Visual Queries for Finding Patterns in Time Series Data

Harry Hochheiser
Department of Computer Science
Human-Computer Interaction Lab
University of Maryland
College Park, MD 20742 USA
hsh@cs.umd.edu

Ben Shneiderman
Department of Computer Science
Human-Computer Interaction Lab,
Institute for Advanced Computer Studies, and
Institute for Systems Research
University of Maryland
College Park, MD 20742 USA
ben@cs.umd.edu

Abstract

Few tools exist for data exploration and pattern identification in time series data sets. Timeboxes *are rectangular, direct-manipulation queries for studying time-series datasets. Timeboxes are the primary query tool in our TimeSearcher application, which supports interactive exploration via dynamic queries, along with overviews of query results and drag-and-drop support for query-by-example. This paper describes the TimeSearcher application and possible extensions to the timebox query model, along with a discussion of the use of TimeSearcher for exploring a time series data set involving gene expression profiles.*

Keywords

Information Visualization, Time Series Data, Dynamic Query, Visual Query

1. Introduction

Time series data sets are found in many domains including finance, meteorology, physiology and genetics. To date, most information visualization work on these data sets has focused on display and interactive exploration, often emphasizing the periodic nature of some calendar-based data sets [7, 15].

Work in data mining has addressed the need for additional tools to identify patterns of trends of interest in these data sets. Algorithmic and statistical methods for identifying patterns [1, 2, 3, 6, 10] have provided substantial functionality in a wide variety of situations. In domains such as stock price analysis, familiar patterns have been named and identified as shorthand approaches to identifying trends of interest [18].

Tools for specifying dynamic queries over these data sets have recently been developed: QuerySketch supports query-by-example based on a sketch of a desired profile [26], and Spotfire's Array Explorer 3 supports graphical queries for temporal patterns [25].

This paper introduces timeboxes: visual query operators for time series data sets. Timeboxes are rectangular regions that are placed and directly manipulated on a timeline, with the boundaries of the region providing the relevant query parameters. The use of timeboxes is discussed in the context of TimeSearcher, a data exploration tool for time series data. Related research is discussed, along with a description of planned extensions that will expand the expressive power of timebox queries.

2. Timeboxes: Interactive Temporal Queries

Timeboxes are rectangular query regions drawn directly on a two-dimensional display of temporal data. The extent of the Timebox on the time (x) axis specifies the time period of interest, while the extent on the value (y) axis specifies a constraint on the range of values of interest in the given time period. More specifically, a timebox that goes between (x_1, y_1) and (x_2, y_2) indicates that for the time range $x_1 \leq x \leq x_2$, , the dynamic variable must have a value in the range $y_1 \leq y \leq y_2$ (assuming $y_2 \geq y_1$ and $x_2 \geq x_1$).

We assume that the temporal data is divided into discrete time points of granularity determined by each data set. The discrete nature of the data is enforced by constraining timeboxes to occupy an integral number of time points. Multiple timeboxes can be drawn to specify conjunctive queries. Items in a data set must match all of the constraints implied by the active timeboxes in order to be included in the result

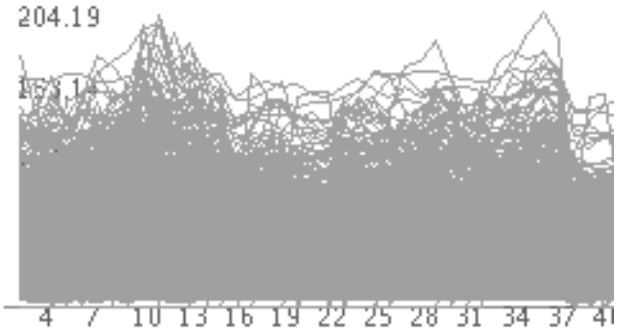

Figure 1. a "graph envelope" overview, formed by superimposing the time series for all of the items in the data set.

