactions with Their Classes

Shigeaki Sakurai[1,2]

[1] *Business Intelligence Laboratory and Advanced IT Laboratory, Toshiba Solutions Corporation, Tokyo 183-8512, Japan*
[2] *Department of Computational Intelligence and Systems Science, Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Kanagawa 226-8502, Japan*

Correspondence should be addressed to Shigeaki Sakurai, sakurai@hrt.dis.titech.ac.jp

Received 21 October 2011; Revised 31 December 2011; Accepted 15 January 2012

Academic Editor: Tzung P. Hong

This paper deals with transactions with their classes. The classes represent the difference of conditions in the data collection. This paper redefines two kinds of supports: characteristic support and possible support. The former one is based on specific classes assigned to specific patterns. The latter one is based on the minimum class in the classes. This paper proposes a new method that efficiently discovers patterns whose characteristic supports are larger than or equal to the predefined minimum support by using their possible supports. Also, this paper verifies the effect of the method through numerical experiments based on the data registered in the UCI machine learning repository and the RFID (radio frequency identification) data collected from two apparel shops.

1. Introduction

Owing to the progress of computer environment and network environment, we can easily collect large amount of data and cheaply store it. We believe that the data includes useful knowledge which can help our decision making. Many researchers tackle on the discovery of the knowledge from the data since the mid-1990s. Various discovery tasks are studied in order to deal with various kinds of data.

The discovery task of frequent patterns composed of items from transactions is one of the tasks. Each transaction is composed of an item set. In the retail field, a receipt and a sales item correspond to a transaction and an item, respectively. In the initial researches, [1] proposes a method that efficiently generates candidate patterns and discovers frequent patterns by using the Apriori property. Here, the property shows that the frequencies of patterns monotonically decrease as items composing the patterns increase. Reference [2] proposes a bitmap index, [3] proposes a vertical ID list, and [4] proposes a frequent pattern tree (FP-tree) in order to speedily access the transactions and efficiently calculate the frequencies of the patterns. It is possible for these improvements to speedily discover the frequent patterns.

However, the discovered frequent patterns are not always the ones that are attractive for analysts. The discovery of patterns with different features is tried. For example, [5] tries to discover patterns whose orders based on the frequency are higher than the predefined order. Reference [6] does closed patterns representing many frequent patterns. Reference [7] does long patterns including many items. Reference [8] does patterns reflecting weights of items. Reference [9] does patterns reflecting hierarchical relationships among items. It is anticipated that more attractive patterns are discovered by these researches.

On the other hand, the discovery of patterns from different kinds of data is tried. For example, [10] tries to deal with tabular structured data. Here, each item is composed of an attribute and an attribute value. It proposes a method that efficiently discovers frequent patterns from the data by using relationships between attributes and attribute values. In the case of the data, missing values are often included. In order to deal with the values, [10–12] propose new evaluation criteria of the patterns.

The discovery task of patterns from transactions is expanding its research field more and more. This paper focuses on the difference of conditions in the data collection as one

of expanding fields. This is because we would like to discover patterns which tend to be overlooked due to the difference. For example, we try to discover patterns of sales items from a chain of retail stores. If we collect the data from many stores, we cannot anticipate that each store sells the same items. Limited stores can sell specific items such as new items and regional items. It is difficult to discover patterns related to the items sold in the limited stores. This is because their sales volume tends to be much smaller than the one of items sold in all stores. Also, we cannot anticipate that each store sells the same items every day. It can sell seasoned items. Patterns related to the seasoned items tend to be overlooked. In addition, we cannot anticipate that each store sells items based on the same sales method. Customers may be able to buy items through the trial of items in a store. Patterns related to the trial tend to be buried in patterns unrelated to the one. We can easily find difference of conditions in the data collection. It is anticipated that more attractive patterns are discovered, if we reflect the difference in the discovery of patterns.

Thus, this paper regards the difference of conditions in the data collection as classes. It proposes a method that efficiently discovers patterns reflecting the classes. Also, this paper applies the method to the data registered in the UCI machine learning repository [13] and the RFID (radio frequency identification) data collected from apparel shops [14]. It compares patterns discovered by the proposed method with patterns discovered by an existing discovery method, verifies the validity of the patterns discovered by the proposed method, and shows the effect of the proposed method.

2. Discovery of Characteristic Patterns

2.1. Transactions with Their Classes. This paper regards an item set with a class as a transaction. That is, the transaction t is defined by Formula (1). In this formula, I is a set of all items, C is a set of all classes, and n is the number of items included in the transaction. C_{it_i} is a subset of C and is composed of classes where the item it_i can be included. This definition shows that the transaction does not have multiple same items. The constraint is introduced by the original discovery task [1] which does not deal with classes. It notes whether an item is sold or not. Also, this definition shows that each item is included in the class cl

$$t = (\mathrm{cl} \mid \mathrm{it}_1, \mathrm{it}_2, \ldots, \mathrm{it}_n), \quad \mathrm{it}_i \in I,\ \mathrm{cl} \in \bigcap_{\forall i} C_{\mathrm{it}_i} \subseteq C, \quad \text{if } k \neq l,\ \text{then } \mathrm{it}_k \neq \mathrm{it}_l. \tag{1}$$

Here, we note that the examples are composed of four classes: "spring", "summer", "autumn", and "winter", and three items: "apple", "watermelon", and "banana". Table 1 shows classes in which each item is sold. The set of classes corresponding to an item is called an item class hereafter. Tables 2 and 3 show legal examples and illegal ones, respectively.

In Table 2, t_1 is a transaction with its class, because "winter" is included in the item classes of "apple" and "banana". Also, t_2 is a transaction with its class, because "summer" is included in the item classes of "watermelon" and "banana". In Table 3, t_3 is not a transaction with its class, because two "apple" are included in t_3. Also, t_4 is not a transaction with its class, because the item class of "apple" does not include "spring". Some readers may think that this restriction is too strict. However, it is not always strict. This is because we can add "spring" to the item class of "apple" if we need to regard t_4 as a transaction with its class. On the other hand, if a valid item class is assigned to each item, we can think of two possibilities. One is that the class in the transaction is wrong, and the other is that wrong items are included in the item set. We can think that some errors occur in the data collection.

Table 1: Item class.

Item	Class
apple	{autumn, winter}
watermelon	{spring, summer}
banana	{spring, summer, autumn, winter}

Table 2: Legal examples.

t_1 = (winter | apple, banana)
t_2 = (summer | watermelon, banana)

Table 3: Illegal examples.

t_3 = (autumn | apple, apple, banana)
t_4 = (spring | apple, watermelon)

2.2. Criteria. This section considers a method dealing with transactions with their classes. Also, it considers a method that discovers important patterns composed of items from the transactions. Here, we note that the numbers of the transactions with specific classes are not always equal to each other. That is, the number of transactions corresponding to a class can be large, and the one corresponding to another class can be small. Therefore, the numbers of items corresponding to the former class tend to be frequent, and patterns composed of the items tend to be frequent.

