Rule Discovery and Matching in Stock Databases

You-min Ha
Department of Computer Science
Yonsei University, Korea
ymha@cs.yonsei.ac.kr

Sanghyun Park
Department of Computer Science
Yonsei University, Korea
sanghyun@cs.yonsei.ac.kr

Sang-Wook Kim
College of Information and Communications
Hanyang University, Korea
wook@hanyang.ac.kr

Jung-Im Won
College of Information and Communications
Hanyang University, Korea
jiwon@hanyang.ac.kr

Jee-Hee Yoon
Division of Information Engineering and Telecommunications
Hallym University, Korea
jhyoon@hallym.ac.kr

Abstract

This paper addresses an approach that recommends investment types to stock investors by discovering useful rules from past changing patterns of stock prices in databases. First, we define a new rule model for recommending stock investment types. For a frequent pattern of stock prices, if its subsequent stock prices are matched to a condition of an investor, the model recommends a corresponding investment type for this stock. The frequent pattern is regarded as a rule head, and the subsequent part a rule body. We observed that the conditions on rule bodies are quite different depending on dispositions of investors while rule heads are independent of characteristics of investors in most cases. With this observation, we propose a new method that discovers and stores only the rule heads rather than the whole rules in a rule discovery process. This allows investors to impose various conditions on rule bodies flexibly, and also improves the performance of a rule discovery process by reducing the number of rules to be discovered. For efficient discovery and matching of rules, we propose methods for discovering frequent patterns, constructing a frequent pattern base, and its indexing. We also suggest a method that finds the rules matched to a query from a frequent pattern base, and a method that recommends an investment type by using the rules. Finally, we verify the effectiveness and the efficiency of our approach through extensive experiments with real-life stock data.

1. Introduction

Around us, there are a variety of objects such as stock prices, temperature values, and money exchange rates whose values change as time goes by. The list of changing values sampled at a fixed time interval is called $time\text{-}series\ data$ [2, 13, 14, 16]. In many applications, an element value in time-series data is affected by its preceding values accumulated[7]. Thus, by analyzing past element values in time-series data, we can find the regularities and also build their model, thereby predicting the values to appear in the future.

Stock price sequences are a typical example of time-series data[3, 10]. Since the goal of stock investors is to maximize their earnings, it would help them achieve successful investments to recommend proper buying and selling points via analysis of the stock price sequences.

Time-series analysis[6] has been a well-known method for predicting stock prices in the future. It is classified into two categories: time domain analysis and frequency domain analysis. *Time domain analysis* is based on the regression model, which assumes that a current value is determined by the regression of its preceding values[6]. *Frequency domain analysis* is primarily used in analyzing stationary time-series data for predicting macroscopic tendencies of months, seasons, or years. However, they cannot reflect the conditions for investments, which could be dynamically changed by investors, and also have a problem of not being appropriate for short-term predictions.

In a machine learning perspective, there have been some

0730-3157/08 $25.00 © 2008 IEEE
DOI 10.1109/COMPSAC.2008.20

methods proposed for predicting future values by analyzing past values with the neural network[17, 18]. For reflecting the various conditions of investors, however, these methods should construct a neural network for each condition, therefore, incur a large storage overhead in main memory. Also, they have a difficulty in applying themselves to a huge database environment due to their inherent scalability problem.

In a database perspective, there have been research efforts on rule discovery and matching in time-series data[9, 15]. Reference [9] proposed a method that transforms time-series sequences into symbol sequences and then discovers rules from them. For transformation, it extracts a number of *windows* of a fixed length from each time-series sequence, and classifies all the windows into multiple groups by using a clustering technique[11]. It assigns a symbol to each window group, and converts every window in a time-series sequence into its corresponding symbol. Finally, all the time-series sequences become a set of symbol sequences. Reference [15] introduced a concept of *elastic rules*, and also proposed a method that discovers the elastic rules effectively by using the *suffix tree*. For suffix tree construction, this method uses the *TAH–tree*[8] in order to convert time-series sequences into symbol sequences.

