An Efficient Stream Mining Technique

Hatim A. Aboalsamh
Hatim@ccis.ksu.edu.sa
Department of Computer Sciences

Alaaeldin M. Hafez
ahafez@ccis.ksu.edu.sa
Department of Information Systems

Ghazy M. R. Assassa
ghazy@ccis.ksu.edu.sa
Department of Computer Sciences

College of Computer and Information Sciences
King Saud Uni., SAUDI ARABIA

Abstract: Stream analysis is considered as a crucial component of strategic control over a broad variety of disciplines in business, science and engineering. Stream data is a sequence of observations collected over intervals of time. Each data stream describes a phenomenon. Analysis on Stream data includes discovering trends (or patterns) in a Stream sequence. In the last few years, data mining has emerged and been recognized as a new technology for data analysis. Data Mining is the process of discovering potentially valuable patterns, associations, trends, sequences and dependencies in data. Data mining techniques can discover information that many traditional business analysis and statistical techniques fail to deliver. In our study, we emphasis on the use of data mining techniques on data streams, where mining techniques and tools are used in an attempt to recognize, anticipate and learn the stream behavior with different directly related or looked unrelated factors. Targeted data are sequences of observations collected over intervals of time. Each sequence describes a phenomenon or a factor. Such factors could have either a direct or indirect impact on the stream data under study. Examples of factors with direct impact include the yearly budgets and expenditures, taxations, local stocks prices, unemployment rates, inflation rates, fallen angels, and rising odds for upgrades. Indirect factors could include any phenomena in the local or global environments, such as, global stocks prices, education expenditures, weather conditions, employment strategies, and medical services. Analysis on data includes discovering trends (or patterns) and association between sequences in order to generate non-trivial knowledge. In this paper, we propose a data mining technique to predict the dependency between factors that affect performance. The proposed technique consists of three phases: (a) for each data sequence that represents a chosen phenomenon, generate its trend sequences, (b) discover maximal frequent trend patterns, generate pattern vectors (to keep information of frequent trend patterns), use trend pattern vectors to predict future factor sequences.

Keywords: Data Mining, Stream Mining, Time Series Mining, Mining Trends, Data Sequences, Association Mining, Maximal Trend Patterns, Global Trends, Local Trends.

1. Introduction

Stream data is a sequence of observations collected over some intervals. For clarity reasons, in our examples, we use time values for demonstrating interval values. Each stream of data describes a phenomenon. For example, daily stock prices could be used to describe the fluctuations in the stock market. In general, for a data stream X with n observations, X is represented as

$$X = (v_1, t_1),\ (v_2, t_2),\ \ldots,\ (v_n, t_n)$$

where v_i and t_i, $1 \le i \le n$, are the observation value and its time stamp, respectively. A data sream [7, 11, 13, 14] can be either regular or irregular. In a regular Stream, data are collected periodically at defined points, while in irregular data streams, data arrive at nondeterministic points. Irregular data stream can have long periods without any data, and short periods with bursts of data.

Analysis on stream data includes discovering trends (or patterns) in a stream sequence [7, 13]. Stream patterns are classified as either systematic patterns, where patterns can be determined at specific points of time, or non-systematic patterns. Many researchers have been working on different algorithms on stream analysis. Two main lines of research have been considered to achieve this goal,

(a) Identifying or describing the pattern of observed stream data. Once the pattern is established, we can interpret and integrate it with other data. The identified pattern can be extrapolated to predict future events [11, 13].
(b) Determining whether two or more given data streams display similar behavior or not. Similar Streams can be clustered or classified into similar groups. Because of the inherent high dimensionality of the data, the problem of similarity search in large stream databases is a non-trivial problem [7, 14].

Most solutions [7, 10, 15] perform dimensionality reduction on the data, then indexing the reduced data with a spatial access method. Singular Value Decomposition (SVD), Discrete Fourier transform (DFT), Discrete Wavelets Transform (DWT), and Piecewise Aggregate Approximation (PAA) are used to reduce dimensionality.

Discovering patterns in a data stream would tell us what patterns are mostly likely to happen, but in some cases, as in non-systematic patterns, it would not tell us when those patterns would occur. Predicting non-systematic patterns needs applying some techniques to discover similarity between two or more stream sequences. Although studying similarity between patterns is an important subject, but it only covers a narrow class of stream applications. To widen the range of applications that could benefit from stream analysis, a new look on the term “similarity” should be considered. Actually, using the word “similarity” does not reflect the true meaning of a synchronized or a dependent behavior of two stream sequences. Two sequences could be completely different (in values, shapes, …, etc.), but they still react to the same conditions in a dependent way.

In the last few years, data mining has emerged and been recognized as a new technology for data analysis. Data Mining is the process of discovering potentially valuable patterns, associations, trends, sequences and dependencies in data [1- 6, 9, 12, 17-22]. Key business examples include analysis for improvements in e-commerce environment, fraud detection, screening and investigation, and product analysis. Data mining techniques can discover information that many traditional business analysis and statistical techniques fail to deliver. In data mining, advanced capabilities that give the user the power to ask more sophisticated and pertinent questions are provided.