For example, transactions and item classes as shown in Table 4 are given. Also, they have three patterns composed of two items: $(\mathrm{it}_1, \mathrm{it}_2)$, $(\mathrm{it}_1, \mathrm{it}_3)$, and $(\mathrm{it}_2, \mathrm{it}_3)$. Their frequencies are 4, 2, and 1, respectively. $(\mathrm{it}_1, \mathrm{it}_2)$ is the most frequent pattern in the three patterns. Here, we note the ratio of the frequencies to the number of transactions where the patterns can be included. $(\mathrm{it}_1, \mathrm{it}_2)$ can be included in all transactions ($t_1 \sim t_8$). This is because both the item class of it_1 and the one of it_2 are $\{\mathrm{cl}_1, \mathrm{cl}_2\}$, and the class set corresponding to $(\mathrm{it}_1, \mathrm{it}_2)$ is $\{\mathrm{cl}_1, \mathrm{cl}_2\}$. The ratio of $(\mathrm{it}_1, \mathrm{it}_2)$ is 0.50(=4/8). On the other hand, $(\mathrm{it}_1, \mathrm{it}_3)$ can be included in three transactions ($t_6 \sim t_8$). This is because the item class of it_3 is $\{\mathrm{cl}_2\}$. The ratio of $(\mathrm{it}_1, \mathrm{it}_3)$ is 0.67(=2/3). Similarly, $(\mathrm{it}_2, \mathrm{it}_3)$ can be included in three transactions ($t_6 \sim t_8$). The ratio of $(\mathrm{it}_2, \mathrm{it}_3)$ is 0.33(=1/3). Therefore, $(\mathrm{it}_1, \mathrm{it}_3)$ is the pattern with the biggest ratio, and it is the most important pattern in the three patterns.

We anticipate that the important pattern based on the ratio is more valid than the frequent pattern because the

Table 4: An example of transactions and item classes (1).

(a) Transaction

$t_1 = (\mathrm{cl}_1 \mid \mathrm{it}_1)$
$t_2 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_2)$
$t_3 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_2)$
$t_4 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_2)$
$t_5 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_2)$
$t_6 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_3)$
$t_7 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_3)$
$t_8 = (\mathrm{cl}_2 \mid \mathrm{it}_2, \mathrm{it}_3)$

(b) Item class

Item	Class
it_1	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$
it_2	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$
it_3	$\{\mathrm{cl}_2\}$

latter one is often a well-known pattern and does not give a big impact to analysts. It is necessary to define a new criterion corresponding to the ratio. Formula (2) shows the definition. It is based on an idea similar to the support for missing values in tabular structured data [10, 11]. The criterion is called a characteristic support hereafter

$$\mathrm{supp}_{\mathrm{char}}(p) = \frac{N_p}{N_{C_p}}, \quad C_p = \bigcap_{\mathrm{it}_i \in p} C_{\mathrm{it}_i}. \tag{2}$$

In Formula (2), p is a pattern composed of items. N_p is the number of transactions including p. C_p is such a class set that all items composing p can be included. N_{C_p} is the number of transactions whose classes are included in C_p. C_p and N_{C_p} are called a pattern class and a total transaction number for C_p hereafter. We note that C_p can be ϕ. However, if $C_p = \phi$, N_p is 0. It is not necessary to calculate the characteristic support of p. Therefore, it is not necessary to care about the possibility $C_p = \phi$. Also, we note that the characteristic support is in $[0, 1]$. This is because N_p is smaller than or equal to N_{C_p}.

We can anticipate that if the discovery method extracts patterns whose characteristic supports are larger than or equal to a predefined threshold, the patterns can reflect the class distribution in transactions. However, the patterns may be accidentally extracted from a very small transaction subset. We cannot always believe that the patterns are characteristic. Thus, this paper introduces the other criterion assuring the minimum frequency of the pattern. The criterion is equal to the representativity introduced by [12].

This paper tries to discover patterns satisfying two conditions: the minimum support and the minimum frequency. That is, if the characteristic supports of the patterns are larger than or equal to the minimum support and their frequencies are larger than or equal to the minimum frequency, the patterns are extracted as characteristic patterns. In the case of transactions without their classes, these conditions are equal to each other. This is because if the support of a pattern p_1 is always larger than the one of the other pattern p_2 in the latter case, the frequency of p_1 is larger than the one of p_2. It is sufficient for the transactions without their classes to evaluate one of the conditions.

Table 5: An example of transactions and item classes (2).

(a) Transaction

$t_1 = (\mathrm{cl}_1 \mid \mathrm{it}_1)$
$t_2 = (\mathrm{cl}_3 \mid \mathrm{it}_3)$
$t_3 = (\mathrm{cl}_2 \mid \mathrm{it}_2)$
$t_4 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_2)$
$t_5 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_2)$

(b) Item class

Item	Class
it_1	$\{\mathrm{cl}_1, \mathrm{cl}_2, \mathrm{cl}_3\}$
it_2	$\{\mathrm{cl}_2\}$
it_3	$\{\mathrm{cl}_3\}$

Next, we note the property of characteristic supports and frequencies. If two patterns p_1 and p_2 satisfy the relationship $p_1 \subseteq p_2$, the frequency of p_1 is clearly larger than or equal to the one of p_2. The frequencies monotonically decrease as patterns grow. However, the monotonic property is not always satisfied in the case of the characteristic supports. For example, we note transactions and item classes as shown in Table 5.

Table 5 shows that the frequency of it_1 is 3 and the total transaction number for it_1 is 5. This is because it_1 is included in three item classes cl_1, cl_2, and cl_3. The characteristic support of it_1 is 0.60(=3/5). Also, Table 5 shows that the frequency of $(\mathrm{it}_1, \mathrm{it}_2)$ is 2 and the total transaction number for $(\mathrm{it}_1, \mathrm{it}_2)$ is 3. This is because a pattern class of $(\mathrm{it}_1, \mathrm{it}_2)$ is $\{\mathrm{cl}_2\}$. The characteristic support of $(\mathrm{it}_1, \mathrm{it}_2)$ is 0.67(=2/3). This example shows that the monotonic property is not satisfied.

The monotonic property is very important for the discovery method of patterns in order to efficiently discover them. The method can avoid their redundant evaluation by using the property. Therefore, it is difficult for the characteristic support to efficiently discover patterns. Certainly, we can try to discover patterns based on the monotonic property of the frequency. However, the frequency is a limited criterion. The minimum frequency tends to be set as a small value. We cannot anticipate that sufficient efficiency is gained by the frequency.

Thus, this paper introduces another criterion. The criterion is based on an idea similar to the possible support proposed by [10]. The support reflects missing values in tabular structured data. This paper redefines the possible support in order to deal with transactions with their classes. Formula (3) shows the definition. In Formula (3), N_{cl_i} is the number of transactions assigned class cl_i. $\min_{\mathrm{cl}_i \in C_p}\{N_{\mathrm{cl}_i}\}$ is called the minimum transaction number for p hereafter

$$\mathrm{supp}_{\mathrm{pos}}(p) = \frac{N_p}{\min_{\mathrm{cl}_i \in C_p}\{N_{\mathrm{cl}_i}\}}. \tag{3}$$

The redefined possible support is larger than or equal to 0 and can be larger than or equal to 1. This is because N_p can

be larger than $\min_{\mathrm{cl}_i \in C_p}\{N_{\mathrm{cl}_i}\}$. Also, the redefined possible support satisfies the monotonic property. The reason is shown in the following. Firstly, we note the numerator of the formula. The numerator is the frequency of a pattern and monotonically decreases. On the other hand, if two patterns p_1 and p_2 have the relationship $p_2 \supseteq p_1$, the relationship $C_{p_2} \subseteq C_{p_1}$ is clearly satisfied. Thus, $\min_{\mathrm{cl}_i \in C_{p_2}}\{N_{\mathrm{cl}_i}\} \geq \min_{\mathrm{cl}_i \in C_{p_1}}\{N_{\mathrm{cl}_i}\}$. That is, the denominator monotonically increases as the patterns grow. Therefore, the possible supports monotonically decrease as the patterns grow.