A rule consists of a *rule head* and a *rule body*. A common feature of the prior methods proposed in a database community is to find both the heads and the bodies of rules in a rule discovery process. This implies that the conditions on the heads and the bodies should be defined earlier than the beginning of the rule discovery process. As mentioned in Section 2, however, such conditions used for recommending stock investment types are highly dependent on dispositions of investors. Thus, the prior methods that discover whole rules inherently have two problems: 1) They produce a large number of rules and 2) they cannot reflect the conditions newly defined by users on the set of rules obtained from the previous rule discovery process.

In this paper, we propose a novel approach for discovering and matching of rules that solves these problems, and discuss how to apply it to stock investment. Unlike the prior methods that discover both heads and bodies of rules in advance, our approach discovers only the rule heads and stores them in a *frequent pattern base*. If a user raises a query on a stock of his/her interests, it finds a rule head matched to a query from a frequent pattern base, and then discovers its rule body at this point. With this strategy, the proposed approach allows users to define various conditions on rule bodies at any time. Our contributions are summarized as follows.

(1) We define a new flexible rule model to recommend stock investment types.

(2) We propose a method to discover frequent stock patterns, each of which corresponds to a rule head.

(3) We propose a method that constructs a frequent pattern base by using the frequent patterns thus discovered and builds its index for efficient rule matching.

(4) We propose a method for efficient rule matching in the frequent pattern base, thereby recommending investment types to users through the matching result.

(5) To show the effectiveness and the efficiency of the proposed approach, we conduct performance evaluation via extensive experiments.

2. Overview

2.1. Motivating example

Figure 1 shows the changing patterns of ten sequences of stock prices, $Stock_i$ $(0 \leq i < 10)$. The vertical axis represents a stock price, and the horizontal axis does a relative time when each pattern occurs. $RH\text{-}Start_i$ and $RH\text{-}End_i$ mean the starting and ending points of the time range named by a rule head. Similarly, $RB\text{-}Start_i$ and $RB\text{-}End_i$ mean the starting and ending points of the time range named by a rule body. In this figure, we see that all the sequences show exactly the same pattern in the time range of the rule head.

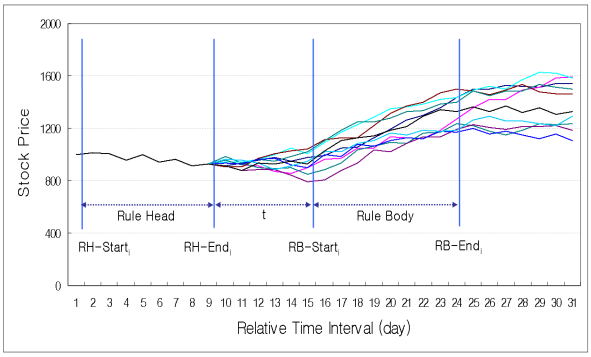

Figure 1. Changing patterns of stock prices.

Let us examine the price changing patterns of ten stock sequences in the time range of the rule body, which follows the rule head after the time interval of t. In every sequence, we observe that the average stock price in the rule body gets more than 20% high in comparison with the last stock price in the rule head. By this observation, we can expect that, when a new sequence shows the same pattern as those in the rule head, it would increase more than 20% after t time interval since then. Suppose that an investor knows this tendency in advance. If his/her stock of interest shows the same pattern as those in the rule head, the investor expects its price will grow significantly after time interval t. Thus,

this investor buys that stock, thereby having a chance to get a high return.

2.2. Rule model

In this paper, we use the following form of a rule to express the trend of changing stock prices. Here, H and B denote a rule head and a rule body, respectively. This rule represents that B happens after time t since H has occurred.

$$H \xrightarrow{t} B(s, c)$$

In stock applications, H is an event corresponding to an appearance of a pattern P within the time range of the rule head as shown in Figure 1. Also, B is an event that corresponds to the characteristics of stock prices within the time range of the rule body as shown in Figure 1. In Figure 1, for example, B can be represented as 'INCREASE' because the average prices in the rule body get 20% high. Like this, investors specify their own conditions, which are related to the investment types for recommendations, on the characteristics of stock prices within the time range of the rule body. Such conditions are called *rule body conditions*, which decide when the characteristics of stock prices are regarded as 'INCREASE'. In the prior example, investors set the rule body condition as "the average stock price in the time range of the rule body increases more than 20% in comparison with the last stock price in the time range of the rule head". In this case, the changing patterns of stock prices in Figure 1 are discovered as a meaningful rule. We note that these rule body conditions vary depending on dispositions of investors.