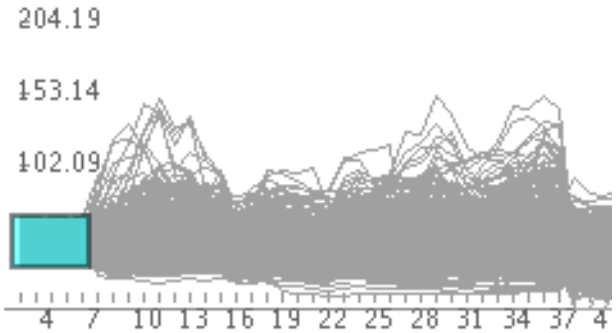

Figure 2. A single timebox query, for items between $28 and $64 during weeks 1-5

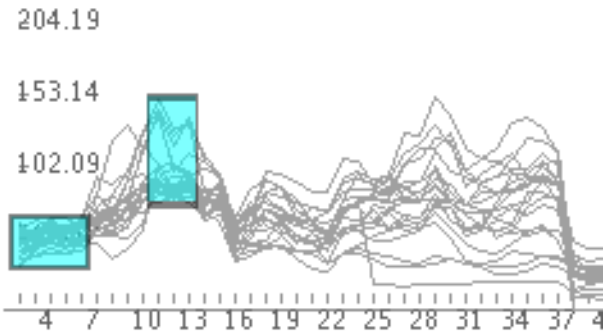

Figure 3. A refinement of the query in Figure 2.

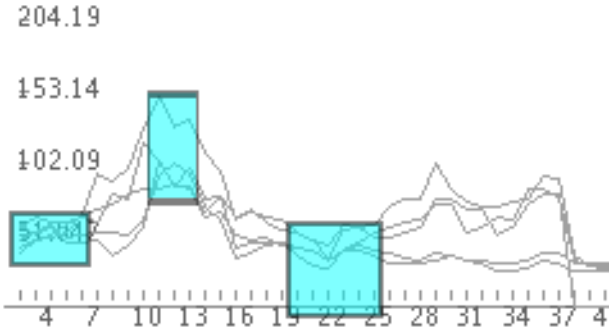

Figure 4. A complex query containing three timeboxes.

.

set.

Creation of timeboxes is straightforward: the user simply clicks on the desired starting point of the timebox and drags the pointer to the desired location of the opposite corner. As this is identical to the mechanism used for creating rectangles in widely used drawing programs, this operation should be familiar to most users. Once the timebox is created, it may be dragged to a new location or resized via appropriate resize handles on the corners, using similarly familiar interactions.

In all cases, query processing occurs on mouse-up. When the user releases the mouse, the current position of the timebox is stored, the query is updated, and the new result set is displayed.

Construction of timeboxes is aided by drawing all of the items in the data set directly on the query area. This "graph envelope" display provides additional insight into the density, distributions, and patterns of change found among items in the data set, in a display that is similar to a parallel coordinates visualization [14] (Figure 1).

The example data set shown in Figure 1 contains weekly stock prices for 1430 stocks and will be used in a brief scenario to illustrate the use of timeboxes. An analyst interested in finding stocks that rose and then fell within a four-month period might start by drawing a timebox specifying stocks that traded between $28 and $64 during the first few weeks. When this query is executed, the graph envelope is updated to show only those records that match these constraints. We can quickly see that this query substantially limits the number of items under consideration, but many still remain (Figure 2).

To find stocks in this restricted set that rose in subsequent weeks, the user draws a second box, specifying items that traded between $73 and $147 during weeks 10-12 (Figure 3). A third box, specifying a lower price range ($0-$56) during weeks 19-24 completes the query (Figure 4).

As timeboxes are added to the query, the graph envelope provides an ongoing display of the effects of each action and an overview of the result set. Once created, the timeboxes can be scaled or moved singly or together to modify the query constraints.

The use of simple, familiar idioms for creation and modification of timeboxes supports interactive use with minimal cognitive overhead. Rapid (<100ms), automatic query processing on mouse-up events provides the virtually instantaneous response necessary for dynamic queries, thus supporting interactive data exploration. Users can easily and quickly try a wide range of queries, modifying these queries to quickly see the effects of changes in query parameters. This ability to easily explore the data is helpful in identifying specific patterns of interest, as well as in gaining understanding of the data set as a whole.

3. TimeSearcher

TimeSearcher uses timeboxes to pose queries over a set of entities with one or more time-varying attributes. Entities have one or more static attributes, and one or more time-varying attributes, with the number of time points and the definition of those points being the same for every entity in a given data set. If there are multiple time-varying attributes, any one of them can be selected for querying, through a drop-down menu that specifies the dynamic attribute being queried. All active queries refer to the same attribute.

