Sketching a Graph to Query a Time-Series Database

Martin Wattenberg
Dow Jones / SmartMoney.com
601 W. 26th St.
New York, NY 10001 USA
+1 212 304-0008
mwattenberg@smartmoney.com

ABSTRACT

Sequential data is easily understood through a simple line graph, yet systems to search such data typically rely on complex interfaces or query languages. This paper presents QuerySketch, a financial database application in which graphs are used for query input as well as output. QuerySketch allows users to sketch a graph freehand, then view stocks whose price histories match the sketch. Using the same graphical format for both input and output results in an interface that is powerful, flexible, yet easy to use.

Keywords

Time series, dynamic queries, database, user interface, stock market, visual query, freehand sketch.

INTRODUCTION

When analyzing a time-series database, such as a collection of historical stock price data or weather information, a frequent goal is to find objects whose series match a given pattern. For instance, an investor who wishes to diversify a portfolio heavy in technology stocks might look for a stock whose price increased in periods when technology companies did poorly. Researchers have uncovered a broad array of efficient algorithms and data structures that support such pattern-matching queries [1,4,5].

Less attention, however, has been paid to creating a simple user interface for these queries. Several systems, for example, require the user to input strings of symbols to represent data patterns [1,6]. This is a relatively cumbersome task that does not correspond to the way the data is almost universally represented: a simple line graph. This paper introduces QuerySketch, an application in which the user performs queries by drawing a freehand graph that is used as a target pattern. Sketching as a query interface is intuitive, easy and enjoyable, while giving the user a high degree of flexibility and power.

IMPLEMENTATION

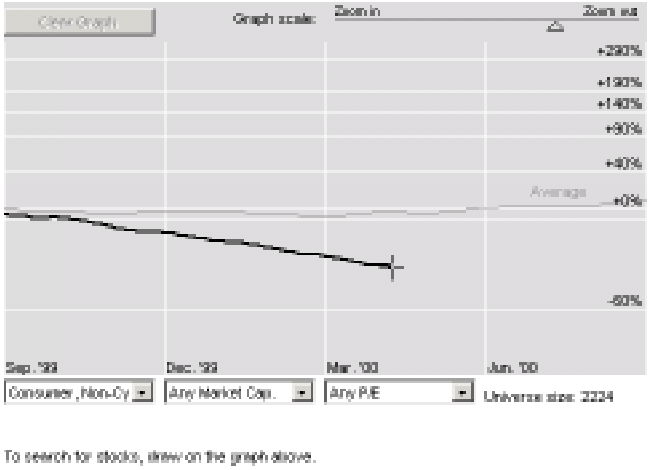
QuerySketch is implemented as a Java applet, viewable within standard web browsers. The applet loads a database of historical stock price data into RAM, so that all processing can be done on the client computer rather than calling back to the web server. This architecture was chosen primarily to allow the fast response times necessary for dynamic queries [2]. Pattern matching is performed using a metric based on the Euclidean distance between sequences of monthly percentage price changes. This metric was selected based on its common use in time series queries [4,5], but the QuerySketch interface is equally compatible with other metrics.

INTERACTION DESIGN

User interaction in QuerySketch has two main steps: performing a query and viewing results.

Query interface

In designing the interface, we wanted to ensure that a variety of queries were possible. For instance, What stock will best diversify my portfolio? What is a technology stock that increased in price in April 2000? Which stocks have fallen dramatically in price over the past year, but have recently begun to go up?



To search for stocks, draw on the graph above.

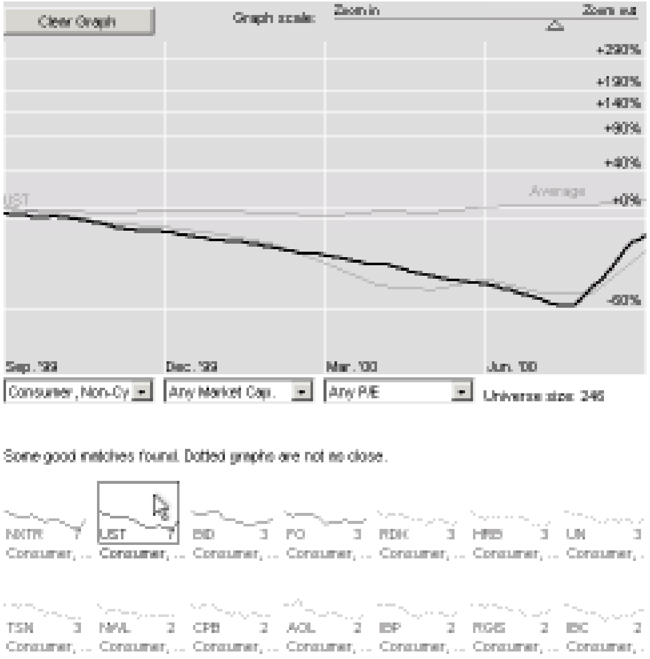
Figure 1: Query in progress. Heavy line is the user's query, light line displays aggregate data for reference.