Past trend patterns in a given data stream can be used to predict future sequences in the same data stream, we will call this a local prediction, and the trend patterns are called local patterns. Predicting future sequences using only local patterns could work but for only specific types of stream sequences, e.g., regular or cyclic. For most real life stream data, predicting sequences using only local trend patterns would tell us what sequences are mostly likely to happen, but it would not tell us when they would occur. In this paper, we introduce the notion of global prediction. In global prediction, we do not use only past local patterns but we also use past patterns in other "dependent" time sequences, those patterns are called global patterns. For each trend pattern, a pattern vector is generated to hold information about those sequences having that pattern.

In this paper, we propose a generic technique that could be used in predicting stream trends. The proposed technique consists of three phases:

(a) Generate trend sequences for all stream sequences under consideration.
(b) Discover frequent sequential patterns and generate maximal frequent trend pattern vectors.
(c) Predict future stream sequences by relating the maximal frequent trend pattern vectors.

In section 2, a formal definition of the problem is given. The approach is introduced in section 3. In section 4, the approach is evaluated. The paper is summarized and concluded in section 5.

2 Problem Definition

Let X be a sequence of n time-stamped observation values, where for each time instant t_i, , $1 \leq i \leq n$, v_i is a value collected at that instant. The data stream X is represented as

$$X = (v_1, t_1), (v_2, t_2), ..., (v_n, t_n)$$

In stream data, statisticians are often challenged with efficient ways of presenting data. Data often exhibit seasonal variation and trends that are sometime difficult to detect by observing plots of the raw data. "Noise" is defined as the variation around the trend in stream data. The amount of "noise" in the measurements often masks these trends. Therefore presentations should clearly indicate the trends and be useful for establishing the sources of variation. As an example in this paper, we use the two streams shown in fig. 1 and fig. 2; the yearly prices of Cotton (fig. 1) and the yearly prices of S&P stocks (fig. 2).

1900	23.11	1914	33.06	1928	31.07	1942	41.84	1956	56.64	1970	53.27	1984	75.21
1901	28.64	1915	30.22	1929	32.02	1943	41.25	1957	57.8	1971	53.56	1985	71.4
1902	24.94	1916	36.33	1930	31.55	1944	41.15	1958	56.87	1972	53.14	1986	72.17
1903	26.9	1917	39.72	1931	26.92	1945	40.99	1959	56.13	1973	54.19	1987	75.52
1904	25.46	1918	41.31	1932	30.33	1946	41.7	1960	55.76	1974	62.57	1988	70.28
1905	21.15	1919	38.84	1933	28.38	1947	44.01	1961	55.66	1975	63.09	1989	72.87
1906	23.13	1920	34.84	1934	32.11	1948	47.51	1962	55.54	1976	63.31	1990	77.63
1907	22.48	1921	35.67	1935	31.46	1949	47.27	1963	55.3	1977	62.91	1991	72.95
1908	22.95	1922	35.57	1936	32.75	1950	45.9	1964	55.05	1978	62.36	1992	71.35
1909	22.59	1923	32.73	1937	33.32	1951	49.79	1965	54.69	1979	73.06	1993	68.38
1910	20.58	1924	33.56	1938	33	1952	49.45	1966	54.39	1980	74.67	1994	59.15
1911	20.58	1925	35.54	1939	31.9	1953	50.24	1967	54.18	1981	74.68	1995	62.27
1912	22.39	1926	37.22	1940	31.79	1954	51.71	1968	53.7	1982	76.9	1996	63.4
1913	25.53	1927	32.14	1941	32.55	1955	51.77	1969	52.66	1983	69.79		

Fig. 1 Cotton Prices (in Dollars) in the Period 1900-1996

1900	1.20	1914	1.20	1928	1.50	1942	0.53	1956	1.23	1970	1.28	1984	0.73
1901	1.55	1915	1.05	1929	1.52	1943	0.60	1957	1.03	1971	1.35	1985	0.83
1902	1.65	1916	1.18	1930	1.10	1944	0.65	1958	1.27	1972	1.52	1986	0.99
1903	1.61	1917	1.12	1931	0.69	1945	0.81	1959	1.37	1973	1.26	1987	0.98
1904	1.23	1918	0.74	1932	0.55	1946	0.65	1960	1.27	1974	0.76	1988	1.00
1905	1.62	1919	0.68	1933	0.79	1947	0.57	1961	1.57	1975	0.88	1989	1.18
1906	1.78	1920	0.62	1934	0.70	1948	0.53	1962	1.29	1976	0.94	1990	1.01
1907	1.56	1921	0.46	1935	0.94	1949	0.55	1963	1.55	1977	0.80	1991	1.17
1908	1.19	1922	0.54	1936	1.23	1950	0.66	1964	1.74	1978	0.75	1992	1.22
1909	1.55	1923	0.68	1937	0.79	1951	0.70	1965	1.82	1979	0.72	1993	1.31
1910	1.63	1924	0.63	1938	0.89	1952	0.73	1966	1.51	1980	0.79	1994	1.21
1911	1.51	1925	0.83	1939	0.87	1953	0.69	1967	1.71	1981	0.65	1995	1.47
1912	1.41	1926	0.88	1940	0.72	1954	0.93	1968	1.80	1982	0.67	1996	1.82
1913	1.33	1927	1.14	1941	0.58	1955	1.21	1969	1.51	1983	0.78		