Next, we note the relationships between the characteristic support and the possible support. The formulae of these supports show that the numerators are equal to each other. Also, they show that the relationship $N_{C_p} \geq \min_{\mathrm{cl}_i \in C_p}\{N_{\mathrm{cl}_i}\}$ is satisfied in any patterns. Therefore, the relationship described in Formula (4) is always satisfied

$$\mathrm{supp}_{\mathrm{pos}}(p) \geq \mathrm{supp}_{\mathrm{char}}. \tag{4}$$

Formula (4) shows that the possible support of a pattern p_1 gives the upper bound to characteristic supports of any super patterns ($\supseteq p_1$). Therefore, if the possible support of p_1 is smaller than the minimum support, the characteristic supports of the supper patterns are smaller than the minimum support. The supper patterns cannot be characteristic patterns. Thus, the discovery method keeps patterns whose possible supports are larger than or equal to the minimum support and whose frequencies are larger than or equal to the minimum frequency in order to grow them. The patterns are called possible patterns hereafter. Also, the discovery method extracts patterns whose characteristic supports are larger than or equal to the minimum support from the possible patterns. The discovery method based on the two-stepwise evaluation method can efficiently discover all characteristic patterns.

2.3. Discovery Method. This paper proposes a method that discovers all characteristic patterns from transactions with their classes. The method is based on the FP tree and the FP growth proposed by [4]. Here, the FP tree is the data structure storing transactions with tree format. The FP growth is a method discovering all frequent patterns by using the FP tree. The FP growth generates the FP trees from transaction subsets and generates new transaction subsets from generated FP trees. Two kinds of generation steps are recursively repeated. This paper proposes the refined FP tree and the refined FP growth in order to deal with transactions with classes.

The original FP tree is composed of nodes assigned an item name and its frequency, links tying to the nodes with the same item name, a header table storing item, and links tying to items in the table with nodes including the item names. In the FP tree, the root node is a special node and is assigned "null". The refined FP tree is basically equal to the original one. But, the refined one is assigned a pattern class and an identification flag to each item in the header table. Here, the flag shows whether the pattern corresponding to each item in the header table is included in a characteristic pattern. The refined FP tree is generated by an algorithm as shown in Table 6 from transactions with their classes, item classes, and a conditioned item set. In the first generation of the refined FP tree, the conditioned item set is ϕ. In the generation except the first one, it is at least possible patterns. In this algorithm, the minimum support ($\in(0,1]$) and the minimum frequency (≥ 1) are predefined. Table 6 describes the difference of the generation method for the original one with bold style.

Table 6: A generation method of the refined FP tree.

(1) Search transaction and calculate frequencies of each item.

(2) **Calculate a product set between a pattern class C_{Its} of a conditioned item set** Its **and an item class** C_{it_i} of **each item** it_i, **and set the product set as a pattern class** C_{p_i} **of a pattern** p_i (=Its $\cup$ it_i).

(3) **Calculate the total number and the minimum number for** p_i.

(4) **Calculate a characteristic support and a possible support of** p_i.

(5) Repeat from step 2 to step 4 for any *i*.

(6) **Extract items whose patterns are possible patterns and generate a list Flist arranged the items in descending order, where the first key is the frequency, the second key is the characteristic support, and the third key is the possible support**.

(7) Regard Flist as a header table H in a refined FP tree T. **Also, tie** C_{p_i} to it_i in H. **In addition, tie an identification flag** fl_i of p_i **to** it_i **in** H. **Here, if the pattern corresponding to** it_i **is a characteristic pattern, the value of** fl_i is "C". **Otherwise the value is "P". That is, "C" shows that the pattern is a characteristic pattern and "P" shows that the pattern is not a characteristic pattern and is a possible pattern**.

(8) Create a root node of T and assign the label "null" to it.

(9) Set the root node as a target node N.

(10) Pick up a transaction t_j. If the transaction cannot be picked up, then this algorithm stops.

(11) Pick up only items included in H from t_j, sort them in the order of items in H, and create a selected and sorted item set Its_{t_j}.

(12) Pick up an item it_{jk} from the top of Its_{t_j}. If the item cannot be picked up, then go to step 10.

(13) If an item name of it_{jk} is assigned to a child node of N, then the frequency of the node is counted up. Otherwise, a new node N' is created. N' is assigned the item name of it_{jk} and is set 1 to the frequency. Also, create a link between N and N', and register N' to it_{jk} in H. In addition, regard a selected child node or N' as a new target node N.

(14) Go to step 12.

In the following, we show an example which generates the refined FP tree from transactions and item classes as shown in Table 7.

In this case, the item set is ϕ. The minimum support is 0.5 and the minimum frequency is 1. The transactions of Table 7 show that frequencies of $\mathrm{it}_1 \sim \mathrm{it}_9$ are 6, 3, 3, 4, 3, 2, 1, 1, and 1, respectively. The item classes of Table 7 show that the total transaction numbers for it_1, it_2, it_4, it_5, and it_9 are 7, the ones for it_3 and it_8 are 3, and the ones for it_6 and it_7 are 4. Also, it shows that the minimum transaction

Table 7: An example of transactions and item classes (3).

(a) Transaction

$t_1 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_2, \mathrm{it}_3, \mathrm{it}_5)$
$t_2 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_3, \mathrm{it}_4)$
$t_3 = (\mathrm{cl}_1 \mid \mathrm{it}_1, \mathrm{it}_2, \mathrm{it}_3, \mathrm{it}_4, \mathrm{it}_8)$
$t_4 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_2, \mathrm{it}_4, \mathrm{it}_7)$
$t_5 = (\mathrm{cl}_2 \mid \mathrm{it}_4, \mathrm{it}_5, \mathrm{it}_6)$
$t_6 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_5, \mathrm{it}_6)$
$t_7 = (\mathrm{cl}_2 \mid \mathrm{it}_1, \mathrm{it}_9)$

(b) Item class

Item	Class
it_1	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$
it_2	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$
it_3	$\{\mathrm{cl}_1\}$
it_4	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$
it_5	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$
it_6	$\{\mathrm{cl}_2\}$
it_7	$\{\mathrm{cl}_2\}$
it_8	$\{\mathrm{cl}_1\}$
it_9	$\{\mathrm{cl}_1, \mathrm{cl}_2\}$

Table 8: A selected and sorted transaction subset.

