DOI: 10.1111/cgf.12512

COMPUTER GRAPHICS *forum*
Volume 34 (2015), number 1 pp. 277–288

Visualizing the Evolution of Communities in Dynamic Graphs

C. Vehlow, F. Beck, P. Auwärter and D. Weiskopf

VISUS, Visualisation Research Center Stuttgart, University of Stuttgart, Germany
{corinna.vehlow, fabian.beck, Daniel.Weiskopf}@visus.uni-stuttgart.de, auwaerpk@student.uni-stuttgart.de

Abstract

The community structure of graphs is an important feature that gives insight into the high-level organization of objects within the graph. In real-world systems, the graph topology is oftentimes not static but changes over time and hence, also the community structure changes. Previous timeline-based approaches either visualize the dynamic graph or the dynamic community structure. In contrast, our approach combines both in a single image and therefore allows users to investigate the community structure together with the underlying dynamic graph. Our optimized ordering of vertices and selection of colours in combination with interactive highlighting techniques increases the traceability of communities along the time axis. Users can identify visual signatures, estimate the reliability of the derived community structure and investigate whether community evolution interacts with changes in the graph topology. The utility of our approach is demonstrated in two application examples.

Keywords: information visualization, graph visualization, dynamic graph, community structure, clustering

ACM CCS: H.5.2 [Information Interfaces and Presentation]: User Interfaces Graphical user interfaces (GUI)

1. Introduction

Relations among objects are usually modelled as a graph consisting of a set of vertices connected by a set of edges. Graphs that represent real systems such as social networks are often not uniform, rather is the distribution of edges inhomogeneous. Hence, they consist of clusters, that is, highly interconnected sets of vertices, whereas the density of edges between these groups is low. The analysis of such structural subunits, also called communities, is of high importance to understand structural and functional properties of the data. In social networks, for instance, these communities capture highly connected circles of friends, co-authors or collaborators. The identification of communities often gives insight into the social system and helps abstract from individuals.

In many application domains, relations change over time, which also affects the community structure of the network. Analysing the evolution of communities can help determine shifting structural properties of a dynamic network. In social networks, communities evolve as individuals change their roles, social status or interests. An important goal of social network visualization therefore is to facilitate the discovery of communities and patterns that govern their evolution.

Existing visualization approaches for dynamic community structures often suffer from overdraw and edge crossings. These problems make it hard to analyse the evolution of communities over time and hence to identify community evolution phenomena, such as merging of communities. Moreover, these visualizations usually do not show the underlying dynamic graph structure, which makes it hard to understand the origin of the derived community evolution or to estimate its reliability. There are also several approaches that visualize dynamic graphs, but no approaches that show both, community and graph evolution, in a single image.

In contrast, our new approach supports the analysis of the dynamic community structure together with the graph structure by overlaying a dynamic community representation with node-link diagrams of the graphs. We decided to overlay both representations instead of using multiple linked views because using the latter approach vertex representatives in both views can only be allocated using additional visual aids (e.g. labels or brushing and linking) and with long eye movements. Our approach avoids these issues of matching through overlay and further minimizes the aforementioned problems of long eye movements as well as overdraw and edge crossings systematically. Furthermore, in contrast to existing visualization approaches for dynamic communities, we apply an optimization method for colour assignment. We discuss visual signatures of typical patterns in the evolution and how they can be identified using our visualization approach and the implemented interaction techniques. Hence, our main contributions include:

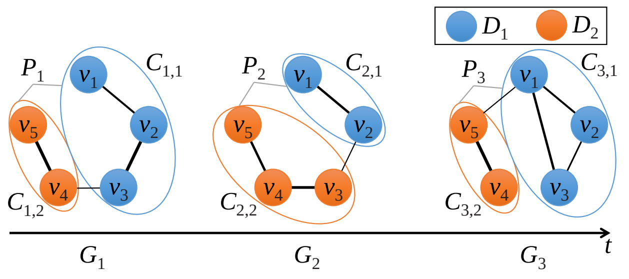

Figure 1: *Example of a dynamic graph $\mathcal{G}$ with $T = 3$ time steps and $L = 2$ dynamic communities. The edge weights w_{e_j} are mapped to the width of the links.*

- the combined visualization of the community evolution and the dynamic graph using a time line,
- the sorting of communities and vertices per time step,
- the colour assignment for dynamic communities and
- an extended discussion of visual signatures.

Two application examples demonstrate how our visualization approach brings together the community evolution and the graph structure to gain relevant insights into the topological evolution of the data.

2. Related Work

Dynamic graphs are often visualized using animation, timeline-based static visualizations [BBDW14] or hybrid approaches such as DiffAni, which combines the first two with a difference representation [RM13] Timeline-based approaches, so far, have only looked at the transition of graphs and considered the community structure as stable [BVB*11, GBD09]. In contrast, for animated diagrams, the evolution of communities has already been visualized: in addition to colour-coding vertices, the drawing space can be partitioned into (not necessarily connected) regions [HKV12, MKH12], convex (potentially overlapping) shapes [FT04, KG06] or nested boxes displaying a cluster hierarchy [MELS95, RPD09]. However, animation can lead to high cognitive load [APP11]; it is difficult to follow multiple community transitions happening at the same time and to track communities or individuals across longer periods. These are the main reasons why we avoid using animation in our approach, but use a timeline-based representation.

There also exist approaches that visualize just the evolution of communities (without considering an additional dynamic graph) using a timeline-based representation [FBS06, OMB*07], [RTJ*11, RB10]. Vertices are usually positioned on a vertical axis and grouped by community containment for each time step individually. These vertically aligned vertices are then plotted from left to right onto a timeline. In some approaches, only the transitions between time steps are visualized using straight links [RTJ*11, SMM12] or splines [RB10]; other approaches depict the communities or vertices using nodes [FBS06, OMB*07]. The work by Reda *et al.* [RTJ*11] is one of the few examples that apply an explicit sorting strategy: they position large and active communities closer to the top based on an influence factor. Sallaberry *et al.* [SPB10] derive a one-dimensional ordering of vertices by minimizing the distance of the representatives of the same vertex across different time steps.

Besides the evolution of network communities, there are visually similar approaches for the comparison of clustering results or set containment in general. Kosara *et al.* [KBH06], for instance, visualize categorical multi-dimensional data using so-called parallel sets, where each categorical dimension is laid out along one axis and categories are represented by boxes laid out along the other axis. The categorical dimensions are connected to show the overlap of data points between any two categories. Similar visualization techniques have been developed to compare multiple groups of clustered data [LSS*12]. Pilhöfer *et al.* [PGU12] discuss sorting strategies for comparing hierarchical clustering results.

Nearly all described approaches either focus on dynamic graph drawing or the temporal evolution of communities; only Sallaberry *et al.* [SPB10] systematically combine both: the community evolution timeline is used for quick navigation through an animated node-link diagram. In contrast to their approach that requires animation and different views, our visualization approach shows both community evolution and dynamic graph within one view using a layered approach. Also the sorting problem has only been addressed by few works [RTJ*11], [SPB10] based on impact factor and optimizing closeness. However, we think that reducing crossings of community transitions could be a better optimization criterion. Besides, our approach uses a special ranking of transitions and an advanced colour assignment to better deal with the remaining and inevitable crossings.