Next, we discuss (s, c) in the rule. A changing pattern can be formed as a rule only when a sufficient number of stock sequences support the pattern. s defined in the following is called a $support$ which means how many times the pattern P corresponding to H appeared in past stock sequences.

$$s = support(H) = \frac{N_{P,H}}{N_{LP,H}} \times 100$$

, where $N_{P,H}$ is the number of occurrences of a pattern P corresponding to H and $N_{LP,H}$ is the number of occurrences of patterns whose lengths are same as that of the pattern P corresponding to H. Also, for being formed as a rule, a set of sequences that satisfy the above support should show a similar tendency in the time range of the rule body. c defined below is a $confidence$, which represents how many stock sequences matched to H satisfy the condition on B together.

$$c = confidence(H, B) = \frac{N_{H,B}}{N_H} \times 100$$

, where $N_{H,B}$ is the number of occurrences of patterns that are matched to H and also satisfied with the condition on B, and N_H is the number of occurrences of patterns that are matched to H. Our approach discovers those rules whose support and confidence are both larger than predetermined thresholds during analyzing the past stock sequences. If a recent changing pattern of an investor's stock of interest is matched to some H, it recommends an investment type by referring to its B. Possible investment types to be recommended are 'BUY', 'SELL', 'HOLD', and 'NO RECOMMENDATION'.

3. Construction of a Frequent Pattern Base

3.1. Discovery of frequent patterns

A patten is called a *frequent pattern* when it has a support larger than a predefined threshold called a *minimum support*. As explained in Section 2.2, only the patterns which are frequent can be used as a rule head in the proposed rule model. Now, we discuss the steps to discover frequent patterns from each sequence stored in a stock database.

Preprocessing step : It is rarely possible for a stock sequence with raw element values to contain frequent patterns. To solve this problem, we take the approach to transform a raw stock sequence into a sequence of *change ratios* and then into a symbol sequence. More specifically, each element $s[i]$ $(0 \leq i < n)$ of a raw stock sequence S is first transformed into $s'[i]$ $(0 \leq i < n-1)$ by the expression shown below:

$$s'[i] \ = \ (\frac{s[i+1] - s[i]}{s[i]}) \times 100$$

A sequence of change ratios S' is then transformed into a symbol sequence S'' via *categorization*. Categorization is an operation which divides a value range of elements into a set of non-overlapping categories. Via categorization, each element of S' is converted to a symbol of the category to which the element belongs. Note that two elements whose values are different from each other may be represented by the same symbol. As a result, the probability for a stock sequence to contain frequent patterns becomes higher. In case of the Korean stock market, the daily price change limit is $\pm$15 percent of the previous day's closing price, and the change ratios between adjacent elements are distributed quite nonuniformly over the range, much more around 0% and very few near -15% or +15%. Therefore, we employ the equi-depth categorization method[12] where each category contains almost same number of elements.

Algorithm to discover frequent patterns : Frequent patterns embedded in a given symbol sequence S'' can be

easily discovered by extracting every subsequence α from S'', computing α's support Sup_α, and comparing Sup_α with a predefined minimum support MinSup. Although this method is simple to implement, it incurs large CPU and disk I/O overheads. To overcome this problem, references [3, 4, 5] proposed the *Apriori* algorithm where the concept of candidate patterns was exploited. A pattern is called a *candidate pattern* when it is not judged as infrequent. To discover frequent patterns from a symbol sequence S'', we utilize the Apriori algorithm as follows. We first discover a set of frequent patterns F_1 of length 1 by scanning S''. We then perform the *self-join* of F_{k-1} ($2 \leq k \leq Len(S'')$) to produce a set of candidate patterns C_k of length k. For each pattern α in C_k, we scan S'' to decide whether the support of α is greater than or equal to a predefined minimum support MinSup. If α's support is greater than or equal to MinSup, then α is inserted into F_k. We repeat the above process with incrementing k until no candidates in C_k are judged as frequent and thus there are no elements in F_k.