$t'_1 = (\mathrm{it}_1, \mathrm{it}_3, \mathrm{it}_2, \mathrm{it}_5)$
$t'_2 = (\mathrm{it}_1, \mathrm{it}_4, \mathrm{it}_3)$
$t'_3 = (\mathrm{it}_1, \mathrm{it}_4, \mathrm{it}_3, \mathrm{it}_2)$
$t'_4 = (\mathrm{it}_1, \mathrm{it}_4, \mathrm{it}_2)$
$t'_5 = (\mathrm{it}_4, \mathrm{it}_5, \mathrm{it}_6)$
$t'_6 = (\mathrm{it}_1, \mathrm{it}_5, \mathrm{it}_6)$
$t'_7 = (\mathrm{it}_1)$

numbers for it_1, it_2, it_3, it_4, it_5, it_8, and it_9 are 3 and the ones for it_6 and it_7 are 4. Possible supports of $\mathrm{it}_1 \sim \mathrm{it}_9$ are 2.00(=6/3), 1.00(=3/3), 1.00(=3/3), 1.33(=4/3), 1.00(=3/3), 0.50(=2/4), 0.25(=1/4), 0.33(=1/3), and 0.33(=1/3), respectively. Therefore, the generation method can select it_1, it_2, it_3, it_4, it_5, and it_6 as possible items. The items are arranged in the descending order of frequencies, characteristic supports, and possible supports. The method generates the list Flist such as $(\mathrm{it}_1, \mathrm{it}_4, \mathrm{it}_3, \mathrm{it}_2, \mathrm{it}_5, \mathrm{it}_6)$. But, indexes of the items are used as the fourth key, if three criteria are equal to each other.

Next, the method picks up possible items from the transactions and arranges the items in order of the Flist. It generates a selected and sorted transaction subset as shown in Table 8. The subset is used in order to generate the refined FP tree.

In the case that t'_1 is selected from the subset, the method generates a part of the refined FP-tree as shown in Figure 1(a). Also, in the case that t'_2 is selected from the subset, it generates a part of the refined FP tree as shown in Figure 1(b). That is, the method adds new nodes or updates the frequencies of the existing node in order from the root node. Lastly, the refined FP tree as shown in Figure 2 is generated.

In the remaining part of this subsection, we explain the refined FP growth. The original FP growth detects a single prefix path. The path is a path linked from the root. Each node included in the path has only a child. If an FP tree has the single prefix path, the original FP growth divides the FP-tree into the single prefix path and the remaining part. The remaining part is added a new root node and a new FP tree is generated. The original FP growth separately discovers frequent patterns from the new FP-tree and from the single prefix path. Also, it combines the patterns based on the tree with the patterns based on the path. The combined patterns are frequent patterns. The original FP growth can discover all frequent patterns based on the combination.

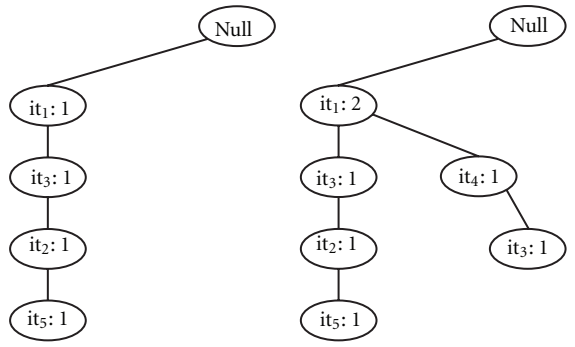
On the other hand, in the case of the refined FP growth, we do not know whether characteristic supports increase as patterns grow. Even if characteristic patterns based on a single prefix path are given and characteristic patterns based on a new refined FP tree are given, their combinations are not always characteristic patterns. It is necessary to check whether the combinations are characteristic patterns by calculating their characteristic supports. The detection of the single prefix path does not give sufficient merits. Thus, the refined FP growth tries to discover characteristic patterns without detecting a single prefix path. In the following, we explain the FP growth without the detection.

The refined FP growth picks up an item it from the rear of Flist in the refined FP tree in order. It generates transactions conditioned by it. That is, the refined FP growth extracts nodes including it from the refined FP tree and extracts paths from upper nodes of the extracted nodes to the root node. A transaction is generated from each path and is assigned frequency which is equal to the frequency of the node including it. The refined FP growth generates a new refined FP tree from the conditioned transactions. it is added to the conditioned item set Its. Here, Its is assigned to the previous refined FP tree and is ϕ in the first process of the refined FP growth. The refined FP growth recursively repeats the generation of refined FP trees and conditioned transactions.

On the other hand, the refined FP growth needs pattern classes in order to calculate characteristic supports and possible supports. The pattern classes can be generated by referring to both items included in the patterns and their item classes. However, if the generation is always started from scratch, it is redundant. We can generate the pattern classes more efficiently by using a relationship among pattern classes. That is, if a pattern p' is generated by the combination of a pattern p and a new item it, a pattern class $(C_{p'})$ of p' is equal to the product set $(C_p \cap C_{\mathrm{it}})$ based on both a pattern class (C_p) of p and an item class (C_{it}) of it. Therefore, the header table keeps pattern classes.

According to the above discussion, the refined FP growth is described as shown in Table 9. Table 9 describes the difference between the original FP growth and the refined one with bold style. The refined FP tree T and Its are input to the refined FP growth.

Lastly, a discovery method of characteristic patterns is described as shown in Table 10. The method appropriately



(a) After t_1' (b) After t_2'

Figure 1: Refined FP tree (1).

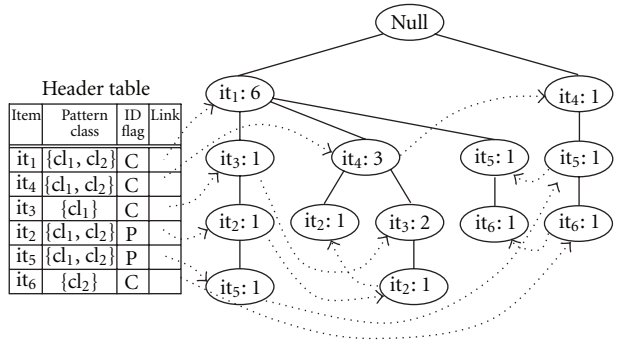


Figure 2: Refined FP tree (2).

Table 9: An algorithm of the refined FP growth algorithm.

(1) Extract an item it from the rear of Flist in a refined FP tree T. If the item is not extracted, this algorithm stops.

(2) Add it to an item set Its and **investigate an identification flag of** it. **If the flag shows a characteristic pattern, then output Its as a characteristic pattern**.

(3) Extract a node N whose item is equal to it from T. If the node is not extracted, then go to step 6.

(4) Extract a path from the upper node of N to the root. Generate an item set composed of item included in the extracted path and extract the frequency assigned to N. Regard the combination of the item set and the frequency as a transaction conditioned by it.

(5) Go to step 3.

(6) Apply a set Trans of the generated transactions to the algorithm shown in Table 6 and generate a new refined FP tree T'. If T' is not generated, then this algorithm stops. Otherwise, this algorithm recursively calls by giving T' and Its to it as arguments.

(7) Go to step 1.

calls the algorithms as shown in Tables 6 and 9. It can generate all characteristic patterns from transactions with their classes, item classes, the minimum support, and the minimum frequency.

Table 10: A discovery method of characteristic patterns.

(1) Search transactions and calculate the number of transactions for each class.

(2) Set ϕ to an item set Its and set all classes to its pattern class.

(3) Generate a refined FP tree T based on the algorithm shown in Table 6.

(4) If T is not generated, then this algorithm stops. Otherwise, it discovers characteristic patterns from T and Its based on the algorithm shown in Table 9.

3. Numerical Experiments

3.1. Data. This paper uses the UCI data sets registered in the UCI machine learning repository for numerical experiments. Also, it uses the RFID data sets collected from two apparel shops. In the case of the UCI data sets, three data sets "car", "hayes", and "nursery" are selected. This is because each data set is a data set for the classification task and is composed of discrete values. These data sets are tabular structured data. Each example in the data sets is composed of attribute values and a class and is regarded as a transaction. Then, the combination of an attribute and an attribute value is regarded as an item. A class of an example is regarded as a class of a transaction. The item classes are generated by checking the combinations of attribute values and classes. Table 11 shows features of transactions generated from the UCI data sets.