3. Data Model and Community Detection

Our visualization approach is based on modelling the data as a sequence of graphs. Communities—groups of vertices of the graphs—are connected to dynamic communities across time steps. In particular, we model an undirected weighted graph $G = (V, E)$ as a set of vertices V and a set of edges $E \subseteq V \times V$, where each edge $e_j \in E$, $1 \leq j \leq m$, $m = |E|$ has a weight $w_{e_j} \in \mathbb{R}$. A dynamic graph $\mathcal{G} := (G_1, \ldots, G_T)$ is defined as a sequence of T subsequent graphs. The community structure of the vertices is summarized in partition set $\mathcal{P} = \{P_1, \ldots, P_T\}$, where each partition P_t consists of disjoint sets of vertices $C_{t,k}$, which are called communities (we use unique consecutive identifiers k for labelling communities such as in Figure 8). Connecting communities across time steps, a set of L dynamic communities $\mathcal{D} = \{D_1, \ldots, D_L\}$ can be derived, where each dynamic community D_l is a sequence of communities $C_{t,k}$ ordered by time, with at most one community for each time step t. We illustrate this basic data model as a node-link diagram for a small example in Figure 1.

The transition of community memberships of vertices between consecutive time steps can be described by an additional set of edges that we call transition edges $\mathcal{E} = \{\mathcal{E}_1, \ldots, \mathcal{E}_{T-1}\}$, where the number of the sets is equal to $T - 1$. Each set $\mathcal{E}_t$ contains transition edges $\epsilon_{t,i} = (v_{t,i}, v_{t+1,i})$ connecting a vertex v_i at consecutive time steps t and $t + 1$.

The community structure of dynamic graphs can be specified manually or derived using automatic approaches. In case of an automatic detection, two general approaches are available [For10]:

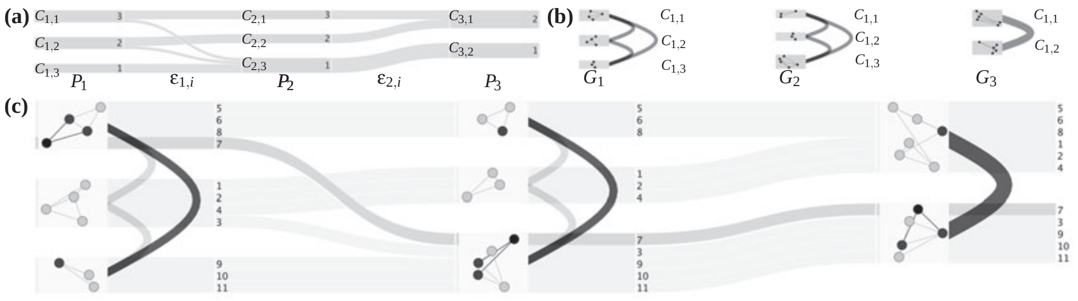

Figure 2: *Evolution of a dynamic graph with $T = 3$ and $L = 3$ dynamic communities D_l. (a) Community Evolution Layer showing the partitions P_t and the transitions $\epsilon_{t,i}$ between them. Communities $C_{t,k}$ are labelled with l according to the dynamic community D_l they are associated with. (b) Graph Layer showing the graph structure at individual time steps, where inter-community edges are aggregated and represented by curves. (c) Joined layers: the Graph Layer (b) is superimposed on the Community Evolution Layer (a); vertices v_i are labelled with i. Vertex v_7 was selected and hence the respective ribbon as well as the respective nodes at all time steps are highlighted in dark grey within the Graph Layer. Besides, also all inter- and intra-community edges, v_7 is involved in, are highlighted and the direct neighbours (v_6, v_8, v_9 and v_{10}) are slightly highlighted at each time step individually.*

First, in two-stage approaches, the community structure is derived for each G_t independently and relationships between communities at different time steps are inferred successively using community tracking methods [GDC11, TFSZ11]. Second, evolutionary community detection approaches incorporate both, the graph structure of the current and of previous time steps, to determine the evolving community structure of $\mathcal{G}$. Which approach is more suitable depends on the application context [For10].

Our visualization approach is independent of the choice of the community detection algorithm. Using evolutionary community detection, the dynamic communities $\mathcal{D} = \{D_1, \ldots, D_L\}$ are derived directly. If, in contrast, the two-step approach is used, they have to be derived by a community tracking procedure that takes the community partitions $\mathcal{P}$ of the vertex sets as input. To support the latter, we incorporate community tracking into our visualization approach. The key concept for the tracking is a notion of similarity between communities at different time steps. Similar to the threshold-based community tracking approach by Greene *et al.* [GDC11], we define that two communities C' and C'' are similar if their Jaccard similarity $\text{sim}(C', C'') = |C' \cap C''|/|C' \cup C''|$ exceeds a given threshold $\theta \in [0, 1)$. The threshold θ can be changed interactively within the interface of our prototype to find a suitable dynamic community structure $\mathcal{D}$—for less stable communities, a lower threshold is more suitable than for more stable communities. While the approach by Greene *et al.* allows many-to-many mappings between similar communities, we only connect pairs of communities across two time steps: if multiple pairs of communities are rated as similar, we select the pair having the highest similarity value. We sweep through the time steps and compare all $C_{t,k}$ of partition P_t to the last members of already partially detected dynamic communities D_l. Using this approach, some communities $C_{t,k}$ may not be assigned to any dynamic community D_l; also some dynamic communities do not cover all time steps and can have gaps.

4. Visualization Technique

The idea of our visualization approach is to show both the evolution of communities and the graph structure at individual time steps. To this end, we assemble the visualization from two superimposed layers: the *Community Evolution Layer* (Figure 2a, Section 4.1) and the *Graph Layer* (Figure 2b, Section 4.2). For the *Community Evolution Layer*, we have developed advanced strategies for the ordering of communities and vertices, for the assignment of colour and for the rendering of transition edges. For dynamic graphs including many time steps, the complete visualization gets too small to analyse individual time steps. Therefore, our tool supports zooming and panning. After introducing both layers, we discuss their interplay, in particular, visual signatures that point to interesting patterns in the data (Section 5).

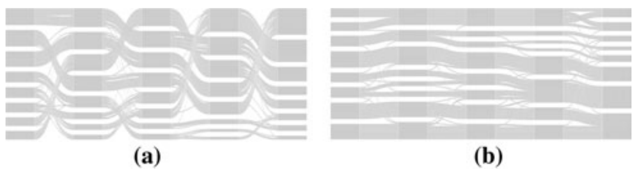

Figure 3: *Demonstration of the effect of ordering communities and vertices: (a) ordered by community size and vertex name; (b) ordered by crossing minimization.*

4.1. Community evolution layer

The dynamic community structure is visualized based on the flow metaphor—already used for group evolution in other contexts [Min69], [PXY*05], [RHF05]—employing a timeline from left to right where communities literally flow from one state to the other, as illustrated in Figure 2(a). The impression of flow is created by grouping the vertices of a community and connecting the same vertices across time steps by smooth curves. Technically speaking, for each time step t, vertices v_i of the graph are rendered as horizontal stripes of a particular height, arranged vertically such that they form blocks representing communities $C_{t,k}$. As a consequence, the heights of the blocks correspond to the size of each community $C_{t,k}$; the blocks are separated by a small gap to clarify that they are distinct. The communities in partitions P_t are arranged side by side, where vertices v_i of consecutive time steps t are connected by links according to transition edges $\epsilon_{t,i}$ to represent the changes in community membership.