Fig. 2 S&P500 stock prices (in Dollars) in the Period 1900-1996

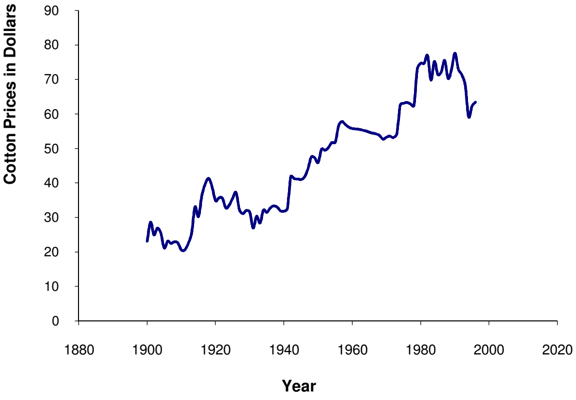
2.1 Moving Average

The moving average is a simple mathematical technique used primarily to eliminate aberrations and reveal the real trend in a collection of data points. A moving average is an average of data for a certain number of time periods that could be used as an indicator for the average value of an observation over a period of time. When a moving average is calculated [7, 10, 13, 14], a mathematical analysis over a predetermined time period is made. As the phenomena value changes, its average value moves up or down. There are five popular types of moving averages: simple (also referred to as arithmetic), exponential, triangular, variable, and weighted. The only significant difference between the various types of moving averages is the weight assigned to the most recent data. Simple moving averages apply equal weight to the values. Exponential and weighted averages apply more weight to recent values. Triangular averages apply more weight to values in the middle of the time period. And variable moving averages change the weighting based on the volatility of values. For data stream *X*, and for a time period *p*, *p-moving average* of value v_i is

$$v'_i = \sum_{j=\left\lceil \frac{p}{2} \right\rceil}^{j=-\left\lfloor \frac{p}{2} \right\rfloor} v_{i-j} / p \ ,$$

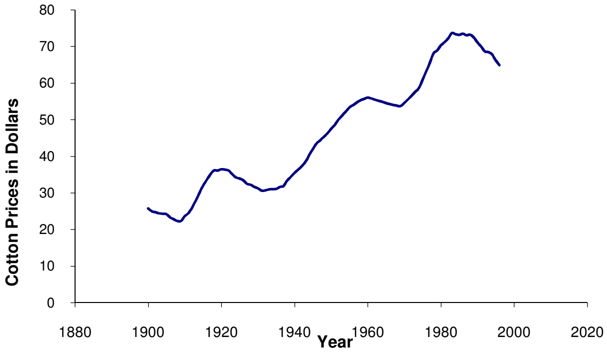
$$\textit{where} \quad 1 + \left\lceil \frac{p}{2} \right\rceil \le i \le n - \left\lfloor \frac{p}{2} \right\rfloor$$

In fig. 3 and fig. 4, we show the effect of moving average calculations on the shape and smoothness on the Cotton prices data and the S&P500 stock prices data.



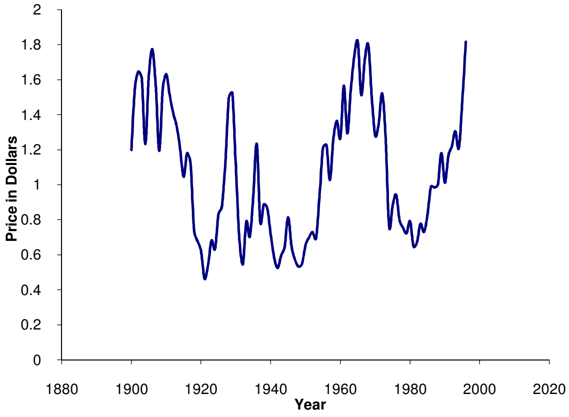
(a) Cotton prices in The Period 1900-1996