Table 11: UCI data sets.

Data set	Item	Transaction	Class
Car	21	1728	4
Hayes	15	160	3
Nursery	27	12960	5

Table 12: RFID data sets.

Shop	No fitting	Fitting
α shop	3695	1372
β shop	7612	1849

The RFID data sets are data sets collected in order to analyze a relationship between sales volume and actions of customers in the shops. RFID readers are set to fitting rooms, shelves, and cash registers, and RFID tags are assigned to sales items. We can grasp the actions of the customers by chasing the items to some extent. In this experiment, we regard the difference between fitting items and no fitting items as classes. This is because the fitting of items is one of the important customer actions but its frequency is not always high. Table 12 shows the distribution of classes in transactions. We can confirm that the number of fitting items is much smaller than the one of no fitting items.

3.2. Method. We compare the proposed method with the existing method [4], but the process of a single prefix path is not incorporated in it. The existing method is a discovery method of frequent patterns. It does not deal with classes. Also, it ignores classes assigned to transactions and item classes corresponding to each item. The proposed method aims at discovering at most hundreds of patterns. The minimum supports and the minimum frequencies are adjusted.

In the case of the UCI data sets, the proposed method discovers characteristic patterns by changing the minimum frequencies. The minimum support is fixed. Also, the proposed method outputs possible patterns in order to evaluate investigated patterns. On the other hand, the existing method discovers frequent patterns by changing the minimum frequencies. The minimum support is directly calculated by referring to the frequencies. The frequencies depend on both the minimum supports and the minimum frequencies of the proposed method. This experiment evaluates relationships between discovered patterns, minimum supports, and minimum frequencies.

In the case of the RFID data sets, this experiment focuses on the difference of characteristic patterns and frequent patterns. The proposed method and the existing method discover patterns by changing the minimum supports. The minimum frequency in the proposed method is fixed. This experiment evaluates the validity of the discovered patterns.

3.3. Experimental Results. Figures 3, 4, and 5 show experimental results in the case of the UCI data sets: "cars", "hayes", and "nursery", respectively. Graph (a) and Graph (b) show the difference of characteristic patterns and frequent patterns. Graph (a) shows the total number of patterns. In this graph, we do not care about the number of items included in patterns. The number is called the length hereafter.

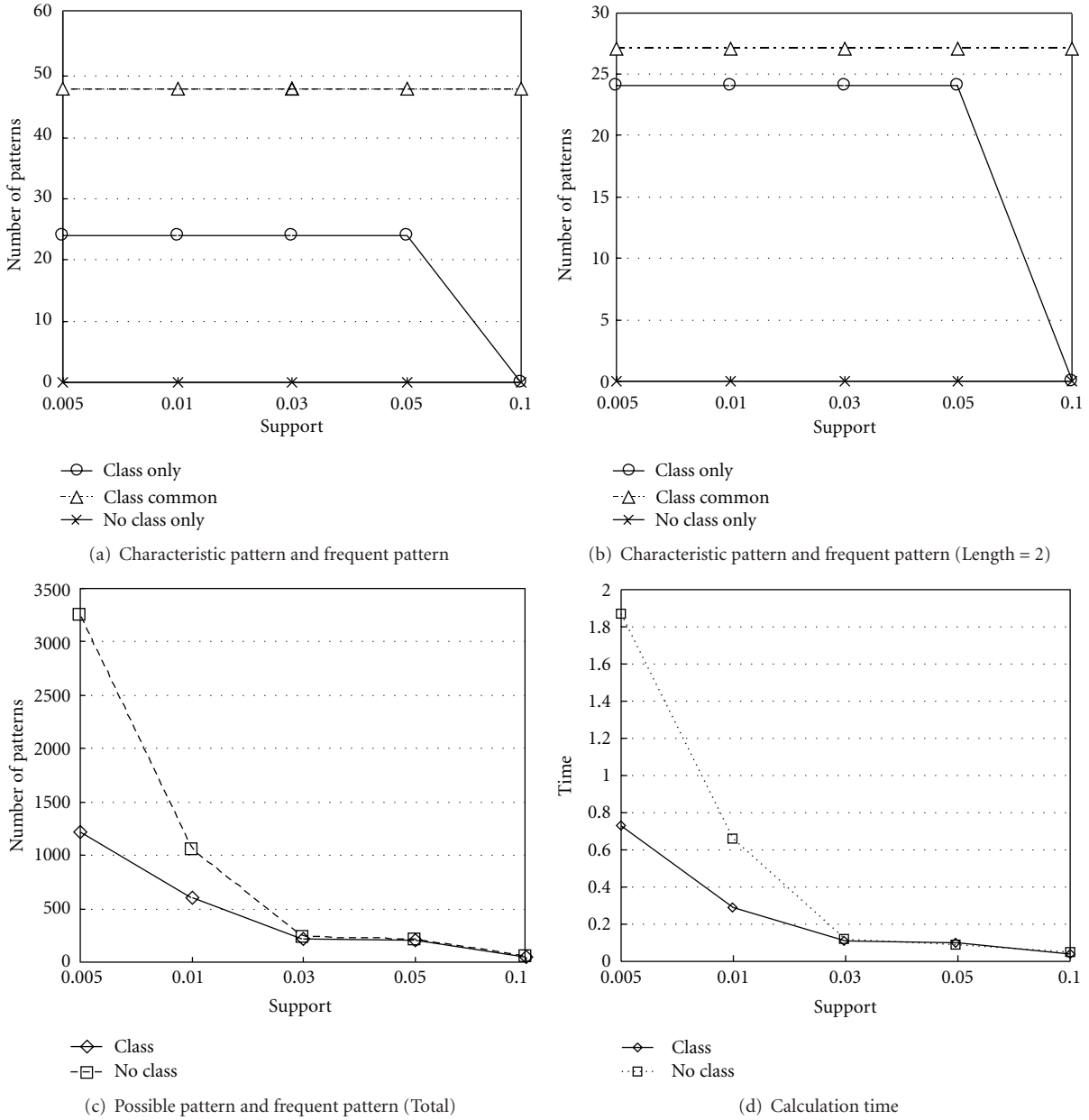
Graph (b) shows the difference of characteristic patterns and frequent patterns in the case of specific length. The lengths in the case of "cars," "hayes," and "nursery," are 2, 2, and 3, respectively. In each graph, "Class only" shows the number of patterns discovered only by the proposed method. "Class common" shows the numbers discovered by both the proposed method and the existing method. "No class only" shows the numbers discovered only by the existing method. But, in the case of the proposed method, the minimum support is fixed to 0.100. Also, the minimum frequencies in the case of "car" are 9, 18, 52, 87, and 173. The ones in the case of "hayes" are 1, 2, 5, 8, and 16. The ones in the case of "nursery" are 65, 130, 389, 649, and 1297. They correspond to the minimum supports of the existing method: 0.005, 0.010, 0.030, 0.050, and 0.100. In the case of the existing method, the minimum frequency for the data sets: "car", "hayes", and "nursery" are fixed and they are 173, 16, and 1297, respectively. They correspond to the minimum support 0.100.