Construction of frequent pattern bases : Frequent patterns discovered in the previous step are stored in a *frequent pattern base*. A frequent pattern base must be organized to support an efficient rule matching. For achieving this goal, we construct a frequent pattern base in a B-tree structure. The entries to be stored in a B-tree are pairs of ⟨frequent pattern, list of positions at which the frequent pattern occurs⟩. Each occurrence position of a frequent pattern α is denoted as a pair of ⟨SID, offset⟩ where SID is the identifier of a stock sequence within which α occurs and offset is the starting point of the stock sequence from which α appears. We could construct only a single pattern base for all frequent patterns from a stock sequence. For more efficient rule matching, however, we classify frequent patterns according to their length and then build a separate index for each class. The physical locations of all B-trees are stored in an index table within main memory.

4. Rule Matching and Recommendation

4.1. Basic idea

Our method mainly consists of two steps. The first step is for rule matching where we search a frequent pattern base for the rule heads matched to a condition given by a stock investor. The second step is for recommending the most promising an investment type after analyzing the rule bodies following the matched rule heads. Queries for requesting recommendation types are specified by the following expression through which a stock investor indicates the stock item of interest and the conditions determining his/her stock trading activities.

Q = (item, QP, t, bodyLen, [minHold, maxHold], minConfidence)

In the above expression, item is the stock item of interest and QP (= $\langle q[0], q[1], ..., q[Len(QP)-1]\rangle$) is a query pattern representing the most recent changing pattern of item. As shown in Figure 1, t denotes the time interval between the end of rule heads and the beginning of rule bodies, and bodyLen denotes the length of rule bodies. For determining investment types, we inspect each of the rule bodies of length bodyLen beginning after t time units from the end of the rule heads matched to QP. [minHold, maxHold] denotes the range of average increase ratio for retaining the current stock item. Given a rule head H and a rule body B, their AIR(*average increase ratio*) is defined by the following expression:

$$AIR(H, B) = \frac{\text{average price of } B \ - \ \text{last price of } H}{\text{last price of } H} \times 100$$

One of such investment types as 'BUY', 'SELL', 'HOLD', and 'NO RECOMMENDATION' is chosen in response to the value of average increase ratio. That is, we recommend 'BUY' when the value of average increase ratio is larger than maxHold, 'HOLD' when it is between minHold and maxHold, and 'SELL' when it is smaller than minHold.

For a rule to be meaningful, the changing patterns of its bodies must be similar. The *confidence* of a rule represents how similar the changing patterns of its bodies are. If the confidence of a rule is less than minConfidence specified in a user query, then the rule is not considered meaningful. 'NO RECOMMENDATION' is delivered to a user in such a case. minConfidence must be set to at least 50% to prevent more than one investment type from being recommended.

Note that the query model explained above is just an illustration. That is, the above query model is for helping readers to understand the basic concept of the proposed method easily. This query model can be adapted in accordance with specific needs of users. For example, according to the dispositions of users, different values can be assigned to query parameters such as t, bodyLen, minHold, maxHold, and minConfidence. Moreover, a new metric rather than the average increase ratio can be employed as a basis for determining the investment type. For example, conservative investors may want to use the MIR(*minimum increase ratio*) rather than the average increase ratio.

$$MIR(H, B) = \frac{\text{lowest price of } B \ - \ \text{last price of } H}{\text{last price of } H} \times 100$$

Since the proposed method enables stock investors to easily adapt the query model to suit their needs or application environments, it provides a fundamental framework for adaptive recommendation systems for stock investment.

4.2. Procedure for rule matching and recommendation

The overall procedure of rule matching and investment recommendation consists of seven steps shown below.

Step 1. Symbolization of a query pattern : To symbolize a query pattern QP (= $\langle q[0], q[1], \ldots, q[Len(QP)-1]\rangle$), we use the same method as the one explained in Section 3.1. That is, we first transform a query pattern QP into a sequence of change ratio QP' (= $\langle q'[0], q'[1], \ldots, q'[Len(QP)-2]\rangle$) and then transform QP' into a symbol sequence QP" (= $\langle q''[0], q''[1], \ldots, q''[Len(QP)-2]\rangle$) via categorization.

Step 2. Searching for matched rule head : We search a frequent pattern base for the rule head RH matched to QP". For this step, we first look up the index table to find out the location of the B-tree corresponding to the length of QP". We then search this B-tree for the leaf node matched to QP". The leaf node has a pointer to the list of the occurrence positions $\langle$SID, offset$\rangle$ of the patterns matched to QP". Here, SID is the identifier of a stock sequence within which a pattern matched to QP" occurs and offset is the offset from the beginning of the stock sequence identified by SID.