When a data set is loaded, entities in the data set are displayed in a window in the upper left-hand corner of the application. Each entity is labeled with its name, and the values of the active dynamic attribute are plotted in a line graph. Complete details about the entity (details-on-demand) can be retrieved by simply clicking on the graph for the desired entity: this will cause the relevant information to be displayed in the upper right-hand window (Figure 5).

The bottom-left corner of the TimeSearcher window is the query input space. This space initially contains an empty grid. To specify a query, users simply draw a timebox in the desired location. Query processing begins as soon as users release the mouse, signifying the completion of the box. Thus, users do not need to press a button to explicitly start a search. When query processing completes, the display in the top half of the application window is updated to show those entities that match the query constraints. For all of these entities, the time points that correspond to the queries are highlighted, in order to simplify interpretation of the display.

Once the initial query is created, the timeboxes can be moved and resized. The hand and box icons on the lower toolbar are used to switch between creating timeboxes and moving/resizing them. As is the case with initial timebox creation, query processing begins immediately upon completion of the movement/resizing of the timebox.

When multiple timeboxes are present, they can be modified individually or simultaneously in groups of two or more. This functionality is particularly useful for searches

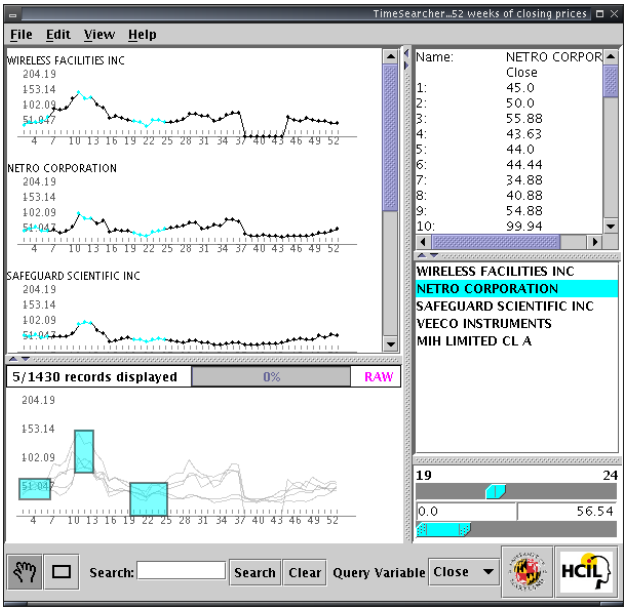

Figure 5. The TimeSearcher application window. Clockwise from upper-left: data items, details-on-demand, item list, range sliders for query adjustment, and query space.

for complex patterns (Figure 4). In these cases, users can select some or all of the timeboxes (using standard lasso and shift-click interactions) and simultaneously apply the same translation and/or scale along either or both axes to all selected timeboxes. This is useful for searching for instances of a pattern that vary slightly in scale or magnitudes, or for modifying queries based on example items.

Timeboxes can also be adjusted via a pair of range sliders in the lower right-hand corner of the screen. When a timebox is selected (or created), these range sliders are initialized with the parameters of the timebox, with the top slider containing time extents and the bottom including values. As each dimension is adjusted separately by its own slider, these controls support a degree of fine-tuning that might be difficult to achieve by dragging the timeboxes. These sliders are disabled when multiple timeboxes are selected.

Much of the research in mining of time series involves queries for items in a data set that are similar to a specified query [1, 3, 6, 10]. TimeSearcher provides a simple drag-and-drop mechanism for these "query-by-example" queries: the user can simply click on an entry in the data display window, drag it into the query window, and release the mouse to drop, thus instantiating a query.

