Temporal and relational data representation by graph morphing

Eloïse Loubier, Bernard Dousset
Institut de Recherche en Informatique de Toulouse
118 route de Narbonne
31062 Toulouse Cedex 9
Ø5 61 55 67 81, France
{loubier;dousset}@irit.fr

ABSTRACT
This article presents a tool namely VisuGraph, designed to support temporal data visualization by means of graph representation that offers users a synthetic view of knowledge, difficult to extract from a data corpus. The temporal dimension adds the possibility to situate past, but also present or future events and strategies.
In this context, we propose a graph-morphing algorithm in order to highlight major tendencies and significant data changes during different periods of time. The goal is to present an animation based on a graph of temporal data, with every period pooled, followed up by each period graph successively. Our approach is based on a space-time analogy like for a clock. For each period, we propose to design a temporal mark, represented in a chronological order in the extremities of the visualization frame, like the hours on a clock. Each element of data is placed near the temporal marks according to its presence during each period. Thus, visualization of the successive periods makes it possible to detect persisting or temporary data and to detect a chronology in the changes in relations.

Keywords
Visualization, relational analysis,, morphing, strategic position, spatio-temporal data placement.

ABSTRACT
Cet article présente l'outil VisuGraph, permettant la visualisation des données temporelles sous forme de graphe et leur analyse, difficilement effective sous forme de texte. La dimension temporelle permet d'identifier et analyser les organisations passées et actuelles des réseaux afin d'en déduire leurs organisations futures.
Dans ce contexte, nous proposons un algorithme de morphing afin de révéler les principales tendances des données relationnelles durant les différentes périodes étudiées. L'objectif est de réaliser une animation basée, dans un premier temps, sur un graphe de données temporelles, toutes périodes confondues, puis dans un second temps sur chaque graphe de période. Notre approche est basée sur l'analogie espace-temps, comme pour les horloges. Chaque période est assimilée à un repère temporel invisible, placé sur le pourtour de la fenêtre de visualisation, comme les heures sur une horloge. Chaque sommet est placé en fonction de son appartenance à chacune des différentes périodes. Ainsi la visualisation successive des différentes périodes permet de détecter les éléments persistants de ceux émergeant ainsi que les changements au niveau des différentes relations au cours du temps.

Mots-clés
Visualisation, analyse relationnelle, morphing, position stratégique, placement spatio-temporel des données.

1. INTRODUCTION

The problem of representing and analyzing temporal data is important in the task of knowledge visualization. The goal is to detect, to identify and to analyze the evolution of relational data. In this context, graphs are used to supplement knowledge extraction visualization of a great quantity of data and to provide the user a maximum of synthetic information, difficult to describe differently.
Moreover, the temporal dimension makes it possible to locate the events, the strategies, and the actions in the past (as a chronology), the present (temporal orientation), and the future (anticipation). Also, successive network organizations (collaborations, alliances, fusions, acquisitions, co-quotations, co-signatures, co-occurrences) are revealed. In this context, the VisuGraph tool offers visualization and interactive classification of relational data. In order to analyze actor or semantic network evolution, we propose to add to VisuGraph a morphing component that emphasizes significant tendencies. Firstly, global visualization of all temporal data is carried out. Next, a dynamic animation between successive time slice visualizations shows data evolution. Our approach is based on the clock principle and the temporal reference marks are fixed and only the needles are moving. We compare a time slice to temporal reference marks fixed in a chronological order on the window circumference. We place each node according to its presence in the various periods. Thus, by the space/time analogy, it is possible to distinguish the persistent elements and the temporary elements and to obtain a chronology of the structural changes in the relations.
The remainder of this paper is structured as follows. In section 2, we introduce VisuGraph tool. In Section 3, we focus on temporal data visualization and the morphing principle is explained and developed. In section 4 we present related work and we discuss our findings.

2. GRAPH DRAWING

Developed in java, VisuGraph is a graph drawing module of the strategic watch platform called Tetralogie (Dousset et al., 1988).

2.1 Data Treatment

VisuGraph uses data obtained from Tetralogie treatments. Data are extracted from corpuses containing documents from data bases, internet, CD-ROMS, theses, papers…

Tetralogie proposes different modules like:

- Treatments : synonymy and term counts for each field,
- Co-occurrence counts and data analysis,
- Interaction and data visualization using graphs: VisuGraph.

Information is synthesized in co-occurrence matrices, used in the various modules proposed by Tetralogie. The basic units of analysis are the term, the field (author, keywords, address, date …) and the document. A field is a basic preset beacon for semi-structured data, as for example author, date, addresses, organization. A field, can have just one value (newspaper) or have different values (author, keyword …).

Data can result from the crossing of two fields, sub-fields or groups of fields in order to obtain co-occurrence matrices. For each of these matrices, crossing between two entities reveals the metric value

of the bond between them. Whatever the entity type (author, newspaper,…), it is possible to cross three fields simultaneously. To consider the temporal aspect, the third dimension represents time.
Crossings between two entities are carried out over several homogeneous temporal segments (or periods), in order to analyze the changes induced in time like: absolute changes, relative changes, accelerations, implosions, clusters evolution,etc.
The example in figure 1 shows co-occurrence matrices between authors obtained by Tetralogie. Crossing between the same author give his publications count for each time slice (in bold in the tables below). Crossing between two different authors shows shared publications (co-authors).