Fig. 3 (Cont.) Cotton Prices

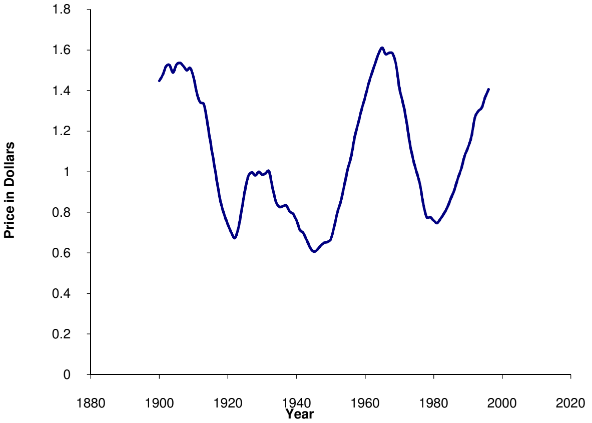


(b) 9-year Moving Average of Cotton Prices

Fig. 3 (Cont.) Cotton Prices



(a) S&P500 Stock Prices in The Period 1900-1996



(b) 9-Year Moving Average of S&P500

Fig. 4 S&P500 Stock Prices

2.2 Trend Calculation

Trends are the identification of value movements on individual stream sequences. The trend of a data stream is not linear and certainly not obvious. Trends are calculated by relating the current value of a phenomenon to its previous value. Several techniques have been proposed to identify trends [7, 10]. It is not essential that every technique have a separate trend filter. Some methods, such as moving averages, incorporate a trend indicator into the entry technique. Others try to predict an imminent change in trend and are therefore entering when the trend is against them at the time.

A simple approach to identify trends is to find the total directional movement over the time period you selected for measurement. If it is positive, the trend is up, if negative, the trend is down, and if 0, the trend is neutral. More sophisticated approaches could use mathematical equations and methods of massaging past values to more precisely determine whether the trend is up or down.
Since our main goal is to focus on the overall technique of determining dependencies between streams, comparing the different approaches of calculating trend indicators is out of the scope of this paper, we do not favor any specific approach for identifying trend indicators. In this paper, we adopt the simple approach of identifying trend indicators. The approach is modified to make the prediction more or less accurate. A *trend scale* is used to determine the size of the space used to choose the values of the trend indicators. For example, if the trend scale equals to 4, the trend indicator space is *{2, 1, 0, -1, -2}*, where *2*, *1*, *0*, *-1*, and *–2* mean *very high*, *high*, *neutral*, *low*, and *very low*, respectively. In fig. 5 we show the trend indicators of the Cotton prices data and the S&P500 stock prices data.

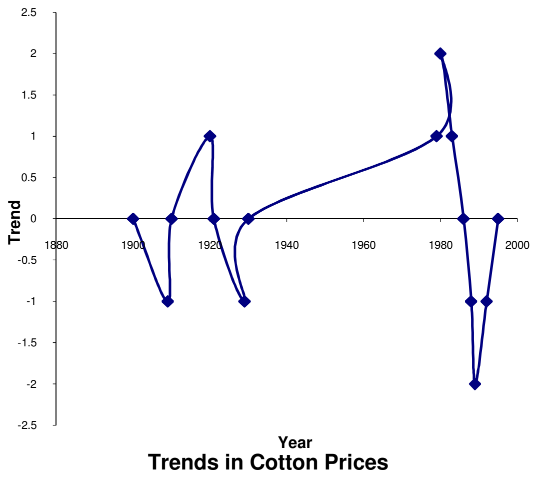
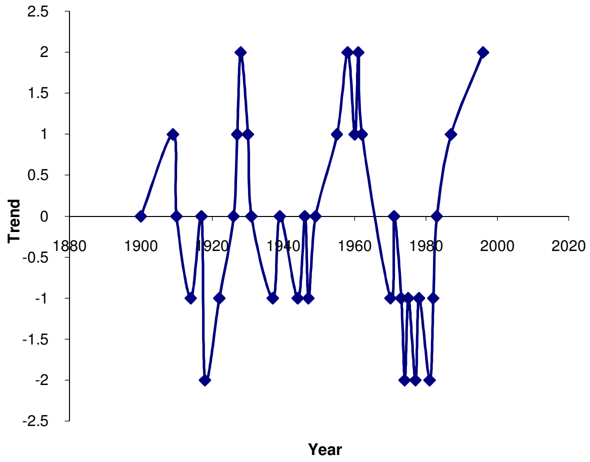


Fig. 5 Trend Indicators



(b) Trends in S&P500 Stock Prices

Fig. 5(cont.) Trend Indicators

2.3 Association Mining

Association mining [2, 3, 16, 18, 22-24] that discovers dependencies among values of an attribute was introduced by Agrawal et al.[2] and has emerged as an important research area. The problem of association mining is originally defined as follows. Let $I = \{i_1, i_2, . . . , i_n\}$ be a set of items and $S = \{s_1, s_2, . . ., s_m\}$ be a set of transactions, where each transaction $s_i \in S$ is a set of items that is $s_i \subseteq I$. An *association rule* denoted by $X \Rightarrow Y$, $X, Y \subset I$, and $X \cap Y = \Phi$, describes the existence of a relationship between the two itemsets *X* and *Y*.