The query resulting from a drag and drop has a separate timebox for each time point in the data set. Each timebox has a width of one interval, with the query values centered around the actual value of the attribute for that entity at the

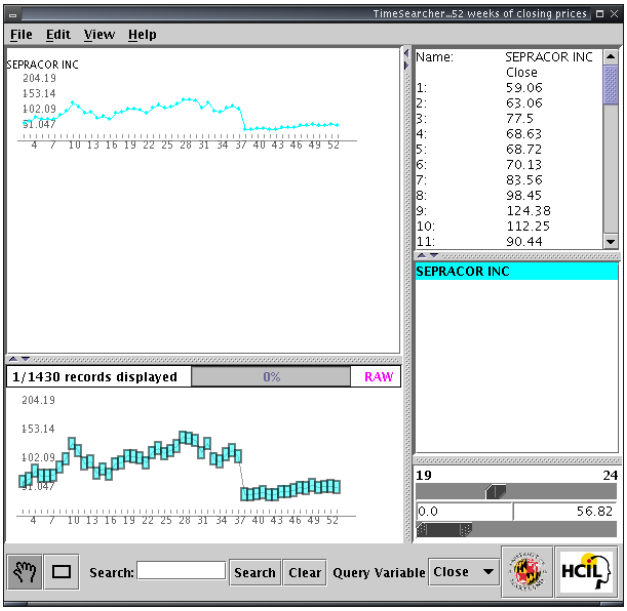

Figure 6. Drag-and-drop query-by-example, with results.

given time point. The height of each timebox is set to be 10% of the total range of the attribute being queried, so each timebox has a range of $v \pm 5\%$ of the total range in the attribute value, where v is the value of the template time series at the given time point (Figure 6).

As the resulting query is composed of multiple timeboxes, it can easily be modified to account for varying levels of similarity. For example, the boxes could be enlarged to allow for a looser definition of similarity, or subsets of the query could be eliminated to focus on items that are similar only at specific time points.

3.1. Overviews

TimeSearcher provides a limited overview display in the upper left-hand window, displaying each of the entities in the data set in a linear list. As this display shows a small number of items at any given time, it is not an effective overview. Another possible overview would display each of the entities in a thumbnail graph. These thumbnails would be displayed in a grid, instead of the linear arrangement shown in Figures 5 and 6. This approach suffers from two shortcomings. For any reasonably sized data set (more than a few dozen items), the limited screen space available would restrict each thumbnail to a tiny area of the screen, rendering it virtually unreadable. Furthermore, displaying each entity in a separate graph may not help users in identifying global trends, such as the extreme values of the time-varying attribute at any given point in time.

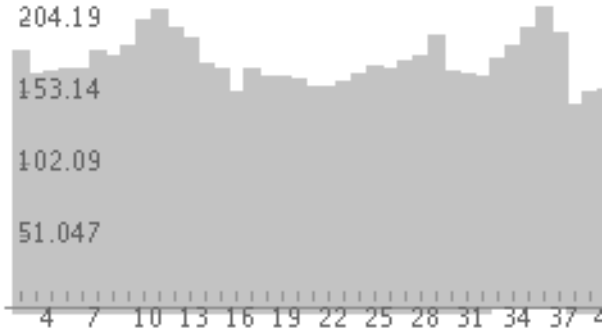

Figure 7. Query window with data envelope.

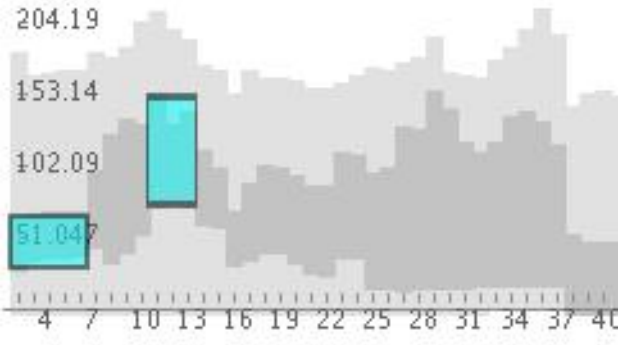

Figure 8. Query display with data and query envelopes.

Another form of overview might be provided by displaying the extreme values that can be found in the data set at each time point. Known as a "data envelope", this overview is optionally shown in the background of the query window as a contour that follows the extreme values of the query attribute at each point in time, thus displaying the range of values that may be queried (Figure 7). When the user executes a query, the data envelope is extended by a "query envelope" - an overlay that outlines extreme values of the entities in the result set (Figure 8). This display provides users with a graphic summary of the relationship between the result set and the data set as a whole.