PERIOD 1

	A	B	C	D
A	**5**	1	3	0
B	1	**2**	0	1
C	3	0	**8**	0
D	0	1	0	**1**

PERIOD 2

	A	B	C	D
A	**2**	1	0	1
B	1	**7**	3	3
C	0	3	**4**	1
D	1	3	1	**7**

PERIOD 3

	A	B	C	D
A	**1**	0	0	1
B	0	**7**	3	1
C	0	3	**3**	0
D	1	1	0	**2**

PERIOD 4

	A	B	C	D
A	**1**	1	0	0
B	1	**9**	3	2
C	0	3	**3**	0
D	0	2	0	**3**

Figure 1: Publication co-occurrence matrices for four authors {A, B, C, D}, obtained by Tetralogie treatment for four periods consecutively.

2.2 From co-occurrence matrices to a graph

The visualization of complex conceptual structures is a key component of support tools for many applications in science and engineering. A graph is an abstract structure that is used to model information. Graphs are used to represent information that can be modeled as objects and connections between those objects. Hence, many information visualization systems require graphs because they are easy to read and understand (DiBattista et al., 1999). The matrices generated by Tetralogie are visualized as graphs.
In VisuGraph, a graph G = (V, E), consists of two components: balanced nodes V and valued arcs E directed or not according to the analysis context. Nodes are generally represented as histograms and arcs are the connections between nodes.
The height of each histogram bar is proportional to the metric value of the node for the period considered. A color is assigned to each histogram bar, corresponding to one specific period. So it is possible to distinguish various node values and to compare data over the periods.
In this article, we focus on the most general class of graphs: undirected graphs, drawn with straight arcs. In a graph, not all the elements (nodes and arcs) have the same importance in the structure. To be able to identify the characteristics of each of these elements, it is essential to introduce visual variables. Therefore, we use color.
The higher the value of the arc between two nodes, the stronger the color used to represent it. The initial value relates to the maximum of the co-occurrence matrices. This functionality is adjustable through a graduated slider, making it possible to vary the color level associated to each arc intensity. To illustrate this representation, we represent the four entities {A, B, C, D} can be visualized as in Figure 1.

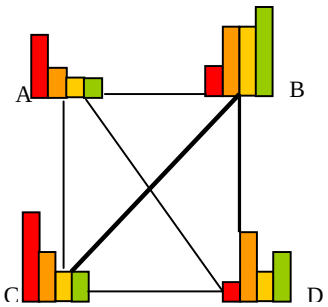

Figure 2: From matrices to graph

2.3 Force-directed placement

The attributes that define a good graph are called aesthetics and have been determined through statistical analysis in (Batini et al., 1985), (Purchase et al., 1996), and (Sugiyama et al., 1981).

The qualities which define a good graph are as follows:

- Minimal crossing – keeping the number of times that lines cross to a minimum,
- Minimal area – keeping the area that the graph takes up to a minimum by producing a compact graph,
- Minimal the sum of the arc lengths,
- A uniform arc length – keeping each of the arcs at the same lengths when possible,
- Minimal bends – keeping the number of times there is a bend to a minimum.

VisuGraph does not explicitly strive for these goals, but it does well at distributing nodes evenly and reflecting symmetry. The aim is to obtain a graph as planar possible. If it is not, the aim is to limit crossing between arcs.

Based on Fruchterman's work (Fruchterman et al., 1991) to draw undirected graphs, we used a spring-embedded model. The basic idea is as follows. "*To embed [lay out] a graph we replace the nodes by steel rings and replace each arc with a spring to form a mechanical system . . . The nodes are placed in some initial layout and let go so that the spring forces on the rings move the system to a minimal energy state*." (Eades, 1984).

In this paradigm, a graph is represented as a system of attractive and repulsive forces and the final graph produced has the smallest energy configuration.

In this way, nodes joined by strong arc attract. In opposition nodes joined by weak arc (or which are not joined) repel. A basic component of a force-directed graph is a need to find the graph with the lowest energy. This can be done using an iterative process.

The above figure shows the analogy between arcs and springs for the example of figure 1, period 1. It is a three step process of what happens in a force-directed drawing procedure and the force model here are springs. The first frame is not at very low energy but in the second frame the springs are more compressed. In the final frame, the graph drawing for the force model is shown.

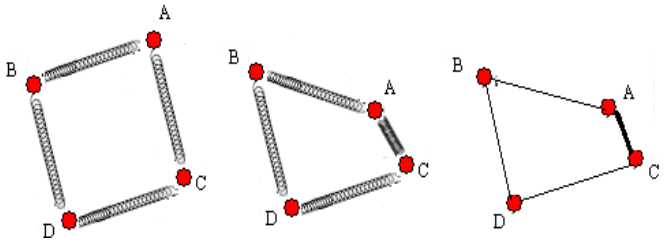

Figure 3: Analogy between arcs and springs for a graph drawing by force-directed placement

VisuGraph offers the possibility to parameterize using sliders attractive and repulsive forces. The more attractive the force level is, the more the nodes are attracted. It is the same for repulsive parameterization.