4.1.1. *Community transitions*

Each transition edge $\epsilon_{t,i}$ is drawn using a Bézier curve instead of a straight link. In doing so, we generate smooth, continuous transitions between successive communities applying the Gestalt principle of continuity [Kof35]. To support the user in following transition edges crossing each other, we render the transitions in a specific order so that small groups of transitions are not covered by larger ones. Transition edges $\epsilon_{t,i}$ are ranked for each $t \in \{1, \ldots, T-1\}$ first and the Bézier curves are then rendered in descending ordered by their rank $r_{t,i}$. Our ranking algorithm assigns high ranks $r_{t,i}$ to transition edges $\epsilon_{t,i}$ that build up groups of transitions $\mathcal{E}$ between communities $C_{t,k}$ belonging to the same dynamic community D_l: $r_{t,i} = 100 \cdot |\mathcal{E}|$. Transition edges $\epsilon_{t,i}$ that are part of a group of transitions connecting two communities belonging to different dynamic communities D_l are assigned medium ranks: $r_{t,i} = |\mathcal{E}|$. The ranks are proportional to the size of these groups of transitions $\mathcal{E}$. For all single transition edges $\epsilon_{t,i}$, small ranks are assigned, where $r_{t,i}$ depends on the vertical distance between the positions of v_i at t and $t+1$: $r_{t,i} = 1 - |y(v_{t,i}) - y(v_{t+1,i})|/\Delta y_{\max}$, where the function $y(v_{t,i})$ returns the vertical position of a vertex $v_{t,i}$ at a particular time step t and $\Delta y_{\max}$ describes the maximal vertical distance between vertices. Using this ranking, larger groups of transitions are drawn first, and single transitions that bridge the highest vertical distance are drawn last. Moreover, we render the Bézier curves with halos to further improve the readability of crossing transition edges, in particular for crossing edges having the same colour.

4.1.2. *Ordering of communities and vertices*

The visual traceability of communities in the visualization also depends on the ordering of the communities and contained vertices. As in node-link diagrams, in particular, many crossings of transition edges $\epsilon_{i,j}$ reduce readability: Figure 3 illustrates this by comparing a simple community ordering by size (and vertex name within the communities) to one with minimized edge crossings; here, the number of crossings is reduced from 6434 to 541. The minimization of edge crossings in layered node-link diagrams has been already studied, for example, as part of the Sugyama layout [STT81]. However, in our case, vertices cannot be moved independently as the community structure needs to be preserved. Holten and van Wijk [HvW08] discussed crossing minimization for two clustered sets of vertices; we extend their approach to multiple time steps. An alternative optimization goal would have been to optimize for closeness of communities and vertices as proposed by Sallaberry [SPB10]. We decided against this criterion because it seems that crossing edges are more harmful than long edges with respect to traceability.

Like other linear arrangement problems in context of layered graphs and partitions [HvW08, SPB10], our problem is NP-complete because the NP-hard problem of one-sided crossing minimization [MUV01] is contained. Hence, we implemented a heuristic solution that is, in particular, based on a barycentre approach [STT81]. We sweep through the layers and optimize the order of communities $C_{t,k}$ and vertices v_i of each two consecutive time steps: the layer that has already been sorted is kept fixed, while the order of communities and vertices is optimized in the other layer in a one-sided crossing minimization. The applied optimization procedure first sorts the communities and then considers the order of the vertices within the communities. To further improve the result of this heuristic optimization, we sweep forth-and-back or back-and-forth (randomized) through the time steps until the number of crossing cannot be reduced further. The full optimization procedure is repeated with randomized starting orderings for communities and vertices. Finally, the solution with least crossings is selected.

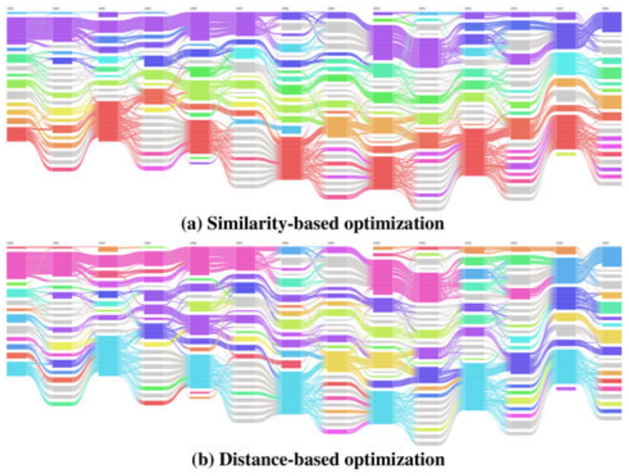

Figure 4: *Colour assignment for dynamic communities D_l based on two contrary optimization criteria.*

4.1.3. *Vertex colouring*

Using colour is not required for our approach, but colour can encode further useful information on the vertices and communities. Our visualization approach, in particular, allows users to switch between three different colouring schemes:

- based on the dynamic communities D_l (Figure 4, communities not belonging to any dynamic community are grey),
- based on the stability of a dynamic community over time (Figure 5 b) and
- based on the vertex stability over time (Figure 8).

For colouring based on communities (not vertices), transition edges $\epsilon_{i,j}$ might connect communities of different colour. The respective curves are therefore rendered with a colour gradient, where the centre of the colour gradient is shifted more towards the next time step to indicate the direction of time.

Dynamic Community Colouring: Previous visualization approaches for community structures in static as well as in dynamic graphs often represent (dynamic) communities by different colours. Which colour is assigned to which (dynamic) community, however, is arbitrary in those approaches. In contrast, we investigate

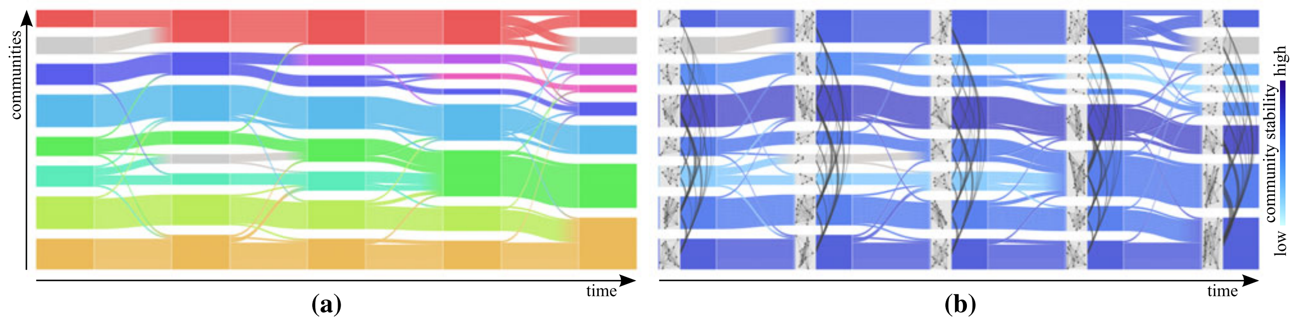

Figure 5: *An evolving community structure of a dynamic graph over five points in time. Communities are visualized as blocks of vertices and connected by curves to visualize transitions between communities. The communities are coloured based on (a) detected sequences of communities (grey if not part of any sequence) or (b) the stability of communities. While (a) just depicts the evolution of communities, (b) overlays node-link diagrams revealing the underlying dynamic graph structure.*

optimization approaches for an informed colour assignment. Others have already addressed the problem in a different context: for instance, Zang *et al.* [ZFMAQ13] optimized the colour assignment for the arcs in their Circos diagram; Jianu *et al.* [JRFL09] optimized the colour assignment to edges such that overlapping or close links in a node-link diagram have perceptually opposing colours; and Chuang *et al.* [CWM09] used optimization in the CIELAB colour space to identify colours that are easy to distinguish in generic visualizations and that reduce the energy consumption of displays. However, to the best of our knowledge, none of the available visualization approaches for dynamic communities assigns colours using an optimization strategy.