Graph (c) shows the difference of possible patterns and frequent patterns. The thresholds in the proposed method are equal to the case of Graph (a) and Graph (b). The minimum supports in the existing method are changed according to the frequencies in the proposed method. In each graph, "Class" shows the total number of possible patterns and "No class" shows the total number of frequent patterns.

Graph (d) shows the calculation time of both the proposed method and the existing method. But, the calculation time of the existing method is estimated by modifying the program code of the proposed method. That is, the calculation of the characteristic support and the possible support, and the evaluation based on them are gotten rid of the program code. On the other hand, other additional processes are left because their calculation cost is comparatively small. The modified program code is used to the estimation of the calculation time for the existing method. In the experiment, the calculation time is measured by using the DOS command "time" with millisecond unit. The experiment uses the computer environment such as Windows 7 Home Premium Service Pack 1 and TOSHIBA dynabook Satellite PXW/59KW loading Intel Core(TM)2 Duo CPU T9600@2.80 GHz and 4.00 GB RAM memory. The program code is described by C language and is compiled by gcc command in MinGW-5.1.6.exe. In the case of the proposed method, the minimum support is fixed to 0.100 and the minimum frequencies are changed. The frequencies are used in the evaluation of the existing method.

In these graphs, horizontal axes are the minimum supports, and vertical axes except Graph (d) are the number of patterns. In Graph (d), vertical axes are the calculation time, and their calculation unit is millisecond.

Figure 6 shows experimental results in the case of the RFID data sets. In this experiment, the minimum frequency is 2 in the proposed method. The minimum supports are changed from 0.001 to 0.005 per 0.001 in the proposed method and the existing method. Graph (a) and Graph (b) show the total number of patterns in the case of α shop and β shop, respectively. In these graphs, "Class only",



(a) Characteristic pattern and frequent pattern

(b) Characteristic pattern and frequent pattern (Length = 2)

(c) Possible pattern and frequent pattern (Total)

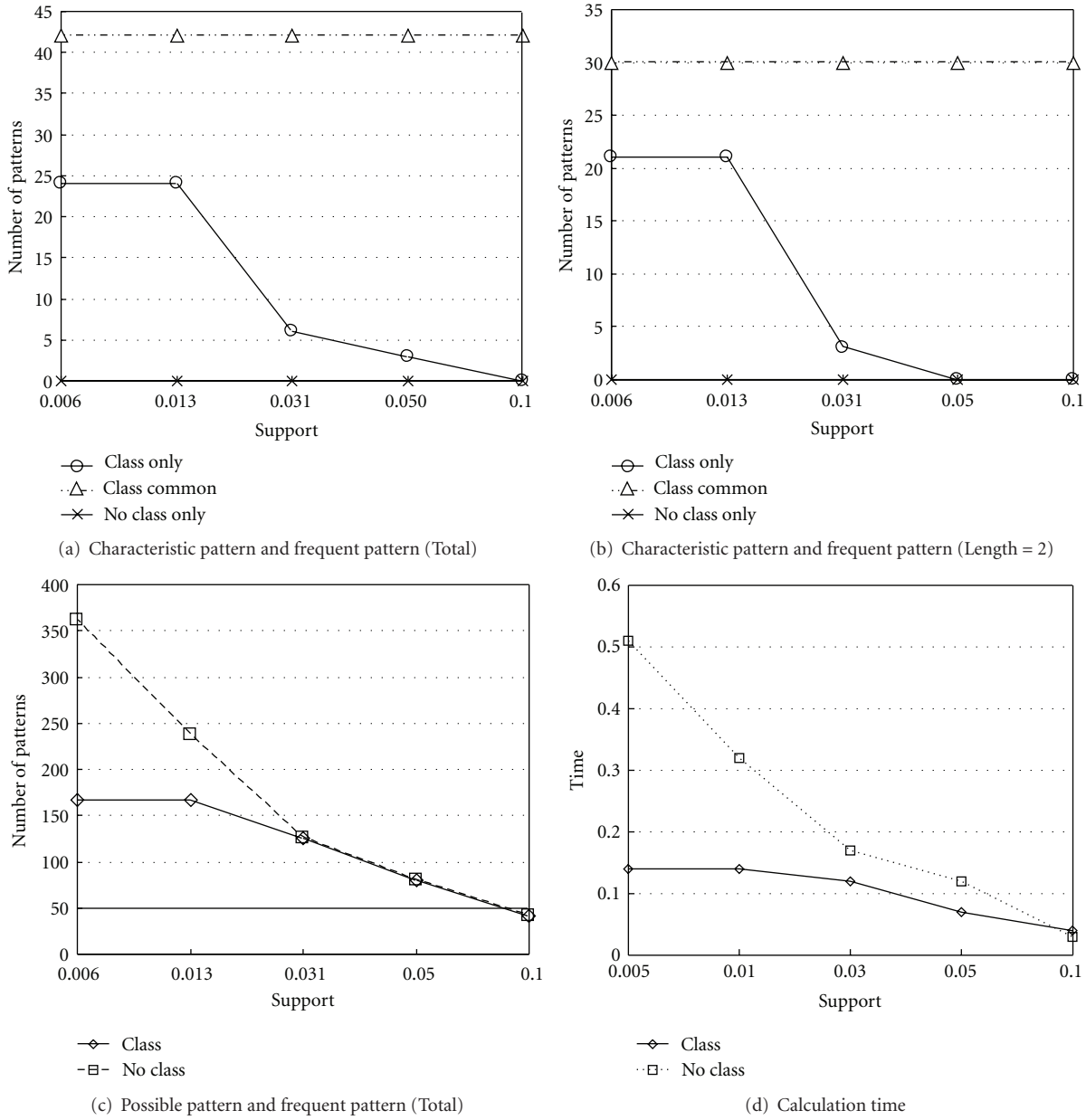
(d) Calculation time

Figure 3: Experimental result in the case of the data set "car".

"Class common", "No class", and the axes correspond to the ones in the case of the UCI data sets. On the other hand, Tables 13 and 14 show patterns discovered only by the proposed method from each shop. In these tables, the minimum support is 0.001, and the length is larger than or equal to 2.

3.4. Discussions. This section discusses the effect of the proposed method due to seven viewpoints: "difference of discovered patterns", "validity of discovered patterns", "frequency of items", "calculation time", "complexity of the refined FP tree", "pre-processing method", and "generality".

3.4.1. Difference of Discovered Patterns. Graph (a) and Graph (b) show that there are many patterns discovered only by the proposed method. Especially, when the minimum frequencies in the proposed method are small, the numbers of the patterns are big. We can anticipate that analysts are interested in the patterns because the ratios included in a specific class set are comparatively high. On the other hand, it is difficult for the existing method to discover the patterns. If the existing method tries to discover them, it is necessary to set smaller minimum frequencies. However, the smaller minimum frequencies tend to discover many frequent patterns. The existing method may overlook important patterns because the patterns are hidden in many frequent patterns.