Without any timeboxes present, the data envelope highlights areas that would be fruitful for query creation, while leaving empty areas unmarked. For example, the data envelope in Figure 7 does not extend to the upper right-hand corner, so queries in that region would not return useful results. When a timebox is created, the updated query envelope shows the differences between the current result set and the data set as a whole, thus clarifying the range of values excluded by the timebox. The query envelope also guides the creation of additional timeboxes, as queries outside this envelope will not match any records.

The graph envelope (Section 2) provides further support for browsing the data set. When the user mouses over a graph envelope line, the line is highlighted, thus displaying the individual items in the context of the larger data set. At the same time, the item list, item display window, and details-on-demand window are updated to display on the selected item. This tight coupling in response to lightweight mouse movement will encourage exploration based on visual examination of the graph envelope overview.

4. Implementation

TimeSearcher was implemented in Java 2, using the Swing toolkit for user-interface widgets. Drawing and scenegraph control in the data and query displays, along with functionality for moving and rescaling timeboxes, is provided by Jazz, a zooming toolkit written in Java [5]. Timeboxes, graphs of each item, and query and data envelopes are implemented as Jazz widgets. As these widgets are implemented as Java classes, they are easily extended for specialized use.

Timebox queries are processed via a modified orthogonal range tree query algorithm [8]. Each of the MN data points in the set (M entities having measurements at each of N time points) is stored in a two-dimensional orthogonal range tree. A timebox is then used to generate a query, which identifies all of the data points that fall within the timebox. For each point that is identified, a counter is incremented in the entity associated with that point. Once all points are processed, all of the entities are examined to find those that have a count that is equal to the width of the timebox - indicating that all of the values for that entity during the specified time range where in the specified value range.

Since the time dimension covers a known range, and each entity has a value at every time point, we use a linear array in place of the range tree for the time dimension. The start and endpoints in this array can be found in constant time, and the value indices associated with each included time points are then searched. The resulting algorithm processes queries of width w in $O(w \log MN + k)$ time, where k is the number of points found within the timebox.

5. Application Example: Gene Expression Levels in DNA Microarray Experiments

The advent of DNA microarray technology has led to the possibility of experiments that examine the response of an entire genome to some event or stimulus [11]. Many of these experiments examine the changes in gene expression that occur over the course of time [9, 13]. These often analyses involve identification of similar profiles in order to group genes that have similar expression patterns. We have been investigating the use of TimeSearcher as a tool to support the process of identifying profiles of interest.

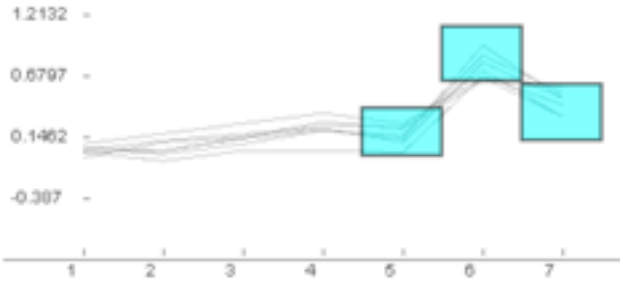

Figure 9. Yeast genes with peak expression levels at the 6th measurement - 19 hours after placement in new medium.

In order to study the shift between anaerobic to aerobic metabolism, DeRisi, Iyer, and Brown examined gene expression changes in yeast (*Saccharomyces cerevisiae*) cells at several points in time after their placement in fresh medium [9]. Microarray measurements were made every 2 hours between 9 and 21 hours after initial placement, for a total of 7 time points. Figure 9 shows a sample query, identifying genes with expression levels that peaked at the sixth measurement (19 hours).

This work and other time series investigations [19] in microarray data present exciting opportunities for the use of TimeSearcher in bioinformatics research. In this regard, TimeSearcher might be particularly useful when used alongside visualizations based on clustered analysis of gene expression profiles [17].

6. Related Work

Traditional time-series graphs are among the most familiar data displays. Visualizations of time-series data attempt to improve the utility of these common graphs, through the use of techniques such as increased data density or polar-coordinate displays that emphasize the serial periodic nature of the data set [7], or by distorting the time axis to realize denser information displays [21]. A recent survey of linear temporal visualizations is found in [24]. Generally, these tools focus on visualization and navigation, with relatively little emphasis on querying data sets.