Several measures have been introduced to define the *strength* of the relationship between itemsets X and Y such as *SUPPORT*, *CONFIDENCE*, and *INTEREST* [1-6, 8, 9, 12, 16]. The definitions of these measures, from a probabilistic viewpoint, are given below.

I. $SUPPORT(X \Rightarrow Y) = P(X,Y)$, or the percentage of transactions in the database that contain both *X* and *Y*.

II. $CONFIDENCE(X \Rightarrow Y) = P(X,Y)/P(X)$, or the percentage of transactions containing *Y* in those transactions containing *X*.

III. $INTEREST(X \Rightarrow Y) = P(X,Y)/P(X)P(Y)$ represents a test of statistical independence.

Agrawal et al [3], introduced the problem of mining sequential patterns over such databases. Two algorithms, *AprioriSome* and *Apriori-Like* [3], have been presented to solve this problem, and their performances have been evaluated using synthetic data. The two algorithms have comparable performances. For more details about Apriori-Like algorithms, please refer to [3, 15].

Usually, most of the data collected in data series sequences could have some outliers. We need to have a mechanism to decide which mismatched patterns (i.e., do not agree on all trend indicators) are considered nearly matched and treated as matched patterns, and which are not. We add one modification to the Apriori-like algorithm to satisfy the nearly matched cases. A *match factor* value is needed to decide at what level of accuracy we could consider two mismatched patterns as matched patterns. In section 3, we give a definition of the *match factor* parameter.

3. The Dependency Mining Technique

In this paper, we propose the Dependency Mining technique. As we mentioned in section 2, the proposed technique consists of three phases:

1- Generate trend sequences for all streams sequences under consideration. This phase has two steps:

a. Smooth the original data stream sequences by calculating moving averages. Moving averages are used to remove any short-term fluctuations.

b. For the smoothed stream values, calculate trend indicator values. Trend indicators are used to describe the behavior of the data. There are many ways to define trend indicators, as an example, a trend indicator could describe the slope of the data curve at a specific time, a value of +2 may indicate a sharp rise, while a value of –2 may indicate a sharp fall.

2- Discover frequent sequential patterns and generate maximal trend pattern vectors. By using any association mining technique, we generate frequent trend patterns for each data stream under consideration. A *match parameter* is specified to define how much difference between two trend patterns to be considered the same. Each frequent trend pattern should be accompanied by its frequency and a trend vector of start times of all occurrences of that pattern.

3- Predict future stream sequences by relating the trend pattern vectors. A frequent trend pattern A in stream 1 is compared to frequent trend patterns B's. If the frequent trend pattern B is in stream 1, B is called, with respect to A, a local pattern. If the frequent trend pattern B is in stream 2, B is called, with respect to A, a global pattern. As a result of the trend comparison, a *dependency function* is generated. A dependency function for trend patterns A and B is a mapping function from the time difference of every two succeeding occurrences of A to the corresponding occurrences in B. The dependency function could be either *a linear function* or *a non-linear function*. The *dependency factor* of A and B is calculated for the differences between the values generated for B by the dependency function and the real values of B. Depending on a pre-specified threshold of dependency value, A and B are considered dependent or not.

For a stream $X = (v_1, t_1), (v_2, t_2), ..., (v_n, t_n)$, let *TR(X)* be a sequence of *m* time stamped trends . For each time instant τ_i; $1 \le i \le m$, r_i represents the trend at that instant. The trend series *TR* is written as,

$$\boldsymbol{TR(X) = (r_1, \tau_1),\ (r_2, \tau_2), ..., (r_m, \tau_m)}$$

Definition: (*Support Set*)

Let $P = p_1, p_2, ..., p_k$; $1 \le k \le m$, be a non-empty sequence of trends. The *support-set* of *P* is defined as

$$Support\_Set(P) = \{i \ \mid P \ \ is \ \ true \ \ in \ \ r_i, \ r_{i+1}, \ ..., \ r_{i+k-1}, \ and \ \forall l \in Support\_Set(P), \ \mid i-l \mid \ \ge \ k\}$$

Definition: (*Support*)

Let $P = p_1, p_2, ..., p_k$; $1 \le k \le m$, be a non-empty sequence of trends. The *support* of *P* is defined as

$$\boldsymbol{Support(P) = \frac{|Support\_Set(P)|}{m}}$$

$|\boldsymbol{Support\ \_Set(P)}|$ is the cardinality of $\boldsymbol{Support\_Set(P)}$.

Definition: (*match factor*)

Let $P = p_1, p_2, ..., p_k$, and $Q = q_1, q_2, ..., q_l$, $1 \le k,l \le m$, be two non-empty sequences of trends. Trend sequence *Q* matches trend sequence *P* with *match factor f*, if

$$p_{i_1} = q_{j_1}, \ \ p_{i_2} = q_{j_2}, \ \ ..., \ \ p_{i_F} = q_{j_F} \quad where$$
$$i_1 < i_2 < ... < i_F \quad and \quad j_1 < j_2 < ... < j_F \quad and \quad f = \frac{F}{k}$$

Definition: (*Trend Vector*)

For a frequent trend sequence *Y*, *Trend Vector T* of *Y* is defined as

$$\boldsymbol{T(Y) = \begin{pmatrix} t_1 \\ t_2 \\ . \\ . \\ . \\ t_k \end{pmatrix}}$$

where t_i; $1 \le i \le k$, is the starting time of occurrence *i* of sequence *Y*.

