Fast Approaches to Simple Problems in Financial Time Series Streams *

Yunyue Zhu
Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, New York, NY 10012
yunyue@cs.nyu.edu

Dennis Shasha
Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, New York, NY 10012
shasha@cs.nyu.edu

ABSTRACT

Financial time series streams are watched closely by millions of traders. In this paper, we consider two simple problems: finding high correlations among all pairs of time series and finding unusually high bursts of events. We explain their applicability to time series and our basic approach.

1. INTRODUCTION

Intraday financial time series data have become more and more accessible to general investors. For example, the Trade and Quote (TAQ) time series published by the New York Stock Exchange (NYSE) contain intraday transaction data for all securities listed on the New York Stock Exchange (NYSE) and American Stock Exchange (AMEX), as well as Nasdaq National Market System (NMS) and SmallCap issues.

There are about 50,000 securities trading in the United States, and every second up to 100,000 quotes and trades (ticks) are generated. A quote gives the information that a market participant is willing to trade an equity at some price. There are bid quotes and ask quotes. For example, a bid of $26.55 on MSFT(Microsoft) with volume 1000 says that the market participant wants to buy 1000 shares of the stock MSFT at the price of $26.55 per share. Similarly, an ask of $26.58 on MSFT with volume 1000 says that the market participant wants to sell 1000 shares of the stock at the price of $26.58 per share. A trade can be executed when the bid price meets the ask price. The quotes (bids and asks) and trades are the datafeeds of financial data streams. Each quote and trade is also called a tick in the financial world.

Many computer trading strategies (also called technical trading strategies) revolve around the following ideas:

1. **Pairs Trading** Forecasting price movement is essential to a trading strategy. However, an individual stock's price movements obey more or less a random walk model (i.e., the differences between consecutive stock prices are independent Gaussian random variables). The key phrase is "more or less". If a stock price movement followed this model precisely, it would be impossible to forecast it. Fortunately, the price movements of different stocks (in the same industry) are correlated. If we observe that the price of ABC is going up while the price of the usually correlated XYZ stays the same, we can speculate that the price of XYZ will follow the price movement of ABC and make a profit.

2. **Index replication** The idea of index replication is to replicate a given index, for example the Nasdaq index, with some of its constituents, thus avoiding transaction costs in low volume equities.

3. **Hammer Discover** Market Makers are large holders of stocks. Whenever they buy or sell in large volumes, they exercise some control (their "hammer") over the security [1]. To avoid calling attention to themselves, they partition the purchases/sales into a number of small trades to avoid paying more when buying or receiving less when selling. But such a hammer might be discovered using some intelligent burst detection from the trading volumes. Once one discovers such hammer, he can infer the stock price movement and profit from it by purchasing when the hammer is purchasing.

2. CORRELATION DISCOVERY

From the above examples, we can see that the correlation plays an important role in technical trading strategies. In fact, pairs trading is also known as correlation trading. Effective pairs trading requires the continuous monitoring of those highly correlated time series streams. Such correlation includes synchronized correlation and time-lagged correlation. Because the number of data streams is very large, we should avoid the naive algorithm that computes all pair-wise correlations.

Index replication is also based on the correlation computation. Let $\vec{f}$ be the time series of an index, and the time series $\vec{g}_1, \vec{g}_2, ..., \vec{g}_k$ be the components that make up the index. We want to choose only a few components that match the index best. This is done by a matching pursuit algorithm. First we approximate $\vec{f}$ by projecting it on a unit

*Work supported in part by U.S. NSF grants IIS-9988636 and N2010-0115586.

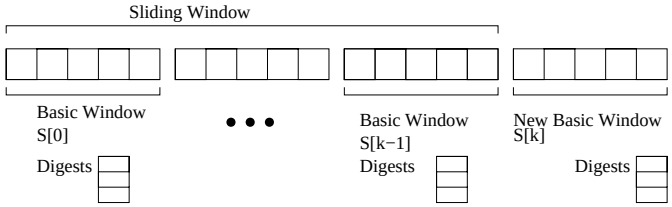

Figure 1: Sliding windows and basic windows

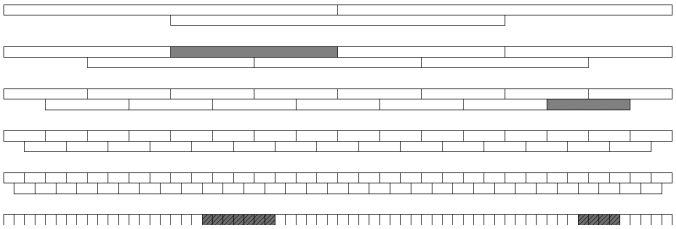

Figure 2: Examples of the windows that include subsequences in the shifted wavelet tree