A few tools have been developed for querying time-series data. MIMSY [22] provided an early example of searches for temporal patterns in stock market data, using text entry fields, pull-down menus, and other traditional widgets to specify temporal constraints. QuerySketch is an innovative query-by-example tool that uses an easily drawn sketch of a time-series profile to retrieve similar profiles,

with similarity defined by Euclidean distance [26]. Although the simplicity of the sketch interface is appealing, the use of Euclidean distance as a metric can lead to non-intuitive results [16].

Spotfire's Array Explorer 3 [25] supports graphically editable queries of temporal patterns, but the result set is generated by complex metrics in a multidimensional space. This potent approach produces useful results, but users may wish to constrain result sets more precisely.

As two-dimensional query widget, timeboxes are similar to earlier proposed models for two-dimensional dynamic query widgets [23]. These earlier proposals used two-dimensional widgets to pose simultaneous constraints on two variables - for example, the number of bedrooms and the price of a house for sale. Like these widgets, timeboxes are graphically two-dimensional. However, timeboxes are more expressive, as each timebox poses constraints on the value for each of the w time points contained within it.

The data mining community has developed a wide variety of innovative techniques for algorithmically extracting interesting patterns from time series. Useful approaches including dynamic time warping [6] and Discrete Fourier Transforms (DFT) in combination with spatial queries [10]. To date, the focus in data mining work has been on the development of search algorithms, with relatively little attention to query specification or interactive systems. One exception is Agrawal et al.'s Shape Definition Language, which specifies queries in terms of natural language descriptions of transition profiles [2]. Although a user interface is not described, an interactive system might support creation of queries through combinations of Shape Definition Language primitives. Support for progressive refining of queries was addressed by Keogh and Pazanni, who suggested the use of relevance feedback for results of queries over time series data [16].

7. Discussion

The power of the timebox model lies in its simplicity. Timeboxes are drawn and modified using standard drawing manipulations, with the graph plot providing a familiar space for construction and modification of queries. This combination of well-known components leads to an interface that is easily understood with a minimal cognitive load.

The expressive power of the timebox model lies in the ability to specify multiple constraints with one query widget: by drawing a single rectangle, the user specifies both a value range and a range of time periods during which the items must have values within that range. If a time series with n data points is viewed as an n-dimensional data set [10], each timebox can be seen as specifying constraints over a contiguous subset of these n dimensions. Thus, the creation or a modification of a timebox of width w leads to the simultaneous specification of w constraints. This represents a substantial improvement over single-attribute query widgets, which would have required $O(w)$ individual interactions to specify the same number of parameters.

Timeboxes also differ from traditional dynamic query widgets in their construction and manipulation directly on the data space. As timeboxes are drawn directly on a graph space suitable for plotting a time series, the queries are easily interpreted at a glance. Complex queries containing multiple timeboxes provide visual feedback that illustrates the pattern defined by the query (Figure 4). The data, query, and graph envelopes drawn directly on the two-dimensional query space provide additional feedback that can aid the process of creating queries and interpreting result sets.

Although their development was motivated by interest in querying time series data sets, timeboxes are more general. For example, any ordinal dimension might be used on the axis, thus providing opportunities to query sequential data such as DNA sequences, or any general function. Timeboxes might also be used to specify queries in data sets using parallel coordinates [14], if the dimensions involved have common value ranges and some appropriate ordering.

TimeSearcher provides an initial implementation of a query tool based on the timebox model. We see several opportunities for building on this work to expand the power of both the tool and the model.

7.1. Extending Query Expressiveness

The timebox queries supported in the current TimeSearcher implementation are relatively simple. Specifically, users are limited to queries that specify a fixed range of values that must be met during a fixed time period. For many applications - including exploration of gene expression levels (Section 5) - these queries are sufficient. For other tasks involving more general queries, additional expressive power may be needed.

One class of queries that would require extensions to the timebox model involves relative changes in values or times. For example, a financial analyst might be interested in finding stocks that increased by some relative amount in any interval. The important feature of this query is the amount of the change: unlike existing timebox queries, no absolute value ranges are specified. Similarly, we might imagine queries that search for transitions occurring during arbitrary time periods (e.g., during any 2-month interval). In focusing on the direction and magnitude of changes, these queries would be similar to those supported by SEQ [2].