Definition: (*Distance Vector*)

For a frequent trend sequence *Y* and a trend vector *T(Y), Distance Vector D* of *Y* is defined as

$$D(Y) = \begin{pmatrix} d_1 \\ d_2 \\ . \\ . \\ . \\ d_k \end{pmatrix}$$

where $d_i = t_i - base\ time$; $1 \le i \le k$.

The *base time* value is used to standardize the starting time of all sequences.

Definition: (*Distance Scaling Factor*)

For two frequent trend sequences *Y* and *Y'*, with distance vectors *D(Y)* and *D(Y'),* respectively, the *Distance Scaling factor* of *Y* and *Y'* is defined as

$$DS(Y,\ Y') = \frac{Max\ (|D(Y)|\ ,\ |D(Y')|)}{Min\ (|D(Y)|\ ,\ |D(Y')|)},$$

where |*D(Y)*| and |*D(Y')*| are the cardinality of *D(Y)* and *D(Y')*, respectively.

In order to compare two frequent trend sequences *Y* and *Y'*, we need to normalize them. The normalization process is to make their distance vectors *D(Y)* and *D(Y')* having the same cardinality. To do that, we map the distance vector with the larger cardinality; m*ax(*|*D(Y)*|, |*D(Y')*|*)*, to a corresponding distance vector with cardinality m*in(*|*D(Y)*|, |*D(Y')*|*)*. Let *D(Y)* and *D(Y')* be the distance vectors of sequences *Y* and *Y'*, respectively, and |*D(Y)*| ≥ |*D(Y')*|. *D(Y)* is normalized to *D'(Y)* such that each element d'_i in *D'(Y)* , $1 \le i \le |D(Y')|$, equals to element *1+(i-1)*DS(Y,Y')* in *D(Y);*

$$d'_i = d_{1+\lfloor (i-1)*DS(Y,Y') \rfloor}$$

Definition: (*Dependency Function*)

Two normalized frequent trend sequences *Y* and *Y'* are dependant if there exists a *dependency function f* such that, for the two normalized distance vectors *D(Y)* and *D(Y'),*

$$D(Y') = f(D(Y))$$

or for all elements d_i and d_i', $1 \le i \le |D(Y)|$, in *D(Y)* and *D(Y'),* respectively,

$$d'_i = f(d_i), \qquad 1 \le i \le |D(Y)|$$

Definition: (*Dependency factor*)

Two frequent sequences *D* and *D'* are dependant with dependency factor α; α-dependant , where

$$\alpha = 1 - \frac{\|D' - f(D)\|}{max(\|D'\|\ ,\ \|f(D)\|)}$$

Definition: (*Linear Dependency*)

Two frequent sequences *D* and *D'* are *linearly dependant* if there exist a dependency function *f* such that

$$D' = f(D) = a\ D + b \quad or$$
$$d'_i = a\ d_i + b, \ 1 \le i \le |D|$$

for some constants *a* and *b*.

Table 1 describes the Dependency Mining technique. The Dependency Mining technique discovers the dependency between two streams X and X'.

Dependency Mining (X, X')

1- Smooth data sequences *X* and *X'*; generate *Z* and *Z'* by calculating the moving averages of *X* and *X'*, respectively.
2- Identify trends; calculate trend indicator values *W* and *W'* of *Z* and *Z'*, respectively.
3- Generate all frequent trend patterns; discover frequent sequences in *W* and *W'*.
4- For each frequent pattern *Y*, generate its trend vector *T(Y).*
5- Calculate the distance vectors *D(Y)'s* of all frequent patterns. A user defined base time value is used to calculate the differences between the time stamp values and the base time value.

```
6-  For each pair of distance vectors D(Y) and D(Y') of two frequent patterns Y and Y', respectively, do
 a. Normalize. By calculating the distance scaling factor, we normalize that distance vector with the largest cardinality to be in the same cardinality of the other distance vector.
 b. Calculate the dependency factor $\alpha$ of the normalized D(Y) and D(Y').
7-  Choose only those frequent patterns Y and Y' with $\alpha \le \alpha min$.
```

Table 1. Outlines of the Dependency Mining Technique

4. Performance Evaluation

In section 4.1, we give the complexity of the Dependency Mining technique. The experimental results are given in subsection 4.2.