Our approach optimizes the colour assignment with respect to one of two contrary optimization criteria (Figure 4). The similarity-based criterion tries to assign similar hues to similar dynamic communities D_l (i.e. dynamic communities that exchange many vertices). In contrast, the distance-based criterion tries to assign different hues to dynamic communities D_l that are close in space to increase the hue contrast. The optimization problem of assigning different hues to all D_l can be considered as a circular arrangement problem, where the nodes (here, D_l) of a graph have to be positioned on a circle to minimize the total angular edge length. This problem, also known as minimum circular arrangement problem, is NP-complete [GJ79]. To find a heuristic solution for this problem, we calculate the arrangement of dynamic communities D_l on the hue circle by extending the barycentre-based method of Gansner and Koren [GK07]. We incorporate edge weights to achieve that pairs of D_l whose relation has strong weights are positioned close to each other on the circle. This method is based on polar coordinates, where the node positions are rescaled before the colours are assigned such that they are equally distributed on the circle.

We developed six weighting schemes—three for each optimization criterion—describing the edge weights for all pairwise combinations of D_l. Figure 4 illustrates the colouring results of two such weighting schemes. To find out which of the colourings users find more aesthetically pleasing, we performed an electronic survey at our institute. In the survey, we asked the participants to rank the six colourings presented in random order using a five-point Likert scale (the higher the rank, the higher the participant's preference) and to choose the colouring they like most. With high significance (t-test, $t = 3.85$, $p = 0.001$), participants preferred similarity-based optimization results ($\text{aver}_{\text{rank}} = 3.56$, $\sigma_{\text{rank}} = 0.94$) over distance-based ones ($\text{aver}_{\text{rank}} = 2.47$, $\sigma_{\text{rank}} = 0.94$). Besides, 20 of 22 participants choose one of similarity-based optimization results as their favourite. This result suggests that the vertical gap between the communities is sufficient to differentiate between them. Besides, the similarity-based approaches are more aesthetically pleasing. Therefore, we decided to use a similarity-based optimization for the remainder of this paper.

Community Stability Colouring: Dynamic communities exchange members with each other. The stability of the dynamic communities describes the degree of exchange, where high stability values represent a low exchange. We define the stability of a dynamic community D_l as the mean Jaccard similarity (Section 3) between each pair of subsequent constituent communities according to Greene *et al.* [GDC11]:

$$\text{stab}(D_l) = \frac{1}{T-1} \sum_{n=1}^{|D_l|-1} \text{sim}(C(D_l, n), C(D_l, n+1)),$$

where function $C(D_l, n)$ returns the n^{th} community of dynamic community D_l. To distinguish the stability of gapped or partial sequences of communities from non-gapped or full sequences spanning the complete time frame, we divide the term by $T-1$ but not $|D_l| - 1$. The highest similarity $\text{stab}(D_l) = 1$ is reached only if all communities belonging to D_l are identical and $|D_l| = T - 1$. We use a colour map created with ColorBrewer [Col11] that maps $\text{stab}(D_l)$ to a range from bright to dark blue as depicted in Figure 5(b).

Vertex Stability Colouring: The third colouring approach considers stability from the perspective of vertices. The stability of a vertex v describes its tendency to belong to similar communities over time. We define it based on Jaccard similarity of communities, similar to Takaffoli *et al.* [TFSZ11]:

$$\text{stab}(v) = \frac{1}{a-1} \sum_{n=1}^{a-1} \text{sim}(C(v, n), C(v, n+1)),$$

where function $C(v, n)$ returns the n^{th} community to which v belongs and a is the number of communities v is part of, that is,

$a \leq T$. If the dynamic community to which a vertex belongs does not change over time, $\mathrm{stab}(v) = 1$ is maximal. To visually discern the vertex from community stability, here, we use a colour map from bright to dark green. Since $\mathrm{stab}(v)$ is a property of the vertices themselves, all horizontal stripes and transition edges representing the vertex can be coloured using the same value as shown in Figure 8.

4.2. Graph layer

Visualizing only community evolution, users cannot analyse why the community structure changed and whether changes are due to noise or drastic changes in graph topology. To allow users investigate the reliability of the derived dynamic community structure, the *Graph Layer* (Figure 2 b) can be superimposed on the *Community Evolution Layer*. Each individual graph G_t is visualized as a node-link diagram on top of the community structure of time step t (see Figure 2 c). The node-link diagrams contain two types of edges: intra-community edges (edges of the intra-community graph, i.e. within the communities) and inter-community edges (edges connecting vertices from different communities). In both cases, the width of the links encodes the edge weights.

Each community contains a rectangular area where the intra-community graph is plotted. Within this area, small circular nodes representing vertices are connected by straight lines according to the graph structure. The vertical position of the nodes is determined by the ordering of communities and vertices (Section 4.1.2). However, nodes can be moved horizontally to optimize the layout. Therefore, we employ a force-based Fruchterman–Reingold model [FR91] and adapt it by restricting node movements to the horizontal axis.

As an important property of the intra-community graph, we visualize its relative density in the background colour of the enclosing rectangle: if dynamic community colouring is applied, the saturation of the original background colour is varied, otherwise the relative density is mapped to a grey scale. The relative density of a community $C_{t,k}$ is based on the intra-cluster density of the community subgraph (internal degree: $d_{int}(C_{t,k})$), that is, the ratio between the number of internal edges of $C_{t,k}$ and the number of all possible edges in $C_{t,k}$, and the inter-cluster density (external degree $d_{ext}(C_{t,k})$), that is, the ratio between the number of edges connecting vertices of $C_{t,k}$ with the rest of the graph G_t [For10]:

$$\rho(C_{t,k}) = \frac{d_{int}(C_{t,k})}{d_{int}(C_{t,k}) + d_{ext}(C_{t,k})}.$$

The relative density within a community may be sufficient to explain why a community dissolves or splits up. The intra-community graph can therefore also be hidden on demand.

For depicting the inter-community edges, straight links could have been used as well to connect the vertices, but they would clutter the display considerably. For this reason, we decided to aggregate those edges based on the community structure: all edges linking vertices from the same pair of communities are aggregated to a single edge between the two communities (edge weights are summed). These aggregated inter-communities edges are then visualized connecting the rectangular areas of the communities by curved links as depicted in Figure 3(b). Both types of edges—intra-community and inter-community edges—can be filtered interactively using sliders to concentrate on strong edges only.

4.3. Interactive highlighting

To support the user in analysing community lifetime phenomena and identifying their causes, our visualization approach incorporates standard brushing-and-linking techniques that help the user focus on particular nodes or communities of the dynamic graph. The selection of a vertex v_i within the *Community* or *Graph Layer* will result in highlighting all visual objects associated with this vertex over the complete period of time, that is, the ribbon and the respective nodes representing v_i within the node-link diagrams. This is achieved using a more saturated colour for selected elements and a less saturated colour for all others (see Figure 2c). In the *Graph Layer*, in addition to the selected node itself, also its neighbours as well as its relations will be highlighted showing them in medium saturation. The highlighted relations include the node's inter-community links and all aggregated intra-community links (curves) the vertex contributes to. The selection of a complete community $C_{t,k}$ will highlight all vertices v_i it includes as described before.

5. Visual Signatures

Visual signatures—sometimes also referred to as visual patterns—are typical patterns of a visualization that identify significant phenomena in the data. For community evolution, some signatures have already been discussed elsewhere: formation, growth, contraction, split and dissolution of communities and merge with other communities [APU09, For10, LSC*07]. However, our combined visualization of the community evolution and the graph structure introduces new signatures that enable a more informed and in-depth analysis of the community evolution as we discuss in the following.

The low-level evolution of communities alone can be largely described by a small number of visual signatures that are illustrated in Figure 6 (two-step signatures) summarized from literature [APU09, CLT*11, For10, LSC*07]: *stable* (or *continued*) communities form thick continued lines; at a *split*, a single community falls into at least two parts and a *merge* is a reverse *split*; if vertices of a community are not continued, they *extinct* or, on the contrary, they *emerge*. These signatures describe pure scenarios—often, in real data sets, mixed scenarios or slightly deviating patterns can be observed. Visualizing community stability, as possible with our approach using colour mapping (Section 4.1.3), supports estimating the degree of stability. While patterns can be detected automatically and highlighted [CLT*11], we decided against this option to not overload the diagram and to not limit the users' interpretation of patterns.