Extending the timebox model to handle these queries will require additional mechanisms for specification of queries and presentation of the results. Specifically, additional widgets for expressing an allowable range of time or attribute values will be needed, along with visualizations

that clearly indicate the relationship between queries and results.

Our work with DNA microarray data sets (Section 5) has led to the identification of another interesting extension to the query model. Much of the effort in microarray analyses involves the identification of transcription factors: genes that regulate the expression of other genes. When viewed as time series plots, these transcription factors appear to "lead" the genes they regulate. The regulated genes often have expression profiles that are similar to those of the transcription factors, but slightly delayed.

We have begun implementation of query facilities for supporting the identification of these transcription factors and their downstream targets. The use of these tools begins with the creation of a set of timeboxes that specifies the desired pattern - perhaps the expression profile of the transcription factor. The user then makes a menu selection to note this query as a "leader" pattern. This causes the items that marked the original query to be highlighted, and a new query created with the same values as the original, but times slightly offset. The new query boxes can then be moved forward and backwards in time, scaled, or otherwise adjusted to find items that "lag" in the desired fashion. We believe that this "leaders and laggards" facility has general utility for other fields.

7.2. Additional Data Types

Alternative interpretations of timeboxes and query space present intriguing opportunities for extending the query model. For example, the value axis might be interpreted as describing discrete categories, instead of real-numbered values. This modification would support the use of timeboxes for searching over categorical data sets such as those found in medical records [20, 21].

Timebox queries might also be extended to temporal data sets, such as those found in video applications [12], medical data [20], market-basket data [3], and other applications. In these cases, events have arbitrary duration, and may occur simultaneously. Users may want to formulate queries that co-occur, follow or precede each other, or have other time relationships [4]. Maintaining rapid query evaluation for these more complicated queries may be a significant challenge.

7.3. Scaling

In the current TimeSearcher implementation, time series are effectively limited in length by the screen space available for drawing the plots. For many applications, this limit of a few hundred time points is overly constraining: physiologic, financial, and other long-term data sets often generate tens of thousands of time points. Ideally, a time series query tool would gracefully handle these longer time series.

A first approach to increasing the length of manageable data sets would be to divide each item into a number of pieces, each containing a number of time points that can be handled by the software. For example, climate data with readings taken every minute over the course of many days might be divided into blocks of three hours in length. While certainly possible, this approach is far from ideal, as it obscures the continuity of the data and potentially complicates interpretation.

A preferable approach might be to provide displays and query tools that examine long time series at progressively greater levels of detail. Such a tool would begin by presenting a condensed view of each time series, with the average of n points from the original data presented as 1 point. The user would then create queries and narrow down to certain time periods of interest, restricting the display to a smaller time range that could be displayed in greater detail. Eventually, the user would be able to examine a small subset of the original time series in its original form.

Alternative possibilities include tools for filtering out time periods that are of little or no interest. Some data sets - EKG data, for example - are characterized by long periods of relatively little change with occasional periods of intense activity. In reviewing these data sets, analysts frequently want to ignore the periods of inactivity and concentrate on the interesting changes. Additional query widgets might be provided to support this filtering. For example, range sliders might be used to eliminate time periods with little change. Once such a selection was made, each time series could be reduced to a smaller set of interesting data, and the intervening time points need not be displayed.

8. Conclusion

The power of the timebox query model lies in its simplicity: through manipulations familiar to users of modern GUIs, users can simultaneously specify two dimensions of a query over time series data. The combination of the ability to use multiple timeboxes to specify a complex query and dynamic query updating provides users with a fast and easy tool for exploring these data sets.

Our implementation of the TimeSearcher prototype and our work with users has provided preliminary validation of the timebox query model. Extensions to the tool have been informed by our collaborations with molecular biologists (Section 5) and other motivated users. Further assessments of the tool, in the form of usability evaluations and empirical studies are planned. We are particularly interested in examining the performance of timeboxes relative to combinations of more traditional one-dimensional range sliders.

9. Acknowledgments

Thanks to Eric Baehrecke, Hillary Hutchinson, Hyunmo Kang, and Martin Wattenberg for valuable assistance. The first author was supported by a fellowship from America Online.