4.1. Complexity

The complexity of the Dependency Mining technique is divided into three components, each corresponds to one of the three phases of the technique. The first component gives the complexity of smoothing the data series sequence and generating the trend indicators. For two streams $X = (v_1, t_1), (v_2, t_2), ..., (v_n, t_n)$ and $X' = (v'_1, t'_1), (v'_2, t'_2), ..., (v'_{n'}, t'_{n'})$ of n and n' time-stamped observation values, the number of iterations needed to smooth and generate the trend indicators is

$$2*(n+n'); (O(n+n'))$$

The complexity of the second phase depends on the association mining algorithm used. For Apriori-Like algorithms, the complexity is

$$(l+1)*n+(l'+1)*n',$$

where $l$and l' are the maximum lengths of the discovered frequent patterns in X and X', respectively.

In the third phase, let K and K' be the number of frequent trend patterns discovered in X and X', respectively, during the association mining process, and d be the length of the distance vector used in calculations. The complexity of the third phase is

$$K*K'*d$$

4.2. Experimental Results

Two stream sequences are used in our experiments. The two streams are the Cotton prices during the period 1900-1996, and the S&P500 stock prices during the same period of time. The two streams are shown in fig 1 and fig. 2. Among the set of experiments that have been conducted on the Mining Dependency technique, and because of the space limitations, we choose to show only one class of experiments, with the following parameter values,

minsup (minimum support) = 0.2
match factor = 0.9
scale (trend indicator space) = *{2, 1, 0, -1, -2}*

On this class of experiments, we choose the minimum dependency factor = 0.8. In table 2, we give the results of our experiments. The results show a list of dependent maximal frequent trend patterns. The trend values are drawn from the trend space *{2, 1, 0, -1, -2}*. For each pair of maximal frequent trend sequences from the Cotton prices stream and the S&P500 Stock prices stream, we give their dependency factor and dependency function (i.e., $D'=f(D)=a D + b$) parameters.

	Cotton	**S&P500 Stock**	**Dependency Factor**	***a***	***B***
1	{-1, 0}	{-1, 0}	0.97333	1.5	-23.2
	1909==>1919	1914==>1917			
	1929==>1978	1944==>1946			
	1992==>1996	1982==>1986			
2	{-1, 0}	{1}	0.977465	2.3	-4.95
	1909==>1919	1909==>1909			
	1929==>1978	1955==>1957			
	1992==>1996	1987==>1995			

Table 2. The Experimental Results of The Mining Dependency Technique

	Cotton	S&P500 Stock	Dependency Factor	*a*	*B*
3	{1}	{-1, 0}	0.979411	0.508475	9.08475
	1920==>1920	1914==>1917			
	1979==>1979	1944==>1946			
	1983==>1985	1982==>1986			
4	{1}	{1}	0.938083	0.779661	15.2712
	1920==>1920	1909==>1909			
	1979==>1979	1955==>1957			
	1983==>1985	1987==>1995			
5	{0, -1}	{-1, 0}	0.90127	1.42857	-30.6667
	1900==>1909	1914==>1917			
	1921==>1929	1944==>1946			
	1986==>1988	1982==>1986			
6	{0, -1}	{1}	0.892004	2.19048	-13.2857
	1900==>1909	1909==>1909			
	1921==>1929	1955==>1957			
	1986==>1988	1987==>1995			

Table 2 (Cont.). The Experimental Results of The Mining Dependency Technique

The results show, using a linear dependency function, that there is a strong dependency between the changes in the Cotton prices and in the S&P500 stock prices. The *Dependency Mining* technique not only discovers the trivial dependencies between the two stream sequences (e.g., when the Cotton price increases, the S&P500 price decreases, and visa versa; cases 2, 3, 5 and 6), but also discovers those non-trivial dependencies (e.g., there is a dependency with high factor between the increase in the Cotton prices and the S&P500 stock prices; cases 4, and also between the decrease in the Cotton prices and the S&P500 stock prices; case 1).

5. Conclusions

In this paper, we have introduced the *Dependency Mining* technique. The proposed technique adapts and innovates data mining techniques to analyze stream data. We consider past frequent patterns of the same time sequences (*local patterns*) and of other "dependent" time sequences (*global patterns*). Instead of only studying the problem of finding similarities between stream sequences, we have widened the scope of our research by introducing the notion of dependency between stream sequences. Real life stream could be completely different (in values, shapes, …, etc.), but they could react to the same conditions in a dependent way. By using data mining techniques, maximal frequent patterns are mined and used in predicting future sequences or trends, where trends describe the behavior of a sequence. The proposed technique combines several techniques in an integrated one. We use moving average techniques and trend generation techniques as one part. The results generated in the first part are mined to discover those maximal frequent trend patterns that satisfy a minimum support value. Finally, the maximal frequent patterns are processed to discover those dependant patterns that satisfy certain dependency factor threshold.

We have discussed the complexity of the *Dependency Mining technique* in terms of number of iterations. In our preliminary experimental results, the *Dependency Mining technique* has shown a significant potential usage. We have shown the significance of the knowledge discovered by using the *Dependency Mining technique*. Our technique not only discovers those dependencies that can be noticed by human eyes (i.e., trivial), but it also discovers non-trivial dependencies that are not obvious for the human eyes (non-trivial).