Step 3. Retrieval of actual element values of rule heads and bodies : Using the list of the occurrence positions obtained in Step 2, we retrieve from a stock database the actual element values of the corresponding rule head RH and body RB. If there are n instances of the rule head matched to QP", n pairs of $\langle RH_i, RB_i\rangle$ $(0 \le i < n)$ are retrieved in this step.

Step 4. Calculation of average increase ratio : Suppose that a user wants to employ the average increase ratio as a basis for recommending investment types. Then, for each pair of $\langle RH_i, RB_i\rangle$ $(0 \le i < n)$, we calculate its average increase ratio by comparing the end price of RH_i to the average price of RB_i.

Step 5. Calculation of confidence : For each pair of $\langle RH_i, RB_i\rangle$, we first compare its average increase ratio with [minHold, maxHold] to choose one from the three possible change patterns, 'INCREASE', 'DECREASE', and 'UNCHANGED'. More specifically, we choose 'DECREASE' when the average increase ratio is less than minHold, 'INCREASE' when it is larger than maxHold, and 'UNCHANGED' when it is between minHold and maxHold. We then compute how many pairs of $\langle RH_i, RB_i\rangle$ support 'DECREASE', 'INCREASE', and 'UNCHANGED', respectively. We finally compute the confidences of 'DECREASE', 'INCREASE', and 'UNCHANGED'. The confidence of 'DECREASE' is defined as the percentage of the pairs of $\langle RH_i, RB_i\rangle$ supporting 'DECREASE' over all pairs of $\langle RH_i, RB_i\rangle$. The confidences of 'INCREASE' and 'UNCHANGED' are defined analogously.

Step 6. Rule generation : We choose from 'INCREASE', 'DECREASE', and 'UNCHANGED' the one whose confidence is larger than or equal to minConfidence. We then generate a rule with the chosen one as a rule body. Again, note that minConfidence is required to be at least 50% to avoid more than one investment type from being recommended. We do not generate a rule when all three confidences are less than minConfidence. Such a case indicates that the patterns after t time interval from the rule head do not show consistent changing patterns.

Step 7. Recommendation of an investment type : We recommend an investment type in accordance with the rule generated in Step 6. That is, we recommend 'BUY' when the rule supports 'INCREASE', 'SELL' when it supports 'DECREASE', and 'HOLD' when it supports 'UNCHANGED'. We do not recommend anything when no rules are generated in Step 6. In addition to an investment type, we provide the support and confidence values of the underlying rule in order to show its reliability and usefulness.

5. Performance Evaluation

For performance evaluation, we performed a series of experiments on real-life stock data. We selected 10 blue-chip stock items included in the Korean Composite Stock Price Index(KOSPI)[1]. For each stock item, we collected a sequence of more than 5,500 real numbers that represent the daily closing stock prices for 20 years from 28 May 1985 to 27 May 2005. In experiments, we call this data set $KOSPI\text{-}Data$. Each raw sequence was converted into a symbol sequence by categorization as described in Section 3.1. The hardware platform was the Pentium IV 2.6 GHz PC equipped with 512 MB main memory and 80 GB hard disk of 7,200 RPM. The software platform was Redhat Linux version 2.4.

In Experiment 1, we evaluated the proposed rule discovery and matching model. As a performance measure, we used the satisfaction ratio and the recommendation ratio. The *satisfaction ratio* indicates how much fraction of recommendations obtained by processing queries would satisfy stock investors, and is defined in the following.

$$satisfaction\ ratio = \frac{N_{r,s}}{N_r}$$

,where $N_{r,s}$ is the number of recommendations satisfying

the stock investors and N_r is the total number of recommendations obtained from processing queries.

The *recommendation ratio* indicates how much fraction of users' queries produce meaningful recommendations as their results, and is defined as follows.

$$recommendation\ ratio = \frac{N_{q,m}}{N_q}$$

, where $N_{q,m}$ is the number of queries that produce meaningful recommendations and N_q is the total number of queries issued by investors.