Other signatures cover several time steps, such that dynamic communities have a *growth* or *contraction* phase [For10]. We also observed further signatures, not yet described in literature, that are shown in Figure 6 (n-step signatures): a *re-merge* merges together the same vertices that have been split before and a *re-split* is the

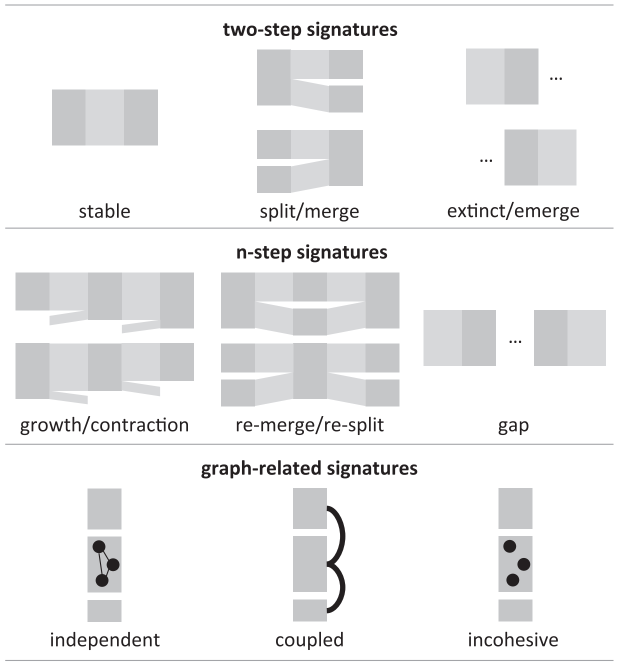

Figure 6: *Visual signatures in community evolution.*

analogous operation for a split; a *gap* is the extinction at any time step t' and emerge of the same vertices at $t'' > t' + 1$ as a community. For detecting these n-step signatures, it is particularly helpful that we compute dynamic communities D_l and assign the same colour to each community $C_{t,k} \in D_l$. For instance, it is much easier to check that a *re-split* really falls into the same communities that existed before based on those colours.

Layering graph information on top of communities, which is a unique feature of our approach, allows for identifying additional signatures. The graph structure provides details on the communities showing how coupled they are between each other and how cohesive they are themselves (Figure 6, graph-related signatures): an *independent community* is not coupled to other communities by inter-community links, but its vertices form a cohesive and dense graph structure; in contrast, a *coupled community* is highly connected to other communities, whereas vertices of an *incohesive community* are not reasonably connected among each other.

The graph-related signatures—in particular, when combined with the time step signatures—open up a new perspective on analysing community evolution: while community assignment is an either/or decision, the dynamic graph layer allows for taking a "look behind the scenes" and reveals uncertainties with respect to community assignment. For instance, it could make a considerable difference if, after a split, the new communities are coupled or independent; in the first case, the split has much less relevance than in the second. When communities are automatically detected, the visualization might even be used for debugging or configuring the detection algorithm because graph structures and communities can be connected.

6. Application Examples

To illustrate the usage of the visualization approach in general and of the visual signatures in particular, we discuss two application examples: the evolution of groups of national soccer teams playing against each other and the evolution of communities in a social network of conference attendees. While the first scenario is based on a large data set and focuses more on the high-level community evolution, the second data set is smaller, yet it allows for studying graph-related signatures in more detail. We print all terms referencing visual signatures in bold font. In both scenarios, we have derived communities independently for each time step using the infomap community detection tool for undirected networks [RB08]; the tool does not require defining a certain number of clusters but fits the number automatically. Taking the graph structure and detected communities as input, our implementation of the visualization approach first detects the dynamic communities before it produces interactive graphics of the data sets. The communities and vertices were sorted using our optimization procedure (Section 4.1.2) using 100 randomized starting orderings. We used interaction and different configurations (e.g. different colour mappings) for exploring the data sets. The Supporting Information (available under: https://www.vis.uni-stuttgart.de/index.php?id=dyncomm) contains a more detailed description and images for the individual analysis steps as well as a video.

6.1. International soccer matches

The first data set summarizes world-wide international soccer matches of 219 national teams between 1992 and 2005. Two teams are connected if they played against each other in the respective year (8783 edges in total). Each of the 14 years forms an independent graph. Figure 7 shows the data set visualized with our approach using colour-coding by dynamic community to highlight community evolution. Dynamic communities D_l are labelled l in the figure. Considering the restricted image space in this paper, we have switched off graph details within the communities (a high-resolution image of Figure 7 including vertex labels is included in the supplemental material).

In general, the colour coding in Figure 7 reveals a number of **stable** communities across several time steps, for instance, the purple group in 1992 – 1997 (D_1). None of the communities, however, is preserved over the complete period of 14 years. Interactively investigating the most **stable** groups, we find out that they often represent continental groups of national teams; for example, the community D_1 summarizes African teams. Colouring as well as interactively highlighting communities also helps detect that **re-merges** happen in the data set after a number of time steps. For instance, the African group D_1 **re-merges** in 2000 after being **split** for 2 years, but also the period of 1993–1996 can be considered as a **re-merge** (Figure 7(1)). Graph-related signatures, however, indicate that African subgroups are still **coupled** among each other in those years of a **split**, for example, 1994/1995 or 1998/1999; there still seems to be a considerable connection although they were assigned to different communities by the algorithm. Also the relative density $\rho(C_{t,k})$ of the communities decreases in those years (the saturation of the dynamic community colour in the inter-community graph area

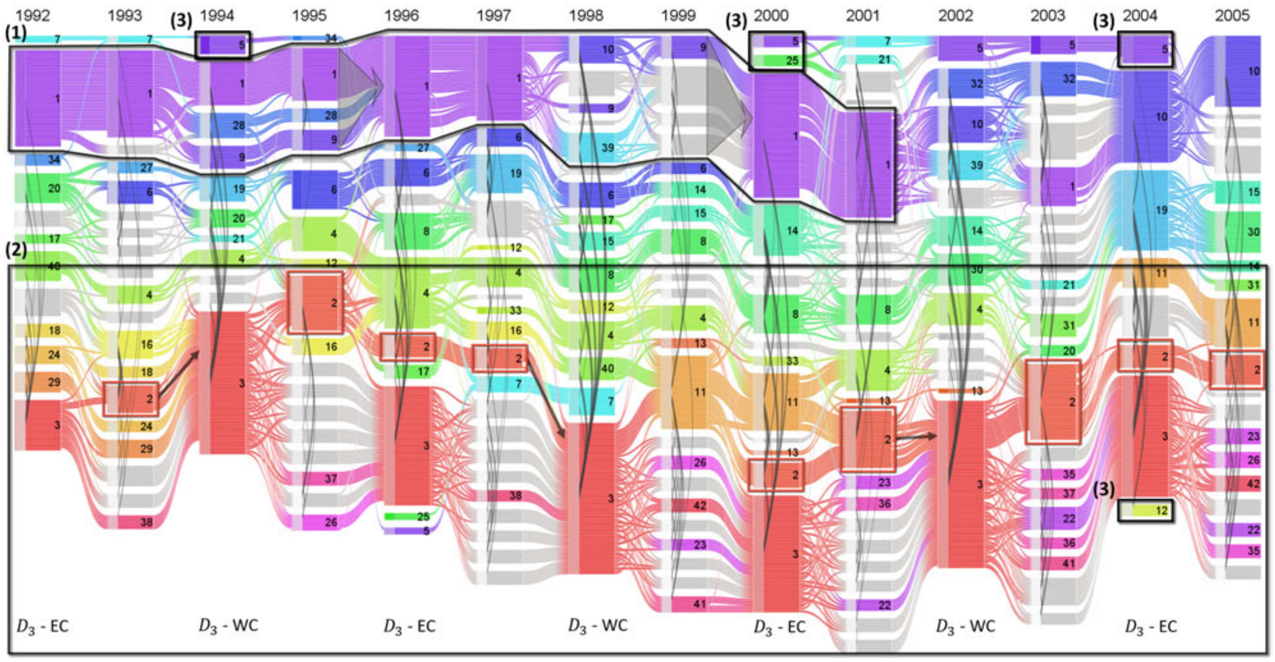

Figure 7: *Worldwide soccer matches from 1992 to 2005 of national teams, which are connected or grouped if they played against each other. Communities are labelled and coloured with respect to the dynamic community D_l they belong to (l), where dynamic communities were derived with $\theta = 0.3$.*