As a future work, the Dependency Mining technique will be tested with different datasets that cover a large spectrum of different stream data sets, such as, medical data, financial data, and environmental data.

References:

[1] S. Agarwal, et. al., On the Computation of Multidimensional Aggregates, In Proc. 1996 Int. Conf. Very Large Databases, Bomaby, India, Sept. 1996.

[2] R. Agrawal, et. al., Mining Association Rules between Sets of Items in Large Databases, Proc. of the ACM SIGMOD Int'l Conf. On Management of data, May 1993.

[3] R. Agrawal and R. Srikant, Mining Sequential Patterns, In Proc. 11th Intl. Conf. On Data Engineering, Taipi, Taiwan, March 1995.

[4] R. Agrawal, and R. Srikant, Fast Algorithms for Mining Association Rules, Proc. Of the 20th VLDB Conference, Santiago, Chile, 1994.

[5] C. Agrawal, and P. Yu, Mining Large Itemsets for Association Rules, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 1997.

[6] D. Berndt and J. Clifford, Using Dynamic Time Wrapping to Find Patterns in Time Series, Discovery in Databases, pp. 359-370, Seattle, Washington, July 1994.

[7] P. Buono, et al., Similarity-Based Forecasting with Simultaneous Previews: A River Plot Interface for Time Series Forecasting, *Proc. 11th International Conference on Information Visualisation*. Zurich, Switzerland; 2-6 July, 2007.

[8] J. Chen, et al., *Palmprint Authentication Using* Time Series, In proceedings of the 5th International Conference on Audio- and Video-Based Biometric Person Authentication. Hilton Rye Town, NY. July 20-22, 2006.

[9] B. Chiu, et al., Probabilistic Discovery of Time Series Motifs, In the *9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. August 24 - 27, 2003. Washington, DC, USA.

[10] C. Faloutsos, et. al., Fast Sequence Matching in Time-Series, Proc. of the ACM SIGMOD Int'l Conference on Management of Data, May 1994.

[11] E. Keogh, et. al., HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence, In Proc. of the 5th IEEE International Conference on Data Mining (ICDM 2005), Houston, Texas, Nov 27-30, 2005.

[12] Ming-Chuan Hung, et. al. Efficient Mining of Association Rules Using Merged Transactions Approach, WSEAS TRANSACTIONS on COMPUTERS, Issue 5, Volume 5, 916-923, May 2006.

[13] E. Keogh, et al., Clustering of Time Series Subsequences is Meaningless: Implications for Past and Future Research, In *proceedings of the 3rd IEEE International Conference on Data Mining* . Melbourne, FL. Nov 19-22, 2003.

[14] Ioannis N. Kouris, et. al. A Spatiotemporal View of Transactional Data for Data Mining, WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS, Issue 8, Volume 2, 1179-1183, August 2005.

[15] N. Kumar, et al., Time-series Bitmaps: A Practical Visualization Tool for working with Large Time Series Databases, In proceedings of SIAM International Conference on Data Mining (SDM '05), Newport Beach, CA, April 21-23, 2005.

[16] J. Lin, et al., Visually Mining and Monitoring Massive Time Series, In proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Seattle, WA, Aug 22-25, 2004.

[17] Jiangbo Liu and Yufen Huang. Healthcare Data Analysis using Data Mining Algorithms, WSEAS TRANSACTIONS on COMPUTERS, Issue 6, Volume 5, 1389-1397, , June 2006..

[18] Jelena Mamcenk and Regina Kulvietiene. Data Mining Technique for Collaborative Server Log File Analysis, WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS, Issue 8, Volume 2, 1111-1115, August 2005.

[19] P. Patel, et al., Mining Motifs in Massive Time Series Databases, In proceedings of the 2002 IEEE International Conference on Data Mining. Maebashi City, Japan. Dec 9-12.

[20] C. Ratanamahatana, et al., A Novel Bit Level Time Series Representation with Implications for Similarity Search and Clustering, PAKDD 05, 2005.

[21] W. Roth, MIMSY: A System for Analyzing Time Series Data in Stock Market Domain, University of Madison, Madison, 1993. Master Thesis.

[22] Y. Tanaka and K. Uehara, *Discover Motifs in Multi Dimensional Time-Series Using the Principal Component Analysis and the MDL Principle*, In proceedings of the 3rd International Conference on Machine Learning and Data Mining in Pattern Recognition, 2003

[23] Li Wei, et al., Assumption-Free Anomaly Detection in Time Series, In Proc. of the 17th International Scientific and Statistical Database Management Conference (SSDBM 2005), Santa Barbara, CA, U.S.A.

[24] M. Zaki, et. al., New Algorithms for Fast Discovery of Association Rules, Proc. Of the 3rd Int'l Conf. On Knowledge Discovery and data Mining (KDD-97), AAAI Press, 1997.