2.4 Visual functions and parameterization

Ergonomic improvement of the tool is based on the accessibility to the various functions by selection and the capacity to apply them. We carried out a parameterization frame in two distinct parts:

- functions which give to the user the possibility to specify the methods to apply to the representation (circular, optimized or reduced graph, icons representing nodes, nodes and arcs coloring, visibility of the nodes names, seeking a specific top, choice of screen color, etc.). With VisuGraph, sub-graph can be extracted, making it possible to study the specific environment of a node. Sub-graph is obtained by transitivity after one node selection. Thus, information is obtained as: the number of nodes connected for each transitivity degree, distance average, centrality. The increase (reduction) in the transitivity degree generates neighbor apperance (suppression) step by step.
- Parameters, which allows graph drawing to be controlled with the possibility to modify the transitivity depth (number 1 in figure 4), the coefficients of the force-directed placement (number 2 in Figure 4, the node appearance threshold according to their weight (number 3 in figure 4), the arc intensity slider (number 4 in figure 4), partitioning granularity (number 5 in Figure 4). Slider (number 6 in Figure 4) gives the possibility to activate graph morphing (in section 3.3). Parameterization frame dimensions are limited, allowing more space for graph drawing.

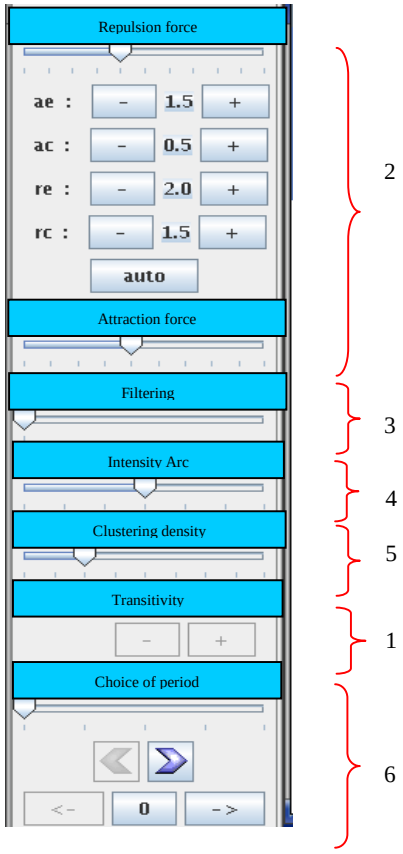

Figure 4: Parameterization frame translated from the original.

2.5 From graph to textual data

Graphs give a synthetic view of temporal and relational data. It is important to have the possibility to go back to non-synthetic data. For example, in a graph of the number of papers written by authors, nodes are authors (the height of the histogram bars depends upon the number of papers) and arcs are assimilated to their collaboration, when several authors write a paper together. In the graph, we can highlight author collaborations, number of papers written for each of them, if the authors write more papers or fewer.

In order to improve interactivity, it is important to include the possibility to extract textual data from the graph (Loubier et al., 2007). In the example of authors, users must be able to obtain papers written by a specific author. We developed this function. When a user clicks on an author node, all papers written by that person appear in a new window. As seen in section 2.1, the data come from various bases. In order to distinguish them, each textual information from a different base appears in a new window. In the end, there are as many windows as bases which contain papers of the selected authors.

3. DATA VISUALIZATION BY GRAPH MORPHING

It is easier to visualize data in the form of a graph than by studying textual data corpuses. If we consider the temporal dimension, it is important to choose the best representation to obtain the right synthetic information.

3.1 From static to dynamic representation

In the temporal context, if time slices are represented on the same graph, static graph interpretation could be false. Moreover, the total graph provides only general information about all periods of time pooled, which can not specify the role of each actor for each time slice. For example, in the static case, the graph could reveal a related class but if each time slice is visualized separately, this related class might never have existed. This mistake is based on cumulative information. In fact, all temporal data are represented in the same graph and give information for all periods together. If the data analyzed are not cumulative the interpretation in a global graph could be false. For example, a graph visualizing people and their residence city during ten years gives strong information in a static graph. One person could live in city “A” during the first year and in city “B” during the last year. A global graph interpretation will be “this person lived at the same time in cities A and B”. Time slice graphs will give the true information and the person will never appear as living in two different cities at the same time.

Another example, given Figure 5, shows articles published for four periods. On global graph, we can see a related class but this is not confirmed in each graph period. So, a global graph gives wrong information. Dynamic graphs are not fixed in time, but can evolve through local changes of the graph. They present changing information in a manner that makes the information easy to understand and it will assist users in efficient and effective completion of their task. The ability of dynamic graphs to depict a large quantity of information over time makes them very different from static graphs where all the information is presented at once.

On the one hand, a static graph can be the origin of wrong interpretation, so it would be interesting to use a dynamic graph. On the other hand, a static graph visualizes all the information at once and gives a global graph structure and tendencies. In fact, it would be necessary to find a solution which successively represents data in a static graph and next, in a dynamic sub-graph of one period.