(a) Characteristic pattern and frequent pattern (Total)

(b) Characteristic pattern and frequent pattern (Length = 2)

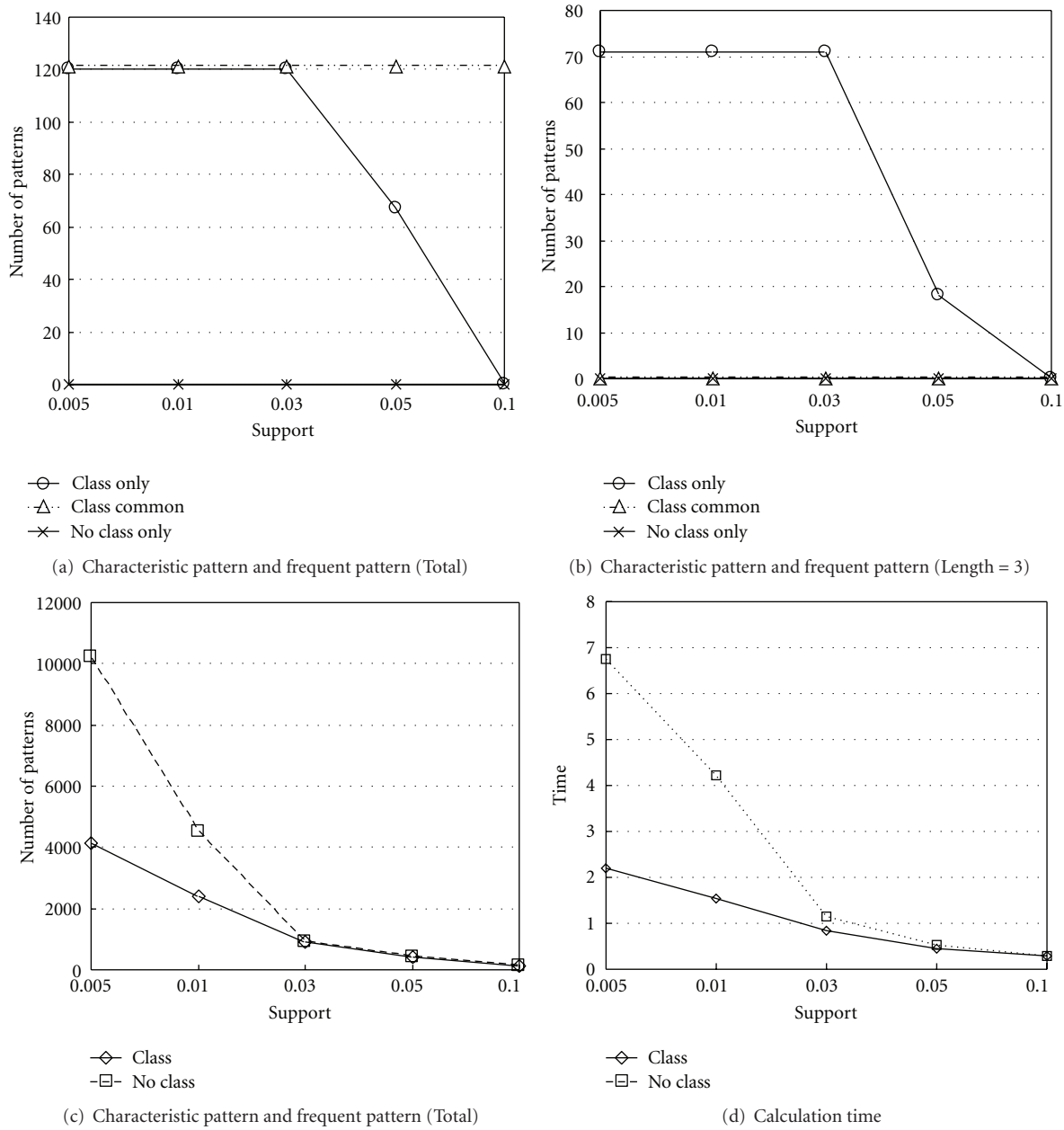
(c) Possible pattern and frequent pattern (Total)

(d) Calculation time

Figure 4: Experimental result in the case of the data set "hayes".

In order to avoid the overlook, we may be able to perform a two-stepwise method. The method combines the existing method with a postprocessing method. That is, the first step discovers frequent patterns by applying the existing method to its smaller minimum frequencies. The second step extracts characteristic patterns from the discovered patterns by evaluating their characteristic supports. However, Graph (c) shows that the number of possible patterns is smaller than the number of frequent patterns. The former numbers are 0.376 times, 0.461 times, and 0.404 times as small as the latter numbers in the case of "car", "hayes", and "nursery", respectively. The proposed method can discover all characteristic patterns by investigating possible patterns whose number is less than half as small as the numbers based on the existing method. The proposed method is more efficient method with calculation cost.

3.4.2. Validity of Discovered Patterns. Tables 13 and 14 show that the proposed method discovers many patterns composed of a top wear and a bottom wear. If a customer is interested in only the top wear, the top wear is not usually fitted in a fitting room. However, if the customer is interested in the combination of the top wear and the bottom wear, the customer tends to fit the combination. The discovered patterns show the combination. On the other hand, the fitting is one of troublesome actions for the customer. The sales volume without the fitting is much larger than the one



(a) Characteristic pattern and frequent pattern (Total)

(b) Characteristic pattern and frequent pattern (Length = 3)

(c) Characteristic pattern and frequent pattern (Total)

(d) Calculation time

Figure 5: Experimental result in the case of the data set "nursery".

with the fitting. It is difficult for the existing method to discover patterns related to the fitting. We can think that the proposed method discovers more valid patterns than the existing method does. In future work, we will try to show the patterns to retailers in the apparel shops and verify the validity of the patterns in detail.

3.4.3. Frequency of Items. In the case of "car" and "nursery", characteristic patterns whose length is 1 completely correspond to frequent patterns whose length is 1. Characteristic patterns whose length is more than equal to 2 include patterns which the existing method cannot discover. This is because characteristic supports increase as the number of classes included in pattern classes decreases. The proposed method discovers the characteristic patterns related to the combination of specific classes. We think that the feature of the proposed method is experimentally confirmed.

3.4.4. Calculation Time. The results in Figures 3–5 show that the shapes of Graph (c) are similar to the ones of Graph (d). That is, the calculation time depends on the number of the patterns. It does not almost depend on the additional calculation cost related to the proposed method. The additional calculation cost of the proposed method is

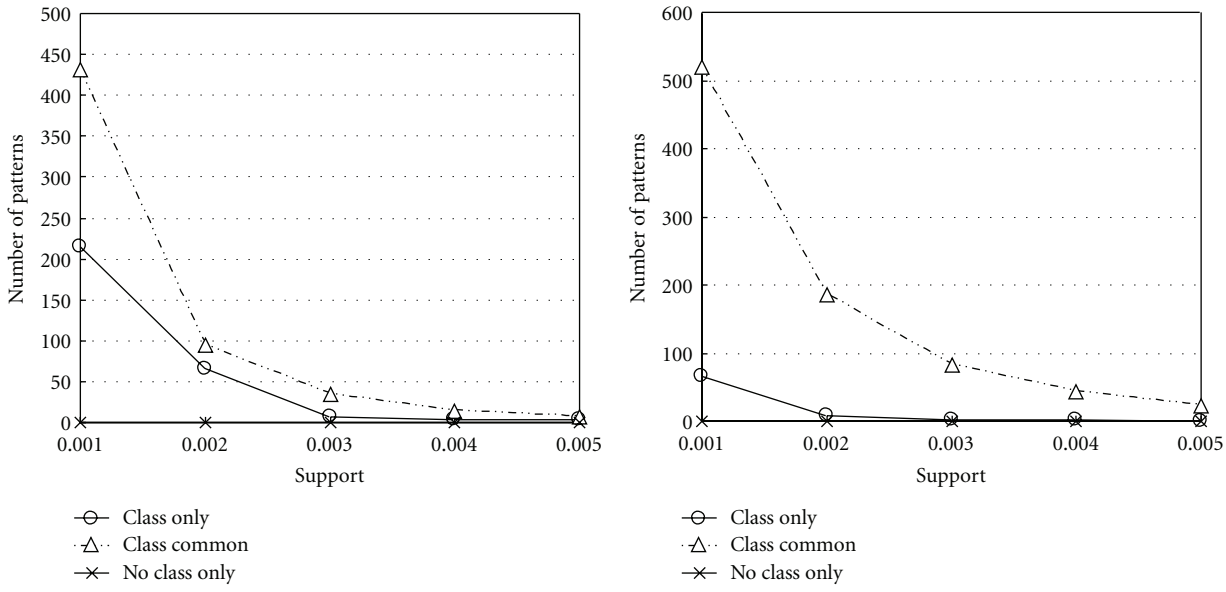


Figure 6: Experimental result in the case of the RFID data sets.