70% of $KOSPI\text{-}Data$ was used for constructing a frequent pattern base, and 30% of $KOSPI\text{-}Data$ was used for forming query sequences. The minimum support for a frequent pattern base was set to 2%. As meaningful investment types, 'SELL' and 'BUY' were used. Also, the basic values of parameters in all the following experiments were set as follows: the number of symbols of 3, the minimum support of 2%, the range of the average increase ratio of [-2%, 2%], the time interval between the rule head and body of 0, the rule body length of 1, the minimum confidence of 60%, and the satisfaction level of 60%.

Figure 2 shows the tendency of satisfaction ratio and recommendation ratio with different average increase ratios in the rule body. We observe that satisfaction ratio is nearly constant around 80% even though the range of the average increase ratio changes. The recommendation ratio, however, decreases abruptly as the average increase ratio grows. In particular, when the range of the increase ratio is [-3%,3%], the recommendation ratio was shown to be only 3.3%.

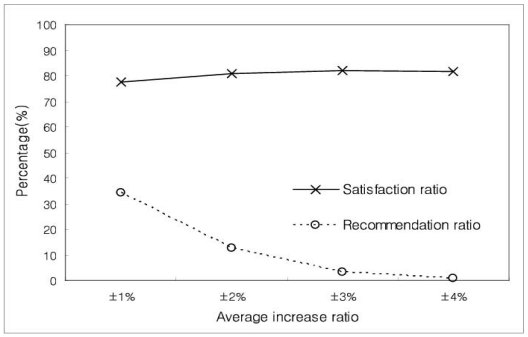

Figure 2. Satisfaction ratio and recommendation ratio with different ranges of average increase ratio.

High recommendation ratio is not always beneficial to investors. However, low recommendation close to 0 implies that the system is not that useful since it hardly recommends investment types. In this work, we aim at developing a stock investment system that provides reasonable recommendation ratio and high satisfaction ratio. From the results, we observe that our system recommends meaningful investment types for 13% of queries, and 80% of recommendations are satisfied by investors when the range of the increase ratio was set to [-2%,2%].

Figure 3 shows the tendency of satisfaction ratio and recommendation ratio with different time intervals t between the rule head and rule body. As the time interval increases, both satisfaction ratio and recommendation ratio decrease. This result can be easily explained from the fact that if the time interval between the rule head and rule body increases, then the association between them diminishes. The results show that the best satisfaction ratio and recommendation ratio are obtained when t is 0.

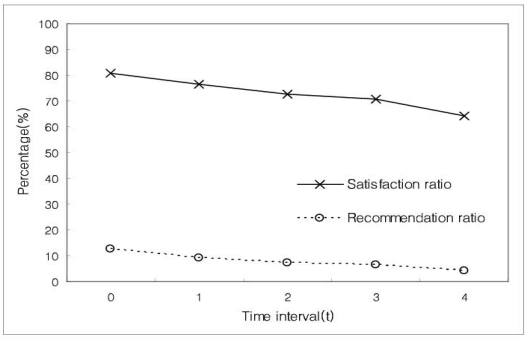

Figure 3. Satisfaction ratio and recommendation ratio with different time intervals t.

In Experiment 2, we analyzed the time for creating a frequent pattern base and the time for query processing in order to evaluate the efficiency of the proposed approach by using $KOSPI\text{-}Data$. To construct a frequent pattern base, we used the minimum support of 2%. Basic settings for parameters are as follows: the number of symbols of 3, the minimum support of 2%, the range of average increase ratio of 2%, the time interval t of 0, the length of a rule body of 1, and the minimum confidence of 60%.

Figure 4 shows the tendency of the time for constructing a frequent pattern base and the time for query processing with a changing data size. To generate large data sets, we duplicated $KOSPI\text{-}Data$ 2, 3, and 4 times. In a horizontal axis, $1S$ denotes pure $KOSPI\text{-}Data$, and $2S$, $3S$, and $4S$ do its duplicated versions. Owing to this duplication, satisfaction ratio is entirely preserved when the same query of the minimum support and the minimum confidence is performed with different sizes of data sets. The result shows that the time for building a frequent pattern base and the time for query processing both increase in proportion to a data size. This is because the occurrences of a frequent pat-

tern double when a data size gets twice as large as before. Consequently, the time for constructing a frequent pattern base and the time for query processing get doubled.