Static graphs can give important information. In the publications example, it could be important to see if authors have written many papers over a whole duration. One author, who has written just one paper during each period, could not seem to have published a lot of papers. If we cumulate ten periods, his

number of papers will be higher than that of an emergent author who has written five papers on just one period and nothing else during other time slices. In contrast, individual period analyses suggest that the second author is more prolific than the first. So the global graph and the period graph both contribute to a good analysis. We have added this possibility to VisuGraph. First, all the information is visualized on a global graph, giving general information. Then, each period is visualized individually, to show the specificities and the main characteristics of each time slice.

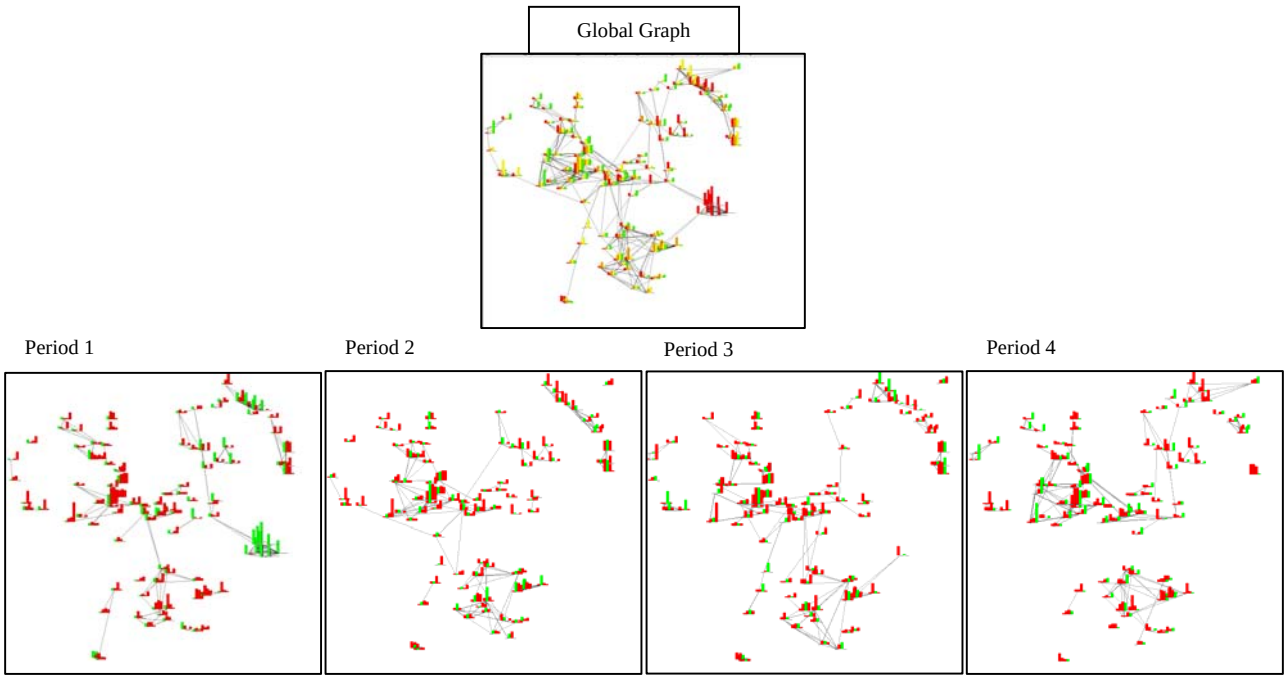

Figure 5: Global graph shows a related class. This information is wrong and it is refuted by period graphs.

3.2 Temporal placement

The qualities required for a good graph in section 2.3 concern general visualization and we propose to complete them in order to consider temporal aspect.

Temporal data must be placed in a strategic position, which reveals their temporal characteristics. For this reason, time slices are associated to invisible virtual nodes which are placed in a chronological order at the window perimeter. Each node has an initial temporal position, computed as a function of the reference marks.

The higher the metric value of a time slice, the closer it will be to the corresponding temporal mark.

Each node position is calculated as follows:

$$\left(\frac{\frac{1}{n}\sum \mathbf{repx}^{k} * \mathbf{m}_{i}^{k}}{\frac{1}{n}\sum \mathbf{m}_{i}^{k}} \; ; \; \frac{\frac{1}{n}\sum \mathbf{repy}^{k} * \mathbf{m}_{i}^{k}}{\frac{1}{n}\sum \mathbf{m}_{i}^{k}} \right)$$

$repx^k$ and $repy^k$ are the co-ordinates of temporal reference marks for period k.

m_i^k is the metric value for node v_i for the period k.

We developed a function to avoid node superposition. Two nodes can have identical metric values during all the various periods considered. According to our formulas, these two nodes have the same co-ordinates. That means that they are superimposed, preventing their distinction.
For each node co-ordinate, we check if the position is not already allotted to another. In this case, the co-ordinates are modified in order to avoid superposition.
Strategic placement of the nodes allows to be located them in time but also their persistence to be evaluated and their tendency to be deduced. Each node is joined initially to temporal reference marks. For a given time slice, the higher the value of a node, the higher the value of the arc between it and the reference mark. Hence, the user can choose to apply the force- directed placement algorithm.
Nodes are attracted by temporal reference marks. Thus, each part of the window, where the graph is drawn, corresponds to a part of time. So, it becomes easier to analyze the temporal data.
With temporal reference marks the user can see on a global graph the temporal tendencies and can visualize temporal data evolution. The position indicates in which periods the data belongs.
For example, fives data are analyzed for four periods in figure 6. Nodes are located according to temporal placement. For the sake of simplicity, we considerer that each node has the same metric value.