Table 13: Discovered patterns from α shop.

Pattern	Class
A-type jacket, A-type skirt	Fitting
X-type stole, Y-type gift box	No fitting
B-type one-piece wear, B-type jacket	Fitting
C-type jacket, D-type skirt	Fitting
C-type jacket, C-type skirt	Fitting
E-type jacket, E-type pants	Fitting
B-type jacket, B-type skirt	Fitting
F-type jacket, F-type skirt	Fitting
F-type jacket, G-type skirt	Fitting
F-type jacket, H-type blouse	Fitting
G-type skirt, H-type blouse	Fitting
F-type jacket, G-type skirt, H-type blouse	Fitting

Table 14: Discovered patterns from β shop.

Pattern	Class
I-type one-piece wear, H-type necklace	Fitting
C-type jacket, C-type pants	Fitting
K-type jacket, K-type skirt	Fitting

not so big. We can anticipate that the proposed method speedily discovers the patterns because it investigates small combinations of items.

3.4.5. Complexity of the Refined FP-Tree. The refined FP-tree has pattern classes and ID flags in the header table. It requires additional memory. However, the nodes composing the refined FP-tree occupy the greater part of memory. This is because the number of transactions and the one of items are much larger than the number of classes. We think that the additional memory is not a big problem.

3.4.6. Preprocessing Method. Even if the existing method directly cannot deal with classes, a combination method of the existing method and a preprocessing method may be able to deal with them. That is, the preprocessing method can divide transactions into some transaction subsets by referring to the classes. The existing method can discover frequent patterns from divided transaction subsets. The combination method may be able to discover all characteristic patterns. However, if the number of classes is large, the combinations of classes exponentially increase. In addition, we do not usually have the knowledge for useful combinations of classes. It is necessary to discover patterns from various transaction subsets. The combination method requires large amount of calculation cost. On the other hand, the proposed method can discover characteristic patterns without requiring the additional knowledge. Therefore, the proposed method is more efficient than the combination method.

3.4.7. Generality. We can regard various conditions as classes. For example, in a retail field, the difference of sales period, shop, and salesperson is regarded as classes. Also, in a sensor network field, the difference of area, data collection interval, and data detection range is regarded as classes. Therefore, we think that the proposed method has high generality. The proposed method can analyze data with various viewpoints by using the classes.

According to the discussions, we believe that the proposed method can perform more efficient analysis for transactions.

4. Summary and Future Work

This paper proposed a method that discovers characteristic patterns from transactions with their classes. The method regards the difference of conditions in the data collection as classes. This paper redefined the possible support and the characteristic support in order to efficiently discover the patterns. Lastly, it verified the effect of the proposed method through numerical experiments based on three data sets registered in the UCI machine learning repository and two data sets collected by RFID readers and RFID tags set in apparel shops.

In future work, we try to improve the proposed method with a viewpoint of calculation speed. For example, we consider a method that calculates smaller possible supports in order to decrease the number of possible patterns. Concretely, we try to use relationships between the growth of patterns and their pattern classes. We believe that the relationships can estimate the possible supports as smaller values. Also, we try to reconsider the use of the single prefix path. In this paper, we ignore the path because characteristic patterns cannot be simply discovered from the combination of the single prefix path and the remaining FP tree. However, similar combinations may be useful for the speedy discovery. It is necessary to investigate the feature of the combinations in detail. In addition, we will consider to use the class information in nodes of a refined FP tree. The use may lead to the discovery of more valid patterns. On the other hand, it leads to a more complex FP tree with high calculation cost. It is important to clarify the relationships between the validity of patterns and the complexity of the FP tree. Lastly, we will try to apply the proposed method to many data sets in order to verify its effect in detail.

The discovery task from transactions without classes is expanding the research field more and more. The expansion may be useful for the discovery task from transactions with their classes. We try to incorporate the expansion in the proposed method.

References

[1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proceedings of the 20th International Conferance on Very Large Data Bases*, pp. 487–499, Santiago, Chile, 1994.

[2] T. Morzy and M. Zakrzewicz, "Group Bitmap Index: a structure for association rules retrieval," in *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pp. 284–288, New York, NY, USA, 1998.

[3] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New algorithms for fast discovery of association rules," in *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pp. 283–286, Newport Beach, Calif, USA, 1997.

[4] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1–12, Dallas, Tex, USA, June 2000.

[5] Z. H. Deng and G. D. Fang, "Mining top-rank-K frequent patterns," in *Proceedings of the 6th International Conference on Machine Learning and Cybernetics (ICMLC '07)*, pp. 851–856, Hong Kong, August 2007.

[6] X. Yan, J. Han, and R. Afshar, "CloSpan: mining closed sequential patterns in large datasets," in *Proceedings of the SIAM International Conference on Data Mining*, pp. 166–177, San Fransisco, Calif, USA, 2003.

[7] R. J. Bayardo Jr., "Efficiently mining long patterns from databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 85–93, Seattle, Wash, USA, June 1998.

[8] C. H. Cai, A. W. C. Fu, C. H. Cheng, and W. W. Kwong, "Mining association rules with weighted items," in *Proceedings of the International Database Engineering and Applications Sympoium*, pp. 68–77, Cardiff, UK, 1998.

[9] M. Pater and D. E. Popescu, "Multi-level database mining using AFOPT data structure and adaptive support constrains," *International Journal of Computers, Communications & Control*, vol. 3, pp. 437–441, 2008.

[10] S. Sakurai and K. Mori, "Discovery of characteristic patterns from tabular structured data including missing values," *International Journal of Business Intelligence and Data Mining*, vol. 5, no. 3, pp. 213–230, 2010.

[11] A. Ragel and B. Crémilleux, "Treatment of missing values for association rules," in *Proceedings of the 2nd Pacific-Asia Conference on Research and Development in Knowledge Discovery and Data Mining*, pp. 258–270, Melbourne, Australia, 1998.

[12] T. Calders, B. Goethals, and M. Mampaey, "Mining itemsets in the presence of missing values," in *Proceedings of the ACM Symposium on Applied Computing*, pp. 404–408, Seoul, Korea, March 2007.

[13] University of California Irvine, UCI Machine Learning Repository, 2011, http://archive.ics.uci.edu/ml/.

[14] S. Sakurai, "Prediction of sales volume based on the RFID data collected from apparel shops," *International Journal of Space-Based and Situated Computing*, vol. 1, no. 2-3, pp. 174–182, 2011.