becomes lower from 1992 to 1994 and higher from 1994 to 1997)—the communities seem to be somewhat **incohesive**.

Re-merges after a single year are easier to spot, even without colour-coding. Nevertheless, the chosen colours instantly show a sequence of **re-merges** every even year for the big red community (D_3), which represents the European teams (Figure 7(2)). An explanation for this recurring visual signature is that either the World Cup (WC) (D_3 in 1994, 1998 and 2002) or the European Championship (EC) (D_3 in 1992, 1996, 2000 and 2004) happened in those years—European teams play many test matches against each other to prepare for the Championship in the respective year. For the odd years in between, the teams fall into groups determined by their randomly drawn continental qualification groups; as an example, we checked manually that the clusters in 2003 (the ones drawn below D_2) roughly coincide with the qualification groups of the 2004 European Championship. Unlike African or European teams, American teams form less stable communities: they can be found in the light red–orange–yellow–green communities in the middle of the diagram (D_2, D_4, D_{11}, D_{16} and others). It is interesting to note that a number of those teams, in particular teams of D_2, are regularly **merged** with European teams D_3 every 4 years whenever a World Cup takes place (1994, 1998, 2002) (the evolution of D_2 is highlighted in Figure 7(2)); interactive highlighting of communities confirms that these are roughly the same top teams every 4 years.

Since not all teams, in particular no teams of very small countries, play international soccer matches every year, some ribbons end or start somewhere in the middle of the timeline. In a few cases, also whole communities (nearly) become **extinct** (e.g. D_5 in 1994 or D_{12} in 2004) or **emerge** (e.g. D_5 and D_{25} in 2000 or D_{12} in 2004) (Figure 7(3)). **Gap** signatures, in contrast, cannot be found in their pure form because it is very unlikely that a community becomes **extinct** and then **emerges** in the same constellation after a while—only D_5, a cluster of South Sea islands, nearly completely disappears in 1994 and comes back in 2000.

6.2. Social network of conference attendees

The second application example studies the interaction of 113 conference participants at ACM Hypertext 2009. People are connected if they had face-to-face contact of at least 20 s (5870 edges in total). These data were recorded using RFID badges and is publicly available in anonymized form [ISB*11]. We divide the data by day and show the evolution of communities of participants over the 3 days of the conference in Figure 8. For this application example, we mainly used vertex stability colouring (stab(v), Figure 8) but also community stability colouring to find specifically stable vertices and communities. In Figure 8, both intra-community graphs and inter-community edges are superimposed on the *Community Evolution Layer*, but we have filtered out weak inter-community edges interactively to improve the readability of the diagram. In this example, communities $C_{t,k}$ are labelled with unique identifiers k.

The overall impression is that communities in this data set are more unstable than we have seen in the soccer example: in Figure 8, the colour-coded vertex stability shows that most vertices are contained in different dynamic communities over the three time steps (i.e. most vertices are coloured in light green). Only two dynamic communities are quite **stable** over the 3 days ($D_1 = \{C_{1,10}, C_{2,26}, C_{3,45}\}$ and $D_2 = \{C_{1,13}, C_{2,28}, C_{3,46}\}$)

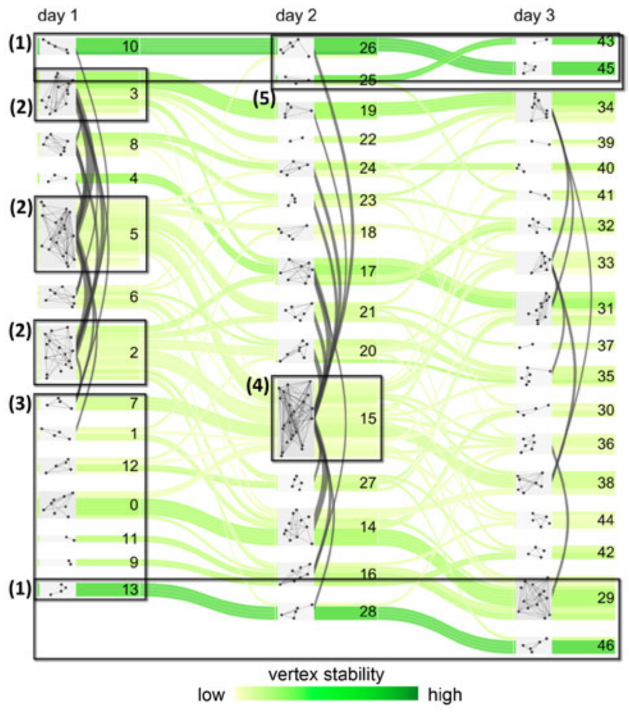

Figure 8: *Evolution of communities of conference attendees at 3-day ACM Hypertext 2009. Vertices (ribbons), here conference attendees, are coloured based on their stability (stab(v)).*

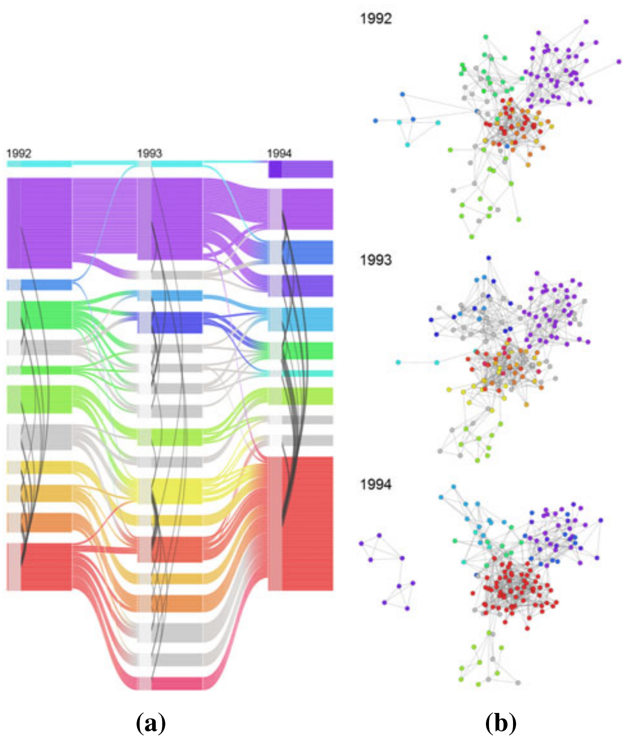

Figure 9: *Comparison of our approach with small multiples of node-link diagrams based on the first three time steps of the soccer data example (Section 6.1).*

(Figure 8(1)), which also becomes apparent using the community stability colouring. The inter-community edges show that both are only weakly **coupled** to one of the bigger communities. This indicates that these two groups of conference participants were quite **independent** from others. However, as the other communities are much more unstable, it seems as if attendees successfully connected to many people during this conference.