Nodes	Period
1	1
2	2
3	3
4	4
5	1,4
6	1,2,3,4

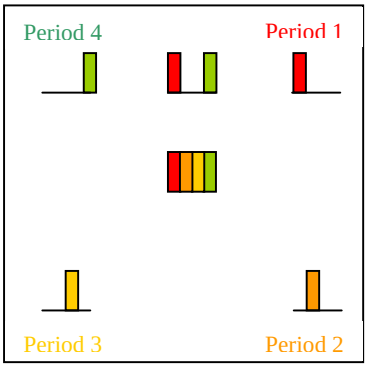

Figure 6: Graph according to temporal placement

3.3 Graph morphing

Morphing refers to "*an animated transformation of one image into another by gradually distorting the first image so as to move certain chosen points to the position of corresponding points in the second image*".(www.wordreference.com)
Morphing leads to an animation which transforms, in the most natural and fluid way possible, an initial graph into a final graph. Based on the space/time analogy, the aim is to detect tendencies, major actors or weak actors.
The aim of graph morphing is to carry out an intuitive reading of data evolution. Morphing is based on a global graph (as seen in section 3.1) and gives the possibility to fluidly animate the graph for each period. This function is activated by a slider, as seen in section 2.4.
Nodes are positioned in a strategic way, as considered in section 3.2. Graph structure can be observed on this global graph. Node positions give important information. If the node is situated in the window center, we can say it belongs to several periods. In contrast, if the node appears near a window edge, we can say it belongs to specific periods.
Specific temporal data characteristics are given by each histogram bar which reveals metrics values (as seen in section 2.2). The higher the metric nodes, the more the node plays an important role during the considered period. A global graph gives global information, so it is really important to analyze the graph for each period individually.

So, each period is visualized masking the global graph nodes and arcs which do not belong to the period considered.
Also by morphing, the graph contains only nodes and arcs of the period studied. Thus, the period graph is not centered and each node is situated near the temporal reference mark. During this step, each node position is memorized in a table.
Then, the force-directed placement algorithm is applied to the period graph, to minimize arc crossing and to obtain a graph as planar as possible (as seen in section 2.3). The user can interact with the system to choose the best representation, as seen in section 2.4. They can manually place and fix nodes on a specific position, chosen by themselves.
In this way, the user has total control of the representation. The graph obtained is organized to reveal various organizations, alliances, collaborations.
When the user decides to change period, they change slider (number 6 on figure 4). Nodes return to the position memorized in the table. Thus, the period graph returns to its position in the global graph. After this step, the global graph reappears to remind the user of the global graph structure, to preserve the user's mental map. The next period is visualized in the same way. Each node and arc which does not appear during the period is masked. The period graph is centered and force-directed placement algorithm is applied.
Successive animation visualizations of various periods, in a chronological order is similar to the space/time analogy of a clock. This animation reveals temporal data evolution. Thus graph morphing is a good compromise between maintaining the user's mental representation on the one hand and layout legibility on the other.
For an illustration of graph morphing, see Figure 7. This example is about the number of papers published per author. The global graph is in the middle and is the reference for the morphing application. Based on this graph, the user can see the different tendencies. Four periods are considered. The first period is designated by dark bars. The second period is designated by clearer color bars than the first. The third period is designated by clear colored bars. And the last period is designated in another color. Authors appear for just one period (they are represented by just one bar) but most belong to several periods simultaneously. Animation between the global graph and the first period graph reveals first period characteristics. All nodes not present during this period are masked. This graph shows the importance of the period inside the global temporal period (see graph 1). If the force-directed placement algorithm is applied to this period graph, we obtain graph (1.1). This graph is planar and we can see the different collaborations. For each author, histogram bar height shows the node's metric value for the period analyzed. For the first period (graphs 1 and 1.1) bar height reveals number of papers written per author during this period. In graph 1.1, we can see that the group present for this period only is represented by a large bar. So, we can deduce that these authors have written many papers during the first period. Nodes are ever represented by temporal histograms, so it is easy to compare each author's metric value between the first and the other periods.
Many authors write papers alone because they are not joined to others. In contrast we can see an author group, which commonly writes papers during only the first period (they are visualized only by one bar). Whereas graph (1) is complicated to analyze (number of arc crossings is high because of temporal data placement), graph (1.1) is planar and each alliance can be visualized. It is not important to consider the temporal reference mark in graph (1.1) because this graph is only about the first period.