norm time series $\vec{g}_{r_0}$:

$$\vec{f} =< \vec{f}, \vec{g}_{r_0} > \vec{g}_{r_0} + R\vec{f}.$$

Since the residual $R\vec{f}$ is orthogonal to $\vec{g}_{r_0}$, we have

$$||\vec{f}||^2 = | < \vec{f}, \vec{g}_{r_0} > |^2 + ||R\vec{f}||^2.$$

We can minimize the norm of the residual $R\vec{f}$ by choosing the $\vec{g}_{r_0}$ that maximizes the $| < \vec{f}, \vec{g}_{r_0} > |$. This process is iterated until the norm of the residual is below some threshold, in which case, $\vec{g}_{r_0}, \vec{g}_{r_1}, ..., \vec{g}_{r_c}$ replicate the index. Because the inner product between two normalized time series is their correlation coefficient, the problem of index replication can be reduced to finding the highly correlated time series.

We have developed StatStream[6] for real time monitoring of correlations among time series streams. The goal of StatStream is to detect time series stream pairs with high correlation over sliding windows. For example, the user might ask, "which pairs of stocks were correlated with a value of over 0.9 for the last hour?" We solve this problem by a window hierarchy shown in figure 2. The use of the intermediate time interval that we call *basic window* provides a tradeoff between throughput and response time. Results of user queries need not be delayed more than the basic window duration. In our example, the user will be told about correlations between 2 PM and 3 PM by 3:02 PM and correlations between 2:02 PM and 3:02 PM by 3:04 PM.

The raw data streams are too large for efficient data management and processing. They must be reduced to digests for real time data mining. We maintain digests of time series associated with basic windows and sliding windows. The digests include the average, variance of the time series and some coefficients to capture the raw shape of the time series. Such coefficients can be the Discrete Fourier Transform (DFT) coefficients [2], the Discrete Wavelet Transform (DWT) coefficients [5] or Singular Value Decomposition (SVD) coefficients [4]. The DFT is very attractive in the data stream context because the DFT coefficients can be updated incrementally. We can show that the DWT coefficients could also be updated incrementally with some overhead. Sometimes in financial analysis, it is the correlation between the *deltas*(or *returns*) of time series, instead of the time series itself, that is of interest. In this case, the DFT, DWT and SVD work poorly, because the delta time series of a random walk time series is white noise (Gaussian random variable). So, a few DFT, DWT or SVD coefficients cannot capture enough information of the time series. In such cases, sketch-based methods, for example [3], come into play. Sketches are the random projections of a time series. The inner product between time series can be approximated by the inner product between their sketches within a probabilistic guaranteed error bound. By keeping the sketches as the digests of time series, we allow the time series under consideration to be any type of signal.

3. BURST DISCOVERY

Correlation involves the comparison of different time series, whereas a burst reflects the transient behavior of single time series. In the hammer discovery strategy, the challenge is to discover in the quote stream bursts of bid/ask volume from a particular market participant for a particular security. Because only recent ticks are relevant, we search for bursts in a sliding window fashion. A non-trivial problem for the sliding window model is to decide the size of the sliding window. To overcome this problem, we introduce the framework of Elastic Burst Detection [7], where the user needs to specify only the range of the sliding window sizes and the threshold for each window size, and will be notified of all those window sizes in the range having bursts. We have designed a data structure based on wavelets, called a Shifted Wavelet Tree, for efficient elastic burst detection. The Shifted Wavelet Tree can be used to detect bursts with different window sizes simultaneously. In figure 2, we show examples of how bursts with different window sizes are detected.

4. CONCLUSION

Simple problems can be done slowly or fast. We show how to do two of them fast.

5. REFERENCES

[1] http://www.cybertrader.com/cybertrader/hammer.asp.

[2] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient Similarity Search In Sequence Databases. In *FODO 1993*, pages 69–84.

[3] P. Indyk, N. Koudas, and S. Muthukrishnan. Identifying representative trends in massive time series data sets using sketches. In *VLDB 2000*, pages 363–372.

[4] F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently supporting ad hoc queries in large datasets of time sequences. In *SIGMOD 1997*, pages 289–300, 1997.

[5] I. Popivanov and R. J. Miller. Similarity search over time series data using wavelets. In *ICDE*, 2002.

[6] Y. Zhu and D. Shasha. Statstream: Statistical monitoring of thousands of data streams in real time. In *VLDB 2002*, pages 358–369.

[7] Y. Zhu and D. Shasha. Efficient elastic burst detection in data streams. In *Submitted for publication*, 2003.