On the first day, three big communities are identified: $C_{1,2}$, $C_{1,3}$ and $C_{1,5}$ (Figure 8(2)). As the graph-related visual signatures tell, they are **cohesive** but not **independent** because they are considerably coupled among each other (this becomes apparent by the thick curves connecting $C_{1,3}$–$C_{1,5}$ and $C_{1,5}$–$C_{1,2}$) as well as to a number of smaller communities. Hence, there seemed to be a quite lively exchange among the conference participants on the first day. Only a few groups such as $C_{1,0}$, $C_{1,9}$, $C_{1,11}$, $C_{1,12}$ or $C_{1,13}$ seem to be almost **independent** because there are no strong inter-community links involving any of these communities (Figure 8(3)). On the second day, the visual signature of the communities somewhat changes: a central bigger community is identified ($C_{2,15}$), which is strongly **coupled** to nearly all other communities (Figure 8(4)). This community merges parts of the members of two of the bigger communities of the first day and forms a 'core'; it has a higher relative density $\rho(C_{t,k})$ than other communities (darker grey). The strong coupling of a single community ($C_{2,15}$) further suggests that conversations already started to settle while they had been more chaotic at the first day. However, also independent communities of the first day ($C_{1,0}$ and $C_{1,12}$) get integrated into the communication as they become connected to the central community $C_{2,15}$. A new community ($D_6 = \{C_{2,25}, C_{3,43}\}$) **emerges** on the second day (Figure 8(5)), that is, a group of people who missed the first day of the conference. On the last day, the groups fall into small, quite **independent** communities (there are almost no inter-community links)—conference attendees may have found their group of peers they preferred to talk to and the willingness to get in touch with other attendees might have had decreased.

7. Discussion

Dynamic graphs are commonly visualized using small multiples or animation of node-link diagrams [BBDW14]. Force-directed layout algorithms can reveal clusters: vertices that are highly connected are positioned close to each other. Besides spatial proximity, colour is often used to convey to which community a vertex belongs [APF*06]. We compared our visualization approach with small multiples and animation of node-link diagrams (see supplemental material for detailed comparison). Figure 9 shows the first three time steps of our first application example, the international soccer matches. Nodes are positioned based on a force-directed layout of the super-graph, that is, the union of all graphs G_t, of the four time steps. Similar to Figure 7, nodes are coloured based on the dynamic community they belong to at a given time step t (see Section 4.1.3).

Small multiples: To investigate the community membership of a single vertex over time, the respective node needs to be found in each node-link diagram—which is improved by constant node position but still ambiguous if several other nodes are positioned close—and its colour has to be compared over different time steps which can be difficult over long distances. To identify lifetime phenomena, communities have to be identified first—by mentally grouping all nodes of the same colour—and then they have to be compared with respect to their size and composition. To investigate a change of topology in the dynamic graph, that is, the appearance and disappearance of edges, is just as hard as investigating the community membership of a single vertex: the respective pair of nodes needs to be found in each node-link diagram to check whether they are connected by a link.

Animation: A switch of community membership of a single vertex between two successive time steps can be easily identified by focusing the respective node on the screen while browsing through time steps—a change in colour will indicate a change in membership. However, the analysis of community membership of individual vertices over the complete time period produces much more cognitive load as we need to remember earlier memberships and changes. Compared to small multiples, the identification of community lifetime phenomena poses an even more demanding task as we need to identify communities, remember them, and again compare them. Similar to the analysis of community membership of single vertices, the identification of topological changes is easy for two successive time steps but relatively hard for the complete time period because we have to remember earlier states.

Our approach: The investigation of community membership of a single vertex over time is easy: we simply need to check whether the colour along the ribbon changes between two successive time steps. Compared to animation, using our approach we can even identify changes in the context of the evolution, that is, it is easier to keep track of switches that happened at earlier time steps. This is because they are still visualized on the screen but—in comparison to small multiples—we do not need to re-identify a vertex for each time step as it is represented as a ribbon. Also the identification of lifetime phenomena is easier compared to small multiples and animation: the representation of communities as spatially divided blocks helps identify communities and the explicit visualization of transitions helps identify changes in the community structure. In our approach, the static graphs of individual time steps are represented using node-link diagrams, overlaid onto the *Community Evolution Layer*. The investigation of topological changes is therefore similar to small multiples: we need to identify the same pair of nodes within the overlaid node-link diagrams for each time step to check whether an edge was introduced or removed. Due to the restricted flexibility to layout the individual graphs, the readability of the static graphs is somewhat decreased. Alternatively to our layered node-link diagrams, each graph G_t could be visualized by representing each community as matrix and connecting them using links—similar to the NodeTrix approach [HFM07]. The difficulty in investigating topological changes would remain, but the densely connected clusters could be represented less cluttered. However, as graphs may contain communities of greatly varying sizes (see Figure 5), the horizontal area necessary for each time step would largely vary—and hence also the overall width of the visualization increase.

8. Conclusion and Future Work

We developed a visualization approach that combines a dynamic community structure with a dynamic graph in a single image and helps reveal typical life time phenomena of communities. Using our approach, users can estimate the reliability of the derived community structure and investigate whether community evolution interacts with changes in the graph topology. It is therefore possible to investigate whether events, such as merges or splits, were caused by noise or if they are due to drastic changes in the graph topology. We have discussed and demonstrated visual signatures that support these analyses.

The visualization benefits from ordering communities and vertices, which minimizes the number of crossings of transition edges between successive time steps. Our transition ranking approach helps deal with the remaining crossings and to analyse outliers, that is, unstable vertices in the evolution of community memberships: these are drawn last and therefore appear in the foreground. The colour assignment of communities and vertices further facilitates a detailed analysis of the evolution of communities. However, the colouring is not mandatory as communities can also be tracked based on their transition edges and labels; due to sorting, communities that belong to the same dynamic community are arranged at similar vertical positions. Hence, our visualization approach can also be used by people with colour vision deficiency.

The community structure scales well with respect to the number of nodes as the horizontal stripes and transition edges representing them are still clearly visible at small scale. Also, colour-coded relative graph density and aggregated inter-community edges provide a high-level overview of the dynamic graph. The available zooming and panning options as well as the interactive highlighting help investigate the evolution of communities even within larger data sets. However, to improve scalability, in particular with respect to the *Graph Layer*, further interaction techniques are necessary. These could include further filtering options and aggregation opportunities. Alternatively, a semantic zooming approach that allocates more space to a particular graph G_t or community $C_{t,k}$ by compressing other graphs and communities could be used. To analyse the efficiency of our approach, a comparative user study would be necessary.

Acknowledgements

This work was supported by DFG wtihin SFB 716/D.5.

References

[APF*06] Adamcsek B., Palla G., Farkas I. J., Derényi I., Vicsek T.: CFinder: Locating cliques and overlapping modules in biological networks. *Bioinformatics 22*, 8 (2006), 1021–1023.

[APP11] Archambault D., Purchase H. C., Pinaud B.: Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics 17* (2011), 539–552.

[APU09] Asur S., Parthasarathy S., Ucar D.: An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD) 3*, 4 (2009), article no. 16. doi: 10.1145/1631162.1631164.

[BBDW14] Beck F., Burch M., Diehl S., Weiskopf D.: The State of the Art in Visualizing Dynamic Graphs. EuroVis State-of-The-Art Report, Eurographics Association, 2014.

[BVB*11] Burch M., Vehlow C., Beck F., Diehl S., Weiskopf D.: Parallel edge splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2344–2353.