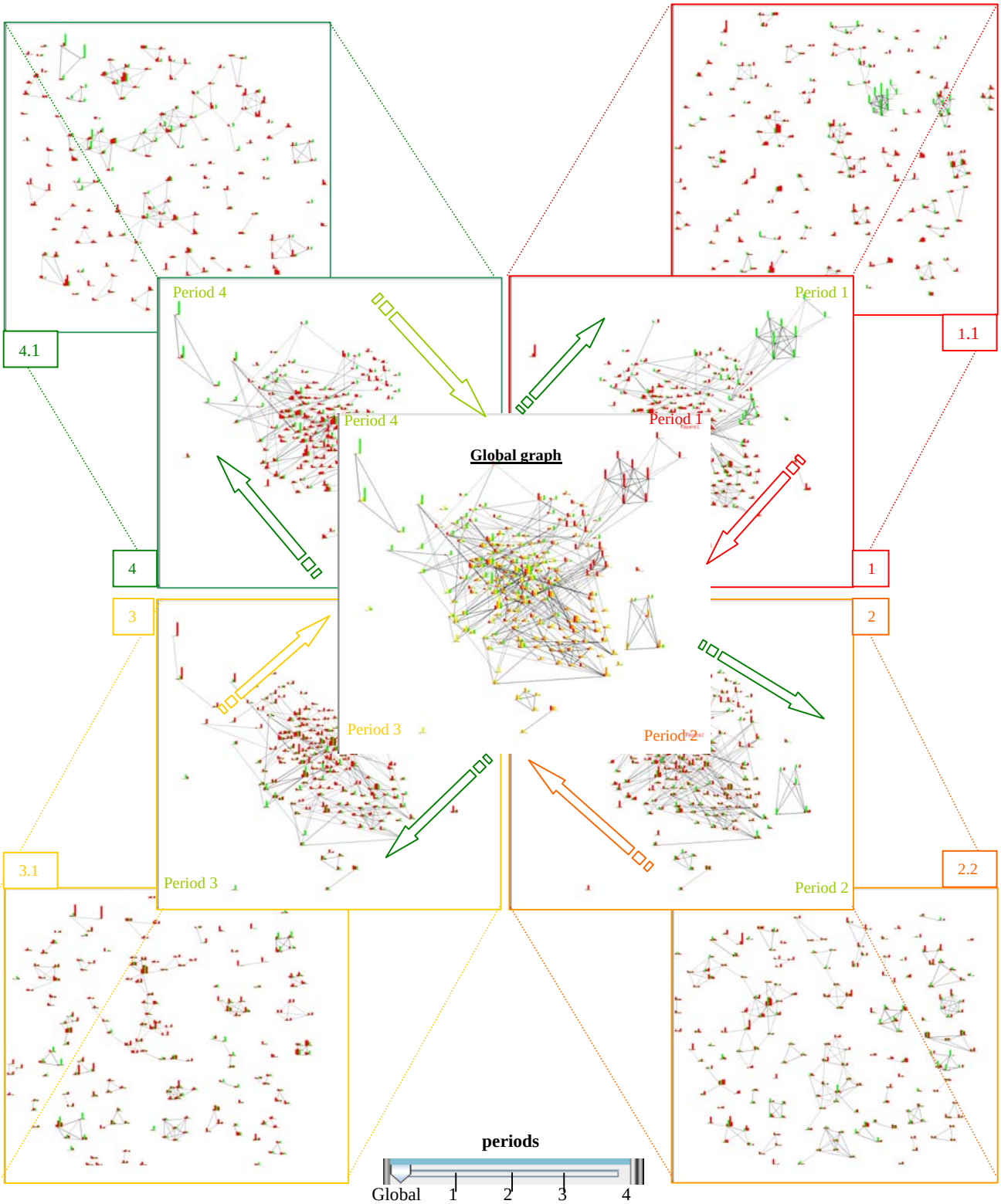

Figure 7: Graph morphing. Global graph is situated in the middle. Each arrow shows the possibility to change visualization (from global graph to period graph). Each period graph is joined to its graph whose force-placement algorithm has been applied.

When the user decides to see period 2, graph (1.1) returns to graph (1) by animation. After this step, the global graph reappears in order to keep the global structure in memory and to see the importance of the last graph (1) compared to the second period inside the global graph. Using the same method, all nodes absent during the second period disappear (graph 2). The force-directed algorithm is applied and each node gets a strategic position (graph 2.2). Compared to the first graph (1.1), we can see that some authors have disappeared. We can observe many collaborations between two or three authors, not in the global graph or in graph (1). Morphing creates an animation between each period, using the graph between each representation. From period 1 to the last period, the graph morphing is smooth. Each period is visualized successively with fluidity. We can distinguish permanent but also temporary graph actors. In graph 4 we can see emergent authors, who only appear during this period. We can deduce they will be present during future periods.
The morphing graph function makes it possible to analyze past events, but also current and especially future tendencies. Each period graph (graphs 1,2,3,4) is situated like hours on a clock and the animation direction is chronological.
Thus, temporal and relational data can be analyzed with temporal critera. Their changes are studied in a specific time context.
This function corrects global graph interpretation mistakes, visualizing each time slice precisely and individually.