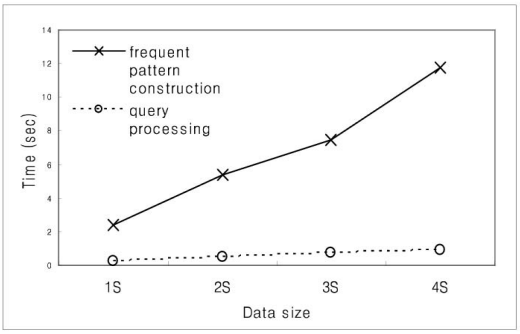

Figure 4. Time for constructing a frequent pattern base and time for query processing with different data sizes.

6. Conclusions

In this paper, we aimed at devising an efficient and flexible approach that recommends appropriate investment types to stock investors by discovering useful rules from past changing patterns of stock prices stored in a database. To achieve our goal, we first proposed a new rule model for recommending stock investment types. For a frequent pattern of stock prices, if its subsequent stock prices are matched to a condition of an investor, the model recommends a corresponding investment type for this stock.

For efficient discovery and matching of rules, we proposed methods for discovering frequent patterns and organizing them into a frequent pattern base. We also suggested a method that searches a frequent pattern base for the rules matched to the conditions given by an investor, and a method that analyzes the matched rules to recommend an investment type. We verified the superiority of the proposed approach by performing various experiments through which we measured satisfaction ratios and response times of rule matching.

Acknowledgments: This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2007-314-D00221) and also partially supported by the MKE(Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement) (IITA-2008-C1090-0801-0040).

References

[1] Koscom data mall. *http://datamall.koscom.co.kr*, 2005.

[2] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. *In Proc. Int'l. Conf. on Foundations of Data Organization and Algorithms(FODO)*, pages 69–84, 8 1993.

[3] R. Agrawal, K. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *In Proc. Int'l. Conf. on Very Large Data Bases, VLDB*, pages 490–501, 10 1995.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. *In Proc. Int'l. Conf. on Very Large Data Bases, VLDB*, pages 487–499, 1994.

[5] R. Agrawal and R. Srikant. Mining sequential patterns. *In Proc. Int'l. Conf. on Data Engineering*, pages 3–14, 1995.

[6] T. Anderson. *The Statistical Analysis of Time Series*. Wiley, 1971.

[7] P. Bloomfield. *Fourier Analysis of Time Series*. Wiley, 2000.

[8] W. W. Chu and K. Chiang. Abstraction of high level concepts from numerical values in databases. *In Proc. AAAI Workshop on Knowledge Discovery in Database*, pages 133–144, 1994.

[9] G. Das, K.-I. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule discovery from time series. *In Proc. Int'l. Conf. on Knowledge Discovery and DataMining*, pages 16–22, 1998.

[10] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. *In Proc. Int'l. Conf. on Management of Data, ACM SIGMOD*, 1:419–429, 5 1994.

[11] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. *Information Systems*, 26(1):35–58, 2001.

[12] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[13] S. W. Kim, S. H. Park, and W. W. Chu. An index-based approach for similarity search supporting time warping in large sequence databases. *In Proc. Int'l. Conf. on Data Engineering, IEEE ICD*, pages 607–614, 2001.

[14] W. K. Loh, S. W. Kim, and K. Y. Whang. A subsequence matching algorithm that supports normalization transform in time-series databases. *Data Mining and Knowledge Discovery Journal*, 9(1):5–28, 7 2004.

[15] S. Park and W. W. Chu. Discovering and matching elastic rules from sequence databases. *Fundamenta Informaticae*, 47(1-2):75–90, 8-9 2001.

[16] S. H. Park, W. W. Chu, J. Yoon, and C. Hsu. Efficient searches for similar subsequences of difference lengths in sequence databases. *In Proc. Int'l. Conf. on Data Engineering, IEEE ICDE*, pages 23–32, 2000.

[17] E. Saad, D. Prokhorov, and D. W. II. Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Trans. on Neural Networks*, pages 1456–1470, 1998.

[18] B. Wah and M. Qian. Constrained formulations and algorithms for stock-price predictions using recurrent fir neural networks. *AAAI/IAAI*, pages 211–216, 2002.