[CLT*11] Cui W., Liu S., Tan L., Shi C., Song Y., Gao Z., Qu H., Tong X.: Textflow: Towards better understanding of evolving topics in text. *IEEE Transactions onVisualization and Computer Graphics 17*, 12 (2011), 2412–2421.

[Col11] ColorBrewer. http://colorbrewer2.org/. Accessed May 2011.

[CWM09] Chuang J., Weiskopf D., Müller T.: Energy aware color sets. *Computer Graphics Forum 28*, 2 (2009), 203–211.

[FBS06] Falkowski T., Bartelheimer J., Spiliopoulou M.: Mining and visualizing the evolution of subgroups in social networks. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence* (Hong Kong, China, 2006), pp. 52–58.

[For10] Fortunato S.: Community detection in graphs. *Physics Reports 486*, 3–5 (2010), 75–174.

[FR91] Fruchterman T. M. J., Reingold E. M.: Graph drawing by force-directed placement. *Software: Practice and Experience 21*, 11 (1991), 1129–1164.

[FT04] Frishman Y., Tal A.: Dynamic drawing of clustered graphs. In *Proceedings of IEEE Symposium on Information Visualization* (Austin, TX, USA, 2004), IEEE, pp. 191–198.

[GBD09] Greilich M., Burch M., Diehl S.: Visualizing the evolution of compound digraphs with TimeArcTrees. *Computer Graphics Forum 28*, 3 (2009), 975–982.

[GDC11] Greene D., Doyle D., Cunningham P.: Tracking the Evolution of Communities in Dynamic Social Networks. Tech. rep., University College Dublin, 2011.

[GJ79] Garey M. R., Johnson D. S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, 1979.

[GK07] Gansner E. R., Koren Y.: Improved circular layouts. In *Proceedings of the 14th International Conference on Graph Drawing* (Karlsruhe, Germany, 2007), pp. 386–398.

[HFM07] Henry N., Fekete, J.-D., McGuffin M.: Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1302–1309.

[HKV12] Hu Y., Kobourov S. G., Veeramoni S.: Embedding, clustering and coloring for dynamic maps. In *Proceedings of the IEEE Pacific Visualization Symposium* (Songdo, South Korea, 2012), IEEE Computer Society, pp. 33–40.

[HvW08] Holten D., van Wijk J. J.: Visual comparison of hierarchically organized data. *Computer Graphics Forum 27*, 3 (2008), 759–766.

[ISB*11] Isella L., Stehlé, J., Barrat A., Cattuto C., Pinton, J.-F., Van den Broeck, W.: What's in a crowd? Analysis of face-to-face behavioral networks. *Journal of Theoretical Biology 271*, 1 (2011), 166–180.

[JRFL09] Jianu R., Rusu A., Fabian A., Laidlaw D.: A coloring solution to the edge crossing problem. In *Proceedings on 13th International Conference on Information Visualisation* (Barcelona, Spain, 2009), pp. 691–696.

[KBH06] Kosara R., Bendix F., Hauser H.: Parallel sets: Interactive exploration and visual analysis of categorical data. *IEEE Transactions on Visualization and Computer Graphics 12*, 4 (2006), 558–568.

[KG06] Kumar G., Garland M.: Visual exploration of complex time-varying graphs. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 805–812.

[Kof35] Koffka K.: Principles of Gestalt Psychology. Harcourt, Brace, 1935.

[LSC*07] Lin Y.-R., Sundaram H., Chi Y., Tatemura J., Tseng B. L.: Blog community discovery and evolution based on mutual awareness expansion. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence* (Silicon Valley, CA, USA, 2007), IEEE Computer Society, pp. 48–56.

[LSS*12] Lex A., Streit M., Schulz H.-J., Partl C., Schmalstieg D., Park P. J., Gehlenborg N.: Stratomex: Visual analysis of large-scale heterogeneous genomics data for cancer subtype characterization. *Computer Graphics Forum 31*, 3 (2012), 1175–1184.

[MELS95] Misue K., Eades P., Lai W., Sugiyama K.: Layout adjustment and the mental map. *Journal of Visual Languages & Computing 6*, 2 (1995), 183–210.

[Min69] Minards C.: Carte figurative des pertes succesives en hommes de l'armée francaise dans champagne de russie, 1869.

[MKH12] Mashima D., Kobourov S., Hu Y.: Visualizing dynamic data with maps. *IEEE Transactions on Visualization and Computer Graphics 18*, 9 (2012), 1424–1437.

[MUV01] Muñoz X., Unger W., Vrt'o I: One sided crossing minimization is NP-hard for sparse graphs. In *Proceedings of Graph Drawing* (Berlin, Heidelberg, 2001), Springer, pp. 115–123.

[OMB*07] Ogawa M., Ma K., Bird C., Devanbu P., Gourley A.: Visualizing social interaction in open source software projects.

In *Proceedings of 6th International Asia-Pacific Symposium on Visualization* (Sydney, Australia, 2007), pp. 25–32.

[PGU12] Pilhöfer A., Gribov A., Unwin A.: Comparing clusterings using bertin's idea. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (2012), 2506–2515.

[PXY*05] Phan D., Xiao L., Yeh R., Hanrahan P., Winograd T.: Flow Map Layout. In *Proceedings of the Symposium on Information Visualization* (Minneapolis, MN, USA, 2005), IEEE Computer Society, p. 29.

[RB08] Rosvall M., Bergstrom C. T.: Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences 105*, 4 (2008), 1118–1123.

[RB10] Rosvall M., Bergstrom C.: Mapping change in large networks. *PloS One 5*, 1 (2010), e8694.

[RHF05] Riehmann P., Hanfler M., Froehlich B.: Interactive Sankey Diagrams. In *Proceeding of the Symposium on Information Visualization* (Minneapolis, MN, USA, 2005), IEEE, pp. 233–240.

[RM13] Rufiange S., McGuffin M.: DiffAni: Visualizing dynamic graphs with a hybrid of difference maps and animation. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2556–2565.

[RPD09] Reitz F., Pohl M., Diehl S.: Focused Animation of Dynamic Compound Graphs. In *Proceedings of the 13th International Conference on Information Visualisation* (Barcelona, Spain, 2009), IEEE Computer Society, pp. 679–684.

[RTJ*11] Reda K., Tantipathananandh C., Johnson A., Leigh J., Berger-Wolf T.: Visualizing the evolution of community structures in dynamic social networks. *Computer Graphics Forum 3* (2011), 1061–1070.

[SPB10] Sallaberry A., Pecheur N., Bringay S., Roche M., Teisseire M.: SequencesViewer: Visualisation de séquences ordonnées de gènes ou comment rendre accessible des motifs séquentiels trop nombreux? In *Extraction et gestion des connaissances (EGC'2010)* (Hammamet, Tunisie, 2010), pp. 387–392.

[STT81] Sugiyama K., Tagawa S., Toda M.: Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cybernetics 11*, 2 (1981), 109–125.

[TFSZ11] Takaffoli M., Fagnan J., Sangi F., Zaiane O.: Tracking changes in dynamic information networks. In *Proceedings of International Conference on Computational Aspects of Social Networks* (Salamanca, Spain, 2011), pp. 94–101.

[ZFMAQ13] Zeng W., Fu C.-W., Müller Arisona, S., Qu H.: Visualizing interchange patterns in massive movement data. *Computer Graphics Forum 32* (2013), 271–280.

Supporting Information

Additional Supporting Information may be found in the online version of this article at the publisher's web site:

S1: International Soccer Matches.

S2: Social Network of Conference Attendees.

S3: Comparison Our approach and common node-link visualizations of dynamic graphs.

S4: High resolution image for the application example "International Soccer Matches".

S5: High resolution image for the application example "Social Network of Conference Attendees".