4. RELATED WORK

Many program visualization tools have been proposed in the past. The aim of these tools is to obtain information in a form that is more readable than a big data corpus.
Created by (Mockus et al., 1999), SoftChange is a tool that extracts complexity, size, purpose and author role changes made to a program and summarize this information in textual web-based reports. Mockus et al. note that to study software changes it is essential to handle large and complex data sets. The volume, complexity, and lack of structure data overwhelm standard statistical analysis tools. SoftChange distinguishes and analyzes with quality temporal data but result restitution is global and not as specific for nodes as VisuGraph graphs.
Among the toolkits related to temporal data visualization, one of the most popular is TGRIP (Erten et al., 2004) making a good base for the visualization of graphs that evolve with time. TGRIP supports two additional attributes: weights of nodes and arcs and time-slice information. The time-slice is a label associated with each node represented in a snapshot of the state of the system. Each time-slice represents the state of the system at a given point in time. The comparison between the different time-slices shows the data evolution. This solution does not offer the possibility to see the global graph, but only period graphs.
Chen (Chen et al., 1999) focused on visualizing the hypertext field based on author co-citation patterns. This software generates an annual snapshot series of the domain structure, and a factor-referenced color-coding scheme to show specificities in the field. The major disadvantage of this approach is the absence of bench marks. Node positions change between different representation and disturb the user's mental map. In addition, the user does not have a global sight of the graph.
(Eick et al., 2002) visualize software changes using mostly traditional views, such as bar-graphs, pie-charts, matrix views, and cityscape views. A large number of different types of statistics can be displayed, allowing changes to the system to be viewed from many different perspectives.

The most significant strength of this system is that it is able to examine extremely large programs, up to several million lines of code. The main challenge is that links between data are not easy to analyze. Importance is given to node changes rather than to arc changes.
A similar technique called Revision Towers is used by (Taylor et al., 2002) that uses colored bars of varying thickness and height to represent the current size, changes and authors of a piece of code. These bars are animated over time to show the development of the software repository.
Rysselberghe and Demeyer used a simple visualization based on information in version control systems to provide an overview of the evolution of systems (Van Rysselberghe et al., 2004). Both to (Gall et al., 1999) and Wu et al. describe an Evolution Spectrograph (Wu et al., 2004) that visualizes a historical sequence of software releases.
(Brandes et al., 2003) presents a system for visualization of networks evolution in three dimensions (3D). The three dimensions are: width (left/right); height (high/low); depth (before/back). The representation is carried out in the form of layers that represent a section of time. Nodes corresponding to an entity remain in similar positions from one layer to another. Thus, the visualization of the evolution is based only on the arcs between the nodes and not on the structure of the graph for each considered period.
Presented by (Collberg et al., 2003), Gevol is a system that visualizes the evolution of software using a novel graph drawing technique for the visualization of large graphs with a temporal component. Gevol extracts information about a Java program stored within a CVS version control system and displays it using a temporal graph visualizer.
(Chen, 2004) presents a generic approach to detecting and visualizing emerging trends and transient patterns in the scientific literature. The contributions of the approach are the nature of an intellectual base algorithmically and temporally identified by emergent research-front terms. The value of a co-citation cluster is explicitly interpreted in terms of research-front concepts.
(Diehl, Görg, 2002) present a generic algorithm for drawing graph sequences. This algorithm is based on dynamic graph drawing in that it considers all graphs in the sequence (offline) instead of just the previous ones (online) when computing the layout for each graph of the sequence.
Each of these tools offers an important solution to analysis in partly temporal data. VisuGraph uses similar techniques and we have added the above properties to offer the user a complete tool. In this case, changes of nodes, arcs, structures, collaborations or alliances during different period of time are assessed.

5. CONCLUSION AND FUTURE WORK

This article describes the Tetralogie module namely VisuGraph which supports mining and representation of temporal and spatial data. In order to analyze temporal and relational data evolution, a graph morphing function is included. Temporal data visualization by graph morphing is based on a first global graph drawing, followed by an extraction of each time slice graph.
Each period of time is assimilated to a temporal reference mark, placed in a strategic position. Temporal data are represented by nodes which are situated according to temporal coordinates. The morphing involves dynamic animation between each time slice visualization, where nodes appear/disappear according to which period of time they belong in. The period graph, extracted from the global graph is organized to obtain a planar graph.
Graph morphing makes it possible to clearly see emergent, persistent or temporary actors. Based on the global graph, the user discovers what periods are the most significant and the most important, according to the number of period nodes.

