Viator - A Tool Family for Graphical Networking and Data View Creation

Stephan Heymann\textsuperscript{1,2}, Katja Tham\textsuperscript{1,3}, Axel Kilian\textsuperscript{2}, Gunnar Wegner\textsuperscript{2}, Peter Rieger\textsuperscript{1,2}, Dieter Merkel\textsuperscript{2} and Johann Christoph Freytag\textsuperscript{1}

\textsuperscript{1} Humboldt-Universität zu Berlin, Unter den Linden 6, D-10099 Berlin, Germany
\textsuperscript{2} Kelman (now Moosbaum) GmbH, Köpenicker Strasse 325, D-12555 Berlin, Germany
\textsuperscript{3} Fachhochschule für Technik und Wirtschaft Treskowallee 8, D-10318 Berlin, Germany

Mail to: heymann@dbis.informatik.hu-berlin.de
Abstract

Web-based data sources, particularly in Life Sciences, grow in diversity and volume. Most of the data collections are equipped with common document search, hyperlink and retrieval utilities. However, users’ wishes often exceed simple document-oriented inquiries. Users wish to comprehend context-sensitive information from a data source. Especially data categories that constitute relationships between two or more items require potent set-oriented content management, visualization and navigation utilities. Moreover, strategies are needed to discover correlations within and between data sets of independent origin. Wherever data sets possess intrinsic graph structure (e.g. of tree, forest or network type) or can be transposed into such, graphical support is considered indispensable. The Viator tool family presented during this demo depicts large graphs on the whole in a hyperbolic geometry and provides means for set-oriented context mining as well as for correlation discovery across distinct data sets at once. Its utility is proven for but not restricted to data from functional genome, transcriptome and proteome research. Viator versions are being operated either as user-end database applications or as template-fed stand-alone solutions for graphical networking.
Design Principles and Functionality (1)

1. Requirement: **Complex Graph Structures dictate Superior Capacity**
   - No. of Nodes >> $10^3$
   - No. of Edges >> $10^3$

   Network representations depict objects (nodes) together with their relationships (edges), whatever field of knowledge they may stem from. In practice, the number of edges and nodes in a network graph may vary considerably. Parametric, Boolean, verbal and other attributes of nodes and edges are used in assisting a user when navigating in and when reducing the network complexity in any dimension, by hiding the mass of query-irrelevant details.

2. Requirement: **An Alternative to Planar Depiction**
   - Approach Node Distribution in a Sphere
   - Inspired by Art “Fish-Eye Mode” (M.C. Escher)

   Multi-node networks are often perplexing if flattened into a plane area of limited extent. To circumvent the problem of too many edge intersections, nodes are being redistributed in a sphere. Network meshes close to the center of the sphere are displayed in high resolution, whereas network components located towards the periphery appear compressed, following a hyperbolic size decrease. Upon mouse-click, details of interest can be shifted, rotated and zoomed.

   The original idea and the powerful API of this art-inspired convenience were created by Tamara Munzner [1]. Several groups have taken over this ingenious approach and extended its functionality into different purpose-driven directions [2, 3], so did we. Our main goal was to union elements belonging together, at the same time representing distinguishable instances of the same object (e.g. allelic versions of a gene; alternative splice products of a transcript, mutated versions of a protein etc.). Therefore, we introduced an important feature [4] briefly outlined in requirement 3.
Design Principles and Functionality (2)

3. Requirement: A Flexible but Consistent Parent-Child Scheme

• By Cross-Hierarchy Propagation of Relationships

Many real world issues reflect hierarchical structures and organization principles. If there is manifest at least one relation between items belonging to a certain level, the Viator ensures the propagation of the corresponding fact to the parent level in the hierarchy, were it persists unresolved. This particular feature enabled us to implement routines for far reaching comparative studies [5].

4. Requirement: Handling Connected and Unconnected Graphs and Graph Components

Complex networks frequently segregate into components. By the aid of the Viator utilities the user toggles the visibility of fictive or hidden connections between distant parts of the graph. Auxiliary root nodes are being created manually or by operating the forest option of the API.

5. Requirement: Reduction of Complexity in Any Dimension

• By Parameters, Attributes, Keywords, Features …
• By Sorting Functions and Colour Coding
• By Unite and Intersect Buttons
• By Set-Oriented Operations

Freedom of choice in operating the before mentioned selection/trigger criteria and settings, alone or in suggestive combinations, allows a user to create specific views on the data behind the edges and nodes. Hyperlinks to primary data sources with their resp. advantages connect of the software to common practice search, fetch and retrieval conveniences. Navigation history records as well as drag-and-drop functions help to meet the users’ cognitive interests, esp. in case of entire groups of nodes to be explored and thus for set-oriented operations.
Design Principles and Functionality (3)

6. Requirement: Correlation Discovery across Huge Independently Monitored Data Sets

• By Superimposing Networks and Trees

Complex systems (like genomes) embrace a variety of hidden interdependencies between their active elements. Partial reflections of such pairwise or group-bound relationships are implicitly contained in data sets stemming from systematic but methodically independent experimental studies, mainly from high-throughput technology based ones. By mapping data set inherent graph structures upon each other, the Viator provides an excellent aid to make transparent hidden correlations if they exist, or to visually prove their absence in the opposite case. Correlation discovery was successfully demonstrated for yeast data [6] by examining publicly available protein-protein interaction results [7] vs. DNA chip measurements of transcript copy numbers in cell cycle stimulation experiments [8].

7. Requirement: Usability Stand-Alone as well as DB Interactive

• Convenient Templates for External Use
• DB-Interfaces
• Data Links to Primary Sources

The Viator tool was initially developed as part of the GUI for an IBM DB2 based Life Science Computation Platform, to retrieve and to display gene-to-gene interrelationships. It has then been used successfully for partial result shipment purposes and for use apart from the stationary system. Afterwards, a series of suitable templates has been created, to provide a user with all prerequisites for feeding the Viator with private data of any nature. We encourage colleagues from any domain of science to taste the potency of the Viator software.
Fig 1  Correlation Mining in Yeast Data

Sets of yeast genes the products of which are known to undergo pairwise physical interactions (protein-protein interactions, data taken from [7]) and which at the same time show transcriptional co-regulation acc. to microarray-based mRNA copy number measurements [8, data normalized and hierarchically clustered] in yeast cultures under the influence of cell cycle regulators.

a) Good Correlation in a set of yeast genes functionally related to cell growth.

b) Bad Correlation in a set of yeast genes of unknown function.
Screenshots of Use Cases

Fig 2 The link structure of data sources provided by the European Bioinformatics Institute. Screenshot of a navigation-friendly network representation.

References:
4. Patents pending, 011152303.3-2201 and 01115234.5-2201 (European Patent Agency)