The primary tool for making such queries is the panel at the top of Figure 1. Users select a universe of companies to search, and a reference graph of aggregate performance is displayed. Users then draw their own pattern on the same grid with the mouse. The scale of the graph is logarithmic. This is critical: if a user wishes to look for a price history with a steady growth rate, it is far easier to draw a straight line than an accurate exponential curve.

Users may also draw graphs that are only partially defined, or edit their existing queries. For instance, they may wish to find a stock that went up in January and down in July, but not care about price movement at other times. In this case they can draw an upward graph for January, then refine the query by adding a downward stroke during July.

Viewing Results of a Query

Each time the user releases the mouse while drawing on the graph entry panel, the applet performs the query and displays the results as a series of thumbnail graphs in the lower portion of the screen, as seen in Figure 2. Several considerations turned out to be important in designing the result display.



Some good matches found. Dotted graphs are not so close.

Figure 2: Query results. Thumbnail graphs represent results, with selected thumbnail overlaid full-size on search grid.

First, it is helpful to present miniature thumbnail graphs of the stocks retrieved by the query. These snapshots provide instant, comprehensible feedback on the quality of the results. In addition, moving the mouse over a thumbnail displays the full version of the thumbnail overlaid on the main graph entry panel. This allows the user to compare the result directly with their query. Each thumbnail is also accompanied by a numerical indicator of how well it matches the target pattern.

Second, performing the query on mouse release seems to be the most desirable degree of interactivity. Forcing the user to click a “submit” button adds a gratuitous step to each query. Dynamic updates as the mouse moves cause an uninformative and distracting shuffling of the results.

Third, it is important to allow users instant access to “drill down” on the thumbnails for more information. Since the point of the query is to find stocks to invest in, we designed the applet so that clicking on a thumbnail brought the user to SmartMoney.com’s stock research web pages.

FUTURE DIRECTIONS

There are many avenues for future research. First, free-form sketching queries can be applied to time series data other than stock histories. The ability to sketch in response to a reference graph may be especially useful in data sets with possible causal relationships among the various elements, such as populations of different organisms in a natural environment. Second, other types of pattern-matching can be explored, including methods that allow for patterns that are independent of time scale. Third, it would be useful to investigate interfaces for defining queries that involve inequalities: a major limitation of the current implementation is that it is hard to look for stocks that move above a certain rate, rather than exactly at that rate.

ACKNOWLEDGMENTS

Thanks to Marc Frons, Joon Yu, and Ben Shneiderman for helpful comments on the QuerySketch implementation.

REFERENCES

1. Agrawal, R.; Psaila, G.; Wimmers, E. L.; and Zait, M. 1995b. Querying shapes of histories. *Proc. of the 21st Int'l Conference on Very Large Databases.*
2. Ahlberg, C., Williamson, C., and Shneiderman, B., Dynamic queries for information exploration: An implementation and evaluation, *Proc. ACM CHI'92: Human Factors in Computing Systems* (1992), 619-626.
3. Catarci, T., Costabile, M.F., Levialdi, S. and Batini, C., Visual Query Systems for Databases: A Survey, *Technical Report SI/RR-95/17, Dipartimento di Scienze dell'Informazione, Universita' di Roma "La Sapienza"*, 1995
4. Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. Fast subsequence matching in time-series databases. *Proc. of the ACM SIGMOD Conference on Management of Data*, May 1994.
5. Keogh, E.J., Pazzani, M.J, A simple dimensionality reduction technique for fast similarity search in large time series databases, *Proc. of Pacific- Asia Conf. on Knowledge Discovery and Data Mining*, 2000.
6. Xia, B. Similarity Search in Time Series Data Sets. Master thesis, Simon Fraser University, 1997.