Analysis can be done on global data (all data included) or on specific data. Moreover, the user can access textual information about a piece of data. They just have to click on the node which represents the data, and textual information will appear in a new window.
VisuGraph is an interactive tool where the user is free to decide which function to apply to the graph representation. They can also change the position and the color of the nodes. They can change force-directed function parameters. In this case, they can choose to come nearer connected nodes or to distinguish (by distance) non-connected data. Using the morphing graph parameter, the user can choose the period they want to see. They can decide to see the next period graph but also the last one.
By these function VisuGraph is a good tool for the visualization of synthesized temporal data. Graphic representations enable the user to obtain an image of structural information about big data corpuses. With graph morphing the user is able to rapidly reach a correct good conclusion about temporal data evolution from their spatial position.
Future work will involve improvement of the graph morphing. Nodes are placed in relation with temporal reference mark. With this solution, the global graph can lose the qualities required for a good graph (Batini et al., 1985), (Purchase et al., 1996), (Sugiyama et al., 1981) and crossing is not minimized. In order to overcome this limit, we must find a solution to apply force-directed placements according to temporal position. However, VisuGraph is limited by the number of data. Indeed, the higher data volume the less the efficiency. Thus, it would be advisable to improve the performance of the tool.
Moreover, graph animation lacks fine control because the nodes are strongly attracted by temporal reference marks, to the detriment of their initial relations. It would be interesting to find a compromise for a more flexible animation between two periods. Lastly, the graph animation is conditioned by the user's point of view. For example it can be directed towards detection of strong or persistent signals or of weak signals (appearances, disappearances, reorganizations of actors which can be potentially interesting). Thus, we must target the user's problems precisely, to highlight the topics of current interest. This could be done by a highlighted visualization of the structures preferred by the user.

6. REFERENCES

Brandes U., Corman S. Visual unrolling of network evolution and the analysis of dynamic discourse. InfoVis'02, *IEEE Computer Society Press*. Vol. 2, N°1, p. 40-50, (2003).

Batini C., Furlani L., Nardelli E., What is a Good Diagram? A Pragmatic Approach, *In Proceeding 4th Internat. Conf. on the Entity Relationship Approach* (1985).

Chen C., Carr L. *Visualizing the evolution of subject domain: A case study*. InfoVis'99, *IEEE Computer Society Press*. p. 449-452 (1999).

Chen C. Searching for intellectual turning points:Progressive Knowlarc Domain Vizua-lisation. Proceedings of the National Academy of Sciences of the United States of America, 101(suppl. 1), p. 5303-5316 (2004).

Collberg C., Kobourov S., Nagra J., Pitts J., Wampler K., system for graph-based visualization of the evolution of software, Software Visualization archive. Proceedings of the 2003 ACM symposium on Software visualization, pp 77 - ff, (2003).

Di Battista G., Tollis I., Eades P.,Tamassia R. Graph Drawing: Algorithms for the Visualization of Graphs, Prentice Hall, (1999).

Diehl S., Görg C. Graphs, They Are Changing -Dynamic Graph Drawing for a Sequence of Graphs. *Proceedings Graph Drawing*, pp. 23-30, Irvine, CA, USA, (2002).

Dousset B., Benjamaa T. *Tétralogie : logiciel d'analyse de données*, Conférence sur les systèmes d'informations élaborées : Bibliométrie – Information Stratégique – Veille technologique (1988).

Eades P. A heuristic for Graph Drawing. *Congressus Numerantium*, vol. 42, p. 149-160 (1984).

Eick S. G., Graves T. L., Karr A. F., Mockus A., Schuster, P. *Visualizing software changes*. Software Engineering 28, 4, 396{412. (2002).

Erten C., Harding P., Kobourov S., Wampler K., Yee G. Exploring the computing literature using temporal graph visualization. Conference on Visualization and Data Analysis (2004).

Fruchterman TMJ., Reingold EM. Graph *drawing by force_directed placement. Software* – Practice and experience, 21, p. 1129-1164 (1991).

Gall H., Jazayeri M., Riva C. Visualizing software release histories: The use of color and third dimension. In *Proceedings of the International Conference on Software* Maintenance, *IEEE Computer Society Press*, Oxford, UK, 99.108 (1999).

Loubier E., Bahsoun W. *La visualisation de données relationnelles au service de la recherche d'informations.* In : Conférence francophone en Recherche d'Information et Applications (CORIA 2007), Saint-Etienne, Association Francophone de Recherche d'Information et Applications (ARIA) (2007).

Mockus A., Eick S., Graves T., Karr, A. *On Measurement and Analysis of Software Changes*. IEEE Transactions on software engineering, VOL. XX (1999).

Purchase H. C., Cohen R. F.,James M. *Validating Graph Drawing Aesthetics*, In F. J. Brandenburg, editor, Graph Drawing , *Proceeding GD '95*, vol. 1027 of Lecture Notes Comput. Sci., pp. 435-446. Springer-Verlag (1996).

Sugiyama K., Tagawa S., Toda M. *Methods for Visual Understanding of Hierarchial System, IEEE Trans. Syst. Man Cybern.*, SMC-11, no. 2, 109-125 (1981).

Taylor C.M.B., Munro M. " Revision Towers" in *Visualizing Software for Understanding and Analysis. Proceedings. First International Workshop on Publication*, pp. 43- 50 (2002).

Van Rysselberghe F., Demeyer S. Studying. Software evolution information by visualizing the change history. *In Proceedings of the 20th International Conference on Software Maintenance*, IEEE Computer Society Press, Chicago, Illinois, USA. (2004).

Wu J., Spitzer C. W., Hassan A. E., Holt R. C. Evolution spectrographs: Visualizing punctuated change in software evolution. In Proceedings of the 7th International Workshop on Principles of Software Evolution, IEEE Computer Society Press, Kyoto, Japan, K. Inoue, T. Ajisaka, and H. Gall, Eds., 57.66, (2004).