References

[1] R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *The VLDB Journal*, pages 490–501, 1995.

[2] R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zait. Querying shapes of histories. In *Proceedings of the 21st International Conference on Very Large Databases*, pages 502–514, 1995.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. L. P. Chen, editors, *Proceedings 11th International Conference on Data Engineering, ICDE*, pages 3–14, Taipei Tawian, March 1995. IEEE Press.

[4] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[5] B. Bederson, J. Meyer, and L. Good. Jazz: An extensible zoomable user interface graphics toolkit in java. In *ACM Symposium on User Interface Software and Technology*, pages 171–180, San Diego, CA, November 2000. ACM Press.

[6] D. J. Berndt and J. Clifford. Finding patterns in time series: A dynamic programming approach. In *Advances in Knowledge Discovery and Data Mining*, pages 229–248. AAAI Press/MIT Press, 1996.

[7] J. V. Carlis and J. A. Konstan. Interactive visualization of serial periodic data. In *ACM Symposium on User Interface Software and Technology*, pages 29–38, San Francisco CA, November 1998. ACM Press.

[8] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Spring-Verlag, 2000.

[9] J. DeRisi, V. Iyer, and P. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278:680–686, 24 October 1997.

[10] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*, pages 419–429, Minneapolis Minnesota, May 1994. ACM Press.

[11] H. Hamadeh and C. A. Afshari. Gene chips and functional genomics. *American Scientist*, pages 508–515, November/December 2000.

[12] S. Hibino and E. Rudensteiner. A visual multimedia query language for temporal analysis of video data. In K. Nwosu, B. Thuraisingham, and P. Berra, editors, *Multimedia Database Systems: Design and Implementation Strategies*, pages 123–159. Kluwer Academic Publishers, 1996.

[13] A. Hill, C. Hunter, B. Tung, G. Tucker-Kellogg, and E. Brown. Genomic analysis of gene expression in c. elegans. *Science*, 290(27):809–812, October 27 2000.

[14] A. Inselberg. Multidimensional detective. In *IEEE Conference on Information Visualization*, Phoenix AZ, October 1997.

[15] D. A. Keim. Pixel-oriented visualizations techniques for exploring very large databases. *Journal of Computational and Statistical Graphics*, March 1996.

[16] E. J. Keogh and M. J. Pazzani. Relevance feedback retrieval of time series data. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval SIGIR '99*, pages 183–190, Berkeley CA, August 1999. ACM.

[17] S. Kim, J. Lund, M. Kiraly, K. Duke, M. Jiang, J. Stuart, A. Elzinger, B. Wylie, and G. Davidson. A gene expression map for *Caenorhabditis elegans*. *Science*, 293:2087–2092, 14 September 2001.

[18] J. Little and L. Rhodes. *Understanding Wall Street*. Liberty Publishing, Inc., Cockeysville MD, 1978.

[19] K.-H. Pan, C.-J. Lih, and S. Cohen. Analysis of DNA microarrays using algorithms that employ rule-based expert knowledge. *Proceedings of the National Academies of Science, USA*, 99(4):2118–2123, February 19 2002.

[20] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. Lifelines: Visualizing personal histories. In *Proceedings of the 1996 Conference Human Factors in Computing Systems. Seattle, WA*, pages 221–227. ACM Press, April 1996.

[21] S. Powsner and E. Tufte. Graphical summary of patient status. *The Lancet*, 344:386–389, 1994.

[22] W. G. Roth. MIMSY: A system for analyzing time series data in the stock market domain. Master's thesis, University of Wisconsin, Department of Computer Science, 1993.

[23] B. Shneiderman. Dynamic queries for visual information seeking. *IEEE Software*, 11(6):70–77, November 1994.

[24] S. Silva and T. Catarci. Visualization of linear time-oriented data: a survey. In *Proceedings of the first International Conference on Web Information Systems Engineering*, Hong Kong, June 2000. IEEE Computer Society.

[25] Spotfire. http://www.spotfire.com.

[26] M. Wattenberg. Sketching a graph to query a time series database. In *Proceedings of the 2001 Conference Human Factors in Computing Systems, Extended Abstracts*, pages 381–382, Seattle WA, March 31- April 5, 2001 2